Modeling of the human trunk: the impact of stiffness gain and lumbar lordosis on stability

By

Valerie Jardon

Submitted to the University of Kansas’ Bioengineering graduate program and faculty in partial fulfillment of the Master of Science in Bioengineering degree

________________________________________
Sara E. Wilson, Ph.D, Chair

________________________________________
Elizabeth A. Friis, Ph.D

________________________________________
Carl W. Luchies, Ph.D

Date Defended: May 4, 2022
The thesis committee chair for Valerie Jardon certifies that this is the approved version of the following thesis:

Modeling of the human trunk: the impact of stiffness gain and lumbar lordosis on stability

________________________
Sara E. Wilson, Ph.D, Chair

Date Approved: May 13, 2022
Abstract

Low back pain and injuries are prevalent and costly musculoskeletal conditions in our society, afflicting most Americans at some point throughout their lifetime. In an effort to develop effective treatment and rehabilitation methods, researchers have continued in their investigation of the potential risk factors and causes of low back pain through use of spine models and experimental data collection.

The focus of this work is the development and utilization of an 18 degree-of-freedom stability-based trunk model with 90 muscle fascicles, reflexes, and a lumped parameter intervertebral disc model to explore the potential impacts stiffness gain magnitude and lumbar lordosis angle have on the model's predictions for stability. Throughout the development of this model, a thorough verification procedure was utilized to minimize errors.

It was determined that as stiffness gain in the short-range muscle stiffness model increased from 0.9 to 40, the magnitude of the required metabolic power and muscle force to stabilize the system decreased. Above stiffness gains with a magnitude of 20, increasing the stiffness gains had little impact on the required metabolic power and muscle force. Additionally, for a stiffness gain of 0.5, the model predicted instability whereas all stiffness gains greater than or equal to 0.9 resulted in a stabilized system. It is evident that stiffness gain has the ability to influence model predictions, including the required metabolic power and muscle force.

In our investigation with lumbar lordosis, it was determined that the hyperlordosis case required more metabolic power and additional recruitment of the flexor muscles in order to stabilize in comparison to the other lordosis cases. The hypolordosis cases required less metabolic power and additional recruitment of extensor muscles to stabilize in comparison to the other lordosis cases. The additional recruitment of the flexor muscles needed to stabilize the hyperlordotic spine and the
addition of the extensor muscles needed to stabilize the hypolordotic spine can be explained by the line of gravity being positioned posteriorly and anteriorly for the lordosis cases respectively.

Future investigations should further explore the impact of lumbar lordosis angle on spine stability, especially experimentally and with EMG-based models, and should consider investigating the impacts of varying stiffness gains on spinal stability. Lastly, additional loading tasks should be simulated with the model, such as asymmetric tasks, differing load magnitudes and differing load application points.
Acknowledgements

I would like to extend my appreciation and thanks to all that have supported me and helped me along the way throughout this research project and my graduate education.

First, I would like to thank Dr. Wilson for her continuous guidance, support, and patience – none of this would have been possible without her. I cannot thank Dr. Wilson enough for her valuable insight and advice, all the knowledge she has shared with me, and her willingness and dedication to support her students in our research, academic and professional endeavors. Additionally, I would like to thank my committee members Dr. Friis and Dr. Luchies for all of their support and the knowledge they have shared with me throughout my undergraduate and graduate career. I truly couldn’t have accomplished this without them.

I would also like to thank Mukui Mutunga for her impactful role as my research mentor throughout my undergraduate career. I am so thankful to have had a mentor so willing to share her passion, kindness, insight, and knowledge. I am thankful that you are still a part of my research journey today. Additionally, I would like to thank Tong Chu for her continuous encouragement and contributions, specifically her dedication, diligence and assistance in literature reviewing. I am thankful to have been a part of her research journey and grateful to have been her mentor!

I would like to thank my fiancé Andrew Yancey for his continuous support and encouragement throughout this research project. During the pandemic, he truly was my acting lab mate, and I am so grateful to have had him working alongside me. I am thankful for his guidance, his willingness to listen, and for his continuous encouragement throughout this journey. Additionally, I would like to thank my family for all they have done to support me and my passions. I am so thankful to have so many wonderful people encouraging me, supporting me, and inspiring me.

Lastly, I would like to thank the faculty of the KU Bioengineering department, Mechanical Engineering department, and the School of Music for all their support, instruction and for opportunities I have been granted. I am truly blessed to have such a wonderful community at KU.
Table of Contents

Chapter 1: Introduction ................................................................................................................................................................................................. 1

1.1 What causes low back pain and injury? .......................................................................................................................................................... 1

1.1.1 Prevalence of Low Back Pain............................................................................................................................................................... 1

1.1.2 Modeling of the Human Trunk ............................................................................................................................................................ 1

1.2 Background: Stability-based Spine Models and Utilization .................................................................................................................. 3

1.2.1 The importance of spine modeling in the investigation of low back pain ............................................................................................. 3

1.2.2 The importance of spine stability analysis in the investigation of low back pain .................................................................................. 4

1.2.3 Stability-based Trunk Models in Literature ....................................................................................................................................... 5

1.2.4 Summary..................................................................................................................................................................................................... 13

1.3 Model Development: Based on the Work of Franklin et al. .................................................................................................................. 15

1.3.1 Model Summary.................................................................................................................................................................................................. 15

1.4 Current Investigation .................................................................................................................................................................................................. 16

1.4.1 Bergmark's Model of Short-range Stiffness ........................................................................................................................................... 16

1.4.2 Hyperlordosis and Hypolordosis ............................................................................................................................................................. 17

1.5 Specific Aims and Hypotheses ........................................................................................................................................................................ 18

Chapter 2: Model Development ............................................................................................................................................................................ 20

2.1 Rigid Bodies........................................................................................................................................................................................................ 20

2.1.1 Dimensions................................................................................................................................................................................................... 22

2.1.2 Body Origins.................................................................................................................................................................................................. 25

2.1.3 Mass and Mass Moment of Inertia ......................................................................................................................................................... 26

2.2 Rotations ........................................................................................................................................................................................................ 27

2.2.1 Lordosis Angles............................................................................................................................................................................................ 27

2.3 Intervertebral Discs ..................................................................................................................................................................................... 28

2.4 Muscle Anatomy .................................................................................................................................................................................................. 30

2.4.1 Calculating Transformed Muscle Anatomy .......................................................................................................................................... 30

2.4.2 Calculating Muscle Length .................................................................................................................................................................. 31

2.4.3 Calculating Muscle Velocity ................................................................................................................................................................ 32
Chapter 3: Code Verification

3.1 Model Parameters

3.2 Initial Transformation of Cholewicki et al. Muscle and Skeletal Coordinates

3.2.1 Muscle and Skeletal Coordinates Verification

3.2.2 Muscle Path Verification

3.3 Import Muscle Geometry and Skeletal Coordinates

3.4 Import External Force

3.5 Rotation Matrices

3.6 Body Origins

3.7 Transform Muscle Geometry and Skeletal Coordinates

3.8 Muscle Properties

3.9 Transform External Force

3.10 Dynamics

3.11 Intervertebral Discs Moments

3.12 Generalized Forces

3.13 Optimization

3.13.1 Constraint Variables

3.13.2 Jacobians

3.13.3 Cost Function

3.14 Muscle Model

3.15 Reflex Model

3.16 Time Delay Analysis and Nonlinear Verification

3.17 Summary
Chapter 4: The impact of stiffness gain on stability ............................................................... 81
Chapter 5: The effects of hyperlordosis and hydolordosis on spine stability ......................... 104
Chapter 6: Conclusions and Future Work .............................................................................. 138
6.1 Model Verification Procedure ...................................................................................... 138
6.2 Stiffness Gain and Spine Stability ................................................................................. 139
6.3 Lordosis Angles and Spine Stability ............................................................................. 139
6.4 Additional Future Work ............................................................................................... 140
Appendix A ........................................................................................................................ 149
A.1 Moment of Inertia Equations ....................................................................................... 149
A.2 Lordosis Angles ............................................................................................................ 150
A.3 Skeletal and Muscle Coordinates ................................................................................. 152
A.4 Rotations ....................................................................................................................... 162
Appendix B ........................................................................................................................ 164
B.1 Code Directory ............................................................................................................. 165
B.2 MAT-file Directory ......................................................................................................... 166
B.3 Personal Communication Summary ............................................................................. 166
Appendix C: Main MATLAB Files ....................................................................................... 170
C.1 spine_main.m: Main Spine File ..................................................................................... 171
C.2 For_verification.m: Verification Tasks in MATLAB ..................................................... 176
C.3 model_parameters.m: Model Parameters ................................................................. 185
C.4 rotationmatrices.mlX: Determine the Rotation Matrices ................................................. 189
C.5 origin_body.mlX: Define origins of the bodies ............................................................ 194
C.6 transform_Cholewicki.mlX: Import and Transform Skeletal Geometry ........................ 197
C.7 load_musclefile.mlX: Import Muscle Anatomy ............................................................ 199
C.8 load_forcefile.mlX: Import External Force File ............................................................ 202
C.9 rotate_anatomy_sym.mlX: Transform Muscle File, Calculate Muscle & Velocity ....... 204
C.10 rotateforce.mlX: Transform External Force File ....................................................... 209
C.11 dynam.mlX: Dynamics .................................................................................................. 211
C.12 generalizedForce.mlX: Generalized Forces ............................................................... 217
C.13 optimize.mlX: Optimization ....................................................................................... 220
C.13.1 Main File .................................................................................................................. 220
C.13.2 Angular Acceleration Functions ................................................................. 224
spineaccP.m file utilized for equilibrium constraint .............................................. 224
spineaccA.m file utilized for equilibrium constraint .............................................. 225
spineacc.m file utilized for equilibrium constraint .............................................. 227
C.13.3 run_fmincon.mlx: Constrained Optimization Solver fmincon (MATLAB) .......... 229
C.13.4 costfun.m: Metabolic Power Function for Constrained Optimization Procedure ........ 230
C.13.5 nlcon.m: Nonlinear Constraints for Constrained Optimization Procedure .......... 231
C.14 musclemodel.mlx: Muscle Model .................................................................. 232
C.15 musclemodelR.mlx: Reflex Model ................................................................. 233
C.16 IVD_calcs.mlx: Intervertebral Disc Moments ................................................ 234
C.17 Time Delay Analysis .................................................................................... 237
C.18 Nonlinear Verification ................................................................................... 238
  C.18.1 Run Nonlinear Simulation ........................................................................ 238
  C.18.2 Spinedyn.mlx: Function Handle for Delay Differential Equation Solver ............ 239
  C.18.3 Determine the Distance of the Normalized State-Space from Equilibrium .......... 240
Appendix D : Additional MATLAB Files .................................................................. 241
  D.1 replace_sym.mlx: Replace symbolic variables with numeric .......................... 241
  D.2 vect2ang.m: Assign angles from input angle vectors to individual variables for function input 242
  D.3 sub_timevar.mlx: Replaces angles in variables with time-dependent angles ............ 243
  D.4 tofuncts.m: Convert Symbolic Outputs to Function Handles .......................... 244
## List of Figures

- **Figure 1-1**: Cholewicki and McGill Model Utilization and Influence.................................9
- **Figure 1-2**: Granata & Wilson Utilization & Influence ..................................................11
- **Figure 1-3**: The expansion of the Wagner et al. model\textsuperscript{13}........................................13
- **Figure 1-4**: Model Process Summary ..............................................................................15
- **Figure 1-5**: Visual of normal lordosis, hyperlordosis and hypolordosis in the lumbar spine region ....18
- **Figure 2-1**: Model rigid bodies......................................................................................20
- **Figure 2-2**: Vertebral body diameters. Not drawn to scale ..............................................22
- **Figure 2-3**: Calculating the required vertebral height ....................................................24
- **Figure 2-4**: Front view of vertebra. Origin defined at inferior vertebral surface.................25
- **Figure 2-5**: Front view of vertebra. Center of mass depicted by red circle .......................26
- **Figure 2-6**: N basis vectors affixed to the ground. Axis 2 points anteriorly .......................27
- **Figure 2-7**: Planar rotation .........................................................................................29
- **Figure 2-8**: Twist rotation ............................................................................................29
- **Figure 2-9**: Muscle Anatomy .........................................................................................30
- **Figure 2-10**: Muscle attachment points visualization ..................................................31
- **Figure 2-11**: Hill-type muscle model .............................................................................33
- **Figure 3-1**: Initializing the verification procedure in spine\_main.m file .........................48
- **Figure 3-2**: Example of exporting verification task results to Excel spreadsheet ..........49
- **Figure 3-3**: "Removing" the default lordosis angles: Transforming coordinates to align the connection points of the vertebrae along Axis 3 ..............................................................52
- **Figure 3-4**: Determining Franklin's initial coordinates prior to transformation for selected attachment points. ................................................................................................................53
- **Figure 3-5**: Verification Tasks 2.2 and 2.4 .....................................................................56
- **Figure 3-6**: Verification Task 2.5 Absolute Error ............................................................56
- **Figure 3-7**: Verification Task 3.2 ...................................................................................57
- **Figure 3-8**: Verification Task 3.2 Absolute Error ...........................................................57
- **Figure 3-9**: Verification Task 4.1 ....................................................................................59
- **Figure 3-10**: Verification Task 4.1 Absolute Errors (presented in the MATLAB command window).........60
- **Figure 3-11**: Verification Task 5.1 ..................................................................................60
- **Figure 3-12**: Verification Task 5.1 Absolute Error ..........................................................61
- **Figure 3-13**: Verification Task 6.1 ..................................................................................62
- **Figure 3-14**: Verification Task 6.1 Absolute Error from MATLAB Command Window ........62
- **Figure 3-15**: Verification Task 7.1 ..................................................................................62
- **Figure 3-16**: Verification Task 7.1 Absolute Error for muscle length ................................62
- **Figure 3-17**: Verification Task 7.1 Absolute Error for muscle velocity ................................63
- **Figure 3-18**: Verification Task 8.1 ..................................................................................63
- **Figure 3-19**: Verification Task 8.1 Absolute error presented in the MATLAB command window ....63
- **Figure 3-20**: Verification Task 9.1 ..................................................................................64
- **Figure 3-21**: Verification Task 9.1 Absolute error ..........................................................64

x
Figure 3-22: Verification Task 9.2 Absolute Error for Gravity Vector ................................................................. 65
Figure 3-23: Verification Task 9.2 Absolute Error for Mass Matrix ................................................................. 65
Figure 3-24: Verification Task 9.2 Absolute Error for Aeq ................................................................. 65
Figure 3-25: Verification Task 9.4 Absolute Error for B ...................................................................................... 66
Figure 3-26: Verification Task 9.4 Absolute Error for beq .................................................................................. 66
Figure 3-27: Verification Task 9.4 Absolute Error for Gr ................................................................................... 67
Figure 3-28: Verification Task 10.1 in the IVD_calcs.mlx script ........................................................................... 68
Figure 3-29: Verification task 10.2 in the IVD_calcs.mlx script ........................................................................... 68
Figure 3-30: Verification Task 11.1 Absolute Error ......................................................................................... 69
Figure 3-31: Verification Task 11.1 Absolute Error ......................................................................................... 69
Figure 3-32: Verification Task 12.1 Absolute Error ......................................................................................... 70
Figure 3-33: Continued verification of 12.1 task ............................................................................................... 71
Figure 3-34: Verification Task 12.1 Absolute Error for Aeq ............................................................................. 71
Figure 3-35: Verification Task 12.1 Absolute Error for beq ............................................................................. 71
Figure 3-36: Verification Task 12.1 Absolute Error for B ................................................................................... 72
Figure 3-37: Verification Task 12.1 Absolute Error for A ................................................................................... 72
Figure 3-38: Verification Task 12.5 Absolute Error ......................................................................................... 73
Figure 3-39: Verification Task 12.3 Absolute Error ......................................................................................... 73
Figure 3-40: Verification Task 12.3 Absolute Error ......................................................................................... 74
Figure 3-41: Verification Task 12.4 Absolute Error ......................................................................................... 74
Figure 3-42: Verification Task 12.4 Absolute Error ......................................................................................... 75
Figure 3-43: Verification Task 12.6 Absolute Error ......................................................................................... 75
Figure 3-44: Verification Task 12.6 Absolute Error ......................................................................................... 75
Figure 3-45: Verification task 13.1 Absolute Error ......................................................................................... 76
Figure 3-46: Absolute error for muscle force ................................................................................................. 76
Figure 3-47: Dss for: simulated delay = 0.99 * delay margin ........................................................................... 77
Figure 3-48: Dss for: simulated delay = 0.75 * delay margin ........................................................................... 78
Figure 3-49: Delay Margin * 0.5 ............................................................................................................... 78
Figure 3-50: Dss for: simulated delay = 1.25 * delay margin ........................................................................... 79
Figure 3-51: Dss for: simulated delay = delay margin ................................................................................... 79

Figure 4-1: Spine model. This model includes five rigid lumbar vertebrae, one rigid body representing the cervical and thoracic spine regions, and a fixed pelvis. 90 muscle fascicles are included. The intervertebral discs are represented by a lumped parameter model ......................................................... 100

Figure 4-2: The relationship between required metabolic power and stiffness gain are presented for a simulation including an external load of 200 N. This load was applied at the T4 level and 20 cm anterior to the trunk. For these simulations, GP = 10, GD = 0 and b ................................................................. 101

Figure 4-3: The relationship between the required muscle force to stabilize the system and stiffness gain is presented. For these simulations, an external load of 200 N was applied at the T4 level and 20 cm anterior to the trunk and GP = 10, GD = 0 and b = 2. The ................................................................. 102
Figure 5-1: The current model incorporates 90 muscle fascicles, one rigid body for each of the lumbar vertebrae and one rigid body representing the thoracic and cervical region. A lumped parameter model is utilized to represent the intervertebral discs...

Figure 5-2: The appearance of hyperlordosis, hypolordosis and normal lordosis in the lumbar spine...

Figure 5-3: The metabolic power required to stabilize each lordosis case. One simulation was unloaded while the other was loaded with a 200 N load anterior to the trunk.

Figure 5-4: Muscle activation for the right-side muscles for the unloaded case

Figure 5-5: The distance of the normalized state-space from equilibrium versus time for the unloaded hyperlordotic spine

Figure 5-6: The distance of the normalized state-space from equilibrium versus time for the unloaded hypolordotic spine

Figure 5-7: The distance of the normalized state-space from equilibrium versus time for the unloaded spine with normal lordosis

Figure 5-8: The distance of the normalized state-space from equilibrium versus time for the hyperlordotic spine with a 200 N load applied anteriorly at the T4 level

Figure 5-9: The distance of the normalized state-space from equilibrium versus time for the hypolordotic spine with a 200 N load applied anteriorly at the T4 level

Figure 5-10: The distance of the normalized state-space from equilibrium versus time for the spine with normal lordosis and a 200 N load applied anteriorly at the T4 level

Figure A-1: Elliptical Cylinder Top View

Figure A-2: Lordosis angle applied around Axis 1

Figure A-3: Franklin vs Cholewicki Axes Definition. Franklin Axis 2 and Cholewicki X Axis point anteriorly

Figure A-4: Rotation around Axis 1

Figure A-5: Rotation around Axis 2

Figure A-6: Rotation around Axis 3

Figure B-1: General Model Flowchart
List of Tables

Table 1-1: Overview of pivotal stability-based trunk models ................................................................. 14
Table 1-2: Magnitude of the stiffness gains in literature ........................................................................ 17
Table 2-1: Spine segment properties ....................................................................................................... 21
Table 3-1: Verification Tasks and Results .............................................................................................. 44
Table 3-2: Franklin muscle paths described in appendix for left side of the body .............................. 54
Table 3-3: Cholewicki and McGill muscle paths described in appendix for left side of the body ........ 55
Table 4-1: Evidence of the varying stiffness gain q magnitudes utilized in previous investigations ...... 103
Table 5-1: The percentage of metabolic power required to recruit certain muscle groups to achieve stability in the unloaded cases .......................................................................................... 136
Table 5-2: Critical stiffness for the hypolordosis (32.1˚), normal lordosis (58.1˚), and hyperlordosis (84.1˚) cases for simulations with a 200 N load applied at level T4 and 20 cm anterior to the trunk ..... 137
Table A-1: Development of the Normal Lordosis Case .............................................................................. 150
Table A-2: Segmental lordosis angles for all lordosis cases studied in Chapter 5 ................................. 151
Table A-3: Skeletal Geometry Coordinates ............................................................................................. 153
Table A-4: Muscle Attachment Points .................................................................................................... 158
Table B-1: Code Directory of the scripts utilized in the model with a general description of their purpose ................................................................................................................................. 165
Table B-2: MAT-file directory for the .mat files utilized in the model. Many outputs of functions were saved as symbolic .mat files, allowing for these outputs to be updated to function handles in the tofuncts.m file ................................................................................................................................. 166
Table B-3: Summary of the MATLAB files received from Dr. Michael Madigan and the files Franklin has published in his thesis ........................................................................................................ 167
List of Abbreviations

AP diameter: Diameter in the anterior-posterior direction

AP radius: Radius in the anterior-posterior direction

DOF: Degrees of freedom

EMG: Electromyography

FEA: Finite Element Analysis

FEM: Finite Element Model

IVD: Intervertebral disc

KE: Kinetic Energy

LBP: Low back pain

LLA: Lumbar lordosis angle

ML diameter: Diameter in the medio-lateral direction

ML radius: Radius in the medio-lateral direction

PE: Potential Energy
Chapter 1: Introduction

1.1 What causes low back pain and injury?

1.1.1 Prevalence of Low Back Pain

Low back pain (LBP) and injuries place significant financial and physical burdens on Americans. The estimated cost of low back pain in the United States surpasses 100 billion dollars annually due to the need for medical intervention and lost income\(^1\). According to the 2015 National Health Interview Survey, approximately one out of four individuals reported experiencing low back pain within a span of three months\(^2\).

The causes of low back pain and injuries have not been fully identified, but researchers have progressed in their understanding of the subject. Researchers have concluded that risk factors, history of low back pain\(^3\), differences in muscle stiffness and recruitment strategies\(^4-6\), and impaired reflexes\(^7,8\) may contribute to an individual’s likelihood to experience low back pain or injury; this list is not exhaustive but rather indicative of the progress that has been made in this area of study. Low back pain and injury is likely not the consequence of one simple cause, resulting in researchers considering and investigating a multitude of parameters that may impact spine behavior and potential injury. The investigation of the causes of low back pain is on-going and is crucial to the development of prevention strategies, and the advancement of rehabilitation techniques and treatment methods.

1.1.2 Modeling of the Human Trunk

Various biomechanical trunk models have been developed to further investigate the behavior of the lumbar spine and improve our understanding of the causes of low back pain and spinal injury\(^9\). Researchers have focused on characteristics such as spinal loading, buckling, spine stability, and muscle
recruitment patterns, but the methods used to model the lumbar spine may impact the model’s ability to adequately predict spine behavior.

For example, in literature, biomechanical models vary in complexity and anatomical accuracy. Some researchers have represented the spine as a single pendulum with a pair of antagonistic muscles\textsuperscript{10–12} or as a double inverted pendula model with more than one muscle pair\textsuperscript{13,14}. This representation is simple in comparison to those that have modeled the spine with greater degrees of freedom and more realistic muscle architecture\textsuperscript{15–17}. It has been determined that representing muscle paths as curved more accurately models the true behavior\textsuperscript{18}; despite this conclusion, some researchers still implement straight muscle paths for simplicity. While more complex models incorporate more realistic anatomy, the development of these models is more elaborate, resulting in the increased likelihood of possible unidentified errors. Additionally, in the complex models, it may be difficult to determine the specific or root cause of the system’s behavior.

For models that incorporate many muscles, the method for determining the muscle activation for each muscle can have limitations. Researchers can employ an optimization procedure that solves for the activation of each muscle, but this has been found to not always adequately predict the coactivation of the muscles\textsuperscript{9,19} and/or its magnitude\textsuperscript{13}. Coactivation has been proven to be crucial to spine stability and will result in an increased compressive load\textsuperscript{13,19,20}. Other researchers utilize EMG-assisted or EMG-assisted optimization procedures to determine muscle activation. These methods have been found to better represent coactivation but introduce errors due to the limitations of surface EMG sensors, such as not being able to directly measure the activation of deep muscles\textsuperscript{9,15}.

In the literature, the classification of spine “stability” and “instability” has been shown to be inconsistent in use across researchers\textsuperscript{21,22}. In this context, the behavior of the spine is of interest prior to and following an applied perturbation\textsuperscript{21}. If the spine is unable to converge to its initial position within a
reasonable range, then this indicates instability of the spine and injury potential\textsuperscript{21}. In the past, some researchers have neglected to include stability requirements in their models and have instead focused solely on the stiffness needed for the system to be in equilibrium following a perturbation. It has since been proven that a spinal system in equilibrium does not necessarily correlate to a stable system\textsuperscript{13,23}, and that when a stability constraint is included in the model, the predicted muscle activations more accurately represent those found in experimental studies\textsuperscript{13}.

Commonly, researchers have neglected to include reflexes in models due to their time-delayed properties and the added complexity of their implementation\textsuperscript{9,24}. It has been determined that for certain loading conditions, reflexes are required for successful spine stabilization\textsuperscript{16}, and that while some loads can be stabilized solely by intrinsic stiffness, reflexes allow for the same loads to be stabilized with less coactivation required\textsuperscript{16,17,25}. Additionally, individuals with low back pain have exhibited a greater reflex delay in comparison to healthy controls\textsuperscript{4,8}. The consequence of this greater delay may be instability or injury, as the reflexes may be too delayed to aid in spine stabilization strategies and the system may be unable to compensate elsewhere to stabilize\textsuperscript{21}. Impaired reflex response has been associated with a heightened risk of LBP and injuries\textsuperscript{3}. Researchers have shown that reflexes impact spine behavior and should continue to be investigated experimentally and through modelling for the investigation of LBP.

Lastly, model predictions need to be verified with experimental findings when possible. For spine model predictions to be beneficial, it is crucial for models to predict realistic spine behavior. If unable to do so, refinement of the model should be explored.

\section*{1.2 Background: Stability-based Spine Models and Utilization}

\subsection*{1.2.1 The importance of spine modeling in the investigation of low back pain}

The complexity of the human spine makes it difficult to isolate certain components of the system and deduce their sole impact on spine behavior with confidence. This directly complicates the ongoing
investigation of the causes of low back pain. Experimental studies have provided researchers with vast information and insight to spine properties and movement, such as spine buckling, muscle activations in loading conditions, and vertebral and intervertebral disc (IVD) dimensions. For some of this information, like muscle recruitment strategies, it may be difficult to determine the specific mechanism or mechanisms that explain why this certain behavior occurred. Experimental studies are crucial for understanding how the spinal system reacts to loading conditions, whereas modelling allows for more insight as to why the system may have responded this way. Thus, it is critical that modeling and experimental data collection continue to be utilized in tandem to further our understanding of the mechanisms behind low back pain.

1.2.2 The importance of spine stability analysis in the investigation of low back pain

Recall that the concept of spine stability is based on the spinal system’s ability to return to equilibrium within a reasonable range following a perturbation\(^{21}\). If unable to do so, this can result in potential LBP or injury. Additionally, it has become evident that when perturbed, the muscle recruitment strategy utilized is influenced by the requirements for spine stability\(^{20}\) and when stability constraints are utilized in trunk models, the predicted activations better match those measured experimentally\(^{13}\). Thus, spine stability should continue to be considered and evaluated in future studies.

As mentioned previously, researchers have addressed the spine as a mechanical system, evaluating the components that contribute to its stability and factors that may result in its buckling\(^{18,26}\). Crisco \textit{et al.}\(^{27}\) determined that a compressive load of approximately 88 N will result in spine buckling, indicating that the spine itself cannot be the only contributor to spine stability\(^{28}\). To avoid buckling, muscles and reflexes are recruited to combat applied loads, maintain posture, and minimize injury potential. Using electromyography (EMG), researchers can measure the activation of the muscle groups in subjects, providing insight to the muscle recruitment strategies utilized by the subjects in different
loading conditions. This has proven to be useful in the comparison between healthy controls and those with LBP. Experimental studies have revealed that individuals with chronic low back pain have an increased reflex time-delay in comparison to healthy controls, which can adversely affect spine stability\textsuperscript{4,7,11}. This may result in individuals with chronic LBP utilizing different muscle recruitment strategies in an attempt to stabilize the spine by other means\textsuperscript{4,6}. In the next section, an overview of spine modeling will be provided and the seminal stability-based trunk models in literature will be discussed.

1.2.3 Stability-based Trunk Models in Literature

Spine models have varied in their utilization, development, and characteristics throughout history. First, equilibrium-based models will be discussed. These models focus on the balancing of forces and moments. For static analysis:

\[ \Sigma Forces = 0 \]  
\[ \Sigma Moments = 0 \]

And for dynamic analysis:

\[ \Sigma Forces = ma \]
\[ \Sigma Moments = la \]

Where \textit{m} is mass, \textit{a} is acceleration, \textit{l} is the mass moment of inertia and \textit{\alpha} is angular acceleration.

Finite element models and link-segment models have often been developed and utilized for equilibrium analysis of the spine\textsuperscript{9}. While these equilibrium-based models assist in the understanding of spine loading, the structural properties of the spine are not evaluated\textsuperscript{9}. To evaluate these structural properties, stability analysis has been implemented into trunk models in addition to the equilibrium
analysis. This implementation of stability analysis in trunk models was validated by a study completed by Granata and Orishimo which indicated that a spine in equilibrium does not necessarily reflect a stable spinal system, and that muscle recruitment strategies obtained experimentally more accurately matched those predicted for a system with a stability requirement\textsuperscript{20}.

Many trunk models were expanded to include stability analysis. These stability-based trunk models often implement the Minimum Potential Energy Method based on the Hessian matrix or Linear Stability Analysis based on the Jacobian matrix. For a system to be stable for the Minimum Potential Energy Method, the principal minors and the determinant of the Hessian matrix need to be positive\textsuperscript{15}. For Linear Stability Analysis, the eigenvalues of the Jacobian must have negative real parts for a system to be stable\textsuperscript{17}. Trunk models evaluating stability commonly consist of an inverted pendulum with a pair of antagonistic muscles or stacked inverted pendula where a single pendulum is used to represent each of the lumbar vertebrae. For the models that incorporate the stacked pendula, a more-detailed muscle architecture is included which allows for the roles of each of the muscle groups to be investigated more clearly.

With these trunk models, researchers often use optimization procedures, measured EMG from subjects, or EMG-based optimization procedures to determine the muscle activation utilized for an assigned task. Often, researchers utilize EMG to measure the muscle activations of human subjects that can be utilized for the model. This allows for realistic muscle recruitment strategies to be utilized in the model as subjects may utilize different strategies for the same task. Optimization procedures that do not require subject input are useful in that the muscle activation necessary to satisfy stability and equilibrium requirements can be determined in an efficient and timely manner. This allows for preliminary studies to be completed which can later be validated with experimental data. Additionally,
through the use of optimization-based modeling, vulnerable populations that may be unable to complete experimental tasks can still be investigated without putting subjects at risk.

In the following sections, the characteristics and utilization of trunk models in literature will be discussed. Due to the nature of the model utilized in this work, the focus will primarily be on lumped parameter stability-based trunk models. Finite element trunk models that include stability analysis will be discussed only briefly due to the differences in modeling methodologies and characteristics.

1.2.3.1 Overview of Finite Element Models
Finite element models (FEM) with stability analysis have been utilized to investigate spine behavior for different tasks$^{23,29-43}$. Kiefer et al. and El-Rich et al.$^{30}$ utilized a FEM to investigate upright standing posture. This model included five rigid bodies representing the lumbar vertebrae, one rigid body representing the thoracic spine, six beams representing the stiffness of the functional spinal units, and 60 and 56 muscle fascicles respectively$^{31}$. This model was also utilized by Arjmand and Shirazi-Adl$^{29}$ to determine stability margins and spinal loading in isometric forward flexion tasks with 56 muscle fascicles included in the model. Hajihosseinali et al.$^{40}$ developed a similar FEM incorporating stability requirements in the optimization procedure and implementing curved muscle paths, resulting in muscle activation predictions more realistically representing coactivation. The Minimum Potential Energy Method was utilized in the Hajihosseinali et al. FEM$^{40,44}$.

Gardner-Morse et al. utilized an FEM model in the investigation of the relationship between intrinsic muscle stiffness and spine stability$^{23}$. This model incorporated 66 muscle fascicles, a fixed pelvis, utilized beam elements to represent the stiffness of the functional spinal units, and a rigid body to represent the thorax$^{23}$. The stability of the system was analyzed using eigenvalue buckling analysis$^{23}$. Additionally, Gardner-Morse and Stokes utilized this model to investigate the possible relationship between
coactivation and spine stability requirements. For these investigations, stability analysis was completed and the critical magnitude of stiffness gain $q$ was determined.

1.2.3.2 Bergmark

The investigation of the mechanical stability of the lumbar spine by Bergmark has been an incredibly influential work, inspiring many researchers to use similar methods to model the trunk and analyze mechanical stability. In Chapter 6 of his work, Bergmark details his lumbar spine model, consisting of five lumbar vertebral rigid bodies and one rigid body representative of the thoracic spine region. These rigid bodies are stacked and connected by springs representative of the intervertebral discs, constrained to a fixed pelvis. 40 muscle fibers are included and represented as linear springs with Bergmark’s model of short-range stiffness (discussed in section 1.4.1), with the global erector spinae being represented with curved paths and the other muscles utilizing straight paths. Through the investigation of unloaded standing and loaded standing with differences in posture, the critical magnitude of $q$ could be determined to result for a stable system. The short-range stiffness model proposed by Bergmark (section 1.4.1) is commonly used for muscles in trunk models that are stability-based.

1.2.3.3 Cholewicki & McGill

Cholewicki and McGill developed a more realistic lumbar spine model that incorporated 90 muscle fascicles and an EMG-assisted optimization procedure. This model is unique in that it incorporates a rigid link body model for the external forces, a distribution-moment muscle model for muscle force approximations based on the measured EMG, and an 18 degree of freedom spine model used to estimate the moments generated by the muscles and passive tissues. Using the EMG-based optimization procedure, the predicted muscle activations were scaled as necessary to ensure that the moments were balanced, and the system was mechanically stable.
This model evaluated the mechanical stability of the spinal system during slow dynamic tasks, utilizing the Minimum Potential Energy Method\textsuperscript{15}. This model allowed for subject-specific data collection, with model parameters adjusting based on a subject’s anatomy and recruitment strategies which can vary between subjects\textsuperscript{15}.

\textbf{Figure 1-1: Cholewicki and McGill Model Utilization and Influence}

This Cholewicki and McGill lumbar spine model\textsuperscript{15} has been utilized to perform stability analysis in other experimental studies. For instance, utilizing the Cholewicki and McGill model\textsuperscript{15}, Cholewicki and Van Vilet\textsuperscript{45} and Kavic\textit{et al.}\textsuperscript{46} investigated the stabilizing potential of the muscle groups during different loading tasks to determine if a sole muscle group is primarily responsible for spinal stability (Figure 1-1). Both groups determined that no sole muscle was primarily responsible for spinal stability but rather other factors, such as the loading task, affected the muscle recruitment required to stabilize the system\textsuperscript{45,46}. In the Van Dieën\textit{et al.} study\textsuperscript{6}, the Cholewicki and McGill model\textsuperscript{15} aided in the comparison of muscle recruitment strategies and stability in healthy controls and individuals with LBP, yielding the
conclusion that recruitment strategies differ between the groups likely due to stability requirements. Additionally, in the Cholewicki, Simons and Radebold study\textsuperscript{47} focused on the influence of the positioning and magnitude of external loads on trunk stability, the Cholewicki and McGill model\textsuperscript{15} was utilized for determining the spine stability based on the subjects’ measured EMG.

Cholewicki \textit{et al.}\textsuperscript{10} developed a single inverted pendulum model with a pair of antagonistic muscles representative of the flexor and the extensor muscle groups, referred to as a simplified Cholewicki and McGill model\textsuperscript{15} (Figure 1-1). The short-range stiffness model from Bergmark\textsuperscript{18} was utilized and the Minimum Potential Energy method was utilized for stability analysis. This model\textsuperscript{10} was expanded and modified by Granata and Orishimo in their investigation of the influence spinal stability has on the neuromuscular response during lifting tasks\textsuperscript{20} through the expansion of the muscle model to include the contractile element and differing upright muscle heights. From the Granata and Orishimo study, it was evident that co-contraction increased with increased heights of the load in the lifting task indicating that the system is influenced by stability requirements\textsuperscript{20}.

1.2.3.4 \textit{Granata & Wilson}

Granata and Wilson developed a novel double inverted pendulum to investigate the relationship between posture and stability\textsuperscript{13}. This model incorporates 12 muscles modelled as linear springs with the short-range stiffness model developed by Bergmark\textsuperscript{18}. The Minimum Potential Energy Method is utilized for the stability analysis. A constrained optimization procedure was utilized to determine the muscle recruitment strategy, requiring that stability and equilibrium conditions be met. Additionally, to verify the predicted muscle recruitment from the model, experimental data collection was performed, affirming that the model adequately predicted coactivation with the stability and equilibrium requirements implemented in the optimization procedure.
Ghiasi et al. adjusted the optimization procedure from the Granata and Wilson model\textsuperscript{13} to incorporating requirements of maximizing the stability index and minimizing the sum of the cubed muscle stresses\textsuperscript{48}. Akhavanfar et al.\textsuperscript{49} developed the Granata and Wilson model\textsuperscript{13} in OpenSim, a musculoskeletal modeling software, and compared the predicted results from the model for a static optimization procedure with and without a stability constraint (Figure 1-2). Akhavanfar et al. determined that when the stability requirement was included in the optimization procedure, the predicted muscle activations were more similar to the experimental measurements obtained by Granata and Wilson\textsuperscript{13}. This verifies the results from Granata and Wilson that stability requirements should be included in the optimization procedure to yield more realistic muscle recruitment strategies\textsuperscript{13}.

Granata, Lee and Franklin\textsuperscript{14} utilized the Granata and Wilson model\textsuperscript{13} in their investigation of spine stability in trunk extension and flexion tasks (Figure 1-2). The Granata and Wilson model\textsuperscript{13} was utilized to determine the gain of the muscle forces required for the system to be stable, in equilibrium, and so the muscle activation matched the experimental measures. Then, excluding the stability requirement, the muscle activation required for the system to be in equilibrium could be determined.
The force required for co-contraction could be calculated as the difference between the muscle force for the system with solely the equilibrium condition and the system with equilibrium and stability conditions. Granata, Lee and Franklin determined that co-contraction was greater in flexion than in extension of the trunk\textsuperscript{14}.

The Granata and Wilson model\textsuperscript{13} was used in the Granata, Slota and Wilson\textsuperscript{50} study (Figure 1-2) to investigate the potential relationship between fatigue and stability of the spine. To implement fatigue into the model, the paraspinal muscle stiffness was reduced. In order for the system to stabilize with the reduced muscle stiffness, a greater magnitude of antagonistic co-contraction was required\textsuperscript{50}.

1.2.3.5 \textit{Franklin et al.}

Franklin \textit{et al.} developed a trunk model that incorporated time-delayed reflexes to evaluate the potential role that reflexes may have on spine stability\textsuperscript{16,17,24}. Previous researchers did not consider or incorporate time-delayed reflexes into the trunk model for simplicity and due to the commonly accepted assumption that intrinsic stiffness has a more crucial role in spine stability and that reflex stiffness may be unnecessary\textsuperscript{16,24}. Franklin \textit{et al.} determined that in certain loading conditions, reflex stiffness was required to stabilize the system\textsuperscript{16}. This conclusion motivated other researchers, such as Liebetrau \textit{et al.}\textsuperscript{11} and Zeinali-Davarani \textit{et al.}\textsuperscript{51}, to expand models to include reflexes and continue the investigation of the mechanisms of low back pain while including reflexes in the analyses. Additionally, experimental works such as those by Radebold \textit{et al.}\textsuperscript{8} and Cholewicki \textit{et al.}\textsuperscript{3} emphasize the importance of including reflexes in spine models as they identified the plausible correlation between greater reflex response times and an increased risk of LBP and injuries in individuals with chronic low back pain in comparison to healthy controls. Clearly, reflexes need to be considered in the investigation of LBP.

In addition to the reflexes, the Franklin \textit{et al.} model includes a fixed pelvis with independent rigid bodies representing each of the lumbar vertebrae and the thorax\textsuperscript{16,17,24}. Each rigid body has the ability to
rotate, resulting in an 18-DOF system. Franklin et al. developed a lumped intervertebral disc model applied between the rigid bodies\textsuperscript{16,17,24}. The skeletal and muscle coordinates are based on those used in the Cholewicki and McGill model\textsuperscript{15}. There are 90 Hill-type muscle fascicles with Bergmark’s model for short-range stiffness\textsuperscript{18} included.

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{figure1.png}
\caption{The expansion of the Wagner et al. model\textsuperscript{12}}
\end{figure}

Liebetrau et al.\textsuperscript{11} expanded the Wagner et al. model\textsuperscript{12} through the incorporation of reflexes based on the work of Franklin et al.\textsuperscript{16} (Figure 1-3). The Wagner et al. model incorporates an inverted pendulum with a pair of Hill-type antagonistic muscles\textsuperscript{12}. Stability is assessed using Linear Stability Analysis based on the eigenvalues of the Jacobian matrix\textsuperscript{12}. Through the investigation of lateral perturbations of healthy controls and individuals with low back pain, Liebetrau et al. determined that reflex delay and gain are inversely related for a stable spinal system.\textsuperscript{11}

1.2.4 **Summary**

Table 1-1 summarizes characteristics of the models discussed in this section for comparison. These models range in complexity and in their characteristics but each model has had a vast impact on modeling techniques and the utilization of spine models to investigate the etiology of low back pain.
Table 1-1: Overview of pivotal stability-based trunk models

For the Minimum Potential Energy Method, a system is stable if the Hessian matrix, based on the second derivative of the system’s potential energy, is positive definite.

For Linear Stability Analysis, after linearizing a system about equilibrium, a system is stable if the eigenvalues of the Jacobian need to have negative real parts.

<table>
<thead>
<tr>
<th>Model Type</th>
<th>Reflexes? (Y/N)</th>
<th>Muscle Paths</th>
<th>Muscle Model</th>
<th>Stability Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bergmark(^\text{18})</td>
<td>Stacked rigid bodies</td>
<td>No</td>
<td>Curved paths for global erector muscles; straight paths for others</td>
<td>Spring with Bergmark’s stiffness model</td>
</tr>
<tr>
<td>Cholewicki &amp; McGill(^\text{15})</td>
<td>Stacked rigid bodies; EMG-Assisted Optimization</td>
<td>No</td>
<td>Centroid line approach; some curved lines of action with nodes</td>
<td>DM Muscle Model</td>
</tr>
<tr>
<td>Cholewicki et al.(^\text{10})</td>
<td>Single Inverted Pendulum</td>
<td>No</td>
<td>Straight</td>
<td>Spring with Bergmark’s stiffness model</td>
</tr>
<tr>
<td>Franklin et al.(^\text{16,17,24})</td>
<td>Stacked rigid bodies; Optimization</td>
<td>Yes</td>
<td>Curved</td>
<td>Hill-Type with Bergmark’s stiffness model</td>
</tr>
<tr>
<td>Granata &amp; Wilson(^\text{13})</td>
<td>Inverted Double Pendulum; Optimization</td>
<td>No</td>
<td>Straight</td>
<td>Spring with Bergmark’s stiffness model</td>
</tr>
<tr>
<td>Liebetrau et al.(^\text{11})</td>
<td>Inverted pendulum</td>
<td>Yes</td>
<td>Straight</td>
<td>Hill-Type</td>
</tr>
<tr>
<td>Wagner et al.(^\text{12})</td>
<td>Single Inverted pendulum; optimization</td>
<td>No</td>
<td>Straight</td>
<td>Hill-Type</td>
</tr>
<tr>
<td>Zeinali-Davarani et al.(^\text{51–53})</td>
<td>Single Inverted pendulum; optimization</td>
<td>Yes</td>
<td>Straight</td>
<td>Force-Length and Force-Velocity</td>
</tr>
</tbody>
</table>
1.3 Model Development: Based on the Work of Franklin et al.

Throughout his graduate career at Virginia Polytechnic Institute and State University, Timothy C. Franklin developed a novel spine model that investigated the impact of reflexes on spinal stability\textsuperscript{16,17,24}. At this point in time, many researchers did not include reflexes in their spine models, primarily for simplicity and/or because it was assumed that the spine may stabilize mainly or completely from intrinsic stiffness without the utilization of reflexes\textsuperscript{24}. Franklin et al. investigated the impact of reflexes on spinal stability and determined that in certain loading conditions, reflexes were required for spine stability to be achieved\textsuperscript{16,17,24}.

Through personal communication with the Musculoskeletal Biomechanics Laboratory at Virginia Polytechnic Institute (Principal Investigators: Dr. Kevin P. Granata and Dr. Michael Madigan), the Human Motion Control Laboratory at the University of Kansas received some of the files developed by Franklin et al. for the model (Michael Madigan, personal communication, July 26, 2008). Through comparison of these files to the files published in the appendix of Franklin’s thesis, it was determined that the files did not match the final version of the model’s files. Therefore, our objective became redevelopment and improvement of the model.

1.3.1 Model Summary

In a brief summary, the Franklin et al. model includes separate rigid bodies representing each of the lumbar vertebrae, one rigid body representing the cervical and thoracic regions of the spine, and a fixed pelvis\textsuperscript{24}. The muscle geometry was based on the coordinates provided by Cholewicki et al., describing the attachment points of 90 muscle fascicles\textsuperscript{15}. Through constrained optimization, the muscle activation that occurs when the metabolic

\begin{figure}[h]
\centering
\begin{tikzpicture}
\node[rectangle, draw] (1) {Load Model Parameters};
\node[rectangle, draw, below of=1] (2) {Transform Model Geometry};
\node[rectangle, draw, below of=2] (3) {Optimization Procedure};
\node[rectangle, draw, below of=3] (4) {Perform Time Delay Analysis};
\node[rectangle, draw, below of=4] (5) {Nonlinear Verification};
\end{tikzpicture}
\caption{Model Process Summary}
\end{figure}
power is minimized and the system is stable and in equilibrium, can be determined for the linearized system. From the results of the optimization procedure, one can determine the time delay in which the system becomes unstable. Lastly, the nonlinear system can be simulated to verify the results from the linearized system.

Franklin et al. determined that reflexes were required to stabilize the spine under certain loading conditions\textsuperscript{16} and determined that an increase in reflex gain resulted in less required metabolic power required for stabilization\textsuperscript{16,17}. Additional conclusions and findings are discussed in their works\textsuperscript{16,17,24}.

1.4 Current Investigation

Building upon the studies completed by Franklin et al.\textsuperscript{16,17,24}, this work focuses on the investigation of the potential impact that dimensionless stiffness gain (discussed in 1.4.1) and lumbar lordosis angle (discussed in 1.4.2) may have on stability.

1.4.1 Bergmark’s Model of Short-range Stiffness

Muscle stiffness has been characterized as demonstrating elastic behavior for small deformations referred to as short-range stiffness\textsuperscript{18,54} (Equation 1-5):

\[
K = \frac{q}{l_o} F
\]

Where \(q\) is the dimensionless stiffness gain, \(F\) is muscle force, and \(l_o\) is the neutral length of the muscle.

Equation 1-5 is known as Bergmark’s model for short-range stiffness\textsuperscript{18}. For the dimensionless stiffness gain, Bergmark\textsuperscript{18} utilized a value of 40 in his analysis. The magnitude of this dimensionless stiffness gain was further investigated by Crisco et al.\textsuperscript{55} which resulted in the conclusion that this coefficient could fall within a range of 0.5 to 42. In past studies, the dimensionless stiffness gain \(q\)
utilized has varied between researchers (Table 1-2), prompting our interest as to how the magnitude of
the dimensionless stiffness gain may impact the force required to stabilize the spine.

Table 1-2: Magnitude of the stiffness gains in literature

Stiffness gain is a dimensionless parameter ranging between 0.5 and 42 in magnitude.\textsuperscript{55} This table
demonstrates how the stiffness gain is inconsistent between studies.

<table>
<thead>
<tr>
<th>Source</th>
<th>Stiffness Gain q</th>
</tr>
</thead>
<tbody>
<tr>
<td>Akhavanfar \textit{et al.}\textsuperscript{49}</td>
<td>5</td>
</tr>
<tr>
<td>Bergmark\textsuperscript{18}</td>
<td>40</td>
</tr>
<tr>
<td>Brown \textit{et al.}\textsuperscript{19}</td>
<td>10</td>
</tr>
<tr>
<td>Cholewicki \textit{et al.}\textsuperscript{10}</td>
<td>30</td>
</tr>
<tr>
<td>Franklin \textit{et al.}\textsuperscript{17}</td>
<td>10</td>
</tr>
<tr>
<td>Granata &amp; Wilson\textsuperscript{13}</td>
<td>5</td>
</tr>
<tr>
<td>Hajihosseinali \textit{et al.}\textsuperscript{40}</td>
<td>5</td>
</tr>
<tr>
<td>Samadi &amp; Arjmand\textsuperscript{56}</td>
<td>1.8, 5, 10</td>
</tr>
<tr>
<td>Shamsi \textit{et al.}\textsuperscript{57}</td>
<td>5</td>
</tr>
<tr>
<td>Stokes &amp; Gardner-Morse\textsuperscript{34}</td>
<td>5</td>
</tr>
<tr>
<td>Vakilzadeh \textit{et al.}\textsuperscript{58}</td>
<td>2, 4, 6, 10</td>
</tr>
<tr>
<td>Zeinali-Davarani \textit{et al.}\textsuperscript{51}</td>
<td>2, 4, 10</td>
</tr>
</tbody>
</table>

1.4.2 Hyperlordosis and Hypolordosis

Researchers have not come to a consensus regarding whether there is a significant correlation
between LBP and lordosis angle or not.\textsuperscript{59} Many of these studies incorporated radiographic
measurements or magnetic resonance imaging but did not investigate the impact that lordosis may have on spine stability through modeling.

Pesenti et al. measured total lumbar lordosis from the superior endplate of S1 to the superior endplate of L1, resulting in a mean standing lordotic angle of $58.1 \pm 13^\circ$. In order to define the lordosis angles for hyperlordosis and hypolordosis, the definition from Fernand et al. was used: hyperlordosis and hypolordosis as lordotic angles whose values fall outside of two standard deviations from the mean. Based on the mean standing lordotic angle presented by Pesenti et al. ($58.1 \pm 13^\circ$), hyperlordosis and hypolordosis would be defined as a lordotic angle greater than $84.1^\circ$ and less than $32.1^\circ$ respectively.

![Figure 1-5: Visual of normal lordosis, hyperlordosis and hypolordosis in the lumbar spine region](image)

1.5 Specific Aims and Hypotheses

**Specific Aim 1:** Redevelop the Franklin et al. spine model and perform code verification.

**Specific Aim 2:** Determine the influence the magnitude of stiffness gain $q$ has on stability of the spine.

*Hypothesis:* Increasing the magnitude of stiffness gain $q$ will allow for the spine to stabilize with less required force.
**Specific Aim 3:** Investigate the potential impact that hyperlordosis or hypolordosis may have on spine stability.

*Hypothesis:* With a load of 0 N applied vertically to T4 at a position 20 cm anterior to the trunk, the hyperlordotic spine will require increased recruitment of flexor muscles to stabilize, consequently increasing the required metabolic power.

*Hypothesis:* With a load of 0 N applied vertically to T4 at a position 20 cm anterior to the trunk, the hypolordotic spine will require increased recruitment of extensor muscles to stabilize, consequently increasing the required metabolic power.

*Hypothesis:* With a load of 200 N applied vertically to T4 at a position 20 cm anterior to the trunk, a greater critical stiffness will be required to stabilize the hypolordotic spine.
Chapter 2: Model Development

Our goal was to reconstruct the Franklin et al. spine model\textsuperscript{16,17,24} and continue further investigation of the impact of model parameters on spinal stability. This model incorporates six rigid bodies, a fixed pelvis, a lumped parameter model representative of an intervertebral disc, 90 Hill-Type muscle fascicles and neuromuscular reflexes\textsuperscript{16,17,24}.

The model was reconstructed in MATLAB R2021b (Mathworks, Natick, MA). The MATLAB Scripts and Live Scripts utilized for the model can be viewed in Appendix C-D. The verification processes for these MATLAB scripts and functions are discussed in Chapter 3.

2.1 Rigid Bodies

The current spine model includes one rigid body representing the cervical and thoracic regions of the spine and five separate lumbar vertebrae rigid bodies, and a fixed pelvis\textsuperscript{16,17,24} (Figure 2-1). Each of the vertebrae can rotate about all three axes, resulting in 18 degree of freedom system.

![Cervical & Thoracic Spine](image1)

![Lumbar Spine](image2)

Figure 2-1: Model rigid bodies

The lumbar vertebral geometry has been measured by a few researchers\textsuperscript{63–72} with discrepancies in the reported measurements most likely due to sample size, sample population characteristics, and methods used\textsuperscript{73}. When determining the rigid body parameters that would be included in our model, sources were reviewed and compared.
Table 2-1: Spine segment properties.

The vertebral spacings for the lumbar vertebrae are based on Cholewicki et al. The coordinates provided by Cholewicki et al. describe the spine in a neutral position, meaning that the lordosis of the spine is influencing these coordinates. To determine the skeletal coordinates without the influence of lordosis, these coordinates were rotated so that the connection points for the vertebrae were aligned along the vertical axis, resulting in the calculated vertebral spacing provided in this table.

For the cervical and thoracic region, the height of the cervical region is calculated based on the lumbar vertebral heights and lumbar intervertebral discs using a method by Gilad et al. The utilized lumbar vertebral heights are from Berry et al. and the calculated intervertebral disc height are based on the vertebral spacing. The height of the thoracic region is based on the vertebral heights from Panjabi et al. and the thoracic intervertebral disc heights from Kunkel et al.

The masses of the vertebrae are based on Liu et al. and the lordosis angles for each body have been calculated based on the work of Pesenti et al. See Appendix A.2 for more information on lordosis angle calculations for each of the bodies.

<table>
<thead>
<tr>
<th>Spine Segment</th>
<th>Body Number</th>
<th>Vertebral Spacing (m)</th>
<th>Mass (kg)</th>
<th>Normal Lordosis Angle (deg)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cervical &amp; Thoracic Region</td>
<td>0</td>
<td>0.4241</td>
<td>16</td>
<td>0</td>
</tr>
<tr>
<td>L1</td>
<td>1</td>
<td>0.0363</td>
<td>2</td>
<td>20.1</td>
</tr>
<tr>
<td>L2</td>
<td>2</td>
<td>0.0386</td>
<td>2</td>
<td>17.5</td>
</tr>
<tr>
<td>L3</td>
<td>3</td>
<td>0.0379</td>
<td>2</td>
<td>9.6</td>
</tr>
<tr>
<td>L4</td>
<td>4</td>
<td>0.0370</td>
<td>2</td>
<td>-2</td>
</tr>
<tr>
<td>L5</td>
<td>5</td>
<td>0.0389</td>
<td>2</td>
<td>-17.8</td>
</tr>
<tr>
<td>Pelvis</td>
<td>p/6</td>
<td>N/A</td>
<td>N/A</td>
<td>-38</td>
</tr>
</tbody>
</table>
2.1.1 Dimensions

The rigid bodies are represented as cylinders with an elliptical cross-sectional area for simplicity\(^{24}\). In the current model, the radii of the trunk in the medio-lateral (ML) and the anterior-posterior (AP) direction are 0.024 meters and 0.017 meters respectively, based on body diameter measurements from Berry \textit{et al.}\(^{64}\) and Panjabi \textit{et al.}\(^{65}\) (Figure 2-2).

These measurements were obtained from the inferior and superior endplates of the lumbar vertebral bodies\(^{64,65}\). The average AP radius for the lumbar vertebrae based on the Panjabi \textit{et al.}\(^{65}\) and Berry \textit{et al.}\(^{64}\) measures was approximately 0.017 m for both studies. For the ML radius, there was more deviance in the results: for Panjabi \textit{et al.}\(^{65}\), the resulting average ML radius was approximately 0.023 m, while for Berry \textit{et al.}\(^{64}\), it was approximately 0.025 m. An average value of 0.024 m was utilized in our model for the ML radius.

![Figure 2-2: Vertebral body diameters. Not drawn to scale](image)

While Franklin \textit{et al.}\(^{24}\) utilized the vertebral heights from Liu \textit{et al.}, a cadaver study that included only one subject\(^{63}\) the current model utilizes vertebral body heights based on the skeletal coordinates provided by Cholewicki \textit{et al.}\(^{54}\). These skeletal coordinates describe a spine in neutral position, indicating
that these coordinates are influenced by lumbar lordosis. In order to be able to adjust the lordosis angle magnitudes as necessary for our study, these coordinates were rotated so that the connection points for all vertebrae were aligned along Axis 3. Each vertebra has a connection point placed at the inferior and superior vertebral surfaces, which describes where the vertebra connects to the vertebra inferior and superior to it. The distance between these connection points will be referred to as the vertebral spacing for that vertebra in this work (Table 2-1).

The height of the cervical and thoracic regions of the spine were computed separately and then summed for a total height.

2.1.1.1 Cervical Spine Region Height

Gilad et al. measured the posterior and anterior height of the lumbar and cervical vertebrae and determined the height relationships between these regions of the spine. Using the vertebral spacing determined previously, the height of the lumbar vertebra and the intervertebral disc that encompasses that spacing needed to be calculated.

For the intervertebral disc heights of the lumbar region, measurements from Tibrewal et al. were utilized. Tibrewal et al. measured the lumbar intervertebral disc heights for 11 healthy controls and provided the range of disc heights. Using the maximum value of the range for the anterior and posterior disc height, the average maximum disc height could be determined. Using these maximum disc heights and the vertebral spacing from the Cholewicki et al. coordinates, the required vertebral height could be determined (Figure 2-3).
The mean lumbar vertebral heights from Gilad et al. were utilized to determine if the required vertebral height (Figure 2-3) was reasonable. Based on our calculations, the vertebral heights of L4 and L5 were sufficient without adjustment. L1, L2, and L3 required an increase in magnitude, correlating to +1.721*STD, +1.22*STD, and +0.239*STD. The average lumbar vertebral heights were adjusted to appropriately fit the vertebral spacing required by the skeletal coordinates.

Using the determined lumbar vertebral heights and intervertebral disc heights, the respective measures could be calculated for the cervical region.

2.1.1.2 Thoracic Spine Region Height

For the thoracic region, the height was determined from the spacing determined in Cholewicki et al. and compared to the values reported in Berry et al. and Kunkel et al.

Many of these sources did not differentiate between sexes in their results, so it is important to address the potential impact that sex may have on the average vertebral geometry measures across the sample. While it has been determined that lumbar vertebral heights are not sex-specific, as the
difference of the heights between sexes has been proven to be insignificant, the ML diameter does
differ significantly between sexes\textsuperscript{71,75}.

2.1.1.3 Modifications to the 2006 Franklin Model

Franklin \textit{et al}.\textsuperscript{24} utilized a ML radius of 0.12 m and an AP radius of 0.06 m for the rigid bodies\textsuperscript{24}. For the current model, the radii have been updated based on the vertebral diameters provided by Berry \textit{et al}.\textsuperscript{64} and Panjabi \textit{et al}.\textsuperscript{65} Additionally, Franklin \textit{et al}. utilized body heights from Liu \textit{et al}.\textsuperscript{63} in his analysis whereas the current model utilized the vertebral spacing defined by Cholewicki \textit{et al}.\textsuperscript{54}

2.1.2 Body Origins

The origin of each vertebra is defined at the inferior vertebral endplate, as shown in Figure 2-4.

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{figure2_4.png}
\caption{Front view of vertebra. Origin defined at inferior vertebral surface}
\end{figure}

The following equation can be used to determine the origin of a vertebral body:

\begin{equation}
\text{origin of vertebral body} = \text{origin of inferior vertebral body} + \text{inferior vertebral spacing}
\end{equation}

The vertebral spacing (Table 2-1, Equation 2-1) is based on the muscle and skeletal coordinates provided by Cholewicki \textit{et al}.\textsuperscript{15} for a body in neutral spine position and is the height between the connection points between vertebrae. Comparing these coordinates to those provided by Franklin \textit{et al}.\textsuperscript{24}, one can deduce that Franklin transformed the coordinates. The skeletal geometry of each vertebra was rotated so that the upper and lower connection points were in alignment along Axis 3 (Figure 2-4).
Additionally, Franklin defined the origin of the vertebra as the inferior connection point of the vertebra, and all body coordinates were translated to the local coordinate system of the vertebra from the global coordinate system\textsuperscript{24}. The origin of the pelvis was defined at (0,0,0) meters.

2.1.3 Mass and Mass Moment of Inertia

Lui \textit{et al.} determined the mass of each vertebra of a cadaver spine\textsuperscript{63}. The lumbar vertebrae resulted in an average mass of approximately 2.0 kg. The cervical and thoracic region resulted in a total mass of approximately 15.3 kg. A mass of 16 kg was used for the cervical/thoracic region of the spine and a mass of 2 kg was used for each lumbar vertebra (Table 2-1).

In our model, each vertebra’s center of mass was located at the center of the vertebral spacing for each vertebra (Figure 2-5).

![Figure 2-5: Front view of vertebra. Center of mass depicted by red circle](image)

The mass moment of inertia for an elliptical cylinder can be determined using the equations in Appendix A.1. The vertebral spacing (Table 2-1) was used in the mass moment of inertia calculation.

2.1.3.1 Modifications to the 2006 Franklin Model

The cervical and thoracic spine region mass has been updated to 16 kg from a mass of 20 kg previously defined in the model by Franklin \textit{et al.}\textsuperscript{24}
2.2 Rotations

The basis vectors depicted in Figure 2-6 are the basis vectors affixed to the ground (N basis). Each vertebra is rotated first around Axis 1, followed by Axis 2, and lastly around Axis 3. This was accomplished using rotation matrices. Matrix multiplication can be used to develop a rotation matrix encompassing the rotation around all three axes:

\[
NRC = NRA * Rab * Rbc
\]

This rotation matrix \((NRC)\) defines the angular relationship between bases \(N\) and \(C\). See Appendix A.4 for more information regarding the development of the rotation matrix.

2.2.1 Lordosis Angles

Lumbar lordosis is the curve of the lumbar region of the spine. The lordosis angle of each vertebra is applied as a rotation around Axis 1 where a counterclockwise rotation is defined as the positive direction. See Appendix A.2 for more information about the Lordosis Angles applied.
2.2.1.1 Modifications to the 2006 Franklin Model

The lordosis angles have been updated based on the average segmental lordosis angles determined by Pesenti et al. In this study, the radiographic data was collected for 119 adult healthy controls with no history of back pain or injury. The updated lordosis angles for each vertebra are recorded in Table 2-1.

2.3 Intervertebral Discs

Franklin et al. developed a lumped parameter model for the modelling of the intervertebral discs with parameters based on the work of Stokes et al. and Izambert et al.

Stokes et al. measured the rotational stiffness of cadaveric discs and Izambert et al. determined that the damping coefficient was approximately 2 orders of magnitude less than the stiffness coefficient. This resulted in a rotational stiffness \( K_{IVD} \) of 50 Nm/rad and a rotation damping \( B_{IVD} \) of 0.5 Nms/rad, which was applied to all three axes of the intervertebral disc model.

By determining the angular position and angular velocity, the magnitude of the moment generated by the intervertebral discs can be determined (Equation 2-3) and applied as equal and opposite moments to the vertebrae.

\[
M = K_{IVD} \theta + B_{IVD} \dot{\theta}
\]  

(2-3)

For this calculation, the angles of rotation need to be determined, which can be achieved through use of a dummy plane; the body below the vertebra being measured will act as the dummy plane. Franklin et al. defined planar rotation to be the rotation of the body that causes the plane and body to not be parallel (Figure 2-7) and twist rotation to be the rotation of the body that causes the plane and body to not be aligned (Figure 2-8).
To determine these angles, the dummy plane must first be rotated to have the same lordosis angle as the vertebra above it. The difference in lordosis angles should not contribute to our angle measures. Once the plane and vertebra have the same lordosis angles, we can determine if the planes are parallel or if there is a planar rotation present. If the planes are parallel, Axis 3 of the vertebrae and plane will be aligned. If Axis 3 is not aligned, then the planar angle can be calculated by determining the angle between Axis 3 of the vertebra and the plane. Using the Rodrigues Rotation Formula, the vertebra is rotated so that it is parallel to the dummy plane.

![Figure 2-7: Planar rotation](image)

Next, we can determine the twist rotation angle. If Axis 1 is aligned for the vertebra and dummy plane, then a twist rotation is not present. If Axis 1 is not aligned, then the rotation angle can be calculated by determining the angle between Axis 1 of the vertebra and the plane. Using the Rodrigues Rotation Formula, the vertebra is rotated. The vertebra should now be aligned with the dummy plane.

![Figure 2-8: Twist rotation](image)
2.4 Muscle Anatomy

This model incorporates 90 muscle fascicles with the attachment points detailed in Appendix A of Cholewicki et al.\textsuperscript{15} Included are the origin, insertion and nodal coordinate points with respect to the skeletal geometry. The muscle coordinates have been transformed into body-fixed coordinates. The adjusted muscle coordinates and the process used to adjust the coordinates is further detailed in Appendix A-3.

![Figure 2-9: Muscle Anatomy](image)

While many models only include muscles with straight lines of action, meaning muscles connecting from the origin directly to the insertion, the nodal points in our muscle anatomy allow for the muscles to have a more curved path as they would in the body\textsuperscript{15} (Figure 2-9).

2.4.1 Calculating Transformed Muscle Anatomy

The transformed muscle anatomy can be determined through use of the rotation matrices and origins of the bodies. Each muscle attachment point will be rotated based on the body it is associated with and that body’s origin:
\[ pt_{\text{transformed}} = N^R_C \ast pt + \text{origin} \]  \hspace{1cm} (2-4)

where \( N^R_C \) is the rotation matrix from the \( N \) to \( C \) bases, and the coordinates of the muscle attachment point \( (pt) \) and origin of the body \( (\text{origin}) \) need to be column vectors.

### 2.4.2 Calculating Muscle Length

Muscle length should be calculated after the bodies have been rotated, so that lumbar lordosis has been implemented. Vectors can be determined between a muscle’s attachment points (Figure 2-10, Equation 2-5):

\[
\vec{p} = ma_{pt1} - ma_{pt2} = \vec{p_1} \hat{c}_1 + \vec{p_2} \hat{c}_2 + \vec{p_3} \hat{c}_3 \]  \hspace{1cm} (2-5)

where \( ma_{pt1} \) is the first attachment point in the set and \( ma_{pt2} \) is the second attachment point.

Using resulting vector \( \vec{p} \), the length of the vector can be calculated:

\[
|\vec{p}| = \sqrt{p_1^2 + p_2^2 + p_3^2} = \text{length of vector} \]  \hspace{1cm} (2-6)

Lastly, the total muscle length can be determined through the summation of the vector lengths between the muscle attachment points.
2.4.3 Calculating Muscle Velocity

In order to calculate the muscle velocity, we must calculate the time-derivative of the rotation matrix:

$$\frac{d^N R^C}{dt} = \frac{\partial^N R^C}{\partial q_1} \frac{dq_1}{dt} + \frac{\partial^N R^C}{\partial q_2} \frac{dq_2}{dt} + \frac{\partial^N R^C}{\partial q_3} \frac{dq_3}{dt}$$  \hspace{1cm} (2-7)

Where $q_1$, $q_2$, and $q_3$ are the angles of rotation around Axis 1, Axis 2 and Axis 3, respectively, and $^N R^C$ is the rotation matrix for each body that incorporates all three body rotations.

For example, for a muscle fascicle with only two attachment points, the following can be used to determine the linear velocity of a muscle attachment point:

$$\text{coord} = \left( \frac{d^N R^C}{dt} \ast \text{coord} + \text{dOr} \right)$$  \hspace{1cm} (2-8)

Where $\frac{d^N R^C}{dt}$ is the time-derivative of the rotation matrix, coord are the muscle attachment point coordinates and dOr is the time-derivative of the origin of the body.

A muscle point velocity vector can then be determined between the two muscle attachment points using Equation 2-9:

$$\vec{v} = \left( \frac{d^N R^C}{dt} \ast \text{coord}_{\text{origin}} + \text{dOr} \right) - \left( \frac{d^N R^C}{dt} \ast \text{coord}_{\text{terminal}} + \text{dOr} \right)$$  \hspace{1cm} (2-9)

Where coord$_{\text{origin}}$ is the origin attachment point and coord$_{\text{terminal}}$ is the terminal attachment point for the muscle fascicle.
The position vector is defined in Equation 2-10 as the vector between the attachment points:

\[ \vec{p} = \text{N}_R^C \ast \text{coord}_{\text{origin}} - \text{N}_R^C \ast \text{coord}_{\text{terminal}} = (p_1, p_2, p_3) \]  

(2-10)

Finding the unit vector of the position vector:

\[ \hat{p} = \frac{\vec{p}}{|\vec{p}|} = \frac{\vec{p}}{\sqrt{p_1^2 + p_2^2 + p_3^2}} \]  

(2-11)

Lastly, by performing the dot product, we can determine the projection of \( \vec{v} \) in the direction of \( \hat{p} \):

\[ \text{muscle velocity} = \vec{v} \cdot \hat{p} \]  

(2-12)

The resulting scalar is the magnitude of the muscle velocity. If the muscle path includes nodes, the velocity can be determined between muscle attachment points along the path of the muscle and summed to yield the total muscle velocity.

### 2.5 Muscle Model

Franklin et al.\textsuperscript{24} incorporated a Hill-Type muscle model\textsuperscript{80} that includes a contractile element, spring and damper, all in parallel (Figure 2-11):

\[ F_m = F_{CE} + K_{PE}x + B_{DE} \dot{x} \]  

(2-13)

where \( K_{PE} \) is stiffness, \( B_{DE} \) is the damping coefficient, \( x \) is muscle length, and \( \dot{x} \) is stretch rate.

![Figure 2-11: Hill-type muscle model](image)
In the Hill muscle model, the contractile element is representative of the actin and myosin crossbridges, the parallel elastic is representative of the connective tissue properties, and the series elastic, which has not been included in this work, is representative of the tendon properties\textsuperscript{61,82}.

The force of the contractile element was determined by scaling the maximum muscle force by muscle activation\textsuperscript{24} (Equation 2-15). Muscle activation ranges between zero and one for all muscles, with a magnitude of one indicating a fully activated muscle and a magnitude of zero indicating an un-activated muscle. Maximum muscle force was determined by the product of maximum muscle stress (46 N/cm\textsuperscript{2}) by the muscle cross-sectional area\textsuperscript{23} (Equation 2-14).

\[
f_{\text{max}} = (46 \text{N/cm}^2) \times \text{CSA}
\] (2-14)

\[
F_{\text{CE}} = f_{\text{max}} \times \alpha
\] (2-15)

The parallel elastic has a stiffness ($K_{PE}$) based on Bergmark’s model of short-range stiffness where $q$ is the stiffness gain\textsuperscript{18}:

\[
K_{PE} = q \times \frac{F_{\text{CE}}}{x_0}
\] (2-16)

Similarly, the muscle damping was scaled by the damping gain $b$:

\[
B_{DE} = b \times \frac{F_{\text{CE}}}{x_0}
\] (2-17)

By substituting in the stiffness and damping coefficients, the equation for the muscle model is:
\[ F_m = f_{\text{max}} \alpha + \left( q \frac{f_{\text{max}}}{x_o} \alpha (t) - x_o \right) + \left( b \frac{f_{\text{max}}}{x_o} \dot{x}(t) \right) \]  

(2-18)

where \( f_{\text{max}} \) is the maximum muscle force, \( q \) is stiffness gain, \( b \) is damping gain, \( x \) is muscle length, \( x_o \) is equilibrium muscle length, \( \dot{x} \) is stretch rate, and \( \alpha \) is muscle activation.

### 2.6 Reflex Model

In simplistic terms, the muscle stretch reflex is the concept in which a stimulus causes the muscle to lengthen and the muscle spindles respond to this lengthening by opposing this stretch through contraction\(^{83,84}\). The muscle spindles have the ability to determine the velocity and length of the muscle, and through communication with the nervous system, oppose motion of the muscle\(^{83,84}\).

This muscle stretch reflex mechanism can be represented through the application of a PD controller. The implementation of a PD controller in our system allows for the system to be responsive to the change in muscle length and velocity, like the muscle stretch reflex, creating a feedback control system\(^{24,85}\). Franklin et al.\(^{24}\) defined muscle activation to be the sum of the steady-state activation \( \alpha_o \) and the reflex activation \( \alpha_r \) (Equation 2-19).

\[ \alpha(t) = \alpha_o + \alpha_r = \alpha_o + \alpha_o \left( G_p \frac{x(t - \tau) - x_o}{x_o} + G_D \frac{\dot{x}(t - \tau)}{x_o} \right) \]  

(2-19)

where \( G_p \) is the proportional gain and \( G_D \) is the differential gain of the PD controller.
2.7 Dynamics

The derivative of the Lagrangian ($L$) was used to determine the equations of motion of the system\textsuperscript{24}. The Lagrangian function is based on the difference between the system’s kinetic ($KE$) and potential energy ($PE$)\textsuperscript{86}:

$$L = KE - PE$$

(2-20)

Franklin\textit{ et al.} only included gravitational energy in the potential energy of the system\textsuperscript{24}, so the general equation for the potential energy of the system is as follows:

$$PE = \sum_{b=1}^{6} mgh$$

(2-21)

where $h$ is the position of the center of mass of the body along Axis 3, $m$ is the mass of the vertebra, and $b$ is the body index.

The kinetic energy of the system includes rotational and translational kinetic energy:

$$KE = \sum_{b=1}^{6} \left( \frac{1}{2}mv^2 + \frac{1}{2}I\omega^2 \right)$$

(2-22)

where $I$ is the mass moment of inertia, and $v$ is the velocity and $\omega$ is the angular velocity of the center of mass of the body.

The muscle forces, external forces and the IVD moments are utilized in the generalized force calculations (Equation 2-23).
\[Q_i = \sum_{b=1}^{6} \left( \sum_{e=1}^{n} F_{be} \right) \frac{\partial v_b}{\partial q_i} + \left( \sum_{e=1}^{n} r_{be} \times F_{be} \right) \cdot \frac{\partial \omega_b}{\partial \dot{q}_i} \]  

(2-23)

where \(i\) is the degree of freedom index for the degree of freedom \(q_i\), \(b\) is the rigid body index, \(e\) is the applied forces index, \(F_{be}\) represents force applied to the body, \(v_b\) is the linear velocity of the body's origin, \(r_{be}\) is a vector between the point where the force is applied and the body's origin, and \(\omega_b\) is the body's angular velocity.

The Lagrangian Derivative yields the equations of motion (Equation 2-24):

\[EOM = \frac{d}{dt} \left( \frac{\partial K_E}{\partial \dot{q}_i} \right) - \frac{\partial L}{\partial q_i} = Q_i \]  

(2-24)

where \(Q_i\) are the generalized forces and \(q_i\) are the generalized coordinates related to the degrees of freedom.

By substituting in Equation 2-20 into Equation 2-24, the following equation is derived:

\[EOM = \frac{d}{dt} \left( \frac{\partial K_E}{\partial \dot{q}_i} \right) - \frac{d}{dt} \left( \frac{\partial P_E}{\partial \dot{q}_i} \right) - \frac{\partial K_E}{\partial q_i} + \frac{\partial P_E}{\partial q_i} = Q_i \]  

(2-25)

Lastly, potential energy of the mechanical system is not a function of velocity, therefore the second term in Equation 2-25 can be eliminated, resulting in our final equation for the generalized forces calculation:

\[EOM = \frac{d}{dt} \left( \frac{\partial K_E}{\partial \dot{q}_i} \right) - \frac{\partial K_E}{\partial q_i} + \frac{\partial P_E}{\partial q_i} = Q_i \]  

(2-26)

The dynamic equations can be used to determine the accelerations through forward dynamic simulation techniques (Equation 2-27):
\[ M_m(q) \cdot \ddot{q} + G_m(q) + C_m(q, \dot{q}) = Q_i \]  

(2-27)

Where \( M_m \) is the mass matrix, \( C_m \) is the Coriolis vector, and \( G_m \) is the gravity vector.

### 2.8 Linearization

The linearized representation of the system was determined through Taylor Series expansion. For a nonlinear system to be linearized, it must be in equilibrium.

Equation 2-28 is the state vector and Equation 2-29 is the differential state vector:

\[ x = [\theta_1, \theta_2, ..., \theta_n, \dot{\theta}_1, \dot{\theta}_2, ..., \dot{\theta}_n] \]  

(2-28)

\[ \dot{x} = f(x) \]  

(2-29)

Where \( \dot{x} = 0 \) when at the equilibrium point \( x = \bar{x} \).

Each state variable was disturbed by angle \( \delta_\theta \) individually (Equation 2-30):

\[ \delta_\theta = f(\theta_1 + \delta_\theta, \theta_2, ..., \theta_n, \dot{\theta}_1, \dot{\theta}_2, ..., \dot{\theta}_n) \]  

(2-30)

The Taylor Series Expansion is performed, neglecting higher order terms:

\[ \delta_\theta = f(\bar{x}) + \frac{\partial f(x)}{\partial x} \bigg|_{x=\bar{x}} \delta_\theta \]  

(2-31)

At equilibrium, \( f(\bar{x}) = 0 \).

\[ \delta_\theta = \frac{\partial f(x)}{\partial x} \bigg|_{x=\bar{x}} \delta_\theta \]  

(2-32)
The $\frac{\partial f(x)}{\partial x} \bigg|_{x=\bar{x}}$ term in Equation 2-32 is the Jacobian matrix. The Jacobian matrix can be obtained through finite differencing (Equation 2-33), which results in an approximate result of a derivative.

$$\frac{\partial \theta_1}{\partial \theta_1} = \frac{f \left( \bar{\theta}_1 + \delta \theta, \bar{\theta}_2, ..., \bar{\theta}_n, \bar{\dot{\theta}}_1, \bar{\dot{\theta}}_2, ..., \bar{\dot{\theta}}_n \right) - f(\bar{x})}{\delta \theta} \quad (2-33)$$

Neuromuscular reflexes are time-delayed, requiring for there to be a Jacobian for the instantaneous components and a separate Jacobian for the delayed components (Equation 2-34):

$$\dot{x}(t) = J_i \cdot x(t) + J_d \cdot x(t - \tau) \quad (2-34)$$

While the separation of these Jacobians is not necessary for the optimization procedure (Section 2.9), it will be necessary for the Linear Time Delay analysis (Section 2.10).

**2.9 Optimization Procedure**

A constrained optimization is performed to determine the muscle activation that meets the following requirements:

1. The system must be stable and in equilibrium.
2. Muscle activation magnitude must be between 0 and 1 for all muscles.
3. Metabolic power must be minimized.

**2.9.1.1 Stability Requirement**

The stability of the system is evaluated using the Jacobian matrix. A system is considered stable if the eigenvalues of its Jacobian have negative real parts\(^{24,79}\).
2.9.1.2 *Equilibrium Requirement*

For a system in equilibrium, acceleration will have a magnitude of zero. For this criterion to be met, the acceleration due to the passive state equilibrium properties ($b_{eq}$) and the product of the active state equilibrium properties ($A_{eq}$) and the muscle activation ($\alpha$) will need to be equivalent:

$$A_{eq} \alpha = b_{eq}$$ (2-35)

This equilibrium condition will be utilized in the optimization procedure to ensure that the system is in equilibrium.

2.9.1.3 *Muscle Activation Range*

The magnitude of the muscle activation for each muscle must fall between 0 and 1.

2.9.1.4 *Minimizing Metabolic Power*

Anderson\textsuperscript{88} defines the maintenance heat rate by the following equation, assuming a proportionate quantity of fast and slow twitch muscle fibers\textsuperscript{24} (Equation 2-36):

$$P = m \left( 74 \times 0.5 \times \sin\left(\frac{\alpha \pi}{2}\right) + 111 \times 0.5 \times \left( 1 - \cos\left(\frac{\alpha \pi}{2}\right) \right) \right)$$ (2-36)

The metabolic power can be minimized through use of the built-in function *fmincon* in MATLAB. This function performs a constrained optimization, which allows for the previously mentioned requirements to be met while solving for the muscle activations. The metabolic power is specified as the parameter that needs to be minimized, while the equilibrium condition and stability condition are specified as constraints. Additionally, the range for the muscle activation, which is between 0 and 1, is specified as a constraint for the Solver. A randomized muscle activation is input into the function, providing a starting point for the Solver. A minimum of five trials were completed, meaning that at least five different starting muscle activation vectors were utilized. Performing multiple trials with different initial activations increased the likelihood of the global minima being found\textsuperscript{24}. 
2.10 Linear Time Delay Methods

The linear time delay methods are based on the work of Chen et al. In this work, a procedure is outlined that allows for the time delay that would cause instability of the system to be determined.

For the optimization procedure, it is assumed that time delay has a magnitude of zero \( \tau = 0 \), so the Jacobian contains the instantaneous and delayed components. After the optimization procedure has been completed, the Jacobian can be separated into the instantaneous and delayed components (Equation 2-34).

Based on Chen’s method involving frequency sweeping in the Laplace domain (Equation 2-37), the time delay in which the system becomes unstable can be determined, which would be when the real part of an eigenvalue of the Jacobian crosses the Imaginary Axis. This is indicated by \( \lambda = 1 \) (Equation 2-38):

\[
j \omega \cdot X = J_i \cdot X + J_D \cdot X \cdot e^{-j \omega \tau}
\] (2-37)

\[
(j \omega \cdot I - J_i) \cdot X = \lambda \cdot J_D \cdot X, \quad \text{where} \quad \lambda = e^{-j \omega \tau} = e^{-j \theta}
\] (2-38)

Using the corresponding phase \( \theta \) and the frequency \( \omega \), the delay margin \( \tau_m \), which is the time delay magnitude as to when the real part of the eigenvalue crosses the imaginary axis, can be computed:

\[
\tau_m = \frac{\theta}{\omega}
\] (2-39)

Using the delay margin, the time delay that will be utilized in the nonlinear simulation (section 2.11) can be computed:
\[ \tau = \tau_m \times 0.99 \]  \hspace{1cm} (2-40)

This time delay \( \tau \) represents a time delay preceding the delay margin/the system becoming unstable. A time delay with a magnitude less than the delay margin should result in a stable system if the linear analysis adequately predicted the nonlinear results. Lastly, if the delay margin is less than 0.06 seconds, then the system is considered unstable on the basis that the neuromuscular reflexes cannot act faster than 0.06 seconds, indicating that they would not be able to contribute to stability at that time\(^7\).

### 2.11 Nonlinear Verification

A nonlinear verification technique is used to ensure that the linear approximation appropriately represented the nonlinear system and successfully predicted its stability conditions. Using the \textit{dde23} MATLAB Solver, the delayed differential equation with a constant delay could be solved. Based on the parameters defined by Franklin et al., each simulation is run for five seconds\(^17\). Following a kinematic disturbance, the system should approach equilibrium; this behavior can be investigated using the normalized state space about equilibrium\(^17\).

To determine if the system is approaching equilibrium during the nonlinear simulation, the difference between the normalized state space and equilibrium can be determined:

\[
D = \sqrt{\sum_{i=1}^{18} \left( \frac{\dot{\theta}(t) - \bar{\dot{\theta}}}{\text{mean}(\dot{\theta}(t) - \bar{\dot{\theta}})} \right)^2 + \left( \frac{\theta(t) - \bar{\theta}}{\text{mean}(\theta(t) - \bar{\theta})} \right)^2} \]  \hspace{1cm} (2-41)
2.12 Summary

The current methodology and model are based on those developed by Franklin et al.\textsuperscript{16,17,24}. This rigid body model with a fixed pelvis allows for the behavior of the lumbar spine to be explored. Using linear stability analysis in the optimization procedure, the muscle activations and required metabolic power can be determined while the equilibrium, stability and muscle activation constraints are met. Following this optimization procedure, the time delay analysis can be performed, resulting in the magnitude of the time delay that will be utilized in the nonlinear verification procedure. This model allows for the investigation of low back stability and the factors that may affect it.
Chapter 3: Code Verification

Throughout the development of the Franklin et al. model,\textsuperscript{16,17,24} it was crucial that the methodology and implementation of this methodology be reviewed. As mentioned previously, based on the published code in the Appendix of Franklin’s thesis\textsuperscript{24} and the code received from the Musculoskeletal Biomechanics Laboratory at Virginia Polytechnic Institute (Michael Madigan, personal communication, July 26, 2008), it was evident that the Human Motion Control Lab did not possess the final version of the model. Through use of the materials and the published information, the model was developed. While adjustments have been made to the model, the methodology is based on Franklin’s works\textsuperscript{16,17,24} so similarities should be expected.

Verification tasks were completed to identify and eliminate potential errors in the codes\textsuperscript{67}. The verification tasks for the main model codes can be seen in Table 3-1. The file \textit{For_verification.m} is used to perform some of the verification tasks.

\textit{Table 3-1: Verification Tasks and Results}

<table>
<thead>
<tr>
<th>Purpose</th>
<th>Variable Output (Current model/Franklin)</th>
<th>Verification Tasks</th>
<th>Successfully Verified?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Define Model Parameters</td>
<td>L0, L1, L2, L3, L4, L5, L6, L7, lb, m0, m1, m2, m3, m4, m5, g, cent1, cent2, cent3, IKp, IDp, Ikt, IDt, IVD, Lord_Angs/...,AAPOA</td>
<td>1.1) Compare variables in Workspace to desired model parameters.</td>
<td>✓</td>
</tr>
<tr>
<td>2. Import Muscle File</td>
<td>M/M</td>
<td>2.1) Compare Cholewicki muscle coordinates to the current model’s muscle coordinates. This will be performed to ensure that data entry errors have not occurred in our Excel spreadsheet and that our MATLAB function imported/read the Excel file appropriately.</td>
<td>✓</td>
</tr>
</tbody>
</table>

Note: This comparison should be made prior to the
transformation aligning all vertebrae along Axis 3 and the application of lordosis angles.

2.2) Compare current model’s muscle coordinates to Franklin’s muscle coordinates. This will allow for the comparison between our Excel spreadsheet and Franklin’s spreadsheet describing the muscle coordinates and paths. Additionally, this will be performed to ensure that our MATLAB function substituted the numeric coordinates on Sheet 2 of the Excel spreadsheet to the appropriate muscle attachment point name on Sheet 1 of the Excel spreadsheet.

Note: This comparison should be made after the transformation is applied so that all vertebrae are aligned along Axis 3, but the lordosis angles should NOT be applied.

2.3) Compare Cholewicki’s muscle paths to the current model’s muscle paths. Compare the muscle paths described on Sheet 1 of our Excel spreadsheet to the muscle paths described in Table A1 (Cholewicki et al.)

2.4) Compare the current model’s muscle paths to Franklin’s muscle paths. Compare the muscle paths described on Sheet 1 of our Excel spreadsheet and on Sheet 1 of Franklin’s Excel spreadsheet to compare the muscle path of each muscle.

2.5) Compare output of the current model’s load_musclefile.m and Franklin’s muscinput.m file

Note: the Excel spreadsheet utilized by the functions should be the same to ensure constant inputs to the functions.

3. Import External Force File

load_forcefile.m

3.1) Manually check that the correct coordinates have been substituted for the assigned attachment points specified on the LoadLift.xls Excel sheet.

3.2) Compare outputs from Franklin’s forcinput and format functions and the current model’s load_forcefile function

4. Determine Rotation Matrices

rotation_matrices, d_rotation_matrices/Rm, dRm

4.1) Compare output from Franklin’s analys1.mex64 file and rotationmatrices.mlx
<table>
<thead>
<tr>
<th>5. Determine Body Origins</th>
<th>origin, d_origin/Or, dOr</th>
<th>5.1) Compare output from Franklin’s analys1 MEX file and origin_body.mlx</th>
<th>✓</th>
</tr>
</thead>
<tbody>
<tr>
<td>6. Transform Anatomy</td>
<td>Ma/Ma</td>
<td>6.1) Compare output from Franklin’s analys1 MEX file and rotate_anatomy_sym.mlx</td>
<td>✓</td>
</tr>
<tr>
<td>rotate_anatomy.m</td>
<td></td>
<td>Note: In order to compare the transformed Ma, the same input M needs to be used for both functions/files.</td>
<td></td>
</tr>
<tr>
<td>7. Muscle Properties</td>
<td>ML, MV/ML, MV</td>
<td>7.1) Compare output from Franklin’s analys1.mex64 file and the rotate_anatomy_sym.mlx script</td>
<td>✓</td>
</tr>
<tr>
<td>rotate_anatomy_sym.mlx</td>
<td></td>
<td>7.2) Perform hand calculations to verify the methodology implemented in the rotate_anatomy_sym.mlx script used to calculate muscle lengths and velocities</td>
<td></td>
</tr>
<tr>
<td>8. Transforming the External Force</td>
<td>Ea/Ea</td>
<td>8.1) Compare output from Franklin’s analys1 MEX file and rotateforce.mlx</td>
<td>✓</td>
</tr>
<tr>
<td>rotateforce.mlx</td>
<td></td>
<td>8.2) Compare output Ea from hand calculations and the current model’s rotateforce.mlx</td>
<td>✓</td>
</tr>
<tr>
<td>9. Dynamics</td>
<td>Cm/Cv</td>
<td>9.1) Compare output from Franklin’s dynmatsc MEX file and dynam.mlx</td>
<td>X</td>
</tr>
<tr>
<td>dynam.mlx</td>
<td></td>
<td>9.3a) Compare output from Franklin’s Mathematica code and dynam.mlx</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>Gm/Gv</td>
<td>9.2) Compare output from Franklin’s dynmatsc MEX file and dynam.mlx</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>Mm/Mv</td>
<td>9.3b) Compare output from Franklin’s Mathematica code and dynam.mlx</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>Gl/Gl</td>
<td>9.4) Compare the “helper variables” from Franklin’s analys1.mex64 and dynam.mlx</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>Gr/Gr</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10. Intervertebral Disc Moment</td>
<td>PRQ/PRQ</td>
<td>10.1) Compare output from Franklin’s intervertc MEX file to the current model’s IVD_calcs.mlx file</td>
<td>X</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------</td>
<td>-------------------------------------------------------------------------------------------------</td>
<td>----</td>
</tr>
<tr>
<td>IVD_calcs.mlx</td>
<td></td>
<td>10.2) Ensure that the body and plane were parallel after planar and twist rotations were applied to the plane.</td>
<td>✓</td>
</tr>
</tbody>
</table>

| 11. Generalized Forces | Q/Q | 11.1) Compare output from Franklin’s geneforcesc MEX file and the current model’s generalizedForce function | ✓ |

| A | 12.1) Compare constraint variables used in fmincon/developed during the optimization procedure in the current model to Franklin’s outputs | ✓ |
| B | Note: this procedure will verify the use of the spineaccA/AccelerationMA and spineaccP/AccelerationMP functions as they are used to determine Aeq and beq |
| Aeq | |
| beq | |
| Un_opt | 12.2) Ensure that all muscle activations fall within the acceptable range between zero and one | ✓ |
| qddP, qddA | 12.3) Compare the angular accelerations for the current model’s spineaccP and spineaccA functions with Franklin’s AccelerationMP and AccelerationMA functions | ✓ |
| Ja | 12.4) Compare the Jacobians Ja and Jp from Franklin’s alg_optimize.m and the current model’s optimize.m | ✓ |
| | 12.5) After the constrained optimization procedure is completed, ensure that the solution adheres to the stability requirement | ✓ |
| P | 12.6) Compare the output of the current model’s costfunct function and Franklin’s costfun function | ✓ |

| 13. Muscle Model | MF, dA | 13.1) Compare Franklin’s muscmodel.m output to the current model’s musclemodel.mlx output | ✓ |

| 14. Reflex Model | MFR, dAr | 14.1) Perform hand calculations to verify the output of the current model’s musclemodelR function | ✓ |
15. Time Delay Analysis

<table>
<thead>
<tr>
<th>File</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tdinvest.m</td>
<td>15.1) Perform nonlinear simulation with delay margin. The linear system should successfully predict a delay margin that will result in a stable system. (Delay that is simulated should still be delay margin * 0.99) ✓</td>
</tr>
<tr>
<td>tdfind.m</td>
<td></td>
</tr>
<tr>
<td>tdfmax.m</td>
<td></td>
</tr>
</tbody>
</table>

16. Nonlinear Verification

<table>
<thead>
<tr>
<th>File</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>spine_dyn.mlx</td>
<td>16.1) Calculate the distance of the normalized state variables from equilibrium Dss and determine if the system is approaching equilibrium. ✓</td>
</tr>
</tbody>
</table>

The verification procedure `For_verification.m` can be initialized in the `spine_main.m` file (Figure 3-1).

```matlab
clear all; close all; clc;
%% Verification
% Create Excel File with verification outputs.
global date
date=datetime('now');
verification_logical=1; % indicates that verification should be performed
save('verification_initialize_vars.mat','date','verification_logical')

if verification_logical==1
    For_verification %has clc and clear commands in file
    clear; clc
end
```

*Figure 3-1: Initializing the verification procedure in spine_main.m file*

For the verification procedure, the `Stateo_Pango_Rand.mat` file was used to load the Stateo and Pango vectors that were utilized as inputs of functions. These Stateo and Pango vectors were randomly generated, with each value in the vector ranging between zero and one, with all values being nonzero.

The `Model_Parameters_Verify.mat` file was also used for the verification procedure. The variables `cent1`
and cent2 needed to have a magnitude of zero for comparison between Franklin’s output and our output, but all other values were set to be nonzero.

The absolute error calculated between our outputs and Franklin’s were calculated in MATLAB (Equation 3-1):

\[
\text{Absolute Error} = |\text{Jardon output} - \text{Franklin output}|
\] (3-1)

The verification tasks completed in the For_verification.m file are exported to an Excel spreadsheet (Figure 3-2). In some instances, for output matrices with large dimensions, the greatest magnitude of absolute error determined in the verification task was solely presented for simplicity, instead of the absolute error of all matrix indices.

```matlab
%% Exporting Verification to Excel Spreadsheet
global date
warning('off','MATLAB:xlswrite:AddSheet');

filename=sprintf('Verification Summary %s.xlsx',date);
filename=replace(filename,':','.'); %cannot have colons in file titles
filename=append('C:\Users\Valerie\Jardon\Documents\Model\FINALIZED_CODE_updated\Simulation Runs Verification Output',filename));
writematrix(date,filename,'Sheet',1,'Range','A1')
writecell({'Rotation Matrices'},filename,'Sheet',2,'Range','C3')
writematrix(Rm_diff,filename,'Sheet',2,'Range','C4')
```

Figure 3-2: Example of exporting verification task results to Excel spreadsheet

3.1 Model Parameters

For the verification of the model parameters, the stored variables in the MATLAB workspace were compared to the intended inputs to ensure input error did not occur. All model parameters defined in the model_parameters.m file were accurately drafted in MATLAB and stored correctly in the MATLAB workspace (Table 3-1, Verification Task 1.1).
3.2 Initial Transformation of Cholewicki et al. Muscle and Skeletal Coordinates

3.2.1 Muscle and Skeletal Coordinates Verification

Franklin et al.\textsuperscript{16,17,24} uses the Cholewicki et al.\textsuperscript{54} muscle geometry and skeletal coordinates. These coordinates are for the spine in neutral position, indicating that lumbar lordosis is present and impacting the coordinate positions, but no additional flexion, extension or bending of the spine is present.

Franklin specified the segmental lordosis angles of the vertebrae in the appendix of his thesis.\textsuperscript{24} These lordosis angles differ from those incorporated into the Cholewicki et al. muscle/skeletal coordinates\textsuperscript{15}, so each vertebra needs to be rotated to “remove” the default lumbar lordosis (process shown in Figure 3-3). This will result in the alignment of the connection points of the vertebra along the vertical axis, indicating that the vertebral faces are parallel.

Franklin provides the body fixed coordinates in the Appendix of his thesis\textsuperscript{24}. For these coordinates, the vertebrae have been rotated so that their connection points align along the vertical axis. The origin was subtracted from each point associated with that body, so that the coordinates would be described by their relative location from the origin.

There are three skeletal coordinates that differed between Franklin’s muscle Excel file \textit{CholeMusc.xls} and current model’s muscle Excel file \textit{CholeMusc_update.xls}. Upon further investigation, it appears that Franklin may have had data entry errors for points \textit{RIB10, RIB21} and \textit{RIB22} (Table 3-1, Verification Task 2.2).

The unadjusted global coordinates from Cholewicki et al.\textsuperscript{15}, following our coordinate system convention, are as follows:

\[
RIB1 = [0, 7.2, 35.5] \quad \text{global origin}
\]
\[ RIB_{10} = [2, (2 - 7.2), (53.5 - 35.5)] = [2, -5.2, 18] \]

\[ RIB_{21} = [6.5, (3.6 - 7.2), (26.4 - 35.5)] = [6.5, -3.6, -9.1] \]

\[ RIB_{22} = [2, (3.6 - 7.2), (24.2 - 35.5)] = [6.5, -3.6, -11.3] \]

By subtracting the origin of the body from these coordinates, the body-fixed coordinates can be determined:

\[ RIB_{10} = [2, (2 - 7.2), (53.5 - 35.5)] = [2, -5.2, 18] \]

\[ RIB_{21} = [6.5, (3.6 - 7.2), (26.4 - 35.5)] = [6.5, -3.6, -9.1] \]

\[ RIB_{22} = [2, (3.6 - 7.2), (24.2 - 35.5)] = [6.5, -3.6, -11.3] \]

As shown in Figure 3-3, the coordinates can then be rotated to align along Axis 3, yielding:

\[ RIB_{10} = [2, -5.36, 17.95] \]

\[ RIB_{21} = [6.5, -3.51, -9.13] \]

\[ RIB_{22} = [6.5, -3.49, -11.33] \]
Figure 3-3: "Removing" the default lordosis angles: Transforming coordinates to align the connection points of the vertebrae along Axis 3.

Next, using the coordinates Franklin provides in the appendix of his thesis\textsuperscript{24} (recall that these coordinates are descriptive of when the vertebral connection points are aligned along the vertical axis), we can determine that these would have been the non-transformed coordinate points he would have sourced from Cholewicki and McGill\textsuperscript{15}:

$$\begin{align*}
RIB_{10} &= [2, -5.2, 19] \\
RIB_{21} &= [6.5, -3.6, 0.9] \\
RIB_{22} &= [6.5, -3.6, -1.3]
\end{align*}$$
Figure 3-4 demonstrates that the transformation of these coordinate points would result in the coordinates provided in Franklin’s thesis.\textsuperscript{24}

**Figure 3-4:** Determining Franklin’s initial coordinates prior to transformation for selected attachment points.

Based on the non-transformed coordinates, it appears that errors in the z-coordinate for these attachment points resulted in the discrepancies between the current model’s calculations and Franklin’s calculations. It seems plausible that in this case, Franklin either adjusted the location of the point from the data provided by Cholewicki \textit{et al.}\textsuperscript{54} or data entry errors occurred. It is assumed that Franklin unintentionally adjusted the coordinates due to no mention that he knowingly adjusted them.

For the current model, the skeletal muscle coordinates that define paths of the muscle have been manually verified with Cholewicki \textit{et al.} prior to transformation (Table 3-1, Verification Task 2.1).
3.2.2 Muscle Path Verification

The muscle paths provided in the appendix of Franklin’s thesis were compared to those provided in the appendix of Cholewicki and McGill, the source of the coordinates and muscle paths, after differences between the muscle anatomy for the current model and Franklin’s model were discovered (Table 3-1, Verification Task 2.4). It became evident that the nodes of muscles ParsL4, ParsL3, ParsL2, ParsL1, IlioLum, LongTP, LongTL5, LongTL4, LongTL3, and LongTL2 on both the left and right side of the body were not consistent between the studies. It appears that Franklin had a compounding data entry error for these muscles.

The $M$ matrix describes the muscle path of each muscle as a row in the matrix, listing the coordinates starting with the origin and terminal, followed by the nodes. Evidently, when drafting his Excel spreadsheet, Franklin introduced the nodes of ParsL2 two rows prior, so they are associated with ParsL4 instead (Table 3-2). Franklin’s muscle paths for ParsL4, ParsL3, ParsL2, ParsL1, IlioLum, LongTP, LongTL5, LongTL4, LongTL3, and LongTL2 for the left side of body (Table 3-2) can be compared to that of Cholewicki and McGill (Table 3-3). The compounding data entry error can be identified.

Table 3-2: Franklin muscle paths described in appendix for left side of the body

<table>
<thead>
<tr>
<th>LParsL4</th>
<th>LPEL9</th>
<th>LL43</th>
<th>LL410</th>
</tr>
</thead>
<tbody>
<tr>
<td>LParsL3</td>
<td>LPEL9</td>
<td>LL33</td>
<td>LL410</td>
</tr>
<tr>
<td>LParsL2</td>
<td>LPEL9</td>
<td>LL23</td>
<td>LL411</td>
</tr>
<tr>
<td>LParsL1</td>
<td>LPEL9</td>
<td>LL13</td>
<td>LL412</td>
</tr>
<tr>
<td>LilioLum</td>
<td>LPEL10</td>
<td>LRIB8</td>
<td>LL413</td>
</tr>
<tr>
<td>LLongTP</td>
<td>LPEL11</td>
<td>LRIB9</td>
<td>LL312</td>
</tr>
<tr>
<td>LLongTL5</td>
<td>LL54</td>
<td>LRIB10</td>
<td>LL212</td>
</tr>
<tr>
<td>LLongTL4</td>
<td>LL44</td>
<td>LRIB11</td>
<td>LL111</td>
</tr>
<tr>
<td>LLongTL3</td>
<td>LL34</td>
<td>LRIB12</td>
<td></td>
</tr>
<tr>
<td>LLongTL2</td>
<td>LL24</td>
<td>LRIB13</td>
<td></td>
</tr>
</tbody>
</table>
For the current model, the muscle paths utilized match those provided by Cholewicki and McGill\textsuperscript{15} (Table 3-1, Verification Task 2.3). In his thesis, Franklin indicates that his muscle anatomy is based on that of Cholewicki and McGill\textsuperscript{15} and that no alterations have been made to the paths provided\textsuperscript{24}, confirming the belief that these adjustments to the muscle paths were most likely unintentional.

### 3.3 Import Muscle Geometry and Skeletal Coordinates

To import the muscle paths and coordinates into MATLAB for the model, Franklin utilized function `muscinput` to read an Excel spreadsheet that described the muscle paths on one sheet and the coordinates on another\textsuperscript{24}. Due to errors determined in Franklin’s muscle paths and skeletal coordinates (section 3.2), an updated copy of Franklin’s Excel spreadsheet was created and utilized to compare the output of the functions used to import the muscle anatomy file.

The absolute error between the muscle anatomy matrix $M$ output for Franklin’s `muscinput` function and the current model’s `load_musclefile` function should be calculated (Table 3-1, Verification Task 2.5).

There are two calculations present for the Verification 2.2 and 2.4 Tasks (Figure 3-5). These verification tasks incorporate comparing the current model’s muscle coordinates and muscle paths to that of Franklin. The muscle paths and coordinates are used in the development of the muscle anatomy.

<table>
<thead>
<tr>
<th>LParsL4</th>
<th>LPEL9</th>
<th>LL43</th>
</tr>
</thead>
<tbody>
<tr>
<td>LParsL3</td>
<td>LPEL9</td>
<td>LL33</td>
</tr>
<tr>
<td>LParsL2</td>
<td>LPEL9</td>
<td>LL23 LL410</td>
</tr>
<tr>
<td>LParsL1</td>
<td>LPEL9</td>
<td>LL13 LL410</td>
</tr>
<tr>
<td>LilioLum</td>
<td>LPEL10</td>
<td>LRIB8 LL411 LL310 LL210 LL19</td>
</tr>
<tr>
<td>LLongTP</td>
<td>LPEL11</td>
<td>LRIB9 LL412 LL311 LL211 LL110</td>
</tr>
<tr>
<td>LLongTL5</td>
<td>LL54</td>
<td>LRIB10 LL413 LL312 LL212 LL111 LRIB23</td>
</tr>
<tr>
<td>LLongTL4</td>
<td>LL44</td>
<td>LRIB11 LL312 LL212 LL111 LRIB23</td>
</tr>
<tr>
<td>LLongTL3</td>
<td>LL34</td>
<td>LRIB12 LL212 LL111 LRIB23</td>
</tr>
<tr>
<td>LLongTL2</td>
<td>LL24</td>
<td>LRIB13 LL111 LRIB23</td>
</tr>
</tbody>
</table>
matrix $M$. Thus, the muscle anatomy matrix $M$ was utilized in these verification tasks. In Figure 3-5, the calculations performed using the `verify` function with the inputs $M$ and $M_F$ involve Franklin’s original Excel file. This run of the verification task will result in unsuccessful verification due to the previous errors in Franklin’s $M$ file. The calculations performed using the `verify` function with the inputs $M$ and $M_F2$ involve use of the corrected Excel file in Franklin’s function. This corrected Excel file should be an identical input that is used in the functions being compared, so that the resulting absolute error is based on the function procedure and not errors related to the functions’ inputs.

```
[M_check_count,~,M_diff,M_max_error]=verify(M(:,1:32),M_F(:,1:32),1e-3) %Verification Task 2.2 & 2.4
[M2_check_count,~,M2_diff,M2_max_error]=verify(M(:,1:32),M_F2(:,1:32),1e-3) %Verification Task 2.2 & 2.4
```

*Figure 3-5: Verification Tasks 2.2 and 2.4*

For the resulting outputs of $M$ from the current model’s `load_musclefile.m` file and Franklin’s `muscininput.m` file, the maximum error occurring between an element of the matrices was calculated, yielding a magnitude of $4.98e-07$ (Figure 3-6). The magnitude of this error was not concerning, and therefore Verification Task 2.5 was satisfied.

```
>> max_M2_diff=max(M2_diff,[],'all')
max_M2_diff =
    1.9769e-07
```

*Figure 3-6: Verification Task 2.5 Absolute Error*
3.4 Import External Force

Similarly to the method used to import the muscle anatomy, an Excel spreadsheet *LoadLift.xls* was used to define the attachment points of the external load applied to the model. The first column of the \( E \) vector defines the number of attachment points describing the force whereas the four columns that follow describe the coordinates and body of the origin and the last four columns define the coordinates and body of the terminal.

The outputs of Franklin’s *forcinput.m* and *format.m* files and the current model’s *load_forcefile.m* file are compared (Figure 3-7), yielding a maximum error of 0 (Figure 3-8), so Verification Task 3.2 is satisfied (Table 3-1).

![Figure 3-7: Verification Task 3.2](image)

\[
\begin{array}{cccccccc}
E_{\text{check\_count}} & E_{\text{check}} & E_{\text{diff}} & E_{\text{max\_error}} \\
\end{array}
\]

\[\text{verify}(E,E_{\text{F}},1\times10^{-15})\]

\[
\begin{array}{cccccccc}
E_{\text{diff}} = & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{array}
\]

![Figure 3-8: Verification Task 3.2 Absolute Error](image)

Additionally, it was manually reviewed and verified that the correct coordinates were substituted for the assigned attachment points specified on the *LoadLift.xls* Excel sheet, and that the external load was being applied in the intended location (Verification Task 3.1).
3.5 Rotation Matrices

Each rigid body is rotated first about Axis 1, then Axis 2, and lastly around Axis 3. To apply these rotations to the bodies, a rotation matrix was used. The current model’s rotation matrices are based on the methods discussed by Yamaguchi\(^{81}\) where a rotation in the counterclockwise direction was defined as positive.

\[ R_{1_{CCW+}} = \begin{bmatrix} \cos \theta & 0 & -\sin \theta \\ 0 & 1 & 0 \\ \sin \theta & 0 & \cos \theta \end{bmatrix} \]

\[ R_{2_{CCW+}} = \begin{bmatrix} \cos \theta & 0 & \sin \theta \\ 0 & 1 & 0 \\ -\sin \theta & 0 & \cos \theta \end{bmatrix} \]

\[ R_{3_{CCW+}} = \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \]

Using matrix multiplication in the order that the rotations are applied to the body, a rotation matrix can be derived that includes all of the rotations applied to a body:

\[ R_{123} = R_{1_{CCW+}} \ast R_{2_{CCW+}} \ast R_{3_{CCW+}} \]

Based on the Mathematica (Wolfram, Champaign, IL) code provided by Franklin in Appendix A of his thesis,\(^{24}\) it is clear that clockwise rotation is defined as positive in this case.

\[ R_{1_{CW+}} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \theta & \sin \theta \\ 0 & -\sin \theta & \cos \theta \end{bmatrix} \]

\[ R_{2_{CW+}} = \begin{bmatrix} \cos \theta & 0 & -\sin \theta \\ 0 & 1 & 0 \\ \sin \theta & 0 & \cos \theta \end{bmatrix} \]

\[ R_{3_{CW+}} = \begin{bmatrix} \cos \theta & \sin \theta & 0 \\ -\sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \]
When Franklin applies the rotation matrices in the Mathematica code, he often applies his rotation matrices using the transpose:

\[ R_{123} = (R_{3_{CW+}} * R_{2_{CW+}} * R_{1_{CW+}})^T \]

This would result in the same rotation matrix you would derive if counterclockwise rotations were defined as positive and the rotation matrices were multiplied sequentially in the order that the rotation was applied to the body:

\[ R_{123} = (R_{3_{CW+}} * R_{2_{CW+}} * R_{1_{CW+}})^T = R_{1_{CCW+}} * R_{2_{CCW+}} * R_{3_{CCW+}} \]

In Franklin’s `analys1.mexw64` file, it is evident that the rotation matrices are in the form:

\[ R_{321} = R_{3_{CW+}} * R_{2_{CW+}} * R_{1_{CW+}} \]

Therefore, in order to be able to compare the current model’s output to Franklin’s output, the rotation matrices needed to be transposed and then the difference calculated between the outputs.

The `rotation_matrices` and `d_rotation_matrices` outputs of the function handles `Rm_funct` and `dRm_funct` were compared to the outputs `Rm` and `dRm` from Franklin’s `analys1.mexw64` file (Figure 3-9).

![Figure 3-9: Verification Task 4.1](image)

The absolute difference for the rotation matrices and differential rotation matrices did not result in a magnitude greater than 1E-15 (Figure 3-10), therefore Verification Task 4.1 was satisfied (Table 3-1).
3.6 Body Origins

The outputs origins and d_origins of the origin_funct and dorigin_funct function handles were compared to the outputs Or and dOr from Franklin’s analys1.mexw64 file (Figure 3-11).

```
280  [Or_check_count, Or_check, Or_diff]=verify(origin,Or,F,1E-15)  %Verification Task 5.1
281  [dOr_check_count, dOr_check, dOr_diff]=verify(dorigin,dOr,F,1E-15)  %Verification Task 5.1
```

Figure 3-11: Verification Task 5.1
The absolute difference for the origins and differential origins did not result in a magnitude greater than $1E^{-15}$ (Figure 3-12), therefore the magnitude of the error was not concerning, and Verification Task 5.1 was considered satisfied (Table 3-1).

![Table of absolute errors](image)

*Figure 3-12: Verification Task 5.1 Absolute Error*

### 3.7 Transform Muscle Geometry and Skeletal Coordinates

The process of transforming the skeletal coordinates/muscle attachment points is dependent on the origin and rotation matrices derived (Chapter 3.5 and 3.6).

Franklin’s rotated muscle anatomy matrix $Ma$ is a 90x33 matrix with each row describing the muscle attachment points for a muscle. Similarly, the current model’s rotated muscle anatomy matrix $Ma$ is a 90x35 matrix with the last two columns allocated to the CSA and the tendon lengths.

The output of the function handle $Ma\_funct$ was compared to the $Ma$ output from Franklin’s *analys1.mexw64* file (Figure 3-13). To complete this verification task, the functions needed to have identical muscle anatomy $M$ inputs. The greatest magnitude of absolute error resulting was $1.11E^{-16}$ (Figure 3-14), therefore Verification Task 6.1 was considered satisfied (Table 3-1).
3.8 Muscle Properties

The muscle lengths and velocities from Franklin’s `analys1.mexw64` file were compared to those obtained from the current model’s function handles `ML_funct` and `MV_funct` (Figure 3-15). The magnitude of error between these measures was 1.11e-16 and 1.94e-16 for muscle length and muscle velocity respectively (Figures 16-17). Verification Task 7.1 was satisfied due to the small magnitudes of error calculated (Table 3-1).
Additionally, hand calculations were performed to determine a few of the muscle lengths and muscle velocities to verify the outputs of the current model’s `rotate_anatomy_sym.mlx` script to ensure that the methodology was implemented in the code correctly. The results of these hand calculations matched the output of the `rotate_anatomy_sym.mlx` script, so the Verification 7.2 Task was successful (Table 3-1).

### 3.9 Transform External Force

The output of the function handle `Ea_funct` was compared to the output `Ea` from Franklin’s `analys1.mexw64` file (Figure 3-18). The resulting absolute error was less than 1e-15 (Figure 3-19), so Verification Task 8.1 was successful (Table 3-1).
Additionally, hand calculations were performed to ensure that the external force attachment points were being rotated correctly (Table 3-1, Verification Task 8.2). The output $E_a$ from the current model’s `rotateforce.mlx` script matched the hand calculations performed for this verification task. Therefore, this verification task was deemed successful (Table 3-1, Verification Task 8.2).

### 3.10 Dynamics

For Verification Task 9.1, the Coriolis vector was compared for the current model’s `Cm_funct` function handle and Franklin’s `dynmatsc.mex64` file (Figure 3-20). As shown in Figure 3-21, the magnitude of the error between the outputs is concerning for verification, with error ranging from zero to 0.1067 in magnitude. Verification Task 9.1 was unsuccessful (Table 3-1).

```
[Cm_check_count, ~,Cm_diff,Cm_max_error]=verify(Cm,Cm_F',1E-15) %Verification Task 9.1
```

*Figure 3-20: Verification Task 9.1*

<table>
<thead>
<tr>
<th>Cm_diff</th>
</tr>
</thead>
<tbody>
<tr>
<td>Columns 1 through 12</td>
</tr>
<tr>
<td>0.0651 0.0031 0.0000 0.0632 0.0676 0.0000 0.0370 0.0703 0 0.0390 0.0380 0</td>
</tr>
<tr>
<td>Columns 13 through 18</td>
</tr>
<tr>
<td>0.0183 0.0187 0.0000 0.0511 0.1067 0</td>
</tr>
</tbody>
</table>

*Figure 3-21: Verification Task 9.1 Absolute error*

In the appendix of his thesis, Franklin provided the Mathematica code he developed and utilized that included the calculation of the Coriolis vector. The outputs for $C_v$ for the Mathematica code and the output yielded from `dynmatsc.mex64` revealed that Franklin’s codes did not yield the same results. The results for the current model’s `Cm_funct` function handle results agreed with the Mathematica output (Table 3-1, Verification Task 9.3a).
For Verification Task 9.2, the gravity vector and mass matrix were compared for the current model’s `Gm_funct` and `Mm_funct` function handles and Franklin’s `dynmatsc.mex64` file (Figure 3-22).

```
[Mm_check_count, ~, Mm_diff, Mm_max_error] = verify(Mm, Mm_F, 1E-15) \%Verification Task 9.2
[GM_check_count, ~, Gm_diff, Gm_max_error] = verify(Gm, Gm_F', 1E-15) \%Verification Task 9.2
```

**Figure 3-22: Verification Task 9.2**

For the gravity vector and mass matrix, the maximum absolute error magnitude was respectively 4.4409e-16 and 5.5511e-17 (Figure 3-23 and 3-24), resulting in successful verification for Task 9.2 (Table 3-1).

```
Gm_diff =
1.0e-15 *
Columns 1 through 12
 0.2220  0.4441  0  0  0.2220  0  0  0  0  0  0.2220  0  0
Columns 13 through 18
 0  0.0278  0  0  0  0  0
```

**Figure 3-23: Verification Task 9.2 Absolute Error for Gravity Vector**

```
Mm_max_error =

5.5511e-17
```

**Figure 3-24: Verification Task 9.2 Absolute Error for Mass Matrix**

Additionally, the gravity vector and mass matrix outputs were compared between Franklin’s Mathematica code and the current model’s `Gm_funct` and `Mm_funct` function handles (Table 3-1,
Verification Task 9.3b). The error between the gravity vectors and mass matrix was small in magnitudes, so Verification Task 9.3b was deemed successful.

For Verification Task 9.4, the partial derivatives \( \frac{\partial \mathbf{v}_b}{\partial \mathbf{q}_i} \) and \( \frac{\partial \omega_b}{\partial \mathbf{q}_i} \) utilized in the generalized force equation (Equation 2-25) were compared for the current model’s Gl_funct and Gr_funct function handles and Franklin’s analys1.mex64 file (Figure 3-27).

![Figure 3-25: Verification Task 9.4](image)

The variables Gl and Gr are 108x3 matrices, so due to their vast size, the maximum absolute error was determined. The maximum absolute error for Gl was 5.2042e-18 and 1.1102e-16 for Gr (Figures 3-28 and 3-29). The magnitudes of these errors were minor, therefore Verification Task 9.4 was considered successful.

![Figure 3-26: Verification Task 9.4 Absolute Error for Gl](image)
When writing the code for calculating the intervertebral disc moments, it became evident that Franklin’s `intervertc.mexw64` file did not rotate the dummy plane correctly. First, the plane should be rotated so that it has the same lordosis angle as the body it is being compared to. The remaining rotation around Axis 1 is deemed the planar rotation and contributes to the moment. Then the plane should be rotated by this planar angle around Axis 1, and the plane and body should be parallel. If Axis 3 of the plane and body are not aligned, then a twist angle is present. The plane should be rotated by the twist angle so that Axis 3 of the plane and body are aligned. For the body and plane to be aligned and parallel, they would end the calculations with the same rotation matrix. This would indicate that their body axes would be aligned, further confirming that all angles have been accounted for in our calculations.

Through recreation of Franklin’s methodology to determine why our codes were not resulting in the same results, Franklin’s error was discovered. The dummy plane and body were not aligned at the end of the code, impacting angle measurements and moment calculations.

Calculating the difference between the outputs of Franklin’s `intervertc.mexw64` code and the current model’s code `IVD_calcs.mlx` for a verification task resulted in a large magnitude of error, deeming Verification Task 10.1 as unsuccessful (Table 3-1).
As mentioned previously, the dummy plane and body should be aligned and parallel if the body was rotated appropriately by the planar angle and twist angle, according to Franklin’s methodology. Therefore, a verification task was implemented in the *IVD_calcs.mlx* script to ensure that the body and dummy plane are aligned and parallel through use of the resulting rotation matrices which describe the orientation of the axes.

```matlab
[PRQ_check_count, ~,PRQ_diff,PRQ_max_error]=verify(PRQ,PRQ_F,1E-15) %Verification Task 10.1

%Check to see if planes are aligned and parallel
planes_check=body1_Rm_unadjusted-body2_Rm;
num_planes_check=numel(planes_check);
failure_vect=zeros(num_planes_check,1);

for j=1:num_planes_check
    if abs(planes_check(j)) > 1E-7
        failure_vect(j)=1;
    else
        failure_vect(j)=0;
    end
end
if sum(failure_vect)==0
    error(['Cannot continue. Planes are not successfully aligned and/or parallel.'])
else
    fprintf('SUCCESS. Iteration %2d. This plane is aligned and parallel.\n',i)
end
```

*Figure 3-29: Verification task 10.2 in the IVD_calcs.mlx script*

This verification task is evaluated each time the *IVD_calcs.mlx* script is called in a simulation, ensuring that the dummy plane and body are parallel and aligned (Table 3-1, Verification Task 10.2).
3.12 Generalized Forces

The resulting generalized force vector $Q$ was compared for the current model's `generalizedForce` function and Franklin’s `genforcesc.mex64` (Figure 3-30). From the calculated absolute difference, the maximum error magnitude was $1.0658 \times 10^{-14}$ (Figure 3-31). Thus, due to the small magnitude of error, Verification Task 11.1 was considered successful (Table 3-1).

![Figure 3-30: Verification Task 11.1]

```
[Q_check_count, ~, Q_diff, Q_max_error] = verify(Q, Q_F, 1E-15) % Verification Task 11.1
```

![Figure 3-31: Verification Task 11.1 Absolute Error]

```
Q_diff =
1.0e-13 *
0.0178
0
0
0.1066
0
0.0711
0.0533
0.0044
0.0178
0.0355
0.0711
0
0.0355
0.0178
0.0408
0.0711
0.0808
0.0355
```
3.13 Optimization

3.13.1 Constraint Variables

For the optimization procedure, fmincon allows for constraints to be defined that the solver must adhere to while minimizing the metabolic power.

For the equilibrium constraint, the passive and active must be balanced:

\[ A_{eq} \alpha = b_{eq} \]

The constraint variables \( A_{eq} \) and \( b_{eq} \) are used for the equilibrium requirement. For Verification Task 12.1, the constraint variables were compared for Franklin’s alg_optimize script and the current model’s optimize.m script (Figure 3-32).

```matlab
[Aeq_check_count, ~,~] = verify(Aeq, Aeq_F, 1E-15) %Verification Task 12.1
[beq_check_count, ~,~] = verify(beq, beq_F, 1E-15) %Verification Task 12.1
```

*Figure 3-32: Verification Task 12.1*

Additionally, in the optimization procedure, \( A \) is an identity matrix multiplied by \(-1\). \( B \) is defined as a zeros vector. With muscle activations falling between a magnitude of zero and one for each muscle, the following relationship should be met:

\[ A \alpha \leq B \]

If this condition is not met, then muscle activations have the incorrect sign assigned to them.

Verification Task 12.1 includes the evaluation of constraint variables \( A \) and \( B \) as well for Franklin’s alg_optimize script and the current model’s optimize.m script (Figure 3-33).
The resulting maximum absolute errors are of a small magnitude, with constraint variable $A_{eq}$ having the greatest error with a magnitude of $6.9122 \times 10^{-11}$, followed by $b_{eq}$ with an error of $8.5265 \times 10^{-14}$ (Figures 3-34 and 3-35). The absolute error magnitude for constraint variable A and B was zero (Figures 3-36 and 3-37). The magnitude of these errors are small, so Verification Task 12.1 has been satisfied (Table 3-1).
Additionally, the constraint for muscle activation is as follows, limiting the muscle activation of each muscle:

\[ 0 \leq \alpha \leq 1 \]

This constraint is implemented in the form of matrices, with the length correlating to the number of muscles in the model. The muscle activation is reviewed for the most optimal solution and to ensure that all muscle activations fall between zero and one (Verification Task 12.2).

Additionally, the nonlinear constraint function \textit{nlcon.mlx} needs to be verified. This function ensures that the Jacobians have negative real parts, indicating that the system is stable. After the constrained optimization is performed and the optimal solution is determined, Verification Task 12.5 is performed (Figure 3-38) to ensure that the system is stable (Table 3-1).
The angular acceleration can be determined using Equation 3-1:

$$\ddot{\theta}(t) = M^{-1}_m (Q - C_m - G_m)$$  \hspace{1cm} (3-1)

In the current model, the angular acceleration $qdd$ is determined in the $\text{spineaccA}$, $\text{spineaccP}$ and $\text{spineacc}$ functions. The angular acceleration is used to build the Jacobians, for example (Equation 3-2):

$$J_{Pi} = \frac{qddP + beq}{(0.00001 \times \frac{\pi}{180})}$$  \hspace{1cm} (3-2)

Therefore the angular accelerations need to be verified (Table 3-1, Verification Task 12.3). The difference between Franklin’s angular acceleration outputs and the current model’s angular acceleration can be determined (Figure 3-39). The dimensions of the $qdda$ matrix is 90x36x36 but for the verification task, only the first page of $qdda$ is compared for more efficient computation and simplicity.

```matlab
% Check conditions once again for the optimal solution
Un_opt=Un_best;
   J=Jp;
   for i=1:36
      J(19:36,i)=J(19:36,i)+(Un_opt'*Ja(:,19:36,i))';
   end

% Check the eigenvalues
   eig_val=eig(J);
   max_eig=max(real(eig_val));
```

Figure 3-38: Verification Task 12.5

### 3.13.2 Jacobians

The angular acceleration can be determined using Equation 3-1:
In the angular acceleration verification task, the maximum absolute error for \( qddA \) resulted in a magnitude of 7.094e-11 and a magnitude of 1.137e-13 for \( qddP \) (Figure 3-40). These errors are of an acceptable magnitude, thus satisfying Verification Task 12.3.

\[
\begin{align*}
qddA\text{ max}\text{ error} &= 7.094e-11 \\
qddP\text{ max}\text{ error} &= 1.137e-13
\end{align*}
\]

*Figure 3-40: Verification Task 12.3 Absolute Error*

During the optimization procedure, the Jacobians are determined. For our verification procedure, the reflex gains \( G_p \) and \( G_d \) are assigned magnitudes of zero. The Jacobians built based on the delayed components follow the same methodology used for the active properties, with the difference being the utilized muscle model. Thus, in Verification Task 12.4, the instantaneous Jacobians from Franklin’s `alg_optimize` script and the current model’s `optimize.m` script can be compared (Figure 3-41) and the absolute error determined (Figure 3-42). These error magnitudes are greater than the majority of those determined in the entirety of this verification procedure. As was discussed with Verification Tasks 12.1 and 12.3, the absolute error for the optimization constraints and the angular accelerations were not of a concerning magnitude.

\[
\begin{align*}
\text{[~, ~,~,Ja_max_error]=verify(Jai(1:90,1:36,1),Ja_F(1:90,1:36,1),1E-15)} \text{ %Verification Task 12.4} \\
\text{[~, ~,~,Jp_max_error]=verify(Jpi,Jp_F,1E-15)} \text{ %Verification Task 12.4}
\end{align*}
\]

*Figure 3-41: Verification Task 12.4*
Notice the term $0.00001 \times \frac{\pi}{180}$ in the denominator of Equation 3-2. By dividing by a number with such a small magnitude, it makes sense that the error would be greater for the Jacobians. Therefore, Verification Task 12.4 will be considered satisfied (Table 3-1).

### 3.13.3 Cost Function

Verification Task 12.6 consists of comparing the metabolic power output $fval$ for Franklin’s `costfun.m` script and the current model’s `costfunct.mlx` script (Figure 3-43). The results absolute error between the two outputs had a magnitude of zero (Figure 3-44), therefore Verification Task 12.6 was successful (Table 3-1).
3.14 Muscle Model

The resulting muscle force from the current model's `muscmodel.mlx` script and Franklin's `muscmodel.m` script were compared for Verification Task 13.1 (Figure 3-45). Verification Task 13.1 was satisfied (Figure 3-46).

```
[MF_check_count, ~,MF_diff,MF_max_error]=verify(fm,MF_F,1E-15) %Verification Task 13.1
```

*Figure 3-45: Verification task 13.1*

```
MF_max_error =
  0
```

*Figure 3-46: Absolute error for muscle force*

3.15 Reflex Model

For Verification Task 14.1 (Table 3-1), hand calculations were performed to compare to the output of the current model’s `musclemodelR` function. This task was performed to ensure that the `musclemodelR` function syntax was written correctly and was being interpreted as it was intended by MATLAB. Verification Task 14.1 was successful (Table 3-1).

3.16 Time Delay Analysis and Nonlinear Verification

To verify the time delay analysis, a nonlinear simulation was performed with a delay magnitude of 99% of the delay margin. For Verification Task 15.1, the nonlinear simulation should be successful if they delay margin was determined adequately. Figure 3-47 shows the $D_{ss}$ vs. Time graph for a hyperlordotic spine with a 200 N applied external load. As time increases, $D_{ss}$ decreases, successfully approaching equilibrium. Verification Task 15.1 has been successfully completed (Table 3-1).
The nonlinear system is simulated to verify the results of the linearized system. To determine if the system is approaching equilibrium, the distance from state space was calculated from the nonlinear simulation results.

For Verification Task 16.1 (Table 3-1), nonlinear simulations have been simulated with different magnitude for time delay. Notice in the following plots that for Figure 3-47 the time duration was 5 seconds instead of 2 seconds like it is for the other cases. Time was reduced in the other cases for faster execution.

The Dss vs. Time plot is shown in Figure 3-47 for a hyperlordotic spine with a 200 N load external load applied. Parameters are defined as $q=20$, $b=2$, $Gp=10$ and $Gd=0$. Delay margin has a magnitude of 0.2006. In Figure 3-47, the simulated time delay is 0.1986, 99% of the delay margin. Notice that Dss is approaching equilibrium as time increases in Figure 3-47.

![Figure 3-47: Dss for simulated delay = 0.99 * delay margin](image-url)
Figure 3-48 has the same loading conditions and model parameters. The simulated delay in this case is 0.1505, which is 75% of the delay margin. At Time=1.97 seconds, \( Dss = 8.89435 \). Visually, \( Dss \) appears to be approaching equilibrium. This was confirmed with the \( Dss \) vector.

![Figure 3-48: Dss for: simulated delay = 0.75 \* delay margin](image)

Figure 3-49 has the same loading conditions and model parameters. The simulated delay in this case is 0.1003, which is half of the delay margin. At Time=1.97 seconds, \( Dss = 9.17 \). \( Dss \) is approaching equilibrium as time increases.

![Figure 3-49: Delay Margin \* 0.5](image)
Figure 3-50 has the same loading conditions and model parameters. The simulated delay in this case is 0.2508, which is 125% of the delay margin. Notice in Figure 3-50 that as Time increases, $Dss$ starts to increase. This indicates that it is moving away from equilibrium.

![Figure 3-50: Dss for: simulated delay = 1.25 * delay margin](image)

Figure 3-51 has the same loading conditions and model parameters. The simulated delay in this case is 0.2006, which is the delay margin. At Time=1.97 seconds, $Dss = 11.0563$. Notice that $Dss$ is still approaching equilibrium even though the delay margin equals the simulated delay.

![Figure 3-51: Dss for: simulated delay = delay margin](image)
In Franklin’s thesis\textsuperscript{24}, the relationship between the state-space slopes and the percent of the delay margin are compared for a few simulations. It is evident that in some cases, a delay with a greater magnitude than the delay margin can be simulated and still pass the nonlinear verification procedure\textsuperscript{24}. It is evident from the simulations he provided that simulations with delays that had a magnitude of 99\% of the delay margin had negative slopes\textsuperscript{24}. If the simulated delay is greater or equal to the delay margin, then the simulation may not pass the nonlinear verification from some conditions.

Verification Task 16.1 is repeated with each completed nonlinear simulation but for our verification purposes, the model is yielding outputs that verify the linear predictions.

### 3.17 Summary

The purpose of this chapter was to verify the scripts and functions used in the current spine model to minimize potential errors. Many of these verification tasks included comparing the outputs of the current model to the model files received from the Musculoskeletal Biomechanics Laboratory at Virginia Polytechnic Institute (Michael Madigan, personal communication, July 26, 2008).

The Verification Tasks are presented in Table 3-1, including a column that indicates whether the task successfully verified the current model’s script or function. Of the 33 verification tasks listed in Table 3-1, only four of the tasks did not result in successful verification. These failed verification tasks allowed for the errors to be reviewed and if necessary, corrected in the current model. By performing verification tasks, potential errors can be avoided or amended, removing the opportunity for unintentional input or methodology errors to impact the model’s predictions.
Chapter 4: The impact of stiffness gain on stability

Modeling of the human trunk: the impact of stiffness gain on stability

Valerie Jardon

Department of Bioengineering, University of Kansas

3138 Learned Hall, Lawrence, KS 66045

valjardon@ku.edu

Dr. Sara E. Wilson¹

Department of Mechanical Engineering, University of Kansas

3138 Learned Hall, Lawrence, KS 66045

sewilson@ku.edu

ASME Membership (if applicable)

ABSTRACT

In this study, the impact of the stiffness gain in Bergmark’s model for short-range stiffness was investigated [6]. A stability-based trunk model was utilized to perform simulations with differing magnitudes of stiffness gain ranging from 0.9 to 40. It was determined that as stiffness gain increased in magnitude, the metabolic power and muscle force required to stabilize the system decreased. When compared to that of stiffness gains less than 10, the variance between the predicted required muscle force and metabolic power required varied less between stiffness gains with a magnitude of 10 or

¹ Corresponding author information: sewilson@ku.edu
higher. This study highlights the impact that stiffness gain can have on model predictions and spine stability.

INTRODUCTION

Prevalence of Low Back Pain. Low back pain (LBP) and injury are prevalent conditions in our society, with an estimated 80% of Americans affected by these conditions in their lifetime [1]. The financial burden of LBP and injury is vast in the United States, with a total annual expense of over 100 billion dollars due to medical intervention costs, lost earnings, and reduced productivity [2]. These substantial financial and physical burdens of LBP and injuries in our society reaffirms the need for continued investigation of the etiology of low back pain.

Spine Modeling and Mechanical Stability. Stability-based trunk models have been utilized by researchers to investigate the mechanisms of low back pain. In this context, spine stability refers to the spine’s ability to return to equilibrium following a perturbation [3-4]. If the spine cannot return to equilibrium, then injury may occur. While early spine models neglected to include stability requirements, it became evident that stability must be considered when implementing a muscle recruitment strategy and that muscle recruitment patterns were more accurately predicted by models with the inclusion of stability criteria [5].

Bergmark’s Model for Short-range Muscle Stiffness. For minor, quick deformations, muscles exhibit elastic behavior referred to as short-range muscle stiffness [6-7]:

\[ K = \frac{q}{l_o} F \]  \hspace{1cm} (1)

Where \( F \) is muscle force, \( l_o \) is the neutral length of the muscle, and \( q \) is the dimensionless stiffness gain.
While Bergmark utilized a stiffness gain $q$ with a magnitude of 40 in his short-range stiffness model [6], Crisco et al. reviewed the literature and determined that the stiffness gain could more reasonably fall within the range of 0.5 to 42 [8]. This plausible range may explain why the dimensionless stiffness gain $q$ utilized in studies varies (Table 1). While many researchers utilize a magnitude of 10 for $q$ based on the average stiffness gain calculated by Crisco et al.[8], the stiffness gain $q$ still varies vastly in the literature.

Due to the inconsistencies of the magnitude of the dimensionless stiffness gain $q$ in the literature, in this work, the impact of dimensionless stiffness gain $q$ on spine stability was evaluated. This relationship needs to be evaluated as muscle stiffness affects the stability analysis of the spinal system.

The hypothesis for this study is as follows: Increasing the magnitude of the stiffness gain will allow for the spine to stabilize with less required force.

TRUNK MODEL DEVELOPMENT

In this section, the 18 degree of freedom trunk model utilized in this investigation will be discussed. This trunk model and the methodology used are based on the model developed by Franklin et al. [9-11].

Vertebrae. The lumbar vertebrae are represented as independent rigid bodies stacked on a fixed pelvis. An additional rigid body is included in the model, representing the grouped cervical and thoracic regions of the spine. All rigid bodies are represented as elliptical cylinders with their body height based on the skeletal coordinates in Cholewicki et al. [12], although this anatomy has been transformed to allow for differing lumbar lordosis angles to be implemented into the model. In this work, the magnitude of the lordosis angle is 58.1°, the mean lordosis angle measured by Pesenti et al.[13] Using measurements from Panjabi et al. [14] and Berry et al. [15], the uniform radii assigned the
rigid bodies were defined with magnitudes of 0.024 m and 0.017 m in the medio-lateral and anterior-posterior directions respectively. Lastly, the masses of the vertebrae are based on the cadaver dissection performed by Lui et al. [16] The center of mass of each rigid body is positioned in the center.

Muscles. The skeletal coordinates from Cholewicki et al. [12] also describe the muscle paths for the 90 muscle fascicles included in the model. The muscles are modelled using the Hill muscle model [17], consisting of a contractile element, spring and damper (Equation 2):

\[ F_m = F_{CE} + Kx + B\dot{x} \]  

Using the product of maximum muscle stress (46 N/cm²), muscle activation and physiological cross-sectional area of the muscles, the force of the contractile element \( F_{CE} \) can be calculated (Equation 3) [18]. The muscle activation of each muscle falls between zero and one, with a magnitude of one representing a fully activated muscle.

\[ F_{CE} = \left( \frac{46 \text{N}}{\text{cm}^2} \right) \times PCSA \times \alpha \]

Bergmark’s model of short-range stiffness is used to define the stiffness coefficient (Equation 1). Recall that for this study, the magnitude of stiffness gain \( q \) varies for each simulation completed. Like the stiffness coefficient, the damping coefficient calculation included a dimensionless damping gain \( b \) (Equation 4):

\[ B = b \times \frac{F_{CE}}{x_o} \]

The stiffness coefficient (Equation 1) and damping coefficient (Equation 4) can be substituted into Equation 2, yielding this equation for muscle force:
\[ F_m = F_{CE} + \left( \frac{q \cdot f_{\text{max}} \cdot \alpha}{x_0} x(t) - x_0 \right) + \left( \frac{b \cdot f_{\text{max}} \cdot \alpha}{x_0} \dot{x}(t) \right) \] (5)

**Reflexes.** A PD controller with proportional gain \( G_P \) and differential gain \( G_D \) is utilized to represent the muscle spindles’ ability to respond to a change in velocity and length of the muscle. Muscle activation is calculated by summing the steady-state activation \( \alpha_o \) and reflex activation \( \alpha_r \) [9-11].

\[ \alpha(t) = \alpha_o + \alpha_r = \alpha_o + \alpha_o \left( \frac{G_P \cdot x(t - \tau) - x_0}{x_o} + G_D \frac{\dot{x}(t - \tau)}{x_o} \right) \] (6)

**Intervertebral Discs.** Franklin et al. developed and implemented a lumped parameter intervertebral disc (IVD) model in their trunk model [9-11] with a rotational stiffness of 50 Nm/rad based on Stokes et al. [19] and rotational damping of 0.5 Nms/rad based on Izambert et al. [20].

The angular displacement and velocity of each rigid body needs to be determined and implemented into Equation 7 to calculate the intervertebral disc moment \( M \):

\[ M = K_{IVD} \cdot \theta + B_{IVD} \cdot \dot{\theta} \] (7)

The methodology for determining the intervertebral disc moments is further explained in Franklin’s thesis [10].

**Dynamics.** The Lagrangian function can be utilized to determine the equations of motion (Equation 8):

\[ L = \text{Kinetic Energy} - \text{Potential Energy} \] (8)

The total potential energy of the system \( PE \) is based on the sum of the gravitational energy of each body (Equation 9):
\[ PE = \sum_{b=1}^{6} mgh \]  

where \( m \) is the mass of the body, \( h \) is the vertical position of the center of mass of the body, and \( g \) is the gravity constant.

The total kinetic energy of the system can be determined by summing the translational and rotation kinetic energy of each body (Equation 10):

\[ KE = \sum_{b=1}^{6} \left( \frac{1}{2}mv^2 + \frac{1}{2}I\omega^2 \right) \]

where \( v \) is the velocity of the center of mass, \( I \) is the mass moment of inertia, and \( \omega \) is the angular velocity of the body.

The generalized forces can be calculated using Equation 11:

\[ Q_i = \sum_{b=1}^{6} \left( \left( \sum_{e=1}^{n} F_{be} \right) \cdot \frac{\partial v_b}{\partial q_i} + \left( \sum_{e=1}^{n} r_{be} \times F_{be} \right) \cdot \frac{\partial \omega_b}{\partial q_i} \right) \]

Where \( b \) is the index of the rigid bodies, \( F_{be} \) represents force applied to the body, \( e \) is the index of the applied forces, \( i \) is the degree of freedom index, \( v_b \) is the linear velocity of the origin of the body, \( \omega_b \) is the angular velocity of the body and \( r_{be} \) is a vector between the point where the force is applied and the origin of the body.

Solving for the equations of motion by performing the Lagrangian Derivative (Equation 12):

\[ EOM = \frac{d}{dt} \left( \frac{\partial L}{\partial q_i} \right) - \frac{\partial L}{\partial \dot{q}_i} = Q_i \]
\[ EOM = \frac{d}{dt} \left( \frac{\partial KE}{\partial \dot{q}_i} \right) - \frac{d}{dt} \left( \frac{\partial PE}{\partial \dot{q}_i} \right) - \frac{\partial KE}{\partial q_i} + \frac{\partial PE}{\partial q_i} = Q_i \] (13)

Using forward dynamic simulation techniques, the dynamic equations can be used to determine the accelerations \( \ddot{q} \) (Eq. 14):

\[ M_m(q) \cdot \ddot{q} + G_m(q) + C_m(q, \dot{q}) = Q_i \] (14)

Where \( M_m \) is the mass matrix, \( G_m \) is the gravity vector and \( C_m \) is the Coriolis vector.

**Optimization Procedure.** The linearized representation of the system was determined using the Taylor Series Expansion (Equation 15):

\[ \delta \dot{\theta} = \frac{\partial f(x)}{\partial x} \bigg|_{x=x_0} \delta \theta \] (15)

Where \( \frac{\partial f(x)}{\partial x} \bigg|_{x=x_0} \) is the Jacobian matrix that can be approximated through finite differencing.

A constrained optimization is performed to minimize the metabolic power required while adhering to stability and equilibrium constraints. The stability constraint is evaluated using the Jacobian. If the eigenvalues of the Jacobian have negative real parts, then the system is stable.

For this optimization procedure, the instantaneous and delayed components act simultaneously (\( \tau = 0 \)), so a total Jacobian can be determined (Equation 16):

\[ J = J_i + J_d \] (16)

This is crucial for the time delay analysis where the delay margin of the system can be determined based on the time delay that results in the system becoming unstable.

Equation 17 represents the metabolic power equation with an assumed equal magnitude of slow-twitch and fast-twitch muscle fibers from Anderson [21]:
\[ P = m \times \left( 74 \times 0.5 \times \sin \left( \frac{\alpha \times \pi}{2} \right) + 111 \times 0.5 \times \left( 1 - \cos \frac{\alpha \times \pi}{2} \right) \right) \]  \hspace{1cm} (17)

After the optimization procedure is completed, the resulting muscle activation determined from the optimal solution can be utilized to separate the Jacobian into instantaneous and delayed components [9-11] (Eq. 18):

\[ \dot{x}(t) = J_i \cdot x(t) + J_d \cdot x(t - \tau) \]  \hspace{1cm} (18)

The Jacobian with the delayed components incorporates the time-delayed reflexes.

**Time Delay Analysis.** The minimum time delay that causes the system to become unstable, also known as the delay margin, can be calculated using a method developed by Chen [22]. The eigenvalues of the Jacobian are utilized for stability analysis. The time delay in which the system becomes unstable can be indicated by the crossing of the imaginary axis by the eigenvalue, violating the stability criteria that all real parts are negative.

A frequency sweeping technique in the Laplace domain can be utilized to determine this crossing event (Eq. 19-20) [9-11,22]. This crossing event is indicated by \( \lambda = 1 \).

\[ j\omega \cdot X = J_i \cdot X + J_d \cdot X \cdot e^{-j\omega \tau} \]  \hspace{1cm} (19)

\[ (j\omega \cdot I - J_i) \cdot X = \lambda \cdot J_d \cdot X, \quad \text{where} \quad \lambda = e^{-j\omega \tau} = e^{-j\theta} \]  \hspace{1cm} (20)

Lastly, the delay margin can be calculated as shown in Equation 21:

\[ \tau = \frac{\theta}{\omega} \]  \hspace{1cm} (21)

If the resulting delay margin has a magnitude of less than 0.060 seconds, then the system is classified as unstable because a delay less than 0.06 is not physiologically realistic for the reflexes [23].
Nonlinear Verification. Lastly, a nonlinear simulation is completed to verify the results predicted by the linear system. Using the dde23 solver in MATLAB (Mathworks, Natick, MA), a delayed differential equation can be solved. The delay margin should be scaled by 0.99 to determine the time delay prior to the system becoming unstable. This is the delay margin that should be input into the dde23 solver. In the simulation, a disturbance of 0.01° is applied to the cervical and thoracic region rigid body and each simulation has a minimum duration of five seconds [9-11].

Using Equation 22, the distance of the normalized state-space from equilibrium can be calculated to determine if the system is approaching equilibrium:

\[
D = \sqrt{\sum_{i=1}^{18} \left( \frac{\dot{\theta}(t) - \bar{\dot{\theta}}}{\text{mean}(\dot{\theta}(t) - \bar{\dot{\theta}})} \right)^2 + \left( \frac{\theta(t) - \bar{\theta}}{\text{mean}(\theta(t) - \bar{\theta})} \right)^2}
\]  

(22)

If the system approached equilibrium, then the linear predictions are verified.

Muscle Force Calculations. The muscle force required to stabilize the system was determined using the maximum muscle force [18] and the muscle activations predicted by the model in the optimization procedure. In Equation 23, CSA is the cross-sectional area of the muscle and \(\alpha\) is the muscle activations.

\[
F = \left( \frac{46 \text{ N}}{cm^2} \right) \times \text{CSA} \times \alpha
\]  

(23)

Simulation Parameters

For all simulations, based on the methodology utilized by Franklin et al. [9-11], an external load of 200 N was applied at the T4 level, 20 cm anterior to the trunk. The assigned magnitudes for the proportional gain \(G_p\), differential gain \(G_d\) and damping gain \(b\) were 10, 0 and 2 respectively. The stiffness gain magnitudes utilized in the simulations ranged from 0.9 to 40, with \(q\) having the following
magnitudes: 0.9, 1, 2, 4, 5, 10, 15, 20, 25, 30, 35, and 40. Using a step of 0.1, it was determined that 0.9 was the critical stiffness $q_{critical}$ for these simulation parameters.

**RESULTS**

**Metabolic Power.** In Figure 2, it is evident that as the magnitude of stiffness gain increases, the metabolic power required to stabilize the system decreases. Of the stiffness gains utilized in the simulations, $q = 0.9$ resulted in the greatest magnitude of metabolic power required with a magnitude of 99.18 W. Stiffness gain with a magnitude of 40 resulted in the least metabolic power required with only 35.07 W required. Above a magnitude of 20, increasing stiffness gains had little impact on the required metabolic power.

**Muscle Force Required.** As shown in Figure 3, as stiffness gain increased, the required muscle force necessary to stabilize the spine decreased. For $q = 0.9$ and $q = 40$, approximately 3615 N and 1583 N of muscle force are required to stabilize the spine respectively (Figure 3). For the smaller magnitudes of stiffness gain, the magnitude of required muscle force varies more than it does for the greater magnitudes of stiffness, such as the range from $q = 20$ to $q = 40$. There is a change in required muscle force of approximately 330 N from $q = 5$ to $q = 10$. Contrastingly, from $q = 35$ to $q = 40$, the change in required muscle force is approximately 8 N.

**DISCUSSION**

Short-range muscle stiffness describes the stiffness of a muscle for a small deformation where the muscle exhibits a more spring-like behavior [6,7,32]. When the stiffness gain magnitudes ranging from 0.9 and 40 were simulated with the 200 N applied external load, it was determined that the smaller magnitudes of stiffness gain required increased magnitudes of metabolic power and muscle force to achieve stability. As the magnitude of stiffness gain increased, the metabolic power and muscle
force required decreased (Figures 2-3). These findings supported our hypothesis. Both the metabolic power and required muscle force calculations include the predicted muscle activations resulting from the constrained optimization procedure and the physiological cross-sectional area of the muscles. Therefore, the similar trends revealed for the required metabolic power versus stiffness gain (Figure 2) and the required muscle force versus stiffness gain (Figure 3) could be expected.

In Figure 2 and Figure 3, it is evident that the required metabolic power and muscle force varies more for the simulations with stiffness gains spanning from $q = 0.9$ to $q = 10$ as opposed to stiffness gains with magnitudes greater than 10. For the magnitudes of $q$ exceeding 10, the trend shown in Figure 2 that the required metabolic power decreases with increased stiffness gain is still prevalent but the magnitude in which the required metabolic power varies is not as vast. These same trends can be viewed in Figure 3, with the larger magnitudes of $q$ resulting in less variance in magnitude of the required muscle force as opposed to the smaller magnitudes of $q$ which varied more.

From these results, it is evident that the magnitude utilized for the dimensionless stiffness gain in Bergmark’s model of short-range stiffness impacts the model’s predictions for the metabolic power and muscle force required to stabilize the system. These findings agree with Bergmark’s acknowledgement that the magnitude of the stiffness gain will impact the magnitudes of the model’s predictions [6], although Bergmark did not explore to what degree the stiffness gain magnitude would affect model predictions. It is evident from these findings that stiffness gain will influence model predictions and the spinal system’s ability to stabilize. For instance, the magnitude of the critical stiffness was determined to be 0.9. This indicates that for a stiffness gain of 0.5, a magnitude in the range provided by Crisco et al. [8], the model would have predicted an unstable system. Investigation needs to be continued regarding the magnitude of stiffness gain that should be utilized when modeling short-range stiffness. Brown and McGill determined an inverse relationship between muscle activation
and stiffness gain [24]. In the future, incorporating a stiffness gain that varies with muscle activation may yield more accurate model predictions.

Additionally, it should be mentioned that the tolerances utilized in the constrained optimization procedure may impact the model’s predicted activations slightly. This may explain the slight increase in required muscle force for $q = 20$ as opposed to that required for $q = 15$ and $q = 25$.

CONCLUSIONS

In this study, it was determined that the magnitude of the stiffness gain may impact a model’s prediction for spinal stability. The magnitude of the metabolic power and muscle force required to stabilize the system decreased as the stiffness gain increased. In our simulations, a stiffness gain with a magnitude of less than 0.9 would have resulted in instability whereas magnitudes equal to or greater than 0.9 would have been predicted to stabilize. This highlights the important of stiffness gain and its influence in stability analysis. Future studies should consider implementing stiffness gains that vary with muscle activation.

ACKNOWLEDGMENT

FUNDING

This publication was made possible by grant number R01AR046111 from the National Institute of Arthritis and Musculoskeletal and Skin Diseases (NIAMS) at the National Institutes of Health. Its contents are solely the responsibility of the authors and do not necessarily represent the official views of NAIMS.
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>KE</td>
<td>Kinetic Energy</td>
</tr>
<tr>
<td>PE</td>
<td>Potential Energy</td>
</tr>
<tr>
<td>IVD</td>
<td>Intervertebral Disc</td>
</tr>
<tr>
<td>q</td>
<td>Stiffness Gain</td>
</tr>
<tr>
<td>b</td>
<td>Damping Gain</td>
</tr>
<tr>
<td>Gp</td>
<td>Proportional Gain</td>
</tr>
<tr>
<td>Gd</td>
<td>Differential Gain</td>
</tr>
<tr>
<td>K</td>
<td>Stiffness coefficient</td>
</tr>
<tr>
<td>xo</td>
<td>Equilibrium length</td>
</tr>
<tr>
<td>α</td>
<td>Muscle Activation</td>
</tr>
<tr>
<td>Ji</td>
<td>Jacobian of the instantaneous components</td>
</tr>
<tr>
<td>Jd</td>
<td>Jacobian of the delayed components</td>
</tr>
<tr>
<td>L</td>
<td>Lagrangian Derivative</td>
</tr>
<tr>
<td>τ</td>
<td>Time Delay</td>
</tr>
</tbody>
</table>
REFERENCES


Figure Captions List

Fig. 1  Spine model. This model includes five rigid lumbar vertebrae, one rigid body representing the cervical and thoracic spine regions, and a fixed pelvis. 90 muscle fascicles are included. The intervertebral discs are represented by a lumped parameter model separating the rigid bodies.

Fig. 2  The relationship between required metabolic power and stiffness gain are presented for a simulation including an external load of 200 N. This load was applied at the T4 level and 20 cm anterior to the trunk. For these simulations, $G_P = 10$, $G_D = 0$ and $b = 2$.

Fig. 3  The relationship between the required muscle force to stabilize the system and stiffness gain is presented. For these simulations, an external load of 200 N was applied at the T4 level and 20 cm anterior to the trunk and $G_P = 10$, $G_D = 0$ and $b = 2$. The required muscle force was calculated utilizing the resulting muscle activations from the optimization procedure in which the metabolic power was minimized while adhering to stability and equilibrium constraints.
Table Caption List

Table 1 Evidence of the varying stiffness gain $q$ magnitudes utilized in previous investigations
Figure 4-1: Spine model. This model includes five rigid lumbar vertebrae, one rigid body representing the cervical and thoracic spine regions, and a fixed pelvis. 90 muscle fascicles are included. The intervertebral discs are represented by a lumped parameter model.
Figure 4-2: The relationship between required metabolic power and stiffness gain are presented for a simulation including an external load of 200 N. This load was applied at the T4 level and 20 cm anterior to the trunk. For these simulations, $GP = 10$, $GD = 0$ and $b$
Figure 4-3: The relationship between the required muscle force to stabilize the system and stiffness gain is presented. For these simulations, an external load of 200 N was applied at the T4 level and 20 cm anterior to the trunk and GP = 10, GD = 0 and b = 2. The
<table>
<thead>
<tr>
<th>Source</th>
<th>Stiffness Gain $q$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Akhavanfar et al. [25]</td>
<td>5</td>
</tr>
<tr>
<td>Bergmark [6]</td>
<td>40</td>
</tr>
<tr>
<td>Brown et al. [26]</td>
<td>10</td>
</tr>
<tr>
<td>Cholewicki et al. [12]</td>
<td>30</td>
</tr>
<tr>
<td>Franklin et al. [11]</td>
<td>10</td>
</tr>
<tr>
<td>Hajihosseinali et al. [27]</td>
<td>5</td>
</tr>
<tr>
<td>Samadi &amp; Arjmand [28]</td>
<td>1.8, 5, 10</td>
</tr>
<tr>
<td>Shamsi et al. [29]</td>
<td>5</td>
</tr>
<tr>
<td>Stokes &amp; Gardner-Morse [30]</td>
<td>5</td>
</tr>
<tr>
<td>Vakilzadeh et al. [31]</td>
<td>2, 4, 6, 10</td>
</tr>
<tr>
<td>Zeinali-Davarani et al. [32]</td>
<td>2, 4, 10</td>
</tr>
</tbody>
</table>

*Table 4-1: Evidence of the varying stiffness gain $q$ magnitudes utilized in previous investigations*
Chapter 5: The effects of hyperlordosis and hydolordosis on spine stability
Modeling of the human trunk: the impact of lumbar lordosis on stability

Valerie Jardon
Department of Bioengineering, University of Kansas
3138 Learned Hall, Lawrence, KS 66045
valjardon@ku.edu

Dr. Sara E. Wilson
Department of Mechanical Engineering, University of Kansas
3138 Learned Hall, Lawrence, KS 66045
sewilson@ku.edu

ABSTRACT
In this study, the potential impact of lumbar lordosis angle on spine stability was investigated using a stability-based trunk model. Simulations were completed for hyperlordosis (84.1°), hypolordosis (32.1°) and normal lordosis (58.1°) for unloaded and loaded erect posture. When compared to the normal lordosis case, it was determined that the hyperlordosis case would require more metabolic power to stabilize and increased recruitment of muscle flexors. Contrastingly, the hypolordosis case required less metabolic power to stabilize with an increased recruitment of muscle extensors.

\[ \text{Corresponding author information: sewilson@ku.edu} \]
Prevalence of Low Back Pain. Low back pain (LBP) and injury cost the United States more than $100 billion annually [1] and affects approximately 80% of Americans in their lifetime [2]. In a 12-month long study of nearly 2,000 American manual material handling workers, 25% of the sample reported experiencing low back pain that lasted at least 7 days and 10% reported the inability to work due to their ailments [3]. This prevalence in our society proves that furthering the investigation of the causes of low back pain and injury is crucial to the expansion and improvement of current rehabilitation techniques and prevention strategies.

Spine Modeling and Mechanical Stability. Researchers have developed and utilized trunk models to investigate the behaviors of the spine and probable causes of low back pain and injury. To prevent potential injury of the spine, it is crucial that the spine is able to return to equilibrium after being disturbed [4-5]. This is the concept of mechanical stability.

The trunk muscles are vital to the stability of the spine, and in many loading conditions, prevent the spine from buckling [6]. Additionally, Franklin et al. determined that time-delayed neuromuscular reflexes are beneficial to the stabilization process and in certain cases necessary for stability, disproving researchers’ initial beliefs that solely intrinsic stiffness may be adequate for stability [7].

In the development of trunk models, it is critical that stability analysis be incorporated to ensure that these models are adequately representing realistic spine behavior. Researchers have previously determined that equilibrium and stability criteria should be included in spine models that utilize optimization to determine muscle activations, so that the resulting muscle activation better represents co-contraction [8,9].

Lumbar Lordosis. Lumbar lordosis is the natural curve of the lumbar spine region. The angles of the vertebrae in this natural curve have been known to vary between humans, with a study by Pesenti et
al. revealing that in a population of approximately 120 healthy adults, the total lordosis angle ranged from 10-89° with a mean value of 58.1° [10].

The possible correlation between lumbar lordotic curvature and LBP is debated amongst researchers, requiring continued investigation and the comparison of study outcomes, parameters and methodologies [11]. Chun et al. completed a meta-analysis and review, with nine of the thirteen studies reporting that subjects with LBP had significantly smaller lumbar lordosis angles when compared to the controls [11]. The range of lordosis angles measured by Pesenti et al. [10] and the statistically significant relationship between lumbar lordosis angle and LBP supported by most researchers [11] inspired the current investigation of the possible influence lumbar lordosis angle may have on spinal behavior and stability.

To determine the magnitudes of lordosis angle to be simulated in our study, the definition of hyperlordosis and hypolordosis from Fernand et al. was used. Fernand et al. [12] defined a hyperlordotic angle as an angle that is greater than the mean lordotic angle plus the magnitude of two standard deviations from the mean. Contrastingly, a hypolordotic angle has been defined as an angle less than the mean lordotic angle minus the magnitude of two standard deviations from the mean. In this investigation, based on the mean total lordosis angle from Pesenti et al. [10], hyperlordosis and hypolordosis were defined as a lordotic angle greater than 84.1° and less than 32.1° respectively. In this study, lumbar lordosis angles of 10°, 32.1°, 58.1° 84.1° and 89° will be simulated. Our hypotheses for this study are as follows:

**Hypothesis 1:** With a load of 0 N applied vertically to T4 at a position 20 cm anterior to the trunk, the hyperlordotic spine will require increased recruitment of flexor muscles to stabilize, consequently increasing the required metabolic power.
Hypothesis 2: With a load of 0 N applied vertically to T4 at a position 20 cm anterior to the trunk, the hypolordotic spine will require increased recruitment of extensor muscles to stabilize, consequently increasing the required metabolic power.

Hypothesis 3: With a load of 200 N applied vertically to T4 at a position 20 cm anterior to the trunk, a greater critical stiffness will be required to stabilize the hypolordotic spine.

TRUNK MODEL DEVELOPMENT

Vertebrae. The spine model and methodology utilized in this investigation is based on the model developed by Franklin et al. [7,13-14]. This 18-DOF model incorporates six rigid bodies with each lumbar vertebra represented by an individual rigid body and one collective rigid body representing the cervical and thoracic regions (Figure 1). Each rigid body is an elliptic cylinder with its height based on the skeletal coordinates defined by Cholewicki et al. [15]. These coordinates have been rotated so that the vertebral connection points are aligned along the vertical axis. This allows for the bodies to be rotated by the segmental lordosis angle necessary for each lordosis case investigated in this study. The radii of the vertebrae are defined as 0.024 m and 0.017 m in the medio-lateral and anterior-posterior directions respectively, based on measurements from Berry et al. [16] and Panjabi et al. [17]. The masses of the vertebrae are based on measurements obtained from Liu et al. of a cadaver spine [18] and the center of mass of each vertebra is positioned in the center of the rigid body, equidistant from the vertebral faces.

Muscles. The muscle paths are described by the skeletal coordinates from Cholewicki et al. [15]. The muscle model utilized in the current model is based on the Hill muscle model [19] and includes a spring, damper, and contractile element in parallel (Eq. 1).

\[ F_m = F_{CE} + Kx + B\dot{x} \] (1)
The product of maximum muscle stress (46 N/cm²), muscle activation $\alpha$ and the physiological cross-sectional area $PCSA$ of the muscle [20] represents the force of the contractile element $F_{CE}$ for that muscle (Eq. 2):

$$F_{CE} = \left(46 \, \text{N/cm}^2\right) \times PCSA \times \alpha$$  \hspace{1cm} (2)

Muscle activation can range in magnitude from zero to one, indicating an inactivated muscle and fully activated muscle respectively.

The stiffness coefficient $K$ of the spring is based on Bergmark’s model of short-range stiffness [21,22] where $q$ is the dimensionless stiffness gain, $F_{CE}$ is the force of the contractile element and $x_o$ is the equilibrium muscle length (Eq. 3).

$$K = q \times \frac{F_{CE}}{x_o}$$  \hspace{1cm} (3)

Crisco and Panjabi [23] provided a range for the dimensionless stiffness gain $q$ reported in literature from 0.5-42 with a mean value of 10. In the current study, a value of 20 was utilized for the dimensionless stiffness gain.

The damping coefficient $B$ was determined through use of the dimensionless damping gain $b$ (Eq. 4). The magnitude of $b$ in this study was 2.

$$B = b \times \frac{F_{CE}}{x_o}$$  \hspace{1cm} (4)

Through the substitution of Equations 2-4 into Equation 1, the resulting muscle force equation can be obtained (Eq. 5):

$$F_m = f_{\text{max}} \times \alpha + \left(q \times \frac{f_{\text{max}} \times \alpha}{x_o} x(t) - x_o\right) + \left(b \times \frac{f_{\text{max}} \times \alpha}{x_o} \dot{x}(t)\right)$$  \hspace{1cm} (5)
Reflexes. The reflex response to the change in length and velocity of the muscle by the muscle spindles is represented by a PD controller with proportional gain $G_P$ and differential gain $G_D$ (Eq. 6).

Franklin et al. [7,13-14] defined the muscle activation to be the sum of the steady-state $\alpha_o$ and reflex activations $\alpha_r$ (Eq. 6).

$$\alpha(t) = \alpha_o + \alpha_r = \alpha_o + \alpha_o \ast \left( G_P \ast \frac{x(t - \tau) - x_o}{x_o} + G_D \ast \frac{\dot{x}(t - \tau)}{x_o} \right)$$  \hspace{1cm} (6)

Intervertebral Discs. The current model utilizes the lumped parameter model developed by Franklin et al. for the intervertebral discs that incorporated a rotational spring and damper that oppose angular motion [7,13-14]. To determine the intervertebral disc moments, the angular displacement and velocity must be determined for each rigid body. These values should be scaled by constants $K_{IVD}$ and $B_{IVD}$, as shown in Equation 7.

$$M = K_{IVD} \ast \theta + B_{IVD} \ast \dot{\theta}$$  \hspace{1cm} (7)

The stiffness coefficient $K_{IVD}$ was obtained by cadaver measurements performed by Stokes et al. [24] and the damping coefficient was based on the relationship between stiffness and damping determined by Izambert et al. [25]. In the current study, $K_{IVD}$ and $B_{IVD}$ have magnitudes of 50 Nm/rad and 0.5 Nms/rad respectively.

Dynamics. The equations of motion were derived using the Lagrangian function $L$, which is defined as the difference between kinetic energy $KE$ and potential energy $PE$ (Eq. 8):

$$L = KE - PE$$  \hspace{1cm} (8)

The potential energy of the system is based on the sum of the gravitational energy of each body where $h$ is the position of the body’s center of mass along the vertical axis, $g$ is gravity and $m$ is the mass of the body (Eq. 9):
The rotational and translational kinetic energy for the system can be described as shown in Equation 10, where \( v \) is the velocity of the center of mass, \( I \) is the mass moment of inertia, and \( \omega \) is the angular velocity of the body:

\[
KE = \sum_{b=1}^{6} \left( \frac{1}{2} m v_b^2 + \frac{1}{2} I \omega_b^2 \right)
\]  

The muscle forces, external forces and the IVD moments are utilized in the generalized force calculations (Eq. 11).

\[
Q_i = \sum_{b=1}^{6} \left( \left( \sum_{e=1}^{n} F_{be} \right) \cdot \frac{\partial v_b}{\partial q_i} + \left( \sum_{e=1}^{n} r_{be} \times F_{be} \right) \cdot \frac{\partial \omega_b}{\partial q_i} \right)
\]

where \( i \) is the degree of freedom index for the degree of freedom \( q_i \), \( b \) is the rigid body index, \( e \) is the applied forces index, \( F_{be} \) represents force applied to the body, \( v_b \) is the linear velocity of the body’s origin, \( r_{be} \) is a vector between the point where the force is applied and the body’s origin, and \( \omega_b \) is the body’s angular velocity.

The Lagrangian Derivative yields the equations of motion (Eq. 12-13):

\[
EOM = \frac{d}{dt} \left( \frac{\partial L}{\partial q_i^\dot{}} \right) - \frac{\partial L}{\partial q_i} = Q_i
\]

\[
EOM = \frac{d}{dt} \left( \frac{\partial KE}{\partial q_i^\dot{}} \right) - \frac{d}{dt} \left( \frac{\partial PE}{\partial q_i} \right) - \frac{\partial KE}{\partial q_i} + \frac{\partial PE}{\partial q_i} = Q_i
\]
Using forward dynamic simulation methods, the dynamic equations can be used to determine the accelerations $\ddot{q}$ (Eq. 14):

$$M_m(q) \cdot \ddot{q} + G_m(q) + C_m(q, \dot{q}) = Q_i$$  \hspace{1cm} (14)

Where $M_m$ is the mass matrix, $C_m$ is the Coriolis vector, and $G_m$ is the gravity vector.

**Optimization Procedure.** The system was linearized about equilibrium through use of the Taylor Series expansion. Neglecting higher order terms and applying $f(\bar{x}) = 0$ for equilibrium, Equation 15 represents the linearized system:

$$\delta \dot{\theta} = \frac{\partial f(x)}{\partial x} \bigg|_{x=\bar{x}} \delta \theta$$  \hspace{1cm} (15)

Where $\frac{\partial f(x)}{\partial x} \bigg|_{x=\bar{x}}$ is the Jacobian matrix that can be approximated through finite differencing.

By nature, reflexes are time-delayed so for our purposes, the Jacobian was separated into instantaneous and delayed components \[7,13-14\] (Eq. 16). This separation of the Jacobian matrix will be necessary for the linear time delay calculations.

$$\dot{x}(t) = J_i \cdot x(t) + J_d \cdot x(t - \tau)$$  \hspace{1cm} (16)

Next, the model utilizes a constrained optimization procedure that minimizes metabolic power while requiring the system to be stable and in equilibrium. The Jacobian matrix is used for stability analysis. Anderson \[26\] defines metabolic power using the following equation, with an even magnitude of fast and slow twitch fibers assumed (Eq. 17):

$$P = m \cdot \left(74 \cdot 0.5 \cdot \sin\left(\frac{\alpha \cdot \pi}{2}\right) + 111 \cdot 0.5 \cdot \left(1 - \cos\frac{\alpha \cdot \pi}{2}\right)\right)$$  \hspace{1cm} (17)
For the constrained optimization procedure, it is assumed that \( \tau = 0 \), meaning that the time-delayed components will act at the same time as the instantaneous components. Assuming \( \tau = 0 \) in this optimization procedure will allow for the delay margin to be determined.

**Time Delay Analysis.** Using a method developed by Chen [27], the time delay in which the system becomes unstable, referred to as the delay margin, can be determined [7,13-14]. To accomplish this task, the eigenvalues of the Jacobian can be used. By definition, a system is stable when the eigenvalues of its Jacobians have negative real parts. The delay margin can be determined by noting when the first real part of one of the eigenvalues crossed the imaginary axis, indicating instability.

Using the frequency sweeping technique in the Laplace domain (Eq. 18) established by Chen [27], the real part of the eigenvalue crossing the imaginary axis is indicated by \( \lambda = 1 \) (Eq. 19) [7,13-14,27]:

\[
\begin{align*}
  j \omega \cdot X &= J_1 \cdot X + J_D \cdot X \cdot e^{-j \omega \tau} \\
  (j \omega \cdot I - J_1) \cdot X &= \lambda \cdot J_D \cdot X, \quad \text{where } \lambda = e^{-j \omega \tau} = e^{-j \theta}
\end{align*}
\]

Using Equation 20, the delay margin can be calculated:

\[
\tau = \frac{\theta}{\omega}
\]

While linear stability is determined based on the eigenvalues, an additional stability criterion should be considered following the time delay analysis: if the delay margin is less than 0.060 seconds, the delay exhibited physiologically for the reflexes [28], then the system is also considered unstable.

**Nonlinear Verification.** Once the time delay analysis has been performed and the delay margin is determined, the nonlinear simulation can be performed to verify the linear results. For the nonlinear verification, the MATLAB dde23 solver (Mathworks, Natick, MA) is used to solve the delayed differential
equation. The time delay input for the dde23 solver should have the magnitude of 99% of the delay margin, indicating a time delay directly preceding instability. In this simulation, the thoracic/cervical regions’ rigid body is disturbed by 0.01° to represent a lifting motion and each simulation was run for a minimum for 5 seconds. The function output can be used to determine if the system is approaching equilibrium by normalizing the state space and calculating its distance from equilibrium (Eq. 21):

\[
D = \sqrt{\sum_{i=1}^{18} \left( \frac{\dot{\theta}(t) - \bar{\dot{\theta}}}{\text{mean}(\dot{\theta}(t) - \bar{\dot{\theta}})} \right)^2 + \left( \frac{\theta(t) - \bar{\theta}}{\text{mean}(\theta(t) - \bar{\theta})} \right)^2}
\]

(21)

SIMULATION PARAMETERS
A standing unloaded simulation was completed for each lordosis angle (10°, 32.1°, 58.1°, 84.1°, and 89°). For the unloaded simulations, the minimal metabolic power required to stabilize the system while adhering to the stability and equilibrium constraints was determined. As mentioned previously, \( q \) was assigned a value of 20 and \( b \) was assigned a value of 2 for the muscle model for the unloaded simulations. For the reflexes, \( G_p \) was assigned a magnitude of 10, and \( G_n \) a magnitude of zero.

Additionally, loaded simulations were completed, with a 200 N external load applied 20 cm anterior to the trunk at T4 [7,13-14]. For these simulations, the critical stiffness gain \( q_{\text{critical}} \) was determined for the 32.1°, 58.1° and 84.1° lordosis cases.

RESULTS
Metabolic Power. For the unloaded simulations, the resulting required metabolic power was the greatest for a lordosis angle of 84.1 degrees and the least for an angle of 32.1 degrees, with magnitudes of 58.2 W and 1.8 W respectively (Figure 3). The lordosis angle of 10 degrees resulted in a
metabolic power of 4.59 W while the 89 degrees lordosis angle resulted in a metabolic power of 49.57 W (Figure 3).

For the loaded simulations, the hypolordosis case resulted in the least metabolic power required with magnitudes of 16.7 W for the 10-degree angle and 16.48 W for the 32.1-degree angle (Figure 3). The hyperlordosis cases required a greater magnitude of required metabolic power in comparison to the normal lordosis case and the hypolordosis cases with a magnitude of 60.83 W and 46.65 W metabolic power required for the 84.1-degree angle and the 89-degree angle respectively (Figure 3).

**Muscle Activation.** For each muscle fascicle, the difference between the resulting magnitude of activation on the left-side and right-side of the body for the unloaded cases was determined. For all simulated lordosis angles, the greatest difference resulted in a magnitude of 0.0253.

The unloaded normal lordosis case resulted in the full recruitment of two multifidus fascicles (MultPL5 & MultL2T12) on each side of the body (Figure 4). While these fascicles were fully activated, they only accounted for 8.7% of the total metabolic power required to stabilize the system (Table 1). The recruitment of the internal obliques (IntOb2) resulted in 32.5% of the total metabolic power required to stabilize (Table 1). Additionally, a total of 44% of the total metabolic power required to stabilize resulted due to the recruitment of multiple Longissimus Thoracis Pars Thoracis fascicles (Figure 4, Table 1).

Similarly, the unloaded hyperlordosis cases resulted in over 30% of the total metabolic power required to stabilize due to recruitment of the internal obliques (Table 1). Additionally, 53.6% and 50.6% of the total metabolic power required to stabilize was due to recruitment of Latissimus Dorsi fascicles for the 84.1-degree and 89-degree lordosis angle respectively (Table 1). Like the normal lordosis case, the hyperlordosis cases also required full activation of some of the multifidi fascicles based on model predictions (Figure 4).
For the unloaded hypolordosis cases, none of the muscle fascicles were fully activated (Figure 4). The majority of muscle fascicles recruited were multifidus and Longissimus Thoracis Pars Lumborum fascicles. For the 10-degree lordosis angle, the multifidus fascicles and Longissimus Thoracis Pars Lumborum fascicles accounted for 68.8% and 24.5% of the total metabolic power required to stabilize, respectively (Table 1). For the 32.1-degree lordosis angle, the multifidus fascicles and Longissimus Thoracis Pars Lumborum fascicles accounted for 54.4% and 15.6% of the total metabolic power required to stabilize, respectively (Table 1).

**Critical Stiffness.** The critical stiffness $q_{\text{critical}}$ was determined for the hypolordosis (32.1°), normal lordosis (58.1°), and hyperlordosis (84.1°) cases for simulations with a 200 N load applied at level T4 and 20 cm anterior to the trunk. The resulting critical stiffness was 9 for the hypolordosis case, 10.9 for the normal lordosis case and 13 for the hyperlordosis case (Table 2).

**Nonlinear Verification.** Nonlinear simulations were performed for lordosis angle cases of 32.1°, 58.1° 84.1°. The nonlinear verification involving the calculated distance between equilibrium and normalized state-space can be seen in Figures 5-10. For the unloaded task, the normal lordosis and hypolordosis cases approach equilibrium as time increases. The unloaded hyperlordosis has an oscillatory behavior that is still settling with an increase of time. In the loaded 200 N cases, the hypolordosis and normal lordosis cases are still settling but indicate a negative slope approaching equilibrium. The hyperlordosis case demonstrates an approach to equilibrium.

**DISCUSSION**

In this study, lumbar lordosis angles of 10°, 32.1°, 58.1° 84.1° and 89° were simulated to investigate the potential impact that lumbar lordosis angles may have on spinal stability and behavior.

As hypothesized, the hypolordosis cases required additional recruitment of extensor muscles to successfully adhere to the equilibrium and stability requirements in comparison to the normal lordosis
case (Figure 4). Unlike the normal lordosis and hyperlordosis cases, the hypolordosis cases did not require as much activation of the internal and external obliques, flexor muscles commonly utilized for stability. This was expected as the line of gravity is positioned anteriorly in the hypolordosis case [29], requiring the increased recruitment of the extensor muscles to stabilize and balance the system [21, 29]. The multifidus has been identified as a primary stabilizer of the lumbar spine [21, 29] with its characteristics, such as PCSA and path, resulting in less required metabolic power in comparison to the magnitude other extensors would require [21]. It has been determined that subjects with low back pain show lesser recruitment of the multifidus [30] which could be detrimental to spine stability if the multifidus is recruited as the main stabilizer in certain loading and activities.

The loaded hyperlordosis case required a greater magnitude of required metabolic power when compared to the other cases. In the hyperlordosis cases, flexor muscles were recruited as was anticipated due to the gravity line being located more posteriorly [29] (Figure 4). Interestingly, the system utilized the recruitment of the latissimus dorsi to stabilize the system, which previously has been neglected as a vital stabilizer for the lumbar spine [21, 31]. Future simulations should explore the ability of the system to stabilize with the exclusion of latissimus dorsi in the model for the hyperlordosis case. If the system is able to stabilize, then the alternative muscle recruitment strategy and the magnitude of the metabolic power would be of interest.

Unfortunately at this point in time, it appears that researchers have focused their investigations on the possible correlation between low back pain and lordosis angle magnitude but not the muscle activations needed to stabilize a hyperlordotic spine in an erect standing posture [32-34]. Thus, the model’s predicted muscle activations for the hyperlordosis case could not be validated with experimental data.
The critical stiffness determined for the hypolordosis (32.1˚), normal lordosis (58.1˚) and hyperlordosis (84.1˚) cases indicated that the hyperlordosis case would be the lordosis case with the greatest magnitude of critical stiffness. This critical stiffness represents the minimal value of the stiffness gain $q$ required to stabilize the spine. Given the required activation and metabolic power required to stabilize the spine, an increased critical stiffness would be reasonable. Based on the range provided for Crisco et al. [23] for the stiffness gain, this would indicate that it would be possible for this lordosis case to be considered unstable with the current loading parameters and a smaller stiffness gain.

Lastly, through the nonlinear verification, it was determined that for all lordosis cases and loading cases, the distance between normalized state-space and equilibrium was decreasing with an increase in time, indicating the approach to equilibrium. The unloaded hyperlordosis case was not as certain as the others but due to the wave height decreases for the longer waves present, it was considered to be evidence of it approaching equilibrium. In the future, longer nonlinear simulations could be completed to investigate this behavior over a greater span of time.

CONCLUSIONS

In this study, the impact of lumbar lordosis angle on spine stability was evaluated for normal lordosis, hyperlordosis and hypolordosis. Based on model predictions, the lumbar lordosis angle may have an impact on the metabolic power required to stabilize the spine and muscle activation. The hyperlordosis cases required more metabolic power in order to stabilize while the hypolordosis cases required the least. Additionally, the hypolordotic spine increased recruitment of the extensors while the hyperlordotic spine increase recruitment of the flexors due to the adjusted line of gravity.

ACKNOWLEDGMENT
FUNDING

This publication was made possible by grant number R03AR061597 from the National Institute of Arthritis and Musculoskeletal and Skin Diseases (NIAMS) at the National Institutes of Health. Its contents are solely the responsibility of the authors and do not necessarily represent the official views of NIAMS.
<table>
<thead>
<tr>
<th><strong>NOMENCLATURE</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>$KE$</td>
</tr>
<tr>
<td>Kinetic Energy</td>
</tr>
<tr>
<td>$PE$</td>
</tr>
<tr>
<td>Potential Energy</td>
</tr>
<tr>
<td>IVD</td>
</tr>
<tr>
<td>Intervertebral Disc</td>
</tr>
<tr>
<td>$q$</td>
</tr>
<tr>
<td>Stiffness Gain</td>
</tr>
<tr>
<td>$b$</td>
</tr>
<tr>
<td>Damping Gain</td>
</tr>
<tr>
<td>$Gp$</td>
</tr>
<tr>
<td>Proportional Gain</td>
</tr>
<tr>
<td>$Gd$</td>
</tr>
<tr>
<td>Differential Gain</td>
</tr>
<tr>
<td>$K$</td>
</tr>
<tr>
<td>Stiffness coefficient</td>
</tr>
<tr>
<td>$x_o$</td>
</tr>
<tr>
<td>Equilibrium length</td>
</tr>
<tr>
<td>$\alpha$</td>
</tr>
<tr>
<td>Muscle Activation</td>
</tr>
<tr>
<td>$J_i$</td>
</tr>
<tr>
<td>Jacobian of the instantaneous components</td>
</tr>
<tr>
<td>$J_d$</td>
</tr>
<tr>
<td>Jacobian of the delayed components</td>
</tr>
</tbody>
</table>
REFERENCES


Figure Captions List

Fig. 1 The current model incorporates 90 muscle fascicles, one rigid body for each of the lumbar vertebrae and one rigid body representing the thoracic and cervical region. A lumped parameter model is utilized to represent the intervertebral discs.

Fig. 2 The appearance of hyperlordosis, hypolordosis and normal lordosis in the lumbar spine.

Fig. 3 The metabolic power required to stabilize each lordosis case. One simulation was unloaded while the other was loaded with a 200 N load anterior to the trunk.

Fig. 4 Muscle activation for the right-side muscles for the unloaded case.

Fig. 5 The distance of the normalized state-space from equilibrium versus time for the unloaded hyperlordotic spine.

Fig. 6 The distance of the normalized state-space from equilibrium versus time for the unloaded hypolordotic spine.

Fig. 7 The distance of the normalized state-space from equilibrium versus time for the unloaded spine with normal lordosis.

Fig. 8 The distance of the normalized state-space from equilibrium versus time for the hyperlordotic spine with a 200 N load applied anteriorly at the T4 level.
Fig. 9  The distance of the normalized state-space from equilibrium versus time for the hypolordotic spine with a 200 N load applied anteriorly at the T4 level

Fig. 10  The distance of the normalized state-space from equilibrium versus time for the spine with normal lordosis and a 200 N load applied anteriorly at the T4 level
Table Caption List

Table 1  The percentage of metabolic power required to recruit certain muscle groups to achieve stability in the unloaded cases

Table 2  Critical stiffness for for the hypolordosis (32.1°), normal lordosis (58.1°), and hyperlordosis (84.1°) cases for simulations with a 200 N load applied at level T4 and 20 cm anterior to the trunk
Figure 5-1: The current model incorporates 90 muscle fascicles, one rigid body for each of the lumbar vertebrae and one rigid body representing the thoracic and cervical region. A lumped parameter model is utilized to represent the intervertebral discs.
Figure 5-2: The appearance of hyperlordosis, hypolordosis and normal lordosis in the lumbar spine
Figure 5-3: The metabolic power required to stabilize each lordosis case. One simulation was unloaded while the other was loaded with a 200 N load anterior to the trunk.
Figure 5-4: Muscle activation for the right-side muscles for the unloaded case
Figure 5-5: The distance of the normalized state-space from equilibrium versus time for the unloaded hyperlordotic spine
Figure 5-6: The distance of the normalized state-space from equilibrium versus time for the unloaded hypolordotic spine
Figure 5-7: The distance of the normalized state-space from equilibrium versus time for the unloaded spine with normal lordosis
Figure 5-8: The distance of the normalized state-space from equilibrium versus time for the hyperlordotic spine with a 200 N load applied anteriorly at the T4 level
Figure S-9: The distance of the normalized state-space from equilibrium versus time for the hypolordotic spine with a 200 N load applied anteriorly at the T4 level.
Figure 5-10: The distance of the normalized state-space from equilibrium versus time for the spine with normal lordosis and a 200 N load applied anteriorly at the T4 level.
<table>
<thead>
<tr>
<th></th>
<th>Hypolordosis: 10°</th>
<th>Hypolordosis: 32.1°</th>
<th>Normal Lordosis: 58.1°</th>
<th>Hyperlordosis: 84.1°</th>
<th>Hyperlordosis: 89°</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ext. Oblique</td>
<td>3.2%</td>
<td>5.3%</td>
<td>0.3%</td>
<td>0.0%</td>
<td>0.1%</td>
</tr>
<tr>
<td>Int. Oblique</td>
<td>0.8%</td>
<td>9.3%</td>
<td>32.5%</td>
<td>34.0%</td>
<td>31.9%</td>
</tr>
<tr>
<td>Pars Lum.</td>
<td>24.5%</td>
<td>15.6%</td>
<td>2.7%</td>
<td>0.0%</td>
<td>2.4%</td>
</tr>
<tr>
<td>Long. Thor.</td>
<td>0.0%</td>
<td>0.0%</td>
<td>44.0%</td>
<td>8.1%</td>
<td>11.0%</td>
</tr>
<tr>
<td>Lat. Dorsi</td>
<td>0.0%</td>
<td>0.0%</td>
<td>11.0%</td>
<td>53.6%</td>
<td>50.6%</td>
</tr>
<tr>
<td>Multifidus</td>
<td>68.8%</td>
<td>54.4%</td>
<td>8.7%</td>
<td>3.8%</td>
<td>4.0%</td>
</tr>
<tr>
<td>Quad. Lum.</td>
<td>0.0%</td>
<td>0.0%</td>
<td>0.7%</td>
<td>0.5%</td>
<td>0.0%</td>
</tr>
<tr>
<td>Ilio. Lum.</td>
<td>2.7%</td>
<td>15.5%</td>
<td>0.2%</td>
<td>0.0%</td>
<td>0.0%</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>100.0%</strong></td>
<td><strong>100.0%</strong></td>
<td><strong>100.0%</strong></td>
<td><strong>100.0%</strong></td>
<td><strong>100.0%</strong></td>
</tr>
</tbody>
</table>

Table 5-1: The percentage of metabolic power required to recruit certain muscle groups to achieve stability in the unloaded cases.
<table>
<thead>
<tr>
<th></th>
<th>Hypolordosis</th>
<th>Normal Lordosis</th>
<th>Hyperlordosis</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q_{critical}$</td>
<td>9</td>
<td>10.9</td>
<td>13</td>
</tr>
</tbody>
</table>

*Table 5-2: Critical stiffness for hypolordosis (32.1°), normal lordosis (58.1°), and hyperlordosis (84.1°) cases for simulations with a 200 N load applied at level T4 and 20 cm anterior to the trunk*
Chapter 6: Conclusions and Future Work

In this work, the Franklin et al. model\textsuperscript{16,17,24} was constructed and utilized to investigate the potential impact of stiffness gain and lumbar lordosis on spinal stability. Throughout the development of the model, the MATLAB code and methodology used for the current model was verified with the Franklin et al. model\textsuperscript{16,17,24} files received through personal communication with the Musculoskeletal Biomechanics Laboratory at Virginia Polytechnic Institute when possible (Michael Madigan, personal communication, July 26, 2008). Additionally, Franklin provided some of the source code in the appendix of his thesis\textsuperscript{24} which was utilized when constructing the model.

6.1 Model Verification Procedure

A main component of this work was constructing the Franklin et al. model\textsuperscript{16,17,24} to utilize in our investigations. In many of the verification tasks, using the Franklin et al. files obtained from the Musculoskeletal Biomechanics Laboratory at Virginia Polytechnic Institute (Michael Madigan, personal communication, July 26, 2008), the function and script outputs could be compared for the Franklin et al. model and the current model. Additionally, the methodology used by Franklin was critically reviewed as the model was constructed. In many instances in this work, the differences between the current model and Franklin’s model are mentioned, usually in reference to updated model parameters or methodology, allowing for the reader will be able to distinguish the updates made to the model and the original model.

Our in-depth verification procedure allowed for errors to be minimized to the best of our ability (Specific Aim 1). In the future, validation tasks should be performed to ensure that model predictions are realistic and correspond with experimental findings for human subjects. To fulfill these validation tasks, experimental findings existing in literature can be utilized or experimental studies can be completed.
6.2 Stiffness Gain and Spine Stability

For Specific Aim 2, the impact of dimensionless stiffness gain \( q \) on spine stability was evaluated. By increasing the magnitude of stiffness gain, it was determined that less metabolic power and force was required to stabilize the system, supporting our hypothesis. Crisco et al. proposed that a range spanning for 0.5 to 42 could be utilized for the stiffness gain in modeling of short-range muscle stiffness. Our results prove that the demand on the muscles to stabilize the spinal greatly differs from that of \( q = 0.9 \) to \( q = 40 \), with the model predicting that an additional approximately 64 Watts of metabolic power are required to stabilize the \( q = 0.9 \) condition. In terms of muscle forces, an additional approximately 2000 N force would need to be recruited from the muscles for the \( q = 0.9 \) case. For stiffness gain magnitudes of less than 0.9, the system was unable to stabilize.

These results indicate that the magnitude of the stiffness gain selected when modeling short-range stiffness does impact the model predictions, such as the magnitude of metabolic power required, the magnitude of the muscle force required, and the ability for the spine to stabilize in general. Additionally, in the future, the concept of varying stiffness gains should be explored.

6.3 Lordosis Angles and Spine Stability

From Specific Aim 3, the impact of lumbar lordosis angle on spine stability was evaluated. Through our simulations, it was determined that the hyperlordotic spine required more metabolic power and the additional recruitment of the flexor muscles in order to stabilize when compared to the normal lordosis case. This supported our first hypothesis. In contrast to our next hypothesis, the hypolordotic spine required less metabolic power to stabilize in comparison to the normal lordosis case. For the hypolordotic spine, there was an increase in recruitment of muscle extensors, as was hypothesized. In
the future, experimental studies should be completed to yield the muscle activations for a hyperlordotic spine in erect posture.

Many researchers have excluded the latissimus dorsi from their stability-based models. Due to the model predictions that the latissimus dorsi would be utilized to stabilize the normal and hyperlordotic spine, future simulations should explore the ability of the system to stabilize with the exclusion of latissimus dorsi in the model. If the system is able to stabilize in the absence of the latissimus dorsi, then the adjusted muscle recruitment strategy and metabolic power required would be of interest.

Studies have determined that pregnant women have increased lordosis during pregnancy related to the anatomical adjustments to their bodies such as weakened abdominal muscles, increased weight and adjusted center of gravity\textsuperscript{91,92}. Liebetrau \textit{et al.} utilized a two-dimensional stability-based model to investigate the behavior of the spine with these anatomical changes\textsuperscript{92}. Morino \textit{et al.} developed a model that can be utilized to predict co-contraction in pregnant women\textsuperscript{91}, assisting in the investigation of the mechanisms causing low back pain during pregnancy.

In the future, the model developed in this work could be expanded to contribute to the investigation of the causes of low back pain during pregnancy. This would involve adjustments to the muscle model to account for the weakened abdominal muscles with adjusted muscle paths, the magnitude of hyperlordosis commonly found in pregnant women be incorporated into the model parameters, and the shifted center of gravity\textsuperscript{91}.

6.4 Additional Future Work

In addition to the tasks suggested in Sections 6.1-6.3, loading during trunk flexion and rotation should be explored in the future as these have been deemed as risk factors for low back pain\textsuperscript{91}. These investigations are crucial for material handlers working in warehouses and in production who utilize these motions frequently. Additionally, asymmetric loading tasks should be explored, including varying
load magnitudes and the point of application of the load. At this point in time, only symmetric loading tasks have been evaluated using this model and the magnitude of the loads have been consistent between researchers\textsuperscript{16,17,24}. 
References


Appendix A

A.1 Moment of Inertia Equations

The following equations can be used to calculate the mass moment of inertia of an elliptical cylinder:

![Figure A-1: Elliptical Cylinder Top View](image)

\[ I = \frac{1}{4} ma^2 + \frac{1}{3} mL^2 \]

*Equation A-1: Moment of Inertia about the X-Axis*

\[ I = \frac{1}{4} mb^2 + \frac{1}{3} mL^2 \]

*Equation A-2: Moment of Inertia about the Y-Axis*

\[ I = \frac{1}{4} m(a^2 + b^2) \]

*Equation A-3: Moment of Inertia About the Z-Axis*

where \( a \) is the radius along the major axis of the ellipse, \( b \) is the radius along the minor axis of the ellipse, \( m \) is the mass of the body, and \( L \) is the length or vertebral height of the body.
A.2 Lordosis Angles

Pesenti et al.\textsuperscript{61} measured the cumulative lordosis angles with respect to the superior endplate of S1 for 119 healthy subjects (Table A-1, Column 1). The sacral slope, measured at 38 degrees, defines the angle between the superior endplate of S1 and the horizontal plane\textsuperscript{61}. In Column 2 of Table A-1, these angles have been adjusted to their measures with respect to Axis 1, which is parallel to the horizontal plane. This will allow for the lordosis angle of each segment to be applied as a rotation around Axis 1 (Figure A-2). This is the method that has been used previously to implement lordosis in the Franklin et al.\textsuperscript{24} model.

Table A-1: Development of the Normal Lordosis Case

Column 1 of this table indicates the cumulative lordosis provided in Pesenti et al.\textsuperscript{61} Adjusting the pelvis angle to -38°, the segmental lordosis for each of the lumbar vertebrae can be redefined using this reference (Column 2). Column 3 represents the lordosis angles Franklin utilized in the original model\textsuperscript{24}

<table>
<thead>
<tr>
<th></th>
<th>Pesenti et al.\textsuperscript{61} (deg)</th>
<th>Pesenti et al.\textsuperscript{61} Adjusted Lordosis (deg)</th>
<th>Franklin et al.\textsuperscript{24} Lordosis (deg)</th>
</tr>
</thead>
<tbody>
<tr>
<td>L1</td>
<td>58.1</td>
<td>20.1</td>
<td>10</td>
</tr>
<tr>
<td>L2</td>
<td>55.5</td>
<td>17.5</td>
<td>15</td>
</tr>
<tr>
<td>L3</td>
<td>47.6</td>
<td>9.6</td>
<td>18</td>
</tr>
<tr>
<td>L4</td>
<td>36</td>
<td>-2</td>
<td>8</td>
</tr>
<tr>
<td>L5</td>
<td>20.2</td>
<td>-17.8</td>
<td>-14</td>
</tr>
<tr>
<td>Pelvis</td>
<td>0</td>
<td>-38</td>
<td>-20</td>
</tr>
</tbody>
</table>

*Measuring the lordotic angle of each vertebra with respect to the superior endplate of S1.
In their study, Pesenti et al. measured the lordosis angles of 81 women and 38 men. If it is determined that sex impacts skeletal anatomy and lumbar lordosis measures significantly, then the mean lordosis angles reported in the Pesenti et al. study may be skewed. At this point in time, results are inconclusive and future investigation is required.

The lordosis angles used previously by Franklin et al. are presented in Column 3 of Table A-2.

Additionally, for our investigation of the possible impact of lordosis angle on spine stability, the segmental lordosis angles for each case can be seen in Table A-3.

*Table A-2: Segmental lordosis angles for all lordosis cases studied in Chapter 5*

<table>
<thead>
<tr>
<th></th>
<th>Hyperlordosis 89°</th>
<th>Hyperlordosis 84.1°</th>
<th>Normal Lordosis 58.1°</th>
<th>Hypolordosis 32.1°</th>
<th>Hypolordosis 10°</th>
</tr>
</thead>
<tbody>
<tr>
<td>L0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>L1</td>
<td>51</td>
<td>46.1</td>
<td>20.1</td>
<td>-5.9</td>
<td>-28</td>
</tr>
<tr>
<td>L2</td>
<td>33.2</td>
<td>42.736</td>
<td>17.5</td>
<td>-7.184</td>
<td>-30</td>
</tr>
<tr>
<td>L3</td>
<td>15.4</td>
<td>30.962</td>
<td>9.6</td>
<td>-11.678</td>
<td>-32</td>
</tr>
<tr>
<td>L4</td>
<td>-2.4</td>
<td>14.142</td>
<td>-2</td>
<td>-18.098</td>
<td>-34</td>
</tr>
<tr>
<td>L5</td>
<td>-20.2</td>
<td>-8.565</td>
<td>-17.8</td>
<td>-26.765</td>
<td>-36</td>
</tr>
<tr>
<td>Pelvis</td>
<td>-38</td>
<td>-38</td>
<td>-38</td>
<td>-38</td>
<td>-38</td>
</tr>
</tbody>
</table>
A.3 Skeletal and Muscle Coordinates

The skeletal coordinates and muscle attachment points provided by Cholewicki et al.\textsuperscript{15} were used. Adjustments were made to the muscle attachment points based on the deduced methodology used by Franklin\textsuperscript{24} (transformation of coordinates discussed in 2.1.2 Body Origins). Franklin provides the muscle attachment and skeletal geometry coordinates used in the appendix of this thesis\textsuperscript{24}, but due to discrepancies between a few of the transformed coordinates, the coordinates used in this work are provided in Tables A-3 and A-4. Attachment points RIB10, RIB21, and RIB22 differ between the two works. This is further discussed in Chapter 3. Additionally, the muscle attachment point coordinates used in our work (Table A-4) differ from that provided in Franklin’s thesis\textsuperscript{24}; it appears that Franklin’s work may have had data input errors as the nodal points are not associated with the appropriate muscle fascicle specified by Cholewicki et al.\textsuperscript{15} Additionally, RPEL1 and LPEL1 were adjusted to (0,0,0), acting as the origin of the pelvis.

The global coordinates were transformed to the local coordinate systems through the following process, where Equation A-1 represents translation and Equation A-2 represents rotation in the transformation:

\[
\text{coord}_{\text{body,local}} = \text{coord}_{\text{body,global}} - \text{origin}_{\text{body,global}} \tag{A-4}
\]

where \text{coord}_{\text{b,global}} is the muscle attachment point in global coordinates, \text{origin}_{\text{b,global}} is the origin of the body in global coordinates, and \text{coord}_{\text{b,local}} is the muscle attachment point in local coordinates.

\[
\text{coord}_{\text{body,local}} = R1 \ast \text{coord}_{\text{body,local}} \tag{A-5}
\]

where \(R1\) is the rotation matrix for a rotation around Axis 1.

Figure A-3 presents the relationships between the defined axes for Franklin\textsuperscript{24} and Cholewicki et al.\textsuperscript{15}: 
Figure A-3: Franklin vs Cholewicki Axes Definition. Franklin Axis 2 and Cholewicki X Axis point anteriorly

Table A-3: Skeletal Geometry Coordinates

These coordinates originate from Cholewicki et al. Each measure is presented in meters.

Nomenclature: In the coordinate name, the first letter signifies what side of the body the point is on (R=right, L=left), followed by the body the coordinate is associated with, followed by the coordinate number. For example, RPEL1: This attachment point is located on the right-side of the body and is the first point associated with the pelvis.

<table>
<thead>
<tr>
<th>RPEL1</th>
<th>0.0000</th>
<th>0.0000</th>
<th>0.0000</th>
<th>Origin</th>
<th>LPEL1</th>
<th>0.0000</th>
<th>0.0000</th>
<th>0.0000</th>
</tr>
</thead>
<tbody>
<tr>
<td>RPEL2</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.1065</td>
<td>Connect L5</td>
<td>LPEL2</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.1065</td>
</tr>
<tr>
<td>RPEL3</td>
<td>0.0300</td>
<td>0.0780</td>
<td>-0.0254</td>
<td></td>
<td>LPEL3</td>
<td>-0.0300</td>
<td>0.0780</td>
<td>-0.0254</td>
</tr>
<tr>
<td>RPEL4</td>
<td>0.1300</td>
<td>0.0350</td>
<td>0.1152</td>
<td></td>
<td>LPEL4</td>
<td>-0.1300</td>
<td>0.0350</td>
<td>0.1152</td>
</tr>
<tr>
<td>RPEL5</td>
<td>0.0000</td>
<td>0.0839</td>
<td>-0.0260</td>
<td></td>
<td>LPEL5</td>
<td>0.0000</td>
<td>0.0839</td>
<td>-0.0260</td>
</tr>
<tr>
<td>RPEL6</td>
<td>0.1250</td>
<td>-0.0001</td>
<td>0.1477</td>
<td></td>
<td>LPEL6</td>
<td>-0.1250</td>
<td>-0.0001</td>
<td>0.1477</td>
</tr>
<tr>
<td>RPEL7</td>
<td>0.1200</td>
<td>0.0644</td>
<td>0.0863</td>
<td></td>
<td>LPEL7</td>
<td>-0.1200</td>
<td>0.0644</td>
<td>0.0863</td>
</tr>
<tr>
<td>RPEL8</td>
<td>0.1300</td>
<td>0.0350</td>
<td>0.1152</td>
<td></td>
<td>LPEL8</td>
<td>-0.1300</td>
<td>0.0350</td>
<td>0.1152</td>
</tr>
<tr>
<td>RPEL9</td>
<td>0.0600</td>
<td>-0.0693</td>
<td>0.1170</td>
<td></td>
<td>LPEL9</td>
<td>-0.0600</td>
<td>-0.0693</td>
<td>0.1170</td>
</tr>
<tr>
<td>RPEL10</td>
<td>0.0680</td>
<td>-0.0804</td>
<td>0.1060</td>
<td></td>
<td>LPEL10</td>
<td>-0.0680</td>
<td>-0.0804</td>
<td>0.1060</td>
</tr>
<tr>
<td>RPEL11</td>
<td>0.0330</td>
<td>-0.0805</td>
<td>0.1050</td>
<td></td>
<td>LPEL11</td>
<td>-0.0330</td>
<td>-0.0805</td>
<td>0.1050</td>
</tr>
<tr>
<td>RPEL12</td>
<td>0.0900</td>
<td>-0.0301</td>
<td>0.1495</td>
<td>LPEL12</td>
<td>-0.0900</td>
<td>-0.0301</td>
<td>0.1495</td>
<td></td>
</tr>
<tr>
<td>RPEL13</td>
<td>0.0360</td>
<td>-0.0671</td>
<td>0.1188</td>
<td>LPEL13</td>
<td>-0.0360</td>
<td>-0.0671</td>
<td>0.1188</td>
<td></td>
</tr>
<tr>
<td>RPEL14</td>
<td>0.0150</td>
<td>-0.0771</td>
<td>0.0776</td>
<td>LPEL14</td>
<td>-0.0150</td>
<td>-0.0771</td>
<td>0.0776</td>
<td></td>
</tr>
<tr>
<td>RPEL15</td>
<td>0.0300</td>
<td>-0.0561</td>
<td>0.1298</td>
<td>LPEL15</td>
<td>-0.0300</td>
<td>-0.0561</td>
<td>0.1298</td>
<td></td>
</tr>
<tr>
<td>RPEL16</td>
<td>0.0600</td>
<td>-0.0419</td>
<td>0.1516</td>
<td>LPEL16</td>
<td>-0.0600</td>
<td>-0.0419</td>
<td>0.1516</td>
<td></td>
</tr>
<tr>
<td>RPEL17</td>
<td>0.0820</td>
<td>0.0441</td>
<td>-0.0222</td>
<td>LPEL17</td>
<td>-0.0820</td>
<td>0.0441</td>
<td>-0.0222</td>
<td></td>
</tr>
<tr>
<td>RPEL18</td>
<td>0.0000</td>
<td>-0.0814</td>
<td>0.1061</td>
<td>LPEL18</td>
<td>0.0000</td>
<td>-0.0814</td>
<td>0.1061</td>
<td></td>
</tr>
<tr>
<td>RPEL19</td>
<td>0.0620</td>
<td>-0.0364</td>
<td>0.1461</td>
<td>LPEL19</td>
<td>-0.0620</td>
<td>-0.0364</td>
<td>0.1461</td>
<td></td>
</tr>
</tbody>
</table>

| RL51    | 0.0000 | 0.0000  | 0.0000 | Origin  | LL51    | 0.0000  | 0.0000  | 0.0000 |
| RL52    | 0.0000 | 0.0000  | 0.0389 | Connect L4 | LL52    | 0.0000  | 0.0000  | 0.0389 |
| RL53    | 0.0500 | -0.0264 | 0.0230 | LL53    | -0.0500 | -0.0264 | 0.0230 |
| RL54    | 0.0200 | -0.0606 | 0.0119 | LL54    | -0.0200 | -0.0606 | 0.0119 |
| RL55    | 0.0050 | -0.0591 | 0.0071 | LL55    | -0.0050 | -0.0591 | 0.0071 |
| RL56    | 0.0150 | -0.0395 | 0.0051 | LL56    | -0.0150 | -0.0395 | 0.0051 |
| RL57    | 0.0230 | -0.0119 | 0.0108 | LL57    | -0.0230 | -0.0119 | 0.0108 |
| RL58    | 0.0000 | 0.0192  | 0.0251 | LL58    | 0.0000  | 0.0192  | 0.0251 |
| RL59    | 0.0000 | -0.0162 | 0.0242 | LL59    | 0.0000  | -0.0162 | 0.0242 |
| RL510   | 0.0000 | -0.0251 | 0.0287 | LL510   | 0.0000  | -0.0251 | 0.0287 |
| RL511   | 0.0360 | -0.0264 | 0.0230 | LL511   | -0.0360 | -0.0264 | 0.0230 |
| RL512   | 0.0200 | -0.0257 | 0.0337 | LL512   | -0.0200 | -0.0257 | 0.0337 |
| RL513   | 0.0240 | -0.0351 | 0.0370 | LL513   | -0.0240 | -0.0351 | 0.0370 |
| RL514   | 0.0000 | -0.0561 | 0.0175 | LL514   | 0.0000  | -0.0561 | 0.0175 |
| RL515   | 0.0000 | -0.0434 | 0.0238 | LL515   | 0.0000  | -0.0434 | 0.0238 |
| RL516   | 0.0000 | -0.0384 | 0.0275 | LL516   | 0.0000  | -0.0384 | 0.0275 |

<p>| RL41    | 0.0000 | 0.0000  | 0.0000 | Origin  | LL41    | 0.0000  | 0.0000  | 0.0000 |
| RL42    | 0.0000 | 0.0000  | 0.0370 | Connect L3 | LL42    | 0.0000  | 0.0000  | 0.0370 |</p>
<table>
<thead>
<tr>
<th>RL43</th>
<th>0.0400</th>
<th>-0.0320</th>
<th>0.0230</th>
<th>LL43</th>
<th>-0.0400</th>
<th>-0.0320</th>
<th>0.0230</th>
</tr>
</thead>
<tbody>
<tr>
<td>RL44</td>
<td>0.0020</td>
<td>-0.0680</td>
<td>0.0130</td>
<td>LL44</td>
<td>-0.0020</td>
<td>-0.0680</td>
<td>0.0130</td>
</tr>
<tr>
<td>RL45</td>
<td>0.0440</td>
<td>-0.0340</td>
<td>0.0270</td>
<td>LL45</td>
<td>-0.0440</td>
<td>-0.0340</td>
<td>0.0270</td>
</tr>
<tr>
<td>RL46</td>
<td>0.0050</td>
<td>-0.0650</td>
<td>0.0040</td>
<td>LL46</td>
<td>-0.0050</td>
<td>-0.0650</td>
<td>0.0040</td>
</tr>
<tr>
<td>RL47</td>
<td>0.0150</td>
<td>-0.0390</td>
<td>0.0010</td>
<td>LL47</td>
<td>-0.0150</td>
<td>-0.0390</td>
<td>0.0010</td>
</tr>
<tr>
<td>RL48</td>
<td>0.0020</td>
<td>-0.0700</td>
<td>0.0130</td>
<td>LL48</td>
<td>-0.0020</td>
<td>-0.0700</td>
<td>0.0130</td>
</tr>
<tr>
<td>RL49</td>
<td>0.0230</td>
<td>-0.0140</td>
<td>0.0070</td>
<td>LL49</td>
<td>-0.0230</td>
<td>-0.0140</td>
<td>0.0070</td>
</tr>
<tr>
<td>RL410</td>
<td>0.0500</td>
<td>-0.0810</td>
<td>0.0120</td>
<td>LL410</td>
<td>-0.0500</td>
<td>-0.0810</td>
<td>0.0120</td>
</tr>
<tr>
<td>RL411</td>
<td>0.0740</td>
<td>-0.0810</td>
<td>0.0120</td>
<td>LL411</td>
<td>-0.0740</td>
<td>-0.0810</td>
<td>0.0120</td>
</tr>
<tr>
<td>RL412</td>
<td>0.0380</td>
<td>-0.0810</td>
<td>0.0120</td>
<td>LL412</td>
<td>-0.0380</td>
<td>-0.0810</td>
<td>0.0120</td>
</tr>
<tr>
<td>RL413</td>
<td>0.0150</td>
<td>-0.0810</td>
<td>0.0120</td>
<td>LL413</td>
<td>-0.0150</td>
<td>-0.0810</td>
<td>0.0120</td>
</tr>
<tr>
<td>RL414</td>
<td>0.0000</td>
<td>0.0180</td>
<td>0.0110</td>
<td>LL414</td>
<td>0.0000</td>
<td>0.0180</td>
<td>0.0110</td>
</tr>
<tr>
<td>RL415</td>
<td>0.0000</td>
<td>-0.0190</td>
<td>0.0120</td>
<td>LL415</td>
<td>0.0000</td>
<td>-0.0190</td>
<td>0.0120</td>
</tr>
<tr>
<td>RL416</td>
<td>0.0000</td>
<td>-0.0260</td>
<td>0.0150</td>
<td>LL416</td>
<td>0.0000</td>
<td>-0.0260</td>
<td>0.0150</td>
</tr>
<tr>
<td>RL417</td>
<td>0.0360</td>
<td>-0.0320</td>
<td>0.0230</td>
<td>LL417</td>
<td>-0.0360</td>
<td>-0.0320</td>
<td>0.0230</td>
</tr>
<tr>
<td>RL418</td>
<td>0.0200</td>
<td>-0.0340</td>
<td>0.0100</td>
<td>LL418</td>
<td>-0.0200</td>
<td>-0.0340</td>
<td>0.0100</td>
</tr>
<tr>
<td>RL419</td>
<td>0.0160</td>
<td>-0.0340</td>
<td>0.0090</td>
<td>LL419</td>
<td>-0.0160</td>
<td>-0.0340</td>
<td>0.0090</td>
</tr>
<tr>
<td>RL420</td>
<td>0.0000</td>
<td>-0.0700</td>
<td>0.0090</td>
<td>LL420</td>
<td>0.0000</td>
<td>-0.0700</td>
<td>0.0090</td>
</tr>
<tr>
<td>RL421</td>
<td>0.0000</td>
<td>-0.0600</td>
<td>0.0070</td>
<td>LL421</td>
<td>0.0000</td>
<td>-0.0600</td>
<td>0.0070</td>
</tr>
<tr>
<td>RL422</td>
<td>0.0000</td>
<td>-0.0700</td>
<td>0.0120</td>
<td>LL422</td>
<td>0.0000</td>
<td>-0.0700</td>
<td>0.0120</td>
</tr>
<tr>
<td>RL423</td>
<td>0.0000</td>
<td>0.0160</td>
<td>0.0030</td>
<td>LL423</td>
<td>0.0000</td>
<td>0.0160</td>
<td>0.0030</td>
</tr>
<tr>
<td>RL424</td>
<td>0.0000</td>
<td>0.0010</td>
<td>0.0060</td>
<td>LL424</td>
<td>0.0000</td>
<td>0.0010</td>
<td>0.0060</td>
</tr>
<tr>
<td>RL425</td>
<td>0.0000</td>
<td>-0.0160</td>
<td>0.0060</td>
<td>LL425</td>
<td>0.0000</td>
<td>-0.0160</td>
<td>0.0060</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RL31</th>
<th>0.0000</th>
<th>0.0000</th>
<th>0.0000</th>
<th>Origin</th>
<th>LL31</th>
<th>0.0000</th>
<th>0.0000</th>
<th>0.0000</th>
</tr>
</thead>
<tbody>
<tr>
<td>RL32</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0379</td>
<td>Connect L2</td>
<td>LL32</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0379</td>
</tr>
<tr>
<td>RL33</td>
<td>0.0300</td>
<td>-0.0324</td>
<td>0.0254</td>
<td>LL33</td>
<td>-0.0300</td>
<td>-0.0324</td>
<td>0.0254</td>
<td></td>
</tr>
<tr>
<td>RL</td>
<td>RL Value 1</td>
<td>RL Value 2</td>
<td>RL Value 3</td>
<td>LL</td>
<td>LL Value 1</td>
<td>LL Value 2</td>
<td>LL Value 3</td>
<td></td>
</tr>
<tr>
<td>------</td>
<td>------------</td>
<td>------------</td>
<td>------------</td>
<td>------</td>
<td>------------</td>
<td>------------</td>
<td>------------</td>
<td></td>
</tr>
<tr>
<td>RL34</td>
<td>0.0020</td>
<td>-0.0715</td>
<td>0.0195</td>
<td>LL34</td>
<td>-0.0020</td>
<td>-0.0715</td>
<td>0.0195</td>
<td></td>
</tr>
<tr>
<td>RL35</td>
<td>0.0380</td>
<td>-0.0388</td>
<td>0.0288</td>
<td>LL35</td>
<td>-0.0380</td>
<td>-0.0388</td>
<td>0.0288</td>
<td></td>
</tr>
<tr>
<td>RL36</td>
<td>0.0050</td>
<td>-0.0662</td>
<td>0.0061</td>
<td>LL36</td>
<td>-0.0050</td>
<td>-0.0662</td>
<td>0.0061</td>
<td></td>
</tr>
<tr>
<td>RL37</td>
<td>0.0150</td>
<td>-0.0431</td>
<td>0.0042</td>
<td>LL37</td>
<td>-0.0150</td>
<td>-0.0431</td>
<td>0.0042</td>
<td></td>
</tr>
<tr>
<td>RL38</td>
<td>0.0020</td>
<td>-0.0734</td>
<td>0.0200</td>
<td>LL38</td>
<td>-0.0020</td>
<td>-0.0734</td>
<td>0.0200</td>
<td></td>
</tr>
<tr>
<td>RL39</td>
<td>0.0210</td>
<td>-0.0165</td>
<td>0.0087</td>
<td>LL39</td>
<td>-0.0210</td>
<td>-0.0165</td>
<td>0.0087</td>
<td></td>
</tr>
<tr>
<td>RL310</td>
<td>0.0760</td>
<td>-0.0832</td>
<td>0.0221</td>
<td>LL310</td>
<td>-0.0760</td>
<td>-0.0832</td>
<td>0.0221</td>
<td></td>
</tr>
<tr>
<td>RL311</td>
<td>0.0390</td>
<td>-0.0832</td>
<td>0.0221</td>
<td>LL311</td>
<td>-0.0390</td>
<td>-0.0832</td>
<td>0.0221</td>
<td></td>
</tr>
<tr>
<td>RL312</td>
<td>0.0150</td>
<td>-0.0832</td>
<td>0.0221</td>
<td>LL312</td>
<td>-0.0150</td>
<td>-0.0832</td>
<td>0.0221</td>
<td></td>
</tr>
<tr>
<td>RL21</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>Origin</td>
<td>LL21</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>RL22</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0386</td>
<td>Connect L1</td>
<td>LL22</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0386</td>
</tr>
<tr>
<td>RL23</td>
<td>0.0270</td>
<td>-0.0337</td>
<td>0.0236</td>
<td>LL23</td>
<td>-0.0270</td>
<td>-0.0337</td>
<td>0.0236</td>
<td></td>
</tr>
<tr>
<td>RL24</td>
<td>0.0020</td>
<td>-0.0710</td>
<td>0.0138</td>
<td>LL24</td>
<td>-0.0020</td>
<td>-0.0710</td>
<td>0.0138</td>
<td></td>
</tr>
<tr>
<td>RL25</td>
<td>0.0380</td>
<td>-0.0398</td>
<td>0.0275</td>
<td>LL25</td>
<td>-0.0380</td>
<td>-0.0398</td>
<td>0.0275</td>
<td></td>
</tr>
<tr>
<td>RL26</td>
<td>0.0050</td>
<td>-0.0678</td>
<td>0.0035</td>
<td>LL26</td>
<td>-0.0050</td>
<td>-0.0678</td>
<td>0.0035</td>
<td></td>
</tr>
<tr>
<td>RL27</td>
<td>0.0150</td>
<td>-0.0396</td>
<td>0.0003</td>
<td>LL27</td>
<td>-0.0150</td>
<td>-0.0396</td>
<td>0.0003</td>
<td></td>
</tr>
<tr>
<td>RL28</td>
<td>0.0020</td>
<td>-0.0729</td>
<td>0.0144</td>
<td>LL28</td>
<td>-0.0020</td>
<td>-0.0729</td>
<td>0.0144</td>
<td></td>
</tr>
<tr>
<td>RL29</td>
<td>0.0200</td>
<td>-0.0187</td>
<td>0.0108</td>
<td>LL29</td>
<td>-0.0200</td>
<td>-0.0187</td>
<td>0.0108</td>
<td></td>
</tr>
<tr>
<td>RL210</td>
<td>0.0780</td>
<td>-0.0822</td>
<td>0.0182</td>
<td>LL210</td>
<td>-0.0780</td>
<td>-0.0822</td>
<td>0.0182</td>
<td></td>
</tr>
<tr>
<td>RL211</td>
<td>0.0410</td>
<td>-0.0822</td>
<td>0.0182</td>
<td>LL211</td>
<td>-0.0410</td>
<td>-0.0822</td>
<td>0.0182</td>
<td></td>
</tr>
<tr>
<td>RL212</td>
<td>0.0150</td>
<td>-0.0822</td>
<td>0.0182</td>
<td>LL212</td>
<td>-0.0150</td>
<td>-0.0822</td>
<td>0.0182</td>
<td></td>
</tr>
<tr>
<td>RL11</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>Origin</td>
<td>LL11</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>RL12</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0362</td>
<td>Connect L0</td>
<td>LL12</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0362</td>
</tr>
<tr>
<td>RL13</td>
<td>0.0260</td>
<td>-0.0367</td>
<td>0.0233</td>
<td>LL13</td>
<td>-0.0260</td>
<td>-0.0367</td>
<td>0.0233</td>
<td></td>
</tr>
<tr>
<td>RL14</td>
<td>0.0020</td>
<td>-0.0714</td>
<td>0.0193</td>
<td>LL14</td>
<td>-0.0020</td>
<td>-0.0714</td>
<td>0.0193</td>
<td></td>
</tr>
<tr>
<td>RL15</td>
<td>0.0360</td>
<td>-0.0395</td>
<td>0.0267</td>
<td>LL15</td>
<td>-0.0360</td>
<td>-0.0395</td>
<td>0.0267</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RL16</td>
<td>-0.0674</td>
<td>0.0087</td>
<td>LL16</td>
<td>-0.0050</td>
<td>-0.0674</td>
<td>0.0087</td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>----------</td>
<td>---------</td>
<td>--------</td>
<td>----------</td>
<td>---------</td>
<td>---------</td>
<td>--------</td>
<td></td>
</tr>
<tr>
<td>RL17</td>
<td>0.0020</td>
<td>-0.0732</td>
<td>0.0201</td>
<td>LL17</td>
<td>-0.0020</td>
<td>-0.0732</td>
<td>0.0201</td>
<td></td>
</tr>
<tr>
<td>RL18</td>
<td>0.0190</td>
<td>-0.0187</td>
<td>0.0096</td>
<td>LL18</td>
<td>-0.0190</td>
<td>-0.0187</td>
<td>0.0096</td>
<td></td>
</tr>
<tr>
<td>RL19</td>
<td>0.0800</td>
<td>-0.0823</td>
<td>0.0242</td>
<td>LL19</td>
<td>-0.0800</td>
<td>-0.0823</td>
<td>0.0242</td>
<td></td>
</tr>
<tr>
<td>RL110</td>
<td>0.0430</td>
<td>-0.0823</td>
<td>0.0242</td>
<td>LL110</td>
<td>-0.0430</td>
<td>-0.0823</td>
<td>0.0242</td>
<td></td>
</tr>
<tr>
<td>RL111</td>
<td>0.0150</td>
<td>-0.0823</td>
<td>0.0242</td>
<td>LL111</td>
<td>-0.0150</td>
<td>-0.0823</td>
<td>0.0242</td>
<td></td>
</tr>
<tr>
<td>RRIB1</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>Origin</td>
<td>LRIB1</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>RRIB2</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.3250</td>
<td>Connect C7</td>
<td>LRIB2</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.3250</td>
</tr>
<tr>
<td>RRIB3</td>
<td>0.0700</td>
<td>0.1180</td>
<td>-0.0039</td>
<td>LRIB3</td>
<td>-0.0700</td>
<td>0.1180</td>
<td>-0.0039</td>
<td></td>
</tr>
<tr>
<td>RRIB4</td>
<td>0.1250</td>
<td>-0.0115</td>
<td>-0.0551</td>
<td>LRIB4</td>
<td>-0.1250</td>
<td>-0.0115</td>
<td>-0.0551</td>
<td></td>
</tr>
<tr>
<td>RRIB5</td>
<td>0.1050</td>
<td>0.0534</td>
<td>-0.0395</td>
<td>LRIB5</td>
<td>-0.1050</td>
<td>0.0534</td>
<td>-0.0395</td>
<td></td>
</tr>
<tr>
<td>RRIB6</td>
<td>0.0700</td>
<td>0.0786</td>
<td>-0.0643</td>
<td>LRIB6</td>
<td>-0.0700</td>
<td>0.0786</td>
<td>-0.0643</td>
<td></td>
</tr>
<tr>
<td>RRIB7</td>
<td>0.0000</td>
<td>0.1178</td>
<td>0.0261</td>
<td>LRIB7</td>
<td>0.0000</td>
<td>0.1178</td>
<td>0.0261</td>
<td></td>
</tr>
<tr>
<td>RRIB8</td>
<td>0.0840</td>
<td>-0.0563</td>
<td>0.0345</td>
<td>LRIB8</td>
<td>-0.0840</td>
<td>-0.0563</td>
<td>0.0345</td>
<td></td>
</tr>
<tr>
<td>RRIB9</td>
<td>0.0500</td>
<td>-0.0528</td>
<td>0.0845</td>
<td>LRIB9</td>
<td>-0.0500</td>
<td>-0.0528</td>
<td>0.0845</td>
<td></td>
</tr>
<tr>
<td>RRIB10</td>
<td>0.0200</td>
<td>-0.0537</td>
<td>0.1795</td>
<td>LRIB10</td>
<td>-0.0200</td>
<td>-0.0537</td>
<td>0.1795</td>
<td></td>
</tr>
<tr>
<td>RRIB11</td>
<td>0.0200</td>
<td>-0.0541</td>
<td>0.2245</td>
<td>LRIB11</td>
<td>-0.0200</td>
<td>-0.0541</td>
<td>0.2245</td>
<td></td>
</tr>
<tr>
<td>RRIB12</td>
<td>0.0200</td>
<td>-0.0495</td>
<td>0.2696</td>
<td>LRIB12</td>
<td>-0.0200</td>
<td>-0.0495</td>
<td>0.2696</td>
<td></td>
</tr>
<tr>
<td>RRIB13</td>
<td>0.0200</td>
<td>-0.0499</td>
<td>0.3116</td>
<td>LRIB13</td>
<td>-0.0200</td>
<td>-0.0499</td>
<td>0.3116</td>
<td></td>
</tr>
<tr>
<td>RRIB14</td>
<td>0.0200</td>
<td>-0.0500</td>
<td>0.3296</td>
<td>LRIB14</td>
<td>-0.0200</td>
<td>-0.0500</td>
<td>0.3296</td>
<td></td>
</tr>
<tr>
<td>RRIB15</td>
<td>0.0720</td>
<td>-0.0370</td>
<td>-0.0003</td>
<td>LRIB15</td>
<td>-0.0720</td>
<td>-0.0370</td>
<td>-0.0003</td>
<td></td>
</tr>
<tr>
<td>RRIB16</td>
<td>0.0050</td>
<td>-0.0539</td>
<td>-0.0135</td>
<td>LRIB16</td>
<td>-0.0050</td>
<td>-0.0539</td>
<td>-0.0135</td>
<td></td>
</tr>
<tr>
<td>RRIB17</td>
<td>0.1200</td>
<td>0.0169</td>
<td>0.1152</td>
<td>LRIB17</td>
<td>-0.1200</td>
<td>0.0169</td>
<td>0.1152</td>
<td></td>
</tr>
<tr>
<td>RRIB18</td>
<td>0.0650</td>
<td>-0.0599</td>
<td>-0.0106</td>
<td>LRIB18</td>
<td>-0.0650</td>
<td>-0.0599</td>
<td>-0.0106</td>
<td></td>
</tr>
<tr>
<td>RRIB19</td>
<td>0.0650</td>
<td>-0.0476</td>
<td>-0.0384</td>
<td>LRIB19</td>
<td>-0.0650</td>
<td>-0.0476</td>
<td>-0.0384</td>
<td></td>
</tr>
<tr>
<td>RRIB20</td>
<td>0.0650</td>
<td>-0.0414</td>
<td>-0.0654</td>
<td>LRIB20</td>
<td>-0.0650</td>
<td>-0.0414</td>
<td>-0.0654</td>
<td></td>
</tr>
<tr>
<td>Muscle</td>
<td>Origin</td>
<td>Terminal</td>
<td>Node1</td>
<td>Node2</td>
<td>Node3</td>
<td>Node4</td>
<td>Node5</td>
<td>Node6</td>
</tr>
<tr>
<td>-------------</td>
<td>--------</td>
<td>----------</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>RRectAb</td>
<td>RPEL3</td>
<td>RRIB3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RExtOb1</td>
<td>RPEL4</td>
<td>RRIB4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RExtOb2</td>
<td>RPEL5</td>
<td>RRIB5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RIntOb1</td>
<td>RPEL6</td>
<td>RRIB6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RIntOb2</td>
<td>RPEL7</td>
<td>RRIB7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R ParsL5</td>
<td>RPEL9</td>
<td>RL53</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R ParsL4</td>
<td>RPEL9</td>
<td>RL43</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R ParsL3</td>
<td>RPEL9</td>
<td>RL33</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R ParsL2</td>
<td>RPEL9</td>
<td>RL23</td>
<td>RL410</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R ParsL1</td>
<td>RPEL9</td>
<td>RL13</td>
<td>RL410</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RilioLum</td>
<td>RPEL10</td>
<td>RRIB8</td>
<td>RL411</td>
<td>RL310</td>
<td>RL210</td>
<td>RL19</td>
<td></td>
<td></td>
</tr>
<tr>
<td>R Long TP</td>
<td>RPEL11</td>
<td>RRIB9</td>
<td>RL412</td>
<td>RL311</td>
<td>RL211</td>
<td>RL110</td>
<td></td>
<td></td>
</tr>
<tr>
<td>R Long TL5</td>
<td>RL54</td>
<td>RRIB10</td>
<td>RL413</td>
<td>RL312</td>
<td>RL212</td>
<td>RL111</td>
<td>RRIB23</td>
<td></td>
</tr>
<tr>
<td>R Long TL4</td>
<td>RL44</td>
<td>RRIB11</td>
<td>RL312</td>
<td>RL212</td>
<td>RL111</td>
<td>RRIB23</td>
<td></td>
<td></td>
</tr>
<tr>
<td>R Long TL3</td>
<td>RL34</td>
<td>RRIB12</td>
<td>RL212</td>
<td>RL111</td>
<td>RRIB23</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R Long TL2</td>
<td>RL24</td>
<td>RRIB13</td>
<td>RL111</td>
<td>RRIB23</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R Long TL1</td>
<td>RL14</td>
<td>RRIB14</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R Quad LP</td>
<td>RPEL12</td>
<td>RRIB15</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Table A-4: Muscle Attachment Points*

The muscle attachment points and physiological cross-sectional areas are from Cholewicki et al.\(^\text{15}\)
<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>RQuadL1</td>
<td>RPEL12</td>
<td>RL15</td>
<td>1</td>
</tr>
<tr>
<td>RQuadL2</td>
<td>RPEL12</td>
<td>RL25</td>
<td>1</td>
</tr>
<tr>
<td>RQuadL3</td>
<td>RPEL12</td>
<td>RL35</td>
<td>1</td>
</tr>
<tr>
<td>RQuadL4</td>
<td>RPEL12</td>
<td>RL45</td>
<td>1</td>
</tr>
<tr>
<td>RLatDP</td>
<td>RPEL16</td>
<td>RRIB17</td>
<td>2</td>
</tr>
<tr>
<td>RLatDL5</td>
<td>RPEL15</td>
<td>RRIB17 RRIB22</td>
<td>2</td>
</tr>
<tr>
<td>RLatDL4</td>
<td>RL48</td>
<td>RRIB17 RRIB21</td>
<td>4</td>
</tr>
<tr>
<td>RLatDL3</td>
<td>RL38</td>
<td>RRIB17 RRIB20</td>
<td>4</td>
</tr>
<tr>
<td>RLatDL2</td>
<td>RL28</td>
<td>RRIB17 RRIB19</td>
<td>4</td>
</tr>
<tr>
<td>RLatDL1</td>
<td>RL17</td>
<td>RRIB17 RRIB18</td>
<td>4</td>
</tr>
<tr>
<td>RMultPL5</td>
<td>RPEL14</td>
<td>RL55</td>
<td>1.4</td>
</tr>
<tr>
<td>RMultPL4</td>
<td>RPEL14</td>
<td>RL46</td>
<td>2.9</td>
</tr>
<tr>
<td>RMultPL3</td>
<td>RPEL13</td>
<td>RL36</td>
<td>2.4</td>
</tr>
<tr>
<td>RMultPL2</td>
<td>RPEL13</td>
<td>RL26</td>
<td>1.5</td>
</tr>
<tr>
<td>RMultPL1</td>
<td>RPEL13</td>
<td>RL16</td>
<td>0.9</td>
</tr>
<tr>
<td>RMultLSL3</td>
<td>RL56</td>
<td>RL36</td>
<td>0.8</td>
</tr>
<tr>
<td>RMultLSL2</td>
<td>RL56</td>
<td>RL26</td>
<td>0.6</td>
</tr>
<tr>
<td>RMultLSL1</td>
<td>RL56</td>
<td>RL16</td>
<td>0.6</td>
</tr>
<tr>
<td>RMultL4L2</td>
<td>RL47</td>
<td>RL26</td>
<td>0.6</td>
</tr>
<tr>
<td>RMultL4L1</td>
<td>RL47</td>
<td>RL16</td>
<td>0.6</td>
</tr>
<tr>
<td>RMultL3L1</td>
<td>RL37</td>
<td>RL16</td>
<td>0.6</td>
</tr>
<tr>
<td>RMultL2T12</td>
<td>RL27</td>
<td>RRIB16</td>
<td>0.6</td>
</tr>
<tr>
<td>RPsoaL5</td>
<td>RPEL17</td>
<td>RL57</td>
<td>4.4</td>
</tr>
<tr>
<td>RPsoaL4</td>
<td>RPEL17</td>
<td>RL49</td>
<td>4.4</td>
</tr>
<tr>
<td>RPsoaL3</td>
<td>RPEL17</td>
<td>RL39</td>
<td>4.4</td>
</tr>
<tr>
<td>RPsoaL2</td>
<td>RPEL17</td>
<td>RL29</td>
<td>4.4</td>
</tr>
<tr>
<td>RPsoaL1</td>
<td>RPEL17</td>
<td>RL18</td>
<td>4.4</td>
</tr>
<tr>
<td>LRectAb</td>
<td>LPEL3</td>
<td>LRIB3</td>
<td>10</td>
</tr>
<tr>
<td>LExtOb1</td>
<td>LPEL4</td>
<td>LRIB4</td>
<td>10</td>
</tr>
<tr>
<td>LExtOb2</td>
<td>LPEL5</td>
<td>LRIB5</td>
<td>9</td>
</tr>
<tr>
<td>LIntOb1</td>
<td>LPEL6</td>
<td>LRIB6</td>
<td>9</td>
</tr>
<tr>
<td>LIntOb2</td>
<td>LPEL7</td>
<td>LRIB7</td>
<td>8</td>
</tr>
<tr>
<td>LParsL5</td>
<td>LPEL9</td>
<td>LL53</td>
<td>5.6</td>
</tr>
<tr>
<td>LParsL4</td>
<td>LPEL9</td>
<td>LL43</td>
<td>6</td>
</tr>
<tr>
<td>LParsL3</td>
<td>LPEL9</td>
<td>LL33</td>
<td>5.7</td>
</tr>
<tr>
<td>LParsL2</td>
<td>LPEL9</td>
<td>LL23</td>
<td>LL410</td>
</tr>
<tr>
<td>LParsL1</td>
<td>LPEL9</td>
<td>LL13</td>
<td>LL410</td>
</tr>
<tr>
<td>LlilioLumLPEL10</td>
<td>LRIIB8</td>
<td>LL411</td>
<td>LL310</td>
</tr>
<tr>
<td>LLongTP</td>
<td>LPEL11</td>
<td>LRIIB9</td>
<td>LL412</td>
</tr>
<tr>
<td>LLongTL5</td>
<td>LPEL11</td>
<td>LRIIB10</td>
<td>LL413</td>
</tr>
<tr>
<td>LLongTL4</td>
<td>LPEL11</td>
<td>LRIIB11</td>
<td>LL312</td>
</tr>
<tr>
<td>LLongTL3</td>
<td>LPEL11</td>
<td>LRIIB12</td>
<td>LL212</td>
</tr>
<tr>
<td>LLongTL2</td>
<td>LPEL11</td>
<td>LRIIB13</td>
<td>LL111</td>
</tr>
<tr>
<td>LLongTL1</td>
<td>LPEL11</td>
<td>LRIIB14</td>
<td>1.3</td>
</tr>
<tr>
<td>LQuadLP</td>
<td>LPEL12</td>
<td>LRIIB15</td>
<td>2</td>
</tr>
<tr>
<td>LQuadL1</td>
<td>LPEL12</td>
<td>LL15</td>
<td>1</td>
</tr>
<tr>
<td>LQuadL2</td>
<td>LPEL12</td>
<td>LL25</td>
<td>1</td>
</tr>
<tr>
<td>LQuadL3</td>
<td>LPEL12</td>
<td>LL35</td>
<td>1</td>
</tr>
<tr>
<td>LQuadL4</td>
<td>LPEL12</td>
<td>LL45</td>
<td>1</td>
</tr>
<tr>
<td>LLatDP</td>
<td>LPEL16</td>
<td>LRIIB17</td>
<td>2</td>
</tr>
<tr>
<td>LLatDL5</td>
<td>LPEL15</td>
<td>LRIIB17</td>
<td>LRIIB22</td>
</tr>
<tr>
<td>LLatDL4</td>
<td>LPEL15</td>
<td>LRIIB17</td>
<td>LRIIB21</td>
</tr>
<tr>
<td>LLatDL3</td>
<td>LPEL15</td>
<td>LRIIB17</td>
<td>LRIIB20</td>
</tr>
<tr>
<td>LLatDL2</td>
<td>LPEL15</td>
<td>LRIIB17</td>
<td>LRIIB19</td>
</tr>
<tr>
<td>LLatDL1</td>
<td>LPEL15</td>
<td>LRIIB17</td>
<td>LRIIB18</td>
</tr>
<tr>
<td>LMultPL5</td>
<td>LPEL14</td>
<td>LL55</td>
<td>1.4</td>
</tr>
<tr>
<td>LMultPL4</td>
<td>LPEL14</td>
<td>LL46</td>
<td>2.9</td>
</tr>
<tr>
<td>LMultPL3</td>
<td>LPEL13</td>
<td>LL36</td>
<td>2.4</td>
</tr>
<tr>
<td>LMultPL2</td>
<td>LPEL13</td>
<td>LL26</td>
<td>1.5</td>
</tr>
<tr>
<td>LMultPL1</td>
<td>LPEL13</td>
<td>LL16</td>
<td>0.9</td>
</tr>
<tr>
<td>LMultL5L3</td>
<td>LPEL14</td>
<td>LL56</td>
<td>LL36</td>
</tr>
<tr>
<td>LMultL5L2</td>
<td>LPEL14</td>
<td>LL56</td>
<td>LL26</td>
</tr>
<tr>
<td>LMultL5L1</td>
<td>LPEL14</td>
<td>LL56</td>
<td>LL16</td>
</tr>
<tr>
<td>LMultL4L2</td>
<td>LPEL14</td>
<td>LL47</td>
<td>LL26</td>
</tr>
<tr>
<td>LMultL4L1</td>
<td>LL47</td>
<td>LL16</td>
<td>0.6</td>
</tr>
<tr>
<td>LMultL3L1</td>
<td>LL37</td>
<td>LL16</td>
<td>0.6</td>
</tr>
<tr>
<td>LMultL2T12</td>
<td>LL27</td>
<td>LRIB16</td>
<td>0.6</td>
</tr>
<tr>
<td>LPsoaL5</td>
<td>LPEL17</td>
<td>LL57</td>
<td>4.4</td>
</tr>
<tr>
<td>LPsoaL4</td>
<td>LPEL17</td>
<td>LL49</td>
<td>4.4</td>
</tr>
<tr>
<td>LPsoaL3</td>
<td>LPEL17</td>
<td>LL39</td>
<td>4.4</td>
</tr>
<tr>
<td>LPsoaL2</td>
<td>LPEL17</td>
<td>LL29</td>
<td>4.4</td>
</tr>
<tr>
<td>LPsoaL1</td>
<td>LPEL17</td>
<td>LL18</td>
<td>4.4</td>
</tr>
</tbody>
</table>
A.4 Rotations

Each vertebra is rotated around Axis 1, followed by Axis 2, and lastly by Axis 3. The counterclockwise direction is defined as positive.

A rotation is applied around Axis 1($\hat{n}_1$) by an angle $q_1$ (Figure A-4).

The following rotation matrix defines the angular relationship between the N and A bases:

$$
^N R^A = \begin{bmatrix}
1 & 0 & 0 \\
0 & \cos(q_1) & -\sin(q_1) \\
0 & \sin(q_1) & \cos(q_1)
\end{bmatrix}
$$

(A-6)

A rotation is applied around Axis 2($\hat{a}_2$) by an angle $q_2$ (Figure A-5).
The following rotation matrix defines the angular relationship between the A and B bases:

\[ R^A_B = \begin{bmatrix} \cos(q_2) & 0 & \sin(q_2) \\ 0 & 1 & 0 \\ -\sin(q_2) & 0 & \cos(q_2) \end{bmatrix} \]  

(A-7)

A rotation is applied around Axis 3(\(\hat{b}_3\)) by an angle \(q_3\) (Figure A-6).

![Figure A-6: Rotation around Axis 3](image)

The following rotation matrix defines the angular relationship between the B and C bases:

\[ R^B_C = \begin{bmatrix} \cos(q_3) & -\sin(q_3) & 0 \\ \sin(q_3) & \cos(q_3) & 0 \\ 0 & 0 & 1 \end{bmatrix} \]  

(A-8)

A rotation matrix was determined that would allow for all three rotations to be applied to the vertebra with one matrix\(^{81}\):

\[ R^C_N = R^A_N * R^B_A * R^C_B \]  

(A-9)

The matrix \(R^C_N\) is the rotation matrix from basis N to basis C.
Appendix B

The following flowchart provides a general overview of the model processes. This was designed to aid the reader in their understanding of the model and its functions. Appendix C and D provide the scripts and files utilized in the model.

Figure B-1: General Model Flowchart
B.1 Code Directory

<table>
<thead>
<tr>
<th>Script</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>costfun.m</td>
<td>Calculates the metabolic power required in the optimization procedure</td>
</tr>
<tr>
<td>For_verification.m</td>
<td>Performs MATLAB-based verification tasks</td>
</tr>
<tr>
<td>load_musclefile.m</td>
<td>Imports the muscle anatomy from Excel spreadsheet</td>
</tr>
<tr>
<td>model_parameters.m</td>
<td>Script that contains the models parameters</td>
</tr>
<tr>
<td>optimize.m</td>
<td>Calculate the constraints for the constrained optimization procedure; call run_fmincon, and determine optimal solution</td>
</tr>
<tr>
<td>spine_main.m</td>
<td>Main spine model file</td>
</tr>
<tr>
<td>spineacc.m</td>
<td>Acceleration used for constructing the Jacobians</td>
</tr>
<tr>
<td>spineaccA.m</td>
<td>Acceleration used for equilibrium constraint in optimization</td>
</tr>
<tr>
<td>spineaccP.m</td>
<td>Acceleration used for equilibrium constraint in optimization</td>
</tr>
<tr>
<td>tfdfind.m</td>
<td>Calculate the time delay</td>
</tr>
<tr>
<td>tdfmax.m</td>
<td>Determine the maximum frequency for time delay analysis</td>
</tr>
<tr>
<td>tdinvest.m</td>
<td>Main time delay file</td>
</tr>
<tr>
<td>tofuncts.m</td>
<td>Convert symbolic outputs to function handles</td>
</tr>
<tr>
<td>vect2ang.m</td>
<td>Input the Stateo and Pango vectors and output individual angles based on these vectors</td>
</tr>
<tr>
<td>verify.m</td>
<td>Used to compare function inputs and determine absolute error between the inputs</td>
</tr>
<tr>
<td>dynam.mlx</td>
<td>Perform the Lagrangian Derivative. Determine the symbolic generalized force partial derivatives, mass matrix, gravity vector, and Coriolis vector</td>
</tr>
<tr>
<td>generalizedForce.mlx</td>
<td>Perform generalized force calculations</td>
</tr>
<tr>
<td>IVD_calcs.mlx</td>
<td>Calculate the intervertebral disc moments</td>
</tr>
<tr>
<td>load_forcefile.mlx</td>
<td>Import the load details from Excel spreadsheet</td>
</tr>
<tr>
<td>musclemodel.mlx</td>
<td>Muscle model</td>
</tr>
<tr>
<td>musclemodelR.mlx</td>
<td>Reflex model</td>
</tr>
<tr>
<td>nlcon.mlx</td>
<td>Nonlinear constraints for constrained optimization</td>
</tr>
<tr>
<td>origin_body.mlx</td>
<td>Determine the symbolic origins of the bodies</td>
</tr>
<tr>
<td>replace_sym.mlx</td>
<td>Function used to replace symbolic variables with numeric values from the Stateo, Pango, and Vars vectors</td>
</tr>
<tr>
<td>rotate_anatomy_sym.mlx</td>
<td>Determine the symbolic rotated muscle anatomy, muscle length and muscle velocity</td>
</tr>
<tr>
<td>rotateforce.mlx</td>
<td>Determine the symbolic rotated external force vector</td>
</tr>
<tr>
<td>rotationmatrices.mlx</td>
<td>Determine the symbolic rotation matrices</td>
</tr>
<tr>
<td>run_fmincon.mlx</td>
<td>Run the constrained optimization procedure</td>
</tr>
<tr>
<td>spinedyn.mlx</td>
<td>Function handle used in delay differential equation solver</td>
</tr>
<tr>
<td>sub_timevar.mlx</td>
<td>Replaces angles in variables with time-dependent angles</td>
</tr>
<tr>
<td>transform_Cholewicki.mlx</td>
<td>Transform the Cholewicki et al. anatomy and align vertebral connection points</td>
</tr>
</tbody>
</table>
B.2 MAT-file Directory

Table B-2: MAT-file directory for the .mat files utilized in the model. Many outputs of functions were saved as symbolic .mat files, allowing for these outputs to be updated to function handles in the tofuncts.m file

<table>
<thead>
<tr>
<th>MAT-file</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dynam_sym_output.mat</td>
<td>Symbolic dynamics file variables</td>
</tr>
<tr>
<td>Ea_sym.mat</td>
<td>Symbolic rotated external force vector</td>
</tr>
<tr>
<td>model_parameters_mat.mat</td>
<td>Model parameters defined</td>
</tr>
<tr>
<td>Model_Parameters_Verify.mat</td>
<td>Model parameters utilized for verification</td>
</tr>
<tr>
<td>optimization_varsA.mat</td>
<td>Optimization variables used to increase efficiency</td>
</tr>
<tr>
<td>origin_sym_OtoP.mat</td>
<td>Symbolic origin 7x3 matrix; first row describes the origin of body 0, continues in order, and last row ends with the pelvis origin</td>
</tr>
<tr>
<td>origin_sym_Pto0.mat</td>
<td>Symbolic origin 7x3 matrix; first row describes the origin of the pelvis, continues in order, and last row ends with body 0</td>
</tr>
<tr>
<td>origin_sym_separate.mat</td>
<td>Separate symbolic origin vectors for each body</td>
</tr>
<tr>
<td>rotate_anatomy_sym_MaMLMV.mat</td>
<td>Symbolic outputs for rotated muscle anatomy, muscle length and muscle velocity</td>
</tr>
<tr>
<td>rotation_matrices_sym_OtoP.mat</td>
<td>Symbolic rotation 21x3 matrix; first three rows describe the rotation matrix of body 0, continues in order, and last three rows describe the pelvis’ rotation matrix</td>
</tr>
<tr>
<td>rotation_matrices_sym_Pto0.mat</td>
<td>Symbolic rotation 21x3 matrix; first three rows describe the rotation matrix of the pelvis, continues in order, and last three rows describe the rotation matrix for body 0</td>
</tr>
<tr>
<td>rotation_matrices_sym_separate.mat</td>
<td>Separate symbolic rotation matrices for each body</td>
</tr>
<tr>
<td>Stateo_Pango_Rand.mat</td>
<td>Stateo and Pango vectors for verification</td>
</tr>
<tr>
<td>transformed_Cholewicki.mat</td>
<td>Muscle coordinates after the transformation that aligns the vertebral bodies along the vertical axis</td>
</tr>
<tr>
<td>verification_initialize_vars.mat</td>
<td>Load the variables that were defined prior to the verification procedure; all preceding variables are cleared during the verification procedure</td>
</tr>
</tbody>
</table>

B.3 Personal Communication Summary

Through personal communication with the Musculoskeletal Biomechanics Laboratory at Virginia Polytechnic Institute (Principal Investigators: Dr. Kevin P. Granata and Dr. Michael Madigan), the Human Motion Control Laboratory at the University of Kansas received some of the files developed by Franklin et al. for the model (Michael Madigan, personal communication, July 26, 2008). It was discovered that the files in our possession were not the finalized versions.
Throughout the redevelopement process, the model files were reconstructed and may vary from the work of Franklin\(^{16,17,24}\). Franklin’s files were used as a reference, mostly to ensure that the model was being developed following a similar methodology, but the majority of the files were redrafted and verified. The exception to this would be the time delay analysis codes sourced from the appendix of Franklin’s thesis. Therefore, while many of the scripts and function may be similar, there may be differences in the files as well. Additionally, we did not utilize the MEX files that Franklin developed in our model, outside of for verification procedures. While this did increase run time for the model, this allowed for the files to be redrafted with more of an explanation as to what the purpose of the procedure was and the methodology. In some cases, the MEX files provided a symbolic output where the input values just needed to be substituted in. This did not allow for a thorough explanation of the methodology. Additionally, when possible, the symbolic output for certain calculations would be saved in .mat files to increase efficiency of the MATLAB calculations. These symbolic outputs can be converted to MATLAB functions using the MATLAB `matlabFunction` as well to increase efficiency.

Table B-3: Summary of the MATLAB files received from Dr. Michael Madigan and the files Franklin has published in his thesis

<table>
<thead>
<tr>
<th>File Names</th>
<th>HMCL possesses</th>
<th>Published in</th>
</tr>
</thead>
<tbody>
<tr>
<td>Provisional file names are <strong>bolded</strong>.</td>
<td><strong>provisional file</strong></td>
<td>Franklin’s Thesis(^{24})</td>
</tr>
<tr>
<td>Non-bolded file names indicate the name used in Franklin’s thesis appendix.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Spine Acceleration Functions</td>
<td><strong>AccelerationMP, AccelerationSP,</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>AccelerationMA,</strong> spineaccP, spineaccA,</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>spineaccR</td>
<td></td>
</tr>
<tr>
<td>Optimization</td>
<td><strong>alg_optimize,</strong> <strong>alg_optimizeR,</strong></td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>optimizationscript</td>
<td></td>
</tr>
<tr>
<td>Cholewicki</td>
<td><strong>CholeMusc.xls,</strong> printed in thesis</td>
<td>X</td>
</tr>
<tr>
<td>Section</td>
<td>Description</td>
<td>Files</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>--------------------------------------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>Transformed Muscle File</td>
<td>Appendix</td>
<td></td>
</tr>
<tr>
<td>Dynamic Simulation File</td>
<td><strong>Dynsim</strong>, spinedyn</td>
<td>X X</td>
</tr>
<tr>
<td>Dynamics</td>
<td><strong>Dynmatsc.mexw64</strong></td>
<td>X X*</td>
</tr>
<tr>
<td>Main File</td>
<td>Execsim, EXECSIM, SolverP, Simulation, Spinesim</td>
<td>X X</td>
</tr>
<tr>
<td>Used in dde23 Solver</td>
<td><strong>Edcheck</strong>, Extrema</td>
<td>X</td>
</tr>
<tr>
<td>Import External Force</td>
<td>Forcinput, Forcmat</td>
<td>X</td>
</tr>
<tr>
<td>Generalized Forces Calculations</td>
<td><strong>Geneforcesc.mexw64</strong></td>
<td>X</td>
</tr>
<tr>
<td>Muscle</td>
<td>Maintenance Heat helperfunctC, costfun</td>
<td>X X</td>
</tr>
<tr>
<td>Intervertebral Disc Calculations</td>
<td><strong>Intervetc.mexw64</strong></td>
<td>X</td>
</tr>
<tr>
<td>Kinematics</td>
<td><strong>kinematicsc.mexw64/analys1.mexw64</strong></td>
<td>X X*</td>
</tr>
<tr>
<td>External Force Excel File</td>
<td><strong>LoadLift.xls</strong></td>
<td>X</td>
</tr>
<tr>
<td>Import Muscle Anatomy from</td>
<td>muscinput, muscmat</td>
<td>X</td>
</tr>
</tbody>
</table>
**Excel Spreadsheet**

<table>
<thead>
<tr>
<th>Muscle Model</th>
<th>Muscmodel</th>
<th>X</th>
</tr>
</thead>
</table>

Nonlinear

constraints for

| fminconhelper1, nlcon | X | X |

optimization

procedure

| Para, Anatomvert, included in spinesim file | X | X |

Model Parameters

| Pointscope1, Pointscope2 | X |

Adjusting

coordinates

Determining length of the muscles

| Propmusc | X |

Rotation Matrices

| Rotate1, Rotate2, Rotate3, Rotate4, Rotate5, Rotate6, Rotate7, rotation matrices included in kinematicsc.mex file | X |

Simulation Plots

| Simgraphs | X |

| Tdfind, tdfind | X | X |

Time Delay

| Tdfmax, tdfmax | X | X |

Analysis

| Tdinvest, tdinvest | X | X |

Model Visualization 3D

| vischeme | X |

*Franklin provided Mathematica code in his appendix for some of the variables included in this MEX file.*
Appendix C: **Main MATLAB Files**

The following MATLAB files and methodology were based on the work of Franklin *et al.*\textsuperscript{16,17,24} Franklin’s files were obtained through means of personal communication (Michael Madigan, personal communication, July 26, 2008) and from the appendix of his thesis\textsuperscript{24}
C.1 *spine_main.m*: Main Spine File

```matlab
% Valerie Jardon
% Department of Bioengineering, University of Kansas
% spine_main.m
% Last Edited: 4/29/22
% Description
% This is the main script used to execute the spine simulation of the
% biomechanical model.

% This file (spine_main.m) is based on the spinesim.m code developed by
% Timothy C. Franklin, published in Appendix B of his thesis (Franklin,

% Reference:
% Franklin, T. C. (2006). Linear System Analyses of the Role of Reflex Gain and Delay in

% This model is based on the spine model developed by Timothy C. Franklin and
% Dr. Kevin P. Granata from Virginia Polytechnic Institute and State
% University (Franklin, 2006).

% The Human Motion Control Laboratory at the University of Kansas received some of the
% files developed by Franklin et al. for the model (Michael Madigan, personal communication, July 26, 2008).

% Franklin's published thesis can be downloaded from the Virginia Tech Electronic
% Theses and Dissertations library:
% https://vtechworks.lib.vt.edu/handle/10919/33605
% Author: Timothy C. Franklin
% Date Issued: 2006-06-06
% Title: Linear System Analyses of the Role of Reflex Gain and Delay in a Dynamic Human Spine Model
% Department: Engineering Mechanics
% Type: Thesis
% Degree Grantor: Virginia Polytechnic Institute and State University
% Committee Chair: Kevin P. Granata
% Committee Members: Michael L. Madigan & Scott Hendricks

clear all; close all; clc;
% Verification Procedure
% Create Excel File with verification outputs.
    global date
    date=datetime('now');
```
% Indicates that verification should be performed
save('verification_initialize_vars.mat', 'date', 'verification_logical')

if verification_logical == 1
    For_verification %has clc and clear commands in file
    clear; clc
end

%% Load the Defined Model Parameters
% If you would like to adjust the model parameters, edit the
% "model_parameters.m" file
load('verification_initialize_vars.mat')
model_parameters;

%% Import Muscle Files

[X]=load_musclefile(); %this loads the CholeMuscl.xls file

%% Import External Force File
% If you would like to adjust the external force applied, adjust the
% "load_forcesfile.m" file or the LoadLift.xls Excel spreadsheet being imported

[E]=load_forcesfile(); %this loads the LoadLift.xls file
E=0; %force magnitude (units: N)

rotateforce % determine the symbolic force vector:

%% Define State Variables
V.Sperm=1; %Number of state variables per muscle
V.States=28+V.Sperm*length(N); %Number of state variables
State(37:V.States,1)=0;

%% Transform the Applied External Force

% yield individual angle variables based on state variables vectors; this
% will be used for function inputs
[A0, A1, A2, A3, A4, A5, Ap, B0, B1, B2, B3, B4, B5, Bp, DA0, DA1, DA2, DA3, DA4, DA5, DAp, DB0, DB1, DB2, 
  DB3, DB4, DB5, DBp, DB0, DB1, DB2, DB3, DB4, DB5, DT0, DT1, DT2, DT3, DT4, DT5, DTp, T0, T1, T2, T3, T4, T5, Tp, ...
  T0dot, T1dot, T2dot, T3dot, T4dot, T5dot, T0dot, T1dot, T2dot, T3dot, T4dot, T5dot, B0dot, 
  B1dot, B2dot, B3dot, B4dot, B5dot, Apdot, Bpdot, Tpdot] = vect2ang(State0, Pango);

% convert symbolic outputs to function handles
[G1_funct, Gr_funct, Mv_funct, Ms_funct, Gm_funct, Gm_funct, Mv_funct, Ms_funct, Rm_funct, Rm_funct, origin_funct, origin_funct] = toFuncts();

% determine the rotated force
Es = Es_funct(A0, Ap, B0, B1, B2, B3, B4, B5, Bp, L1, L2, L3, L4, L5, T0, T1, T2, T3, T4, T5, Tp, cent1, 
  cent2, cent3);

% Ensure that the external force attachment points are aligned as necessary
% (if the lordosis case is adjusted, then the external force coordinates

% also need to be to be adjusted)
if abs(Es(4) - Ee(8)) > 0.0001
    error('Check external force rotated coordinates.')
end
% writing to the Verification Excel Spreadsheet
if verification_logical==1
    filename=printf('Verification Summary %s.xlsx',date);
    filename=replace(filename,":","."); % cannot have colons in file titles
    filename=string(append('C:\Users\Valerie\Gordon\Documents\Model\FINALIZED_CODE_updated\Simulation Runs Verification Output\', dir,filename));
    writecell(('External Force Magnitude'),filename,'Sheet',1,'Range','B21')
    writematrix(MFo,filename,'Sheet',1,'Range','B22')
    writecell(('Rotated External Force'),filename,'Sheet',1,'Range','B24')
    writematrix(Ex,filename,'Sheet',1,'Range','B25')
end

% Determine rotated muscle anatomy and resting muscle length
Ma=Ma_func(A0,A1,A2,A3,A4,A5,Ap,B0,B1,B2,B3,B4,B5,Bp,L1,L2,L3,L4,L5,T0,T1,T2,T3,T4, T5,Tp,cent1,cent2,cent3); ML=ML_func(A0,A1,A2,A3,A4,A5,Ap,B0,B1,B2,B3,B4,B5,Bp,L1,L2,L3,L4,L5,T0,T1,T2,T3,T4, T5,Tp,cent1,cent2,cent3);
L0=ML; %initial muscle length

% Perform Optimization Procedure
optimize

% Time Delay Analysis
% For the time delay analysis, the Jacobians must be separated by
% instantaneous and delayed components
disp('-----Time Delay Analysis Started-----')
%
% Building the Jacobian for the instantaneous components
Ji=Jpi;
for i=1:36
    Ji(19:36,i)=Ji(19:36,i)+Un_opt'*Jai(:,19:36,i)';
end
%
% Building the Jacobian for the delayed components (reflexes)
Jr=Jpr;
for i=1:36
    Jr(19:36,i)=Jr(19:36,i)+Un_opt'*Jar(:,19:36,i)';
end
%
% Determine the delay margin in which the system would become unstable
DrawSet=1;

[Td]=tdinvest(Ji,Jr,DrawSet);

% Display determined time delay and muscle model parameters in command window
fprintf('The Time Delay is: %3.3f
',Td)
fprintf('Muscle Model Parameters: g=%3.3f, b=%3.3f, Gp=%3.3f, Gd=%3.3f
',g,b,Gp,Gd)
Simulation

In this section, the nonlinear simulation is completed to compare the linear analysis results from the optimization procedure.

disp('------Simulation Started------')

StateDist=State00;
StateDist(16)=StateDist(16)+deg2rad(0.01);  % disturbance applied to thoracic/cervical spine rigid body

Tfinal=5;
Tode=[0,Tfinal];  % defining the bounds of integration

% the delay margin (Td) defines the time delay in which the system becomes unstable, therefore the simulation should include the time delay prior to instability
Lags=Td*0.99;

% Delay must be greater than 0.06 seconds to be physiologically representative of reflexes and their contribution to stability
if Lags < 0.06
    error('Simulation cannot be performed. Required lag is not physiologically possible.');
end

% Set Simulation Options
options=odeset('OutputFcn',@plotscheme,'Events',@edcheck,'RelTol',1e-4,'AbsTol',1e-9,'Stats','on');

% Simulation
tic
sol=dde23(@(spinedyn,Lags,StateDist,Tode,options,Un_opt,Fn,Pango,Vars,GI_func,Gr_func,Mm_func,Cm_func,Gm_func,Ha_func,ML_func,NV_func,Ea_func,Rm_func,dRm_func,origin_func,dorigin_func,Look_Angs,VLO,Lo,CSA,EFO,ChSpace);
Time=sol.x;
State=sol.y';
Stated=State(ones(length(Time),1)*State0);

% Display Magnitude of Lag Simulated & Simulation Run Time
disp(['Lag Simulated: ' num2str(Lags(1))]);
elapsedtime=toc;

elapsedtime_min=floor(elapsedtime/60);
fprintf('The simulation took %d minutes and %d seconds to run.',elapsedtime_min,elapsedtime-(60*elapsedtime_min));
%% Calculate the distance from equilibrium using the normalized state-space
%% Determine if the system is approaching equilibrium

position=Stated(:,1:18);
mean_position=mean(position,'all'); % average of all positions
position_Dss=position./mean_position;

velocity=Stated(:,19:36);
mean_velocity=mean(velocity,'all');
velocity_Dss=velocity./mean_velocity;

Dss=zeros(size(Stated,1),18);
for i=1:18
    Dss(:,i)=(position_Dss(:,i)).^2+(velocity_Dss(:,i)).^2;
end

Dss=sqrt(sum(Dss,2)); % sum of each row - each row is a time step
figure(4); plot(Time,Dss); grid on; xlabel('Time (seconds)'); ylabel('Dss'); % title('Dss vs. Time')

% ------------------------ End of spine_main.m script
C.2 For_verification.m: Verification Tasks in MATLAB

```matlab
clear; clc

% 1. Initialize Verification Variables for Jardon Calculations
load('Stateo_Pango_Rand.mat')
load('Model_Parameters_Verify.mat')
cent1=0; cent2=0; % these values need to have a magnitude of zero for comparison with Franklin values
Vars=[L0,L1,L2,L3,L4,L5,La,lb,cent3,g,m0,m1,m2,m3,m4,m5,cent1,cent2];
Lordosis.Angle=[Stateo(16) Stateo(13) Stateo(10) Stateo(7) Stateo(4) Stateo(1) Pango(1)];
% Lordosis Angle Array [L0,L1,...Pelvis]
IVD=zeros(1,4); % initializing the intervertebral disc parameters array
IP=50; %Planar
ID=0.5;
IKt=50; %Twist
IDt=0.5;
IVD=[IP, ID, IKt, IDt]; %IVD parameters
A=Stateo(3):end);

% Verification Vectors/ Matrices
Lo_verification=[1:1:90]'; % initial muscle length array for verification
MX_verification=[1:1:90]'; % muscle force array for verification
EF_verification=1; % external force magnitude for verification
FFQ_verification=[1 2 3 4 5 6 7 8 9 5 4 3 2 1 9 6 2 3 2 1]; %IVD moments
matrix for verification
U_verification=0.5*zeros(90,1);
Fn_verification=zeros(90,1);

大部分人.load_forces(file);

% Jardon function handles
[S1_fun,Gr_fun,Mx_fun,Gr_fun,Mr_fun,Mr_fun,ML_fun, MV_fun,Ga_fun, \% 
RM_fun,DRM_fun,origin_fun, dorlin_fun]=toCjoint();

[A0,A1,A2,A3,A4,A5,Ap,B0,B1,B2,B3,B4,B5,Bp,DA0,DA1,DA2,DA3,DA4,DA5,DAP,B0,B1,B2, \% 
DB3,DB4,DB5,DBp,DT0,DT1,DT2,DT3,DB4,DTr,DTp,T0,T1,T2,T3,T4,T5,Tp, ...]
A0dot,A1dot,A2dot,A3dot,A4dot,A5dot,T0dot,T1dot,T2dot,T3dot,T4dot,T5dot,B0dot, \% 
B1dot,B2dot,B3dot,B4dot,B5dot,Ap0dot,Bp0dot,Tp0dot]=vec2ang(Stateo,Pango);

T0dot=0; B5dot=0; A5dot=0;
T0dot=0; B4dot=0; A4dot=0;
T0dot=0; B3dot=0; A3dot=0;
T0dot=0; B2dot=0; A2dot=0;
T0dot=0; B1dot=0; A1dot=0;
T0dot=0; B0dot=0; A0dot=0;
T0dot=0; Bp0dot=0; Ap0dot=0;

L0=Vars(1); L1=Vars(2); L2=Vars(3); L3=Vars(4); L4=Vars(5); L5=Vars(6);
L6=Vars(7); Lb=Vars(8); gb=Vars(10);
cent1=Vars(17); cent2=Vars(18); cent3=Vars(19);
m0=Vars(11); m1=Vars(12); m2=Vars(13); m3=Vars(14); m4=Vars(15); m5=Vars(16);
```

176
% ----- End of "Initialize Verification Variables for Jardon Calcs" section

%% 2. Perform Jardon Calculations for Verification Tasks

GL=GLI_function(B1,B2,B3,B4,B5,L1,L2,L3,L4,L5,T1,T2,T3,T4,T5);
Gr=Gr_function(B0,B1,B2,B3,B4,B5,T0,T1,T2,T3,T4,T5);
Mr=Mr_function(A0,A1,A2,A3,A4,A5,A6,A7,A8,A9,A10,A11,A12,A13,A14,A15,A16,A17,A18,A19,A20,A21);
Cr=Cr_function(A0,A1,A2,A3,A4,A5,A6,A7,A8,A9,A10,A11,A12,A13,A14,A15,A16,A17,A18,A19,A20,A21);
M0=M0_function(A0,A1,A2,A3,A4,A5,A6,A7,A8,A9,A10,A11,A12,A13,A14,A15,A16,A17,A18,A19,A20,A21);
M1=M1_function(A0,A1,A2,A3,A4,A5,A6,A7,A8,A9,A10,A11,A12,A13,A14,A15,A16,A17,A18,A19,A20,A21);
M2=M2_function(A0,A1,A2,A3,A4,A5,A6,A7,A8,A9,A10,A11,A12,A13,A14,A15,A16,A17,A18,A19,A20,A21);
M3=M3_function(A0,A1,A2,A3,A4,A5,A6,A7,A8,A9,A10,A11,A12,A13,A14,A15,A16,A17,A18,A19,A20,A21);
M4=M4_function(A0,A1,A2,A3,A4,A5,A6,A7,A8,A9,A10,A11,A12,A13,A14,A15,A16,A17,A18,A19,A20,A21);
M5=M5_function(A0,A1,A2,A3,A4,A5,A6,A7,A8,A9,A10,A11,A12,A13,A14,A15,A16,A17,A18,A19,A20,A21);
M6=M6_function(A0,A1,A2,A3,A4,A5,A6,A7,A8,A9,A10,A11,A12,A13,A14,A15,A16,A17,A18,A19,A20,A21);
M7=M7_function(A0,A1,A2,A3,A4,A5,A6,A7,A8,A9,A10,A11,A12,A13,A14,A15,A16,A17,A18,A19,A20,A21);
M8=M8_function(A0,A1,A2,A3,A4,A5,A6,A7,A8,A9,A10,A11,A12,A13,A14,A15,A16,A17,A18,A19,A20,A21);
M9=M9_function(A0,A1,A2,A3,A4,A5,A6,A7,A8,A9,A10,A11,A12,A13,A14,A15,A16,A17,A18,A19,A20,A21);
M10=M10_function(A0,A1,A2,A3,A4,A5,A6,A7,A8,A9,A10,A11,A12,A13,A14,A15,A16,A17,A18,A19,A20,A21);
M11=M11_function(A0,A1,A2,A3,A4,A5,A6,A7,A8,A9,A10,A11,A12,A13,A14,A15,A16,A17,A18,A19,A20,A21);
M12=M12_function(A0,A1,A2,A3,A4,A5,A6,A7,A8,A9,A10,A11,A12,A13,A14,A15,A16,A17,A18,A19,A20,A21);
M13=M13_function(A0,A1,A2,A3,A4,A5,A6,A7,A8,A9,A10,A11,A12,A13,A14,A15,A16,A17,A18,A19,A20,A21);
M14=M14_function(A0,A1,A2,A3,A4,A5,A6,A7,A8,A9,A10,A11,A12,A13,A14,A15,A16,A17,A18,A19,A20,A21);
M15=M15_function(A0,A1,A2,A3,A4,A5,A6,A7,A8,A9,A10,A11,A12,A13,A14,A15,A16,A17,A18,A19,A20,A21);

%% 3. Perform Franklin Calculations for Verification Tasks

% Adjust variables necessary for Franklin's function inputs
P=zeros(1,1); P(1:8)=Pango(1:8);
FRQ_verification=FRQ_verification(1:6,1); % needs to be trimmed for Franklin's MEX file
QPagercontrol(Ma,real(MF_verification),Sa,EP_verification,PRQ_verification,Gl,Gr,origin);
% ----- End of "Perform Jardon Calculations for Verification Tasks" Section
% Determining the M matrix
M_F=@struct;
M_F.muscinput(M_F,'.\ChMusc\CholeMusc.xls'); %Cholewicki muscle file from Franklin

% Additional verification task necessary due to errors in Franklin's
% CholeMusc.xls spreadsheet (discussed in Chapter 3 of Jordon thesis)
M_F2=@struct;
M_F2.muscinput(M_F2,'.\ChMusc\CholeMusc_update.xls'); %Cholewicki muscle file with
user input errors corrected

[E]=load_forcefile();  %this loads the LoadLift.xls file - Jordon function
E_F=@struct;
E_F.forinput(E,F,'LoadLift.xls'); % input external forces
E_F=forcmat(E,F);

[Ma_F,Ja_F,ML_F,MV_F,G1_F,Gr_F,Rm_F,Gr_F,drm_F,dGr_F,rbv_F,drbv_F] = analysis(Stateo, %
P_F,,Vars,M,E);

[M_F,dA]=muscmode(U_verification,Pn_verification,A,ML_F,MV_F,Lo_verification,CSA);
PLQ_F=intervertc(RBV_F,drbv_F,Lord_Angs,IVD);

4. Verification Optimization Variables
% Initialize Jordon variables
load('Stateo_Pango_Rand.mat')
load('Model_Parameters_Verify.mat')
cent1=0; cent2=0;
Vars=[L0,L1,L2,L3,L4,L5,la,lb,cent3,g,m0,m1,m2,m3,m4,m5,cent1,cent2];
Lord_Angs=[Stateo(16) Stateo(13) Stateo(10) Stateo(7) Stateo(4) Stateo(1) Pango(1)]; %
%Lordosis Angle Array (L0,L1,...,Pelvis)
IVD=zeros(1,4); %initializing the intervertebral disc parameters array
IKp=50; %Linear
IPd=0.5;
IKt=50; %Twist
IDt=0.5;
IVD=[IKp,IPd,IKt,IDt];  %IVD parameters
CSA=M(:,end); % last column of M
Lo_verification=[L0:L1:190]; %Initial muscle length array for verification
cps=0.00001*pi/180;
BP_verification=1;
Po=zeros(size(M,1),1);Pn-Po;

% Jordon Function handles
[G1_func,Gr_func,Ma_func,Gm_func,GM_func,Gr_func,Ma_func,ML_func,MV_func,Ba_func, %
Rm_func,drm_func,origin_func,dorigin_func)=tofuncs();

M=load_musclefile();  %this loads the CholeMusc.xls file - Jordon function
% This loads the LoadLift.xls file - Jardon function

% Optimization
length_array=zeros(90,1);

% Initializing the constraints for fmincon
Aeq=zeros(18,90); % equilibrium
beq=zeros(18,1); % equilibrium
A=eye(90);
B=zeros(90,1);

% Passive Properties
% For the passive properties (t=0 & activation=0 & no muscle force)
% Includes intervertebral disc moments, external force
U=zeros(90,1); % activation = 0 for passive

beq=-spineaccP(0,Stateo,Pango,U,E_F_verification,Vars,M,E,Pa,Lord_Angs,IVD,GL_func,Gr_func,Mm_func,Cm_func,Gm_func,Ma_func,ML_func,IV_func,Ba_func,Rm_func,\ %
dRm_func,origin_func,dorIGIN_func,[],'verification');

% Active Properties
% For active properties (t=0 & no external force & no IVD moments)
for i=1:length(length_array)
    U=zeros(90,1); % initializing a vector
    U(1,i)=1;
    Aeq(i,1)=spineaccA(i,0,Stateo,Pango,U,Pa,E_F_verification,Vars,M,E, Pa,Lo_verification,CSA,GL_func,Gr_func,Mm_func,Cm_func,Gm_func,Ma_func,ML_func,IV_func,Ba_func,Rm_func,\ %
dRm_func,origin_func,dorIGIN_func,[],'verification');
end

Jpi=zeros(36,36);
qddP_J=zeros(36,36);
Jpi(1:18,19:36)=eye(18,18);
Jai=zeros(size(M,1),36,36);
qddA=zeros(size(M,1),36,36);
Jpr=zeros(36,36); % no contribution to passive
Jar=zeros(size(M,1),36,36);

for h=1:36
    dStateo=Stateo;
    dStateo(h)=dStateo(h)+eps;
    for i=1:90
        Uo=zeros(size(M,1),1);
        Uo(1,i)=1;
        [qddA,qddP,Jdd,RM_mat(i,1:h),Mv_mat(1,1:h)]=spineacc(i,dStateo,Pango,Pa,0, \ %
Pm,E_F_verification,Vars,Lo_verification,CSA,Lord_Angs,IVD,GL_func,Gr_func,Mm_func, \ %
Cm_func,Gm_func,Ma_func,ML_func,IV_func,Ba_func,Rm_func,\ %
dRm_func,origin_func,dorIGIN_func,[],'verification');
end

179
% Adjust variables necessary for Franklin's function inputs
V = [l0, L1, L2, L3, L4, L5, L6, lb, cent3, g, z, m, m2, m3, m4, m5];
P = zeros(9, 1); P(1:8) = Pango;
S = Stateo; % replaced "Statevar2" with "Stateo"
Pang-P; % replaced "Statevar1" with "P"

% Optimization - Franklin
Aeq_F = [];
Beq_F = [];
len = size(M, 1);
A_F = eye(size(M, 1));
B_F = zeros(len, 1);
U0 = zeros(size(M, 1), 1);
Aeq_F = zeros(18, size(M, 1));
Beq_F = zeros(18, 1);
Beq_F = AccelerationMP(S, S, Pang, Lord_Angs, M, E, Lo_verification, CSA, Uo, Pn, %
EP_verification, Vars, IVD, 'verification');

for i = 1: size(M, 1)
    U0 = zeros(size(M, 1), 1);
    U0(i) = 1;
    Aeq_F(:, i) = AccelerationMA(l0, S, Pang, Lord_Angs, M, E, Lo_verification, CSA, Uo, Pn, %
EP_verification, Vars, IVD);
end

Jp_F = zeros(36, 36);
qddr_F = zeros(36, 36);

Jp_F(1:18, 19:36) = eye(18, 18); % jacobians
Ja_F = zeros(size(M, 1), 36, 36);
qddA_F = zeros(size(M, 1), 36, 36);

for h = 1:36
    Ed = S;
    Sd(h) = Sd(h) + eps;
    for i = 1:size(M, 1)
        U0 = zeros(size(M, 1), 1);
        U0(i) = 1;
        Ja_F(l, 19:36, h) = (AcclerationMA(l0, Sd, P, Lord_Angs, M, E, Lo_verification, CSA, %

\( u_0, \dot{p}_n, \text{BP\_verification, Vars, IVD} = \text{Aeq\_F(:,1)/eps; } \)
\( \text{qddd\_F}(19:36, h) = \text{AccelerationWA(0, Sd, P, Lord\_Angs, M, E, Lo\_verification, } \)
\( \text{CSA, Uo, Pn, BP\_verification, Vars, IVD); } \)
\end{verbatim}
\( \text{end} \)
\( \text{Uo} = \text{zeros(size(M,1),1); } \)
\( \text{qddd\_F}(19:36, h) = \text{AccelerationMP(0, Sd, P, Lord\_Angs, M, E, Lo\_verification, CSA, Uo, Pn, } \)
\( \text{BP\_verification, Vars, IVD, } \text{"verification"}); \)
\( \text{Jp\_F}(19:36, h) = \text{AccelerationMP(0, Sd, P, Lord\_Angs, M, E, Lo\_verification, CSA, Uo, Pn, } \)
\( \text{BP\_verification, Vars, IVD, } \text{"verification"}) + \text{Aeq\_F} / \text{eps; } \)
\end{verbatim}
\( \text{end} \)
\( \text{U\_verification} = 0.5 * \text{zeros(90,1)}; \)
\( \text{Power_F} = \text{costfun}(U\_verification, S, Jp\_F, Ja\_F, Lo\_verification, CSA); \)
\( \text{Power} = \text{costfun}(U\_verification, \text{State}, Jp\_F, Ja\_F, Lo\_verification, CSA); \)
\( \% \text{------ End of "Verification Optimization Variables" Section} \)
\( \% \% 5. \text{Perform Verification Tasks} \)
\( \% \text{Calculate the difference between our output and Franklin's output.} \)
\( [M\_check\_count, \neg, M\_diff, M\_max\_error] = \text{verify}(M(:,1:32), M\_F(:,1:32), 1E^{-3}) \) \( \% \) \text{Verification Task 2.2 \& 2.4} \)
\( [M2\_check\_count, \neg, M2\_diff, M2\_max\_error] = \text{verify}(M(:,1:32), M\_F2(:,1:32), 1E^{-3}) \) \( \% \) \text{Verification Task 2.2 \& 2.4} \)
\( [E\_check\_count, E\_check, E\_diff, E\_max\_error] = \text{verify}(E, E\_F, 1E^{-15}) \) \( \% \) \text{Verification Task 3.2} \)
\( \% \text{for Verification Task 4.1} \)
\( \text{rotation\_matrices} = \text{zeros(size(Rm\_F))}; \)
\( \text{d\_rotation\_matrices} = \text{zeros(size(Rm\_F))}; \)
\( \text{dim\_mat} = \text{size(rotation\_matrices)}; \)
\( j = 1; \)
\( \text{for} \ i = 1: \text{dim\_mat}(1)/3 \)
\( \text{rotation\_matrices}(j:j+2,:) = \text{transpose(rotation\_matrices}(j:j+2,:)); } \)
\( \text{d\_rotation\_matrices}(j:j+2,:) = \text{transpose(d\_rotation\_matrices}(j:j+2,:)); } \)
\( j = j + 3; \)
\( \text{end} \)
\( [Rm\_check\_count, \neg, Rm\_diff, Rm\_max\_error] = \text{verify(rotation\_matricesT, Rm\_F, 1E^{-15}) \% Verification Task 4.1} \)
\( [dRm\_check\_count, \neg, dRm\_diff, dRm\_max\_error] = \text{verify(d\_rotation\_matricesT, dRm\_F, 1E^{-15}) \% Verification Task 4.1} \)
\( \% \% \text{for Verification Task 5.1} \)
\( [Or\_check\_count, \neg, Or\_diff, Or\_max\_error] = \text{verify(origin, Or\_F, 1E^{-15}) \% Verification Task 5.1} \)
\( [dOr\_check\_count, \neg, dOr\_diff, dOr\_max\_error] = \text{verify(d\_origin, dOr\_F, 1E^{-15}) \% Verification Task 5.1} \)
[Ma_check_count, ~,Ma_diff,Ma_max_error]=verify(Ma(:,1:33),Ma_F,1E-15) %Verification Task 6.1

[ML_check_count, ~,ML_diff,ML_max_error]=verify(ML,ML_F,1E-15) %Verification Task 7.1
[MV_check_count, ~,MV_diff,MV_max_error]=verify(MV,MV_F,1E-15) %Verification Task 7.1

[En_check_count, ~,En_diff,En_max_error]=verify(En,En_F(:,1:9),1E-15) %Verification Task 8.1

[Cm_check_count, ~,Cm_diff,Cm_max_error]=verify(Cm,Cm_F(:,1E-15)) %Verification Task 9.1

[Gl_check_count, ~,Gl_diff,Gl_max_error]=verify(Gl,Gl_F,1E-15) %Verification Task 9.4
[Gr_check_count, ~,Gr_diff,Gr_max_error]=verify(Gr,Gr_F,1E-15) %Verification Task 9.4

[Hm_check_count, ~,Hm_diff,Hm_max_error]=verify(Hm,Hm_F,1E-15) %Verification Task 9.2
[Gm_check_count, ~,Gm_diff,Gm_max_error]=verify(Gm,Gm_F,1E-15) %Verification Task 9.2

[PQ_check_count, ~,PQ_diff,PQ_max_error]=verify(PQ,PQ_F,1E-15) %Verification Task 10.1

[Q_check_count, ~,Q_diff,Q_max_error]=verify(Q,Q_F,1E-15) %Verification Task 11.1

[Aeq_check_count, ~,~,Aeq_max_error]=verify(Aeq,Aeq_F,1E-15) %Verification Task 12.1
[Beq_check_count, ~,~,Beq_max_error]=verify(Beq,Beq_F,1E-15) %Verification Task 12.1
[A_check_count, ~,~,A_max_error]=verify(A,A_F,1E-15) %Verification Task 12.1
[B_check_count, ~,~,B_max_error]=verify(B,B_F,1E-15) %Verification Task 12.1

[~,~,qdda_max_error]=verify(qdda(:,1:90,1:36,1),qdda_F(:,1:90,1:36,1),1E-15) %verification Task 12.3
[~,~,qddp_max_error]=verify(qddp(:,1:90,1:36,1),qddp_F(:,1:90,1:36,1),1E-15) %Verification Task 12.3

[~,~,Jn_max_error]=verify(Jn(:,1:90,1:36,1),Jn_F(:,1:90,1:36,1),1E-15) %Verification Task 12.4
[~,~,Jp_max_error]=verify(Jp,Jp_F,1E-15) %Verification Task 12.4

[MF_check_count, ~,MF_diff,MF_max_error]=verify(MF,MF_F,1E-15) %Verification Task 13.1

[Power_check_count, ~,Power_diff,Power_max_error]=verify(Power,Power_F,1E-15) %Verification Task 12.6

% ------ End of "Perform Verification Tasks" Section
% Exporting Verification to Excel Spreadsheet

global date
warning('off','MATLAB:xlswrite:AddSheet');

filename=sprintf('Verification Summary %s.xlsx',date);
filename=replace(filename,:','.''); % cannot have colons in file titles
filename=string(append('C:\Users\Valerie\Jardim\Documents\Model\FINALIZED_CODE_updated\Simulation Runs Verification Output\',
filename));
writematrix(date,filename,'Sheet',1,'Range','A1')

writecell(['Rotation Matrices'],filename,'Sheet',2,'Range','C3')
writematrix(dM_diff, filename, 'Sheet', 2, 'Range', 'C4')

writecell(['Differential Rotation Matrices'], filename, 'Sheet', 2, 'Range', 'G3')
writematrix(dMn_diff, filename, 'Sheet', 2, 'Range', 'G4')

writecell(['Origins'], filename, 'Sheet', 3, 'Range', 'C3')
writematrix(dOr_diff, filename, 'Sheet', 3, 'Range', 'C4')

writecell(['Differential Origins'], filename, 'Sheet', 3, 'Range', 'G3')
writematrix(dOr_diff, filename, 'Sheet', 3, 'Range', 'G4')

writecell(['Rotated Muscle Anatomy (Ma)'], filename, 'Sheet', 4, 'Range', 'C3')
writematrix(Ma_diff, filename, 'Sheet', 4, 'Range', 'C4')

writecell(['Non-Rotated Muscle Anatomy (M)'], filename, 'Sheet', 5, 'Range', 'C3')
writematrix(M2_diff, filename, 'Sheet', 5, 'Range', 'C4')

writecell(['Rotated External Force (Ea)'], filename, 'Sheet', 6, 'Range', 'C3')
writematrix(Ea_diff, filename, 'Sheet', 6, 'Range', 'C4')

writecell(['Non-Rotated External Force (E)'], filename, 'Sheet', 6, 'Range', 'C5')
writematrix(E_diff, filename, 'Sheet', 6, 'Range', 'C6')

writecell(['Gravity Vector (Gm)'], filename, 'Sheet', 7, 'Range', 'C5')
writematrix(Gm_diff, filename, 'Sheet', 7, 'Range', 'C6')

writecell(['Coriolis Vector (Gm)'], filename, 'Sheet', 7, 'Range', 'C8')
writematrix(Gm_diff, filename, 'Sheet', 7, 'Range', 'C9')

writecell(['Mass Matrix (Mm)'], filename, 'Sheet', 7, 'Range', 'C11')
writematrix(Mm_diff, filename, 'Sheet', 7, 'Range', 'C12')

writecell(['(Gl)'], filename, 'Sheet', 8, 'Range', 'C5')
writematrix(GL_diff, filename, 'Sheet', 8, 'Range', 'C6')

writecell(['(Gr)'], filename, 'Sheet', 8, 'Range', 'G5')
writematrix(Gr_diff, filename, 'Sheet', 8, 'Range', 'G6')
writecell('Optimization (A)'), filename, 'Sheet', 9, 'Range', 'C5')
writecell('Optimization (B)'), filename, 'Sheet', 9, 'Range', 'D5')
writecell('Optimization (Aeq)'), filename, 'Sheet', 9, 'Range', 'E5')
writecell('Optimization (beg)'), filename, 'Sheet', 9, 'Range', 'F5')
writecell('Optimization (Jp)'), filename, 'Sheet', 9, 'Range', 'G5')
writecell('Optimization (Ja)'), filename, 'Sheet', 9, 'Range', 'H5')
writecell('Generalized Forces (Q)'), filename, 'Sheet', 10, 'Range', 'C5')
writecell('Muscle Length (ML)'), filename, 'Sheet', 10, 'Range', 'D5')
writecell('Muscle Velocity (MV)'), filename, 'Sheet', 10, 'Range', 'E5')
writecell('Muscle Force (MF)'), filename, 'Sheet', 10, 'Range', 'F5')
writecell('Intervertebral Disc Moments (PRQ)'), filename, 'Sheet', 11, 'Range', 'C5')

% ------ End of "Exporting Verification to Excel Spreadsheet" Section
C.3 *model_parameters.m*: Model Parameters

```matlab
#C.3 model_parameters.m: Model Parameters

%Valerie Jardon
%Department of Bioengineering, University of Kansas
%model_parameters.m
%Last Edited: 4/29/2022
%% Description
% In the model_parameters.m file, the model parameters are defined.
% This file (model_parameters.m) is based on the "Define Simulation Variables", "Intervertebral Disc Parameters",
% "Initialize Position Vectors" and "Physiological Lordosis Angles" sections of the spineSim.m file
% developed by Timothy C. Franklin, published in Appendix B of his thesis (Franklin, 2006, p. 99-100).
% Additionally, this code includes the lordosis angles calculations and incorporates them into the position vectors.

%% Reference:

%% Defining the Model Parameters
% This section is based on the "Define Simulation Variables" section of the spineSim.m file
% developed by Timothy C. Franklin, published in Appendix B of his thesis (Franklin, 2006, p. 99).

L0=0.4241; %body 0 - thoracic and cervical spine (units: m)

% Cholewicki-Defined Spacing in Muscle Geometry
L1_ChSpace=0.03625; %units: m
L2_ChSpace=0.03860; %units: m
L3_ChSpace=0.0379; %units: m
L4_ChSpace=0.0370; %units: m
L5_ChSpace=0.0389; %units: m
ChSpace=[L1_ChSpace; L2_ChSpace; L3_ChSpace; L4_ChSpace; L5_ChSpace];

L1=L1_ChSpace;
L2=L2_ChSpace;
L3=L3_ChSpace;
L4=L4_ChSpace;
L5=L5_ChSpace;

% Mass of bodies
m0=16; %body 0 - thoracic and cervical spine (units: kg)
m1=2; %body 1 - L1 (units: kg)

m2=2; %body 2 - L2 (units: kg)
m3=2; %body 3 - L3 (units: kg)
m4=2; %body 4 - L4 (units: kg)
m5=2; %body 5 - L5 (units: kg)
```

185
l_a = 0.024; % Radius of trunk, ML
l_b = 0.017; % Radius of trunk, AP

cent1 = 0;
cent2 = 0;
cent3 = 0.1067; % Location of L5 pivot in P in the Axis 3 direction

g = 9.8; % Gravity positive

% Creating an array for Model Variables
Vars = [L0, L1, L2, L3, L4, L5, l_a, l_b, cent3, g, m0, m1, m2, m3, m4, m5, cent1, cent2];
%----------------------------------End of Define the Model Parameters Section

% Intervertebral Disc Parameters
% This section is based on the "Intervertebral Disc Parameters" section of the spinesim.m file
% developed by Timothy C. Franklin, published in Appendix B of his thesis (Franklin, 2006, p. 99).
%----------------------------------End of Intervertebral Disc Parameters Section

% Initializing the intervertebral disc parameters array
I_Kp = 50; % Planar
I_Dp = 0.5;
I_Kt = 50; % Twist
I_Dt = 0.5;
IVD = [I_Kp, I_Dp, I_Kt, I_Dt];
%----------------------------------End of Intervertebral Disc Parameters Section

% Initialize Position Vectors
% This section is based on the "Initialize Position Vectors" section of the spinesim.m file
% developed by Timothy C. Franklin, published in Appendix B of his thesis (Franklin, 2006, p. 99).
%----------------------------------End of Initialize Position Vectors Section

% Lordosis Angles
% This section is based on the "Physiological Lordosis Angles" section of the spinesim.m file
% developed by Timothy C. Franklin, published in Appendix B of his thesis (Franklin, 2006, p. 100).

% Franklin's Values
% Panko(1) = deg2rad(-20); % Pelvis Lordosis Angle (units: radians)
% Stateo(1) = deg2rad(-14); % L5 Lordosis Angle (units: radians)
% Stateo(4) = deg2rad(6); % L4 Lordosis Angle (units: radians)
% Stateo(7) = deg2rad(18); % L3 Lordosis Angle (units: radians)
% Stateo(10) = deg2rad(15); % L2 Lordosis Angle (units: radians)
% Stateo(13) = deg2rad(10); % L1 Lordosis Angle (units: radians)
% Stateo(16) = deg2rad(0); % L0 Lordosis Angle (units: radians)
% Normal Lordosis
  Pango(1)=deg2rad(-38); %Pelvis Lordosis Angle (units: radians)
  Stateco(1)=deg2rad(-17.8); %L5 Lordosis Angle (units: radians)
  Stateco(4)=deg2rad(-2); %L4 Lordosis Angle (units: radians)
  Stateco(7)=deg2rad(9.6); %L3 Lordosis Angle (units: radians)
  Stateco(10)=deg2rad(17.5); %L2 Lordosis Angle (units: radians)
  Stateco(13)=deg2rad(20.1); %L1 Lordosis Angle (units: radians)
  Stateco(16)=deg2rad(0); %L0 Lordosis Angle (units: radians)

% Hyperlordosis
  Pango(1)=deg2rad(-38); %Pelvis Lordosis Angle (units: radians)
  Stateco(1)=deg2rad(-8.565); %L5 Lordosis Angle (units: radians)
  Stateco(4)=deg2rad(14.142); %L4 Lordosis Angle (units: radians)
  Stateco(7)=deg2rad(30.962); %L3 Lordosis Angle (units: radians)
  Stateco(10)=deg2rad(42.736); %L2 Lordosis Angle (units: radians)
  Stateco(13)=deg2rad(46.1); %L1 Lordosis Angle (units: radians)
  Stateco(16)=deg2rad(0); %L0 Lordosis Angle (units: radians)

% Hyperlordosis - range of Pesenti values
  Pango(1)=deg2rad(-38); %Pelvis Lordosis Angle (units: radians)
  Stateco(1)=deg2rad(-20.2); %L5 Lordosis Angle (units: radians)
  Stateco(4)=deg2rad(-2.4); %L4 Lordosis Angle (units: radians)
  Stateco(7)=deg2rad(13.4); %L3 Lordosis Angle (units: radians)
  Stateco(10)=deg2rad(33.2); %L2 Lordosis Angle (units: radians)
  Stateco(13)=deg2rad(31); %L1 Lordosis Angle (units: radians)
  Stateco(16)=deg2rad(0); %L0 Lordosis Angle (units: radians)

% Hypo lordosis
  Pango(1)=deg2rad(-38); %Pelvis Lordosis Angle (units: radians)
  Stateco(1)=deg2rad(-26.765); %L5 Lordosis Angle (units: radians)
  Stateco(4)=deg2rad(-18.098); %L4 Lordosis Angle (units: radians)
  Stateco(7)=deg2rad(-11.678); %L3 Lordosis Angle (units: radians)
  Stateco(10)=deg2rad(-1.384); %L2 Lordosis Angle (units: radians)
  Stateco(13)=deg2rad(5.9); %L1 Lordosis Angle (units: radians)
  Stateco(16)=deg2rad(0); %L0 Lordosis Angle (units: radians)

% Hypo lordosis - range of Pesenti values
  Pango(1)=deg2rad(-38); %Pelvis Lordosis Angle (units: radians)

  Stateco(1)=deg2rad(-36); %L5 Lordosis Angle (units: radians)
  Stateco(4)=deg2rad(-34); %L4 Lordosis Angle (units: radians)
  Stateco(7)=deg2rad(-32); %L3 Lordosis Angle (units: radians)
  Stateco(10)=deg2rad(-30); %L2 Lordosis Angle (units: radians)
  Stateco(13)=deg2rad(-28); %L1 Lordosis Angle (units: radians)
  Stateco(16)=deg2rad(-30); %L0 Lordosis Angle (units: radians)

  Lord Ang= [Stateco(15) Stateco(13) Stateco(10) Stateco(7) Stateco(4) Stateco(1) Pango(1)]; %
  %Lordosis Angle Array (L0, L1,...,Pelvis)
  %-----------------------------End of Lordosis Angles Section
% Exporting Model Parameters to Excel Spreadsheet
% Exporting the Model Parameters to the Summary Excel Spreadsheet

if verification_logical==1
    task=2;
else
    task=1;
end

for i=1:task
    if i==1
        filename=sprintf('Model Parameters Summary %s.xlsx',date);
        filename=replace(filename,"\n","\r"'); % cannot have cols in file titles
        filename=string(append(['C:\Users\Valerie\Documents\Model\FINALIZED_CODE_updated\Simulation Runs Model Parameters \
                Summary\',filename]));
        writematrix(date,filename,'Sheet',1,'Range','A1')
    else
        filename=sprintf('Verification Summary %s.xlsx',date);
        filename=replace(filename,"\n","\r"'); % cannot have cols in file titles
        filename=string(append(['C:\Users\Valerie\Documents\Model\FINALIZED_CODE_updated\Simulation Runs Verification Output\',filename]));
        writematrix(date,filename,'Sheet',1,'Range','A1')
    end

    heading={'Body #','Length(m)', 'Mass(kg)', 'Lordosis Angle(rad)'};
    writecell(heading,filename,'Sheet',1,'Range','C3')
    body_str={'L1', 'L2', 'L3', 'L4', 'L5', 'Pelvis'};
    body_num=[0 1 2 3 4 5 6];
    writecell(body_str,filename,'Sheet',1,'Range','B4')
    writematrix(body_num,filename,'Sheet',1,'Range','A4')

    writematrix(VarS(1:6),filename,'Sheet',1,'Range','B4') % Length Array
    writematrix(VarS(11:18),filename,'Sheet',1,'Range','B4') % Mass Array
    writematrix(Lord_Angs,filename,'Sheet',1,'Range','B4') % Mass Array

    writematrix(['cent values'],filename,'Sheet',1,'Range','B12') % cent values
    writematrix([cent1,cent2,cent3],filename,'Sheet',1,'Range','C12') % cent
    writematrix(['Radius ML'],filename,'Sheet',1,'Range','B13') % radius
    writematrix([la,filename,'Sheet',1,'Range','C13']) % radius
    writematrix(['Radius AP'],filename,'Sheet',1,'Range','B14') % radius
    writematrix([lb,filename,'Sheet',1,'Range','C14']) % radius

    heading={'Planar - K', 'Planar - D', 'Twist - K', 'Twist - D'};
    writecell(heading,filename,'Sheet',1,'Range','B16') % IVD
    writematrix(IVD,filename,'Sheet',1,'Range','C16') % IVD
end

%-------------------End of Exporting Model Parameters to Excel Spreadsheet

save('model_parameters.mat',
      'L0', 'L1', 'L2', 'L3', 'L4', 'L5', 'm0', 'mL', 'mZ', 'm3', 'm4', 'm5', 'la', 'lb', 'cent1', 'cent2', 'cent3', 'g', 'IVD', 'VarS', 'L5_ChSpace', 'L6_ChSpace', 'L7_ChSpace', 'L8_ChSpace', 'L9_ChSpace', 'ChSpace')
C.4 *rotationmatrices.mlx*: Determine the Rotation Matrices

Valerie Jardon
Department of Bioengineering, University of Kansas
*rotationmatrices.mlx*

Last Edited: 4/29/2022

**Determining the Rotation Matrix for Each Body**

*Description:* This file (*rotationmatrices.mlx*) can be used to determine the rotation matrix for each body. Each body rotates around Axis 1, then around Axis 2, and lastly around Axis 3. In this file, rotation in the counterclockwise (CCW) direction is considered positive. Lastly, the rotation matrices are compiled into one 21x3 matrix. Note: symbolic variables were used for the symbolic .mat files that were created.

*Notes:*

1) The use of the variable names (such as "Stateo", "Pango", etc.) is based on the work of Timothy C. Franklin (Franklin, 2006).

2) Franklin defines the order of the rotations and explains the use of these body rotations in Chapter 3A of his thesis (Franklin, 2006, p. 9).

*Reference:*


---

**Assigning Angle Values from the State Vectors**

In this section, the values from the vectors Stateo and Pango are assigned as single values to variables. If the angle name begins with "T", then it is the rotation around Axis 1. If an angle begins with "B", then it is rotation around Axis 2. If an angle begins "A", then it is rotation around Axis 3. The second letter in the angle name defines the body. Body 0 is the thoracic/cervical spine region, Body 1 is L1, Body 2 is L2, Body 3 is L3, Body 4 is L4, Body 5 is L5 and Body p is the pelvis.

Example: the angle name "A5" means the angle of rotation around Axis 3 for Body 5.

```matlab
clc

sym Tp Bp Ap T5 B5 A5 T4 B4 A4 T3 B3 A3 T2 B2 A2 T1 B1 A1 T0 B0 A0 % used for creating symbols

Tp=Pango(1);
Bp=Pango(2);
Ap=Pango(3);
T5=Stateo(1);
B5=Stateo(2);
A5=Stateo(3);
```
Defining the Axis Rotations

In this section, the rotations around the axes are defined as follows:

```matlab
% sym = q1 q2 q3
R1=[1 0 0; 0 cos(q1) -sin(q1); 0 sin(q1) cos(q1)]; % Rotation matrix for a rotation around axis
R2=[cos(q2) 0 sin(q2); 0 1 0; -sin(q2) 0 cos(q2)]; % Rotation matrix for a rotation around axis
R3=[cos(q3) -sin(q3) 0; sin(q3) cos(q3) 0; 0 0 1]; % Rotation matrix for a rotation around axis
R123=R1*R2*R3; % Matrix multiplication to determine Body123 rotation matrix

% Pelvis
P_rotate = subs(R123, [q1 q2 q3], [Tp Bp Ap]); % Substituting in rotation angles for the Pelvis

% L5
five_rotate = subs(R123, [q1 q2 q3], [T5 B5 A5]); % Substituting in rotation angles for L5

% L4
four_rotate = subs(R123, [q1 q2 q3], [T4 B4 A4]);

% L3
three_rotate = subs(R123, [q1 q2 q3], [T3 B3 A3]);

% L2
two_rotate = subs(R123, [q1 q2 q3], [T2 B2 A2]);

% L1
one_rotate = subs(R123, [q1 q2 q3], [T1 B1 A1]);

% Thoracic/Cervical Region
zero_rotate = subs(R123, [q1 q2 q3], [T0 B0 A0]);
```
Compiling the rotation matrices for each body into a single 21x3 matrix
In this section, each resulting rotation matrix for each body is compiled into a 21x3 rotation matrix.

```matlab
rotation_matrices=zeros(21,3); %Initializing the 21x3 matrix
rotation_matrices=sym(rotation_matrices); %used for symbolic output

rotation_matrices(1:3,:)=P_rotate;
% P_rotate=double(P_rotate); %double precision function not used with symbolic
rotation_matrices(4:6,:)=five_rotate;
% five_rotate=double(five_rotate);
rotation_matrices(7:9,:)=four_rotate;
% four_rotate=double(four_rotate);
rotation_matrices(10:12,:)=three_rotate;
% three_rotate=double(three_rotate);
rotation_matrices(13:15,:)=two_rotate;
% two_rotate=double(two_rotate);
rotation_matrices(16:18,:)=one_rotate;
% one_rotate=double(one_rotate);
rotation_matrices(19:21,:)=zero_rotate;
% zero_rotate=double(zero_rotate);

% rotation_matrices=double(rotation_matrices);
```

Determining the Rotation Matrices - Differential
Assigning Angle Values from the State Vectors - Differential
In this section, the values from the vectors Stateo and Pango are assigned as single values to variables. See section "Assigning Angle Values from the State Vectors" for more information about naming procedure and meaning.

```matlab
syms DTp DBp DAp DT5 DB5 DA5 DT4 DB4 DA4 DT3 DB3 DA3 DT2 DB2 DA2 DT1 DB1 DA1 DT0 DB0 DA0 %used

% DTp=Pango(4);
% DBp=Pango(5);
% DAp=Pango(6);
% DT5=Stateo(19);
% DB5=Stateo(20);
% DA5=Stateo(21);
```
### Defining the Axis Rotations - Differential

In this section, the total differential is calculated and the rotation matrices are determined. Recall the equation for the total differential for functions with three variables:

\[
dw = \frac{\partial w}{\partial x} \, dx + \frac{\partial w}{\partial y} \, dy + \frac{\partial w}{\partial z} \, dz
\]

```matlab
syms DAngT DAngB DAngA
dR123 = diff(R123, q1)*DAngT + diff(R123, q2)*DAngB + diff(R123, q3)*DAngA; % calculating the total diff

% Pelvis
d_P_rotate = subs(dR123, [q1 q2 q3 DAngT DAngB DAngA], [Tp Bp Ap Dtp Btp Atp]);

% 5
d_five_rotate = subs(dR123, [q1 q2 q3 DAngT DAngB DAngA], [T5 B5 A5 DT5 DB5 DA5]);

% 4
d_four_rotate = subs(dR123, [q1 q2 q3 DAngT DAngB DAngA], [T4 B4 A4 DT4 DB4 DA4]);

% 3
d_three_rotate = subs(dR123, [q1 q2 q3 DAngT DAngB DAngA], [T3 B3 A3 DT3 DB3 DA3]);

% 2
d_two_rotate = subs(dR123, [q1 q2 q3 DAngT DAngB DAngA], [T2 B2 A2 DT2 DB2 DA2]);

% 1
d_one_rotate = subs(dR123, [q1 q2 q3 DAngT DAngB DAngA], [T1 B1 A1 DT1 DB1 DA1]);

% Thoracic/Cervical Region
```
Compiling the rotation matrices for each body into a single 21x3 matrix

In this section, each resulting rotation matrix for each body is compiled into a 21x3 rotation matrix.

```matlab
d_rotation_matrices=zeros(21,3); %initializing the 21x3 matrix
d_rotation_matrices=sym(d_rotation_matrices); %used for symbolic output

% d_rotation_matrices(1:3,:)=d_P_rotate;
% d_P_rotate=double(d_P_rotate); %double precision function not used with symbolic
% d_rotation_matrices(4:6,:)=d_five_rotate;
% d_five_rotate=double(d_five_rotate);
% d_rotation_matrices(7:9,:)=d_four_rotate;
% d_four_rotate=double(d_four_rotate);
% d_rotation_matrices(10:12,:)=d_three_rotate;
% d_three_rotate=double(d_three_rotate);
% d_rotation_matrices(13:15,:)=d_two_rotate;
% d_two_rotate=double(d_two_rotate);
% d_rotation_matrices(16:18,:)=d_one_rotate;
% d_one_rotate=double(d_one_rotate);
% d_rotation_matrices(19:21,:)=d_zero_rotate;
% d_zero_rotate=double(d_zero_rotate);

% d_rotation_matrices=double(d_rotation_matrices);
```

Create .mat file for output

```matlab
save('rotation_matrices_sym_Pto0.mat','rotation_matrices','d_rotation_matrices')

%adjusting the order of rows
rearrange_mat=zeros(size(rotation_matrices)); drearrange_mat=zeros(size(rotation_matrices));
rearrange_mat=sym(rearrange_mat); drearrange_mat=sym(drearrange_mat);

j=1;
for i=1: numel(rotation_matrices)/9
    rearrange_mat(j:j+2,:)=rotation_matrices(((end-j+1):end-j+1,:));
    drearrange_mat(j:j+2,:)=d_rotation_matrices(((end-j+1):end-j+1,:));
    j=j+3;
end
rotation_matrices=rearrange_mat;
d_rotation_matrices=drearrange_mat;

save('rotation_matrices_sym_0toP.mat','rotation_matrices','d_rotation_matrices')
save('rotation_matrices_sym_separate','zero Rotate','one Rotate','two Rotate','three Rotate','
C.5  *origin_body.mlx*: Define origins of the bodies

Valerie Jardon
Department of Bioengineering, University of Kansas
*origin_body.mlx*

Last edited: 4/29/2022

This file (*origin_body.mlx*) is based on the Mathematica Code developed by Timothy C. Franklin, published in Appendix A of his thesis, specifically sections "Define Inertial Coordinate System", "Define Body Fixed (c) Coordinate Systems..." and "Define Origin of Each Body" (Franklin, 2006, p. 78-79).

Determining the Origin of Each Body

Description: This function is used to determine the origin of each body based on the rotation matrices and geometry of the vertebrae.

Notes:

1) The use of the variable names (such as "Orp", "Orf", etc.) is based on the work of Timothy C. Franklin (Franklin, 2006).

Reference:


```
clear; clc
syms cent1 cent2 cent3 g L0 L1 L2 L3 L4 L5 la lb m0 m1 m2 m3 m4 m5

%load rotation matrices
load('rotation_matrices_sym_separate')
```

Defining Global Coordinate System

This section is based on the "Define Inertial Coordinate System" section of the Mathematica file developed by Timothy C. Franklin, published in Appendix A of his thesis (Franklin, 2006, p. 78).

```
n1=[1 0 0]'; %Axis 1
n2=[0 1 0]'; %Axis 2
n3=[0 0 1]'; %Axis 3
```

End of "Defining Global Coordinate System" Section
Define Body Coordinate Systems

This section is based on the "Define Body Fixed (c) Coordinate Systems..." section of the Mathematica file developed by Timothy C. Franklin, published in Appendix A of his thesis (Franklin, 2006, p. 78).

%Pelvis

\[
\begin{align*}
\text{cp\_Axis1} &= \text{P\_rotate}^*\text{n1}; \ %\text{Axis 1 - after rotation} \\
\text{cp\_Axis2} &= \text{P\_rotate}^*\text{n2}; \ %\text{Axis 2 - after rotation} \\
\text{cp\_Axis3} &= \text{P\_rotate}^*\text{n3}; \ %\text{Axis 3 - after rotation} \\
\text{cl5\_Axis3} &= \text{five\_rotate}^*\text{n3}; \ %\text{L5 Axis 3 - after rotation} \\
\text{cl4\_Axis3} &= \text{four\_rotate}^*\text{n3}; \ %\text{L4 Axis 3 - after rotation} \\
\text{cl3\_Axis3} &= \text{three\_rotate}^*\text{n3}; \ %\text{L3 Axis 3 - after rotation} \\
\text{cl2\_Axis3} &= \text{two\_rotate}^*\text{n3}; \ %\text{L2 Axis 3 - after rotation} \\
\text{cl1\_Axis3} &= \text{one\_rotate}^*\text{n3}; \ %\text{L1 Axis 3 - after rotation} \\
\text{cl0\_Axis3} &= \text{zero\_rotate}^*\text{n3}; \ %\text{L0 Axis 3 - after rotation}
\end{align*}
\]

Determine Origin of Each Body

This section is based on the "Define Origin of Each Body" section of the Mathematica file developed by Timothy C. Franklin, published in Appendix A of his thesis (Franklin, 2006, p. 79).

In this section of code, we start by determining the origin of the pelvis, which is positioned at (0,0,0). cent3 is the location of the L5 pivot in body p (pelvis) in the Axis 3 direction. Using the known origin of the pelvis and cent3, we can determine the origin of body 5 (Or5).

Using the known origin of the previous body in the spinal column, the previous body's coordinate system and the height of the previous vertebra, we can determine the origin of the current body, as shown below.

\[
\begin{align*}
\text{Orp} &= \text{8}\,*\text{n1}; \ %\text{pelvis origin} \\
\text{Or5} &= \text{Orp} + \text{cent1}\,*\text{cp\_Axis1} + \text{cent2}\,*\text{cp\_Axis2} + \text{cent3}\,*\text{cp\_Axis3}; \ %\text{L5 origin} \\
\text{Or4} &= \text{Or5} + \text{L5}\,*\text{cl5\_Axis3}; \ %\text{L4 origin} \\
\text{Or3} &= \text{Or4} + \text{L4}\,*\text{cl4\_Axis3}; \ %\text{L3 origin} \\
\text{Or2} &= \text{Or3} + \text{L3}\,*\text{cl3\_Axis3}; \ %\text{L2 origin} \\
\text{Or1} &= \text{Or2} + \text{L2}\,*\text{cl2\_Axis3}; \ %\text{L1 origin} \\
\text{Or0} &= \text{Or1} + \text{L1}\,*\text{cl1\_Axis3}; \ %\text{L0 origin}
\end{align*}
\]
Define Body Coordinate Systems - Differential

Based on the methodology used previously in the "Define Body Coordinate Systems" section.

\[
\begin{align*}
%Pelvis \\
\text{cd\_p\_Axis1=cd\_p\_rotate\_n1; \%Axis 1 - after rotation} \\
\text{cd\_p\_Axis2=cd\_p\_rotate\_n2; \%Axis 2 - after rotation} \\
\text{cd\_p\_Axis3=cd\_p\_rotate\_n3; \%Axis 3 - after rotation} \\
\text{cd\_L5\_Axis3=d\_five\_rotate\_n3; \%L5 Axis 3 - after rotation} \\
\text{cd\_L4\_Axis3=d\_four\_rotate\_n3; \%L4 Axis 3 - after rotation} \\
\text{cd\_L3\_Axis3=d\_three\_rotate\_n3; \%L3 Axis 3 - after rotation} \\
\text{cd\_L2\_Axis3=d\_two\_rotate\_n3; \%L2 Axis 3 - after rotation} \\
\text{cd\_L1\_Axis3=d\_one\_rotate\_n3; \%L1 Axis 3 - after rotation} \\
\text{cd\_L0\_Axis3=d\_zero\_rotate\_n3; \%L0 Axis 3 - after rotation}
\end{align*}
\]

----------------------------------------------------------------------------------------------------------------------------------------End of "Define Body Coordinate Systems - Differential" Section

Determine Origin of Each Body - Differential

Based on the methodology used previously in the "Determine Origin of Each Body" section.

\[
\begin{align*}
\text{d\_Orp=8*n1; \%pelvis origin} \\
\text{d\_Or5=d\_Orp+cent1*cd\_p\_Axis1+cent2*cd\_p\_Axis2+cent3*cd\_p\_Axis3; \%L5 origin} \\
\text{d\_Or4=d\_Or5+L5*cd\_L5\_Axis3; \%L4 origin} \\
\text{d\_Or3=d\_Or4+L4*cd\_L4\_Axis3; \%L3 origin} \\
\text{d\_Or2=d\_Or3+L3*cd\_L3\_Axis3; \%L2 origin} \\
\text{d\_Or1=d\_Or2+L2*cd\_L2\_Axis3; \%L1 origin} \\
\text{d\_Or0=d\_Or1+L1*cd\_L1\_Axis3; \%L0 origin}
\end{align*}
\]

----------------------------------------------------------------------------------------------------------------------------------------End of "Determine Origin of Each Body - Differential" Section

Save output in .mat file

\[
\begin{align*}
\text{origin=zeros(7,3); origin=sym(origin);} \\
\text{origin=[Or0'; Or1'; Or2'; Or3'; Or4'; Or5'; Orp'];} \\
\text{d\_origin=[d\_Or0'; d\_Or1'; d\_Or2'; d\_Or3'; d\_Or4'; d\_Or5'; d\_Orp'];} \\
\text{save('origin\_sym\_pto8.mat', 'origin', 'd\_origin')} \\
\text{origin=flipud(origin); d\_origin=flipud(d\_origin); \%rearrange row order} \\
\text{save('origin\_sym\_pto8.mat', 'origin', 'd\_origin')} \\
\text{Or0=Or0'; Or1=Or1'; Or2=Or2'; Or3=Or3'; Or4=Or4'; Or5=Or5'; Orp=Orp'} \\
\text{d\_Or0=d\_Or0'; d\_Or1=d\_Or1'; d\_Or2=d\_Or2'; d\_Or3=d\_Or3'; d\_Or4=d\_Or4'; d\_Or5=d\_Or5'; d\_Orp=d\_Orp} \\
\text{save('origin\_sym\_separate.mat', 'Or0', 'Or1', 'Or2', 'Or3', 'Or4', 'Or5', 'Orp', 'd\_Or0', 'd\_Or1', 'd\_Or2', 'd\_Or3', 'd\_Or4', 'd\_Or5', 'd\_Orp')}
\end{align*}
\]
C.6 *transform_Cholewicki.mlx*: Import and Transform Skeletal Geometry

Valerie Jarson
Department of Bioengineering, University of Kansas
(transform_Cholewicki.mlx)
Last Edited: 10/24/2021

Transforming Cholewicki's Anatomy

**Description:** This code is used to transform the skeletal geometry provided in Appendix A of Cholewicki and McGill (Cholewicki and McGill, 1996, pp. 10-13).

From Franklin's coordinates provided in Table Ap A.2 and Table Ap A.3 (Franklin, 2006, p. 89-90), we could determine that Franklin rotated and translated the coordinates from Cholewicki and McGill to determine the body-fixed coordinates.

The following methodology is not provided by Franklin, but was determined from the coordinates provided by Franklin.

**References:**


---

```matlab
clear all;clc

% Load the muscle file
[msg, raw]=xlsread('musclefile_cholewicki.xlsx',1); % Load sheet 1
[msg, raw]=xlsread('musclefile_cholewicki.xlsx',2); % Load sheet 2

% Preloading data
% Converting from cell to string
% musclefile=raw(3:end,:); musclefile=str(musclefile);
points=raw(3:end,1);
points=points(:,:);
raw2=raw(3:end,:);

% Define Q1 Q2 Q3
s1=[1 0 0; 1 0 0; 1 0 0];
s2=[1 0 0; 1 0 0; 1 0 0];
s3=[1 0 0; 1 0 0; 1 0 0];

% Rotation matrix for a rotation around axis 1; + CCM rotation
R1=s1*[1 2 3][1 2 3];

% Rotation matrix for a rotation around axis 2; + CCM rotation
R2=s2*[1 2 3][1 2 3];

% Rotation matrix for a rotation around axis 3; + CCM rotation
R3=s3*[1 2 3][1 2 3];

% Convert coordinates to body-fixed coordinates
pt_transform=zeros(size(points));
for i=1:size(pt_transform,1)
    pt_transform(i,:)=pt_transform(i,:)*R3;
end

% For the left side muscle anatomy (based on the right)
for i=122:size(pt_transform,1)
    pt_transform(i,:)=[-1*pt_transform(i-122,1),pt_transform(i-122,2),pt_transform(i-122,3)];
end

% Connection points - check
connect_vec=zeros(14,3);
for i=1:10,36,61,73,85,96
    connect_vec(i,:)=pt_transform(i,:);
    connect_vec(i,:)=pt_transform(i,:);
    j=j+2;
end
```
function [pt_transform]=transform_anatomy(pt_transform,raw2,points,R123,i)
sys (a1 q2 q3)
    if strcmp(raw2(1,2), 'p')
        translate_pt=[0 points(1,2) points(1,3)];
        pt_transform_pt=points(1,:)-translate_pt;
        R123=subs(R123,[a1,q2,a3], [deg2rad(-1.393310999999),0,0]);
        pt_transform(1,:)=R123*pt_transform_pt;
    elseif double(string(raw2(1,2)))==5
        translate_pt=[0 points(20,2) points(20,3)];
        pt_transform_pt=points(1,:)-translate_pt;
        R123=subs(R123,[a1,q2,a3], [deg2rad(21.9891381789114),0,0]);
        pt_transform(1,:)=R123*pt_transform_pt;
    elseif double(string(raw2(1,2)))==4
        translate_pt=[0 points(36,2) points(36,3)];
        pt_transform(1,:)-points(1,:)-translate_pt;
        % not need to be rotated
        elseif double(string(raw2(1,2)))==3
            translate_pt=[0 points(61,2) points(61,3)];
            pt_transform_pt=points(1,:)-translate_pt;
            R123=subs(R123,[a1,q2,a3], [deg2rad(-22.9991700089026),0,0]);
            pt_transform(1,:)=R123*pt_transform_pt;
        elseif double(string(raw2(1,2)))==2
            translate_pt=[0 points(73,2) points(73,3)];
            pt_transform_pt=points(1,:)-translate_pt;
            R123=subs(R123,[a1,q2,a3], [deg2rad(-9.5575750999997),0,0]);
            pt_transform(1,:)=R123*pt_transform_pt;
        elseif double(string(raw2(1,2)))==1
            translate_pt=[0 points(85,2) points(85,3)];
            pt_transform_pt=points(1,:)-translate_pt;
            R123=subs(R123,[a1,q2,a3], [deg2rad(24.4533562999993),0,0]);
            pt_transform(1,:)=R123*pt_transform_pt;
        elseif double(string(raw2(1,2)))==0
            translate_pt=[0 points(96,2) points(96,3)];
            pt_transform_pt=points(1,:)-translate_pt;
            R123=subs(R123,[a1,q2,a3], [deg2rad(0.52866999996543),0,0]);
            pt_transform(1,:)=R123*pt_transform_pt;
        end
    end
end
C.7 *load_musclefile.mlx*: Import Muscle Anatomy

Valerie Jardon
Department of Bioengineering, University of Kansas

%load_musclefile.ml

% Last Edited: 4/29/2022

% This function (load_musclefile.m) is based on the muscinput.m and muscmat.m codes developed by Timothy C. Franklin, Virginia Polytechnic Institute and State University. The muscinput.m and muscmat.m codes were obtained through means of personal communication (Michael Madigan, personal communication, July 21, 2008).

% Load the Muscle File

% Description: This function imports the muscle paths and geometry needed for the model.

% The muscle attachment points and skeletal geometry are based on Cholewicki et al. (Cholewicki and McGill, 1996, p. 12-13). The built-in lordosis angle in Cholewicki's data has been removed so that the connection points of the vertebrae align along the vertical axis. This adjusted muscle data is located in the 'transformed Cholewicki.mat' file.

% Sheet 1 of the Excel spreadsheet ("musclefile_Cholewicki.xlsx") details the muscle paths for the muscles. A row is allocated for each muscle path. The attachment points are listed as strings, meaning it is text. These strings are then replaced by the adjusted muscle data.

% References:


function [N]=load_musclefile()
[N,...,raw]=xlsread('musclefile_Cholewicki.xlsx',1); %load sheet 1; muscle path of each muscle
load('transformed_Cholewicki.mat'); %load muscle points; built in lordosis angle removed ("rotated out" of Cholewicki data...) vertebrae connections pts aligned along vertical axis

%Trimming raw and raw2 to remove the column headings
%Converting from cell to string
musclepath=raw(3:92,:); musclepath=str2num(musclepath);
musclepoints=str2num(NCheckedChangeListener);
% Adjusting the pelvis origin to be aligned with the other connections
% points; Cholewicki has PELX(1) defined as the hip joint which we can
% adjust
musclepoints\{1,3\} = "0";
musclepoints\{\text{size}(musclepoints,1)/2-1,3\} = "0";

% Currently in a row, the first point is the origin, the second point
% is the terminal, and the following points are the nodes.
% In this section, the points for each muscle are rearranged
% so that the first point is the origin, the nodes follow, and the last
% point is the terminal.
% Additionally, the number of attachment points for each muscle is
% determined and stored in the first column of matrix M.

musclepath\text{trim} = musclepath(:,2:9); % trimming the data to not include muscle name,
rest length or CSA
musclepathR = musclepath\text{trim}; % initializing the matrix for reordered muscle paths (in
order from origin to terminal)

for i=1:size(musclepath,1) % an iteration for each muscle
    no_points = sum(~missing(musclepath\text{trim}(i,:))); % count the number of
attachment points in a row (not counting the <missing> elements)

    for j=1:no_points
        if j==1
            musclepathR(i,j) = musclepath\text{trim}(i,1);
        elseif j==no_points
            musclepathR(i,j) = musclepath\text{trim}(i,2);
        else
            musclepathR(i,j) = musclepath\text{trim}(i,j+1);
        end
    end

    M(i,1) = no_points; M = string(M); % storing the number of attachment points for the
muscle in column 1 of matrix M
end

% Creating the M Matrix
j=0; % initializing variable
for h=[2:4:32]
    j=j+1;
    M(:,j) = musclepathR(:,j); % attachment point
end
M(:,34) = musclepath(:,10); % CSA (cm^2)

% Redefining pelvis body number to 6 instead of 'p'
musclepoints = strrep(musclepoints, 'p', '6');
In this section, the strings of the attachment points in matrix M are replaced with the coordinates of these attachment points.

for k=2:4:size(M,2)-3] % these are the columns in matrix M with the attachment points listed
   for j=1:size(M,1) % for each of the 90 muscles
      str=M(j,k);
      for i=1:size(musclepoints,1) % for each of the attachment points
         if musclepoints(i,1)==str
            M(j,k+3)=str2double(musclepoints(i,2:5));
         end
      end
   end
end

X=double(M); % convert to data type double

% converting coordinates from cm to meters
for k=3:size(M,2)
   for j=1:90
      M(j,k+2)=M(j,k+2)/100;
   end
end

% Replacing the "NaN"s in the matrix with zeros
clear index NaN
index NaN=isnan(M); M(index NaN)=0;

%% Verification Task

% Check that the right side and left side coordinates are the same (the only difference should be what side of the body the point is on)
sum_diff=0;
musclepoints trim=double(musclepoints(:,3:5));
musclepoints trim=[size(musclepoints,1)/2+1:end,1]-1*musclepoints trim([size(W (musclepoints,1)/2+1:end,1);

for k=1:size(musclepoints,1)/2)
   vect_diff=musclepoints trim(k,:)-musclepoints trim(k+size(musclepoints,1)/2,:);
   sum_diff=sum_diff+sqrt(sum(vect_diff.^2));
end

fprintf('Verification Task: The right and left side anatomy has a total error of %3.16e\n',sum_diff)
end
C.8 load_forcefile.mlx: Import External Force File

Valerie Jardon
Department of Bioengineering, University of Kansas
load_forcefile.mlx
Last Edited: 4/29/2022

This file (load_forcefile.mlx) is based on the forceinput.m and foromat.m codes developed by Timothy C. Franklin, Virginia Polytechnic Institute and State University. The forcinput.m and foromat.m codes were obtained through means of personal communication (Michael Madigan, personal communication, July 26, 2008).

Reference:

Load the External Force File

**Description:** Using this function (load_forcefile), the external force data can be imported from the specified Excel spreadsheet ("LoadLift.xls"). Sheet 1 details the origin and terminal points for the force. Sheet 2 gives the coordinates and body of the origin and terminal points.

```matlab
function [E]=load_forcefile()
[-,txt,~]=xlsread('LoadLift.xls',1); %load sheet 1
[-,~,raw2]=xlsread('LoadLift.xls',2); %load sheet 2

%Trimming txt and raw2 to remove the column headings
%Converting from cell to string
forcefile=txt(3,:); forcefile=string(forcefile);
points=raw2(3:4,:); points=string(points);

%Redefining Newtonian body number to 7 instead of 'n'
points=strrep(points,'n','7');

% Creating the E array
E(:,2)=forcefile(:,2); %origin
E(:,6)=forcefile(:,3); %terminal
```

In this section, the strings of the points in matrix E are replaced with the coordinates of these points.

```matlab
for k=[2,6] %these are the columns in array E with the points listed as strings
    str=E(1,k);
    for i=1:2
        if points(i,1)==str
            E(1,k+k*3)=str2double(points(i,2:5));
        end
    end
```
In this section, the number of points for each force will be calculated and stored in the first column of array E. Each point has 4 elements due to the coordinates and body number associated with the point.

```matlab
E(1,1)=0; %initializing the first column of the E matrix; this column will be used to indi
num_points=size(E,2)/4; %calculating the number of columns of E and dividing by 4 to deter
num_points=floor(num_points);
E(1,1)=num_points;
end
```
C.9  \textit{rotate\_anatomy\_sym.mlx}: Transform Muscle File, Calculate Muscle & Velocity

Valerie Jardon

Department of Bioengineering, University of Kansas

\textit{rotate\_anatomy\_sym.mlx}

Last Edited: 4/29/2022

This file (\textit{rotate\_anatomy\_sym.mlx}) is based on the \textit{analys1.c} and \textit{kinematics.c.c} codes developed by Timothy C. Franklin, Virginia Polytechnic Institute and State University. The \textit{analys1.c} and \textit{kinematics.c.c} codes were obtained through means of personal communication (Michael Madigan, personal communication, July 26, 2008).

Reference:


Rotating the Muscle Anatomy

Description: This code is used to calculate the muscle rotated anatomy (Ma), the muscle point velocities (Mv), the muscle lengths (ML) and muscle velocities (MV). This code uses the symbolic rotation matrices and origins to yield a symbolic output for the requested parameters.

This section of code was used for the verification process:

```matlab
clear; clc
% load('Stateo_Pango_Rand.mat')
% load('Model_Parameters_Verify.mat')
% Vars=[L0,L1,L2,L3,L4,L5,L6,Lb,cent3,g,m0,m1,m2,m3,m4,m5,cent1,cent2];
```

---

Initializing the Ma Matrix

The methodology used to determine the muscle anatomy matrix (Ma) was based on Franklin's work. This can be viewed on page 16 of Franklin's \textit{analys1.c} file in the \textit{Kinematics()} function.

```matlab
M=load_musclefile();
Ma=zeros(size(M));
Ma(:,1)=M(:,1);  %number of points
Ma(:,2)=M(:,2);  %origin body
Ma(:,6)=M(:,6);  %second point body
Ma(:,10)=M(:,10);  %third point body
Ma(:,14)=M(:,14);  %fourth point body
Ma(:,18)=M(:,18);  %fifth point body
Ma(:,22)=M(:,22);  %sixth point body
Ma(:,26)=M(:,26);
```
Rotating and Translating the Muscle Anatomy

In this section, the muscle attachment points are rotated and translated based on the origin and the rotation matrix of the body it is associated with.

The methodology used to determine the muscle anatomy matrix (Ma) was based on Franklin's work. This can be viewed on page 16 of Franklin's analys1.c file in the Kinematics() function.

```
%M Rotate the Muscle Anatomy
for i=1:90
    num_pts=M(i,1); %number of muscle attachment points
    Ma=point_transform(Ma,Ma,num_pts,i);
end
```

Calculate the Length of Each Muscle

In this section, the length of each muscle is calculated. This is achieved by summing the length of the vectors in the muscle path.

For example, for a muscle with three attachment points, we would find the length of the position vector between the origin and node 1, and the length of the position vector between node 1 and the terminal. Then we could sum the length of both of these vectors to yield the total muscle length.

General Procedure:

First, we must determine the position vector between two muscle attachment points. Then we calculate the magnitude of this vector which is representative of the vector's length. This procedure is used for all muscle attachment points in the order that they appear in the muscle path, starting with muscle origin, then the nodes and finally the terminal point. Then the vector lengths can be summed to determine the total length of the muscle.

The methodology used to determine the lengths of the muscles was based on Franklin's work. This can be viewed on pages 4-5 of Franklin's analys1.c file in the MuscProps() function.

```
%ML=sym('ML');
for i=1:90
    num_pts=Ma(i,1); %number of muscle attachment points
    ML(i,1)=musc_length(Ma,i,num_pts);
end
```

---End of "Calculate the Length of Each Muscle" section
Initializing the Mv Matrix

The methodology used to determine the muscle point velocity matrix (Mv) was based on Franklin's work. This can be viewed on page 16 of Franklin's analysis file in the Kinematics() function.

\[
\text{Mv} = \text{zeros(size(M))};
\]
\[
\text{Mv}((:,1)) = \text{M}((:,1)); \quad \% \text{number of points}
\]
\[
\text{Mv}((:,2)) = \text{M}((:,2)); \quad \% \text{origin body}
\]
\[
\text{Mv}((:,3)) = \text{M}((:,6)); \quad \% \text{second point body}
\]
\[
\text{Mv}((:,4)) = \text{M}((:,10)); \quad \% \text{third point body}
\]
\[
\text{Mv}((:,5)) = \text{M}((:,14)); \quad \% \text{fourth point body}
\]
\[
\text{Mv}((:,6)) = \text{M}((:,18)); \quad \% \text{fifth point body}
\]
\[
\text{Mv}((:,7)) = \text{M}((:,22)); \quad \% \text{sixth point body}
\]
\[
\text{Mv}((:,8)) = \text{M}((:,26));
\]
\[
\text{Mv}((:,9)) = \text{M}((:,30));
\]
\[
\text{Mv}((:,10)) = \text{M}((:,34)); \quad \% \text{CSA Mv}
\]
\[
\text{Mv} = \text{sym(Mv)};
\]

End of "Initializing the Mv Matrix" section

Calculating the Linear Velocity of the Muscle Attachment Points (Mv)

In this section, we calculate the linear velocity of the muscle attachment points for each muscle.

The methodology used to determine the muscle point velocity matrix (Mv) was based on Franklin's work. This can be viewed on page 16 of Franklin's analysis file in the Kinematics() function.

\[
\text{for } \text{i}=1:90
\]
\[
\quad \text{num}_\text{pts} = \text{M}(1,1); \quad \% \text{number of muscle attachment points}
\]
\[
\quad \text{Mv} = \text{point_transform_v(M, Mv, num}\_\text{pts}, \text{i});
\]
\[
\text{end}
\]

End of "Calculating the Linear Velocity of the Muscle Attachment Points (Mv)" section

Calculate the Muscle Velocity

In this section, the velocity of each muscle is calculated. Like the muscle length calculations, this involves using muscle segments, which are based on the muscle attachment points, to perform our calculations.

General Procedure:

First, the velocity vector and position vector between two muscle attachment points must be determined. The position unit vector can be calculated. The dot product of the velocity vector and the position unit vector is used to determine the magnitude of the velocity vector in the direction of the position unit vector. This procedure is used for all muscle attachment points in the order that they appear in the muscle path, starting with muscle...
origin, then the nodes and finally the terminal point. The scalar result of the dot product can be summed for each segment of the muscle path to determine the total velocity of the muscle.

The methodology used to determine the velocities of the muscles was based on Franklin's work. This can be viewed on pages 4-5 of Franklin's analysis file in the MuscProps() function.

```matlab
MV=sym('MV');
for i=1:100
    num_pts=M(i,1); % number of muscle attachment points
    MV(i,1)=musc_vel(MV,Ma,i,num_pts);
end
```

End of "Calculate the Muscle Velocity" section

Save output in .mat file

```matlab
save('rotate_anatomy_sym_MaMLMV.mat','Ma','MV','ML')
```

```matlab
function [Ma]=point_transform(M,Ma,num_pts,i)
load('rotation_matrices_sym_separate')
load('origin_sym_separate.mat')
for n=0:num_pts-1
    if M(i,2+4*n)==6 % pelvis
        Ma(i,3+4*n+5+4*n)=P_rotate*M(i,3+4*n+5+4*n)+0*0;
    elseif M(i,2+4*n)==5 % body five
        Ma(i,3+4*n+5+4*n)=five_rotate*M(i,3+4*n+5+4*n)+0*5;
    elseif M(i,2+4*n)==4 % body four
        Ma(i,3+4*n+5+4*n)=four_rotate*M(i,3+4*n+5+4*n)+0*4;
    elseif M(i,2+4*n)==3 % body three
        Ma(i,3+4*n+5+4*n)=three_rotate*M(i,3+4*n+5+4*n)+0*3;
    elseif M(i,2+4*n)==2 % body two
        Ma(i,3+4*n+5+4*n)=two_rotate*M(i,3+4*n+5+4*n)+0*2;
    elseif M(i,2+4*n)==1 % body one
        Ma(i,3+4*n+5+4*n)=one_rotate*M(i,3+4*n+5+4*n)+0*1;
    elseif M(i,2+4*n)==0 % body zero
        Ma(i,3+4*n+5+4*n)=zero_rotate*M(i,3+4*n+5+4*n)+0*0;
end
end
```

```matlab
function [ML]=musc_length(Ma,i,num_pts)
ML=0; ML=sym(ML);
for n=0:num_pts-2
    ML_coords=Ma(i,3+4*n+5+4*n)-Ma(i,3+4*(n+1)+5+4*(n+1)); % for the distance formula
    ML=ML+sqrt(sum(ML_coords.^2)); % distance formula
end
```

```matlab
function [MV]=point_transform_v(M,MV,num_pts,i)
load('rotation_matrices_sym_separate')
load('origin_sym_separate.mat')

for n=0:num_pts-1
    if M(i,2+4*n)==6 %pelvis
        Mv(i,3+4*n:5+4*n) = d_P_rotate*M(i,3+4*n:5+4*n)' + d_Orp';
    elseif M(i,2+4*n)==5 %body five
        Mv(i,3+4*n:5+4*n) = d_five_rotate*M(i,3+4*n:5+4*n)' + d_Or5';
    elseif M(i,2+4*n)==4 %body four
        Mv(i,3+4*n:5+4*n) = d_four_rotate*M(i,3+4*n:5+4*n)' + d_Or4';
    elseif M(i,2+4*n)==3 %body three
        Mv(i,3+4*n:5+4*n) = d_three_rotate*M(i,3+4*n:5+4*n)' + d_Or3';
    elseif M(i,2+4*n)==2 %body two
        Mv(i,3+4*n:5+4*n) = d_two_rotate*M(i,3+4*n:5+4*n)' + d_Or2';
    elseif M(i,2+4*n)==1 %body one
        Mv(i,3+4*n:5+4*n) = d_one_rotate*M(i,3+4*n:5+4*n)' + d_Or1';
    elseif M(i,2+4*n)==0 %body zero
        Mv(i,3+4*n:5+4*n) = d_zero_rotate*M(i,3+4*n:5+4*n)' + d_Or0';
    end
end
end

function [MV]=musc_vel(Mv, Ma, i, num_pts)
    MV=0; MVv=sym(MV);
    for n=0:num_pts-2
        Ma_coords=Ma(i,3+4*n:5+4*n)-Ma(i,3+4*(n+1):5+4*(n+1));
        MV_coords=Mv(i,3+4*n:5+4*n)-Mv(i,3+4*(n+1):5+4*(n+1));
        Ma_coords_u=1/sqrt(sum(Ma_coords.^2)) * Ma_coords; %unit vector
        MV=MV+dot(MV_coords,Ma_coords_u);
    end
C.10 rotateforce.mlx: Transform External Force File

Valerie Jardon
Department of Bioengineering, University of Kansas
rotateforce.mlx

Last Edited: 4/29/2022

This file (rotateforce.mlx) is based on the analys1.c and kinematics0.c codes developed by Timothy C. Franklin, Virginia Polytechnic Institute and State University. The analys1.c and kinematics.c codes were obtained through means of personal communication (Michael Madigan, personal communication, July 26, 2008).

Transform the External Force

Description: This code is used to transform the external force points with respect to the body the point is associated with. The symbolic output of variable Ea is stored in the 'Ea_sym.mat' file.

Reference:


------------

```matlab
% clear; clc
[E]=load_forcefile(); %load the external force coordinates
Ea=zeros(size(E,2)); Ea=sym(Ea); %initialize the rotated external force vector
num_pts=E(1); %number of points
Ea(1)=E(1);
Ea(2)=E(2);
Ea(6)=E(6);
Ea=point_transform(E,Ea,num_pts);
save('Ea_sym.mat','Ea') %save the symbolic output of Ea

function [Ea]=point_transform(E,Ea,num_pts)
load('origin_sym_separate.mat')
load('rotation_matrices_sym_separate.mat')

for n=0:num_pts-1
    if E(2+4*n)==6 %pelvis
        Ea(3+(4*n):S+(4*n))=P_rotate*(E(3+(4*n):S+(4*n)))+Orp';
    elseif E(2+4*n)==5 %body five
        Ea(3+4*n)=five_rotate*E(3+4*n:5+4*n)+0r5';
    elseif E(2+4*n)==4 %body four
        Ea(3+4*n)=four_rotate*E(3+4*n:5+4*n)+0r4';
    elseif E(2+4*n)==3 %body three
        Ea(3+4*n)=three_rotate*E(3+4*n:5+4*n)+0r3';
    else
        Ea(3+4*n)=E(3+4*n:5+4*n)';
    end
end
```

---------
elseif E(2+4*n)==2  %body two
    Ea(3+4*n:5+4*n)=two_rotate*E(3+4*n:5+4*n)'+Or2';
elseif E(2+4*n)==1  %body one
    Ea(3+4*n:5+4*n)=one_rotate*E(3+4*n:5+4*n)'+Or1';
elseif E(2+4*n)==0  %body zero
    Ea(3+4*n:5+4*n)=zero_rotate*E(3+4*n:5+4*n)'+Or0';
else
    Ea(3+4*n:5+4*n)=E(3+4*n:5+4*n);
end
end
C.11 dynam.mlx: Dynamics

Valerie Jordon
Department of Bioengineering, University of Kansas
dynam.mlx
Last Edited: 4/28/22

This file (dynam.mlx) is based on the Mathematica Code developed by Timothy C. Franklin, published in Appendix A of his thesis (Franklin, 2006, p. 75-81)

Dynamics

Description: This file (dynam.mlx) is utilized for the system dynamics. The center of mass, velocity of the center of mass and the mass moment of inertia are defined for each body. The Lagrangian derivative is used for the equations of motion.

Reference:

clear; clc

Initialize Variables

The symbolic angles represented in the following variables need to be as a function of time for our calculations in this code. The function 'sub_timer' can be used to adjust the angle variables.

lead('origin_syn_separate.mtx')
lead('rotation_matrices_syn_separate.mtx')

% Need angles as a function of time
Gr1=sub timely(Gr1);
Gr2-sub timely(Gr2);
Gr3-sub timely(Gr3);
Gr4-sub timely(Gr4);
Gr5-sub timely(Gr5);
Gr6-sub timely(Gr6);

% Need angles as a function of time
zero rotate-sub timely(zero rotate);
img rotate-sub timely(img rotate);
two rotate-sub timely(two rotate);
three rotate-sub timely(three rotate);
four rotate-sub timely(four rotate);
five rotate-sub timely(five rotate);
nder rotate-sub timely(nder rotate);

sys 1 1 0 cent 1 cent 2 cent 3 g 1 1 1.1 1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9 2.0

----------- End of the "Initialize Variables" Section

Determining the Symbolic General Rotation Matrix

% Rotations
sys 83 42 9
Fl=[1 0 0; 0 cos(q) -sin(q); 0 sin(q) cos(q)]; Rotation axis 1 +CCW Rotation
F2=[cos(q) 0 sin(q); 0 1 0; -sin(q) cos(q)]; Rotation axis 2 +CCW Rotation
F3=[cos(q) -sin(q) 0; sin(q) cos(q) 0; 0 0 1]; Rotation axis 3 +CCW Rotation
F=[F1 F2 F3];

----------- End of the "Determining the Symbolic General Rotation Matrix" Section

Defining the Coordinate Systems

% Define the Inertial Coordinate System
x1=[1 0];
x2=[0 1];

% Define Body Fixed Coordinate Systems
sys a1(t) b1(t) c1(t) d1(t) e1(t) f1(t) T1(t) T4(t) T6(t) A1(t) T2(t) B2(t) A2(t) T3(t) B3(t) A3(t) T4(t) B4(t) A4(t) T5(t) B5(t) A5(t)
cp_Axis1=rot_axis1; % Pelvis Axis 1 - after rotation
cp_Axis2=rot_axis2; % Pelvis Axis 2 - after rotation
cp_Axis3=rot_axis3; % Pelvis Axis 3 - after rotation
cp_Axis4=rot_axis4; % Hip Axis 1 - after rotation
cp_Axis5=rot_axis5; % Hip Axis 2 - after rotation
cp_Axis6=rot_axis6; % Hip Axis 3 - after rotation

cp_Axis1=rot_axis1; % Pelvis Axis 1 - after rotation
cp_Axis2=rot_axis2; % Pelvis Axis 2 - after rotation
cp_Axis3=rot_axis3; % Pelvis Axis 3 - after rotation
cp_Axis4=rot_axis4; % Hip Axis 1 - after rotation
cp_Axis5=rot_axis5; % Hip Axis 2 - after rotation
cp_Axis6=rot_axis6; % Hip Axis 3 - after rotation
Defining the Angular Velocities and Vectors

\[
\begin{align*}
\text{vec: } \text{domega}(t) &= \text{dtheta}(t) \times \text{omega}(t), \quad \text{domega}(t) = \frac{d}{dt} \text{omega}(t) \\
\text{dtheta}(t) &= \text{omega}(t) \times \text{domega}(t), \quad \text{dtheta}(t) = \frac{d}{dt} \text{theta}(t) \\
\text{omega}(t) &= \text{omega}(t), \quad \text{omega}(t) = \frac{d}{dt} \text{omega}(t) \\
\end{align*}
\]

End of the "Defining the Angular Velocities and Vectors" Section

Determine the Center of Mass and the Velocity of the COM for Each Body

Recall that the origin of each rigid body is at the center of the interior vertebral endplate. The center of mass is equidistant from the vertebral plates.

\[
\text{Center of Mass}
\]

End of "Determine the Center of Mass and the Velocity of the COM for Each Body" Section

Define 3D Inertial Matrix of Each Body

The Moment of Inertia about the Centroidal Axes

\[
\begin{align*}
I_x &= \int \rho_x^2 \, dV, \quad I_x = \frac{1}{12} m (a^2 + b^2) \\
I_y &= \int \rho_y^2 \, dV, \quad I_y = \frac{1}{12} m (a^2 + b^2) \\
I_z &= \int \rho_z^2 \, dV, \quad I_z = \frac{1}{12} m (a^2 + b^2)
\end{align*}
\]

End of "Define 3D Inertial Matrix of Each Body" Section

Storing system to use for kinetic energy calc

\[
\begin{align*}
\text{KCS_rel-formulas(NCS_Abaxis)}; \\
\text{KCS_rel-formulas(NCS_Abaxis)}; \\
\text{KCS_rel-formulas(NCS_Abaxis)}; \\
\text{KCS_rel-formulas(NCS_Abaxis)}; \\
\end{align*}
\]

Kinetic and Potential Energy Definitions

The Lagrangian function \(L\) is as follows:

\[
L = \text{Kinetic Energy of the System} - \text{Potential Energy of the System}
\]

Note: Potential Energy only includes the energy based on gravity. The other passive properties are included in the generalized forces.
The Lagrangian Derivative is as follows:
\[
\frac{d}{dt} \frac{\partial L}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i} = Q_i
\]

where \(q_i\) are generalized forces, \(Q_i\) are generalized coordinates, \(\dot{q}_i\) are degrees of freedom.

Substituting in the Lagrangian function into the Lagrangian Derivative:
\[
\frac{d}{dt} \frac{\partial \mathcal{L}(T,V,T(\dot{q}),V(\dot{q}))}{\partial \dot{q}_i} - \frac{\partial \mathcal{L}(T,V,T(\dot{q}),V(\dot{q}))}{\partial q_i} = Q_i
\]

Potential Energy not a function of velocity.

Resulting equation:
\[
\frac{d}{dt} \frac{\partial \mathcal{T}}{\partial \dot{q}_i} - \frac{\partial \mathcal{T}}{\partial q_i} + \frac{\partial \mathcal{V}}{\partial q_i} = Q_i
\]

See local function \(\mathcal{F}\) for the implementation of this formula in our calculations.

Calculating the Mass Matrix, Gravity Vector and Coriolis/Crossterms Vector

\[
\begin{align*}
\text{Mass Matrix:} & & \text{Gravity Vector:} & & \text{Coriolis/Crossterms Vector:} \\
\text{[Mdd,Td}]_{\text{Ge,Te}} & = \text{Matrix} \left[ T_{\text{dS}} \right] & \text{[Mdd,Ed]} & = \text{Matrix} \left[ E_{\text{dS}} \right] & \text{[Mdd,Td]}_{\text{Ge,Te}} & = \text{Matrix} \left[ T_{\text{dS}} \right]
\end{align*}
\]
\[ \text{Calculating the Lagrangian Derivative:} \]

\[ \frac{d}{dt} \left( \frac{d\dot{\theta}}{d\dot{\theta}}, \frac{d\dot{\theta}}{d\dot{\theta}} + \frac{\dot{\theta}}{\dot{\theta}} - \dot{\theta}_i \right) \]

\[ \text{ang and d_ang were both a function of time.} \]
\[ \text{ang_val and d_ang_val were not a function of time.} \]
C.12 \textit{generalizedForce.mlx}: Generalized Forces

Valerie Jordan
Department of Bioengineering, University of Kansas

\textit{generalizedForce.mlx}

Last Edited: 4/20/2002

This file (\textit{generalizedForce.mlx}) is based on the \textit{genforces.c} codes developed by Timothy C. Franklin, Virginia Polytechnic Institute and State University. The \textit{genforces.c} codes were obtained through means of personal communication (Michael Madigan, personal communication, July 26, 2008).

\textbf{Generalized Forces}

\textbf{Description:} In this script, the generalized forces for the degrees of freedom can be determined. First, the intervertebral disc moments are considered. These are applied to the bodies as pure moments. Following, the muscle forces are considered. Lastly, the external forces are considered. The resulting \( Q \) vector will be an \( 18 \times 1 \) vector due to the 18 degrees of freedom:

\[
Q_i = \sum_{j=1}^{N} \left( \sum_{k=1}^{N} \frac{\partial \tau_k}{\partial q_j} \right) f_{k,j} + \sum_{k=1}^{N} \left( \mathbf{r}_{k,j} \times \frac{\partial \omega_k}{\partial q_j} \right)
\]

Where \( i \) is the index of the right body, \( j \) represents force applied to the body, \( f \) is the index of the applied forces, \( \tau \) is the degree of freedom index, \( \mathbf{r} \) is the tensor velocity of the origin of the body, \( v \) is the angular velocity of the body and \( \mathbf{r} \) is a vector between the point where the force is applied and the origin of the body.

\textbf{Reference:}


---

```matlab
function []=generalizedForce(Nm,Mp,Er,Ef,PRQ,O1,O2,O3)

% Initialize Variables
DFT=18; Degrees of freedom
Nm=size(Nm,1); Number of muscles
mp=size(mp,1); Number of external forces
Q=zeros(DFT,1); Initializing the output array

% Intervertebral Disc Moments - applied as pure moments
% Only the second term in the generalized force eqn will be used for IVDs
for b=0.5 Body number
    for j=1:DFT Degrees of freedom
        C1=b*G1(m,3+[b+1,:]);
        Q(b,j)=Q(b,j)+C1(1);
        Q(b,j)=Q(b,j)+PRQ(b+1,1);
        Q(b,j)=Q(b,j)+G1(b+1,1)
    end
end

% Muscle Force
for i=1:Nm Muscle number
    for j=1:DFT Degrees of freedom
        %------- If the muscle has only two attachment points
        if N(i,1)==2 If the muscle has two attachment points (origin and terminal)
            m1=m(i,3); Muscle origin number
            m2=m(i,3); Muscle terminal number
            Q(m1,j)=Q(m1,j)+G1(b+1,1,1)
            Q(m2,j)=Q(m2,j)+G1(b+1,1,1)
            end
end
end
```

---
Xfer the terminal
if b2>6
  [f_vector2,m_vector2]=m_vvecs(muscle_origin_coords,muscle_term_coords,origin_b2,'terminal');
  G_b2=G(b2,1:10,:);
  Gr_b2=G(r_b2,1:10,:);
  Q(j)=Q(j)+F(1)^T*(dot(f_vector2,G_b2)+dot(m_vector2,Gr_b2))
end

%------ If the muscle has more than two attachment points
else
  points_num=Ma(1,1); %Determine the number of points
  node_num=points_num-2; %Calculate the number of nodes

% for the origin
  b1=Ma(1,1); %Muscle origin point body number
  muscle_origin_coords=Ma(1,1:5); %Muscle origin coordinates
  b2=Ma(1,6); %Muscle node 1 body
  muscle_term_coords=Ma(1,7:9); %Muscle node 1 coordinates
  origin_b2=r(1,1,:); %Origin coordinates of body 1
  for n=1:node_num
    b1=Ma(1,2+n); %Point 1 body
    b2=Ma(1,2+n); %Point 2 body
    muscle_term_coords=Ma(1,3+n:5+n); %Point 1 coordinates
    muscle_term_coords=Ma(1,6+n:8+n); %Point 2 coordinates
    origin_b2=r(1,1,:); %Origin coordinates of body 2
  end

% for the terminal
  b1=Ma(1,2+n); %Node before terminal
  node_coords=Ma(1,2+n:5+n); %Node before terminal coordinates
  b2=Ma(1,2+n:5+n); %Terminal point body
  muscle_term_coords=Ma(1,3+n:5+n); %Termial coordinates
  origin_b2=r(1,1,:); %Origin coordinates of body 2
  for n=1:node_num
    [f_vector2,m_vector2]=m_vvecs(node_coords,muscle_term_coords,origin_b2,'terminal');
    G_b2=G(b2,1:10,:);
    Gr_b2=G(r_b2,1:10,:);
    Q(j)=Q(j)+F(1)^T*(dot(f_vector2,G_b2)+dot(m_vector2,Gr_b2))
end
end

% External Forces
for i=1:n %Number of external forces applied
  for j=1:n %Degrees of Freedom
    if En(i,j)==2
      error('Error. External Force needs to have two points.').
    end
    b1=G(a(i,2)); %Force origin body number
    force_origin_coords=Ma(1,1:5); %Force origin coordinates
    b2=Ma(1,6); %Force terminal point body number
    force_term_coords=Ma(1,7:9); %Force terminal coordinates
    Xfer the origin
    if b1>6
      origin_b2=r(1,1,:); %Origin coordinates of body 1
    end
  end
end
end

for i=1:n %Number of external forces applied
  for j=1:n %Degrees of Freedom
    if En(i,j)==2
      error('Error. External Force needs to have two points.').
    end
    b1=G(a(i,2)); %Force origin body number
    force_origin_coords=Ma(1,1:5); %Force origin coordinates
    b2=Ma(1,6); %Force terminal point body number
    force_term_coords=Ma(1,7:9); %Force terminal coordinates
    Xfer the origin
    if b1>6
      origin_b2=r(1,1,:); %Origin coordinates of body 1
    end
  end
end
end

end

end

end

end

end

end

end

end

end

end

end

end

end

end

end

end

end

end

end
function [F_vector,M_vector]=f_vecs(force_origin_coords,force_term_coords,[],origin_Bi,'origin')
G1_Bi=G1(G1Bi);%1
Gr_Bi=Gr(GrBi);%1
Q(i)=Q(j)*((dot(t_vector1,G1_Bi)+dot(t_vector1,Gr_Bi));
end

For the terminal
if B2<6
origin_B2=Or(b2+1,1); %origin coordinates of body 2
[F_vector,M_vector]=f_vecs(force_origin_coords,force_term_coords,[],origin_B2,'terminal');
G1_B2=G1(G1B2);%1
Gr_B2=Gr(GrB2);%1
Q(i)=Q(j)*((dot(t_vector2,G1_B2)+dot(t_vector2,Gr_B2));
end
end
end

function [vect_out]=unit_vect(vect_in)
 vect_mag=sort(sum(vect_in.^2));
 vect_out=vect_in./vect_mag; %Calculate unit vector
end

function [F_vector,M_vector]=f_vecs(muscle_pt_coords,muscle_p2_coords,muscle_p3_coords,origin_origin,pont_type)
% Calculating F and M p = p +
if strcmp(point_type,'origin')
 F_vector=unit_vect(vect_origin); %Vector from point to point
end
origins_diff_vec=muscle_pt_coords-origin Origin; %Difference between body origin coordinates and muscle point coordinates
F_vector=cross3d(muscle_pt_coords,origins_diff_vec,F_vector);
elseif strcmp(point_type,'terminal')
 F_vector=unit_vect(vect_origin); %Vector from point to point
end
origins_diff_vec=muscle_p2_coords-origin Origin; %Difference between body origin coordinates and muscle point coordinates
F_vector=cross3d(muscle_p2_coords,origins_diff_vec,F_vector);
elseif strcmp(point_type,'node')
 F_vector=unit_vect(vect_origin); %Vector from point to point
end
origins_diff_vec=muscle_p3_coords-origin Origin; %Difference between body origin coordinates and muscle point coordinates
F_vector=cross3d(muscle_p3_coords,origins_diff_vec,F_vector);
end
end
end
function [vect_out]=unit_vect(vect_in)
 vect_mag=sort(sum(vect_in.^2));
 vect_out=vect_in./vect_mag; %Calculate unit vector
end

end
end
C.13 optimize.mlx: Optimization

C.13.1 Main File

% Valerie Jordan
% Department of Bioengineering, University of Kansas
% optimize.mlx
% Last Edited: 4/30/2022

% Based on the optimizationscript.m code developed by Timothy C. Franklin,
% Virginia Polytechnic Institute and State University.
% Description
% In the optimize.m file, muscle activation is determined for the
% system while the metabolic power is minimized, and the system is stable
% and in equilibrium. This is achieved by using the MATLAB fmincon
% built-in function which allows for a minimum to be determined while
% meeting certain constraints.
% This file (optimize.m) is based on the optimizationscript.m file
% developed by Timothy C. Franklin, published in Appendix A of his thesis (pages
% 85-87).

% ------ Reference:
% Franklin, T. C. (2006). Linear System Analysis of the Role of Reflex Gain and
% Delay in a Dynamic Human Spine Model... (Master's thesis, Virginia Polytechnic
% Institute and State University) (pp. 85-87). Virginia Tech. Retrieved June 19,
%--------------------------------------------------------

%--------------------------------------------------------
% This section is based on the "Optimization Settings" and "Equality Assembly" sections
% of the optimizationscript.m file.
% developed by Timothy C. Franklin, published in Appendix A of his thesis.
%--------------------------------------------------------

% Initializing Variables
Uo=zeros(size(M,1),1); Un=Uo; %creating zeros matrices
Fo=zeros(size(M,1),1); Fn=Fc; %creating zeros matrices
C&=Ha(1,34);

eps=0.00001*pi/180; %disturbance
length_array=zeros(90,1);

global PrevElig JPrevElig coequal optiter
   PrevElig=zeros(18,1); % global v
   JPrevElig=zeros(36,1); % settings
   optiter=0;
   coequal=zeros(18,1);

%Initializing the constraints for fmincon
Aeq=zeros(18,90); %equilibrium
beq=zeros(18,1); % equilibrium
A=eye(90);
B=zeros(90,1);

% Equilibrium Constraints

disp('------Determining Equilibrium Conditions - Passive------')
% Includes intervertebral disc moments and external force
% Does NOT include muscle force
U=zeros(90,1); % activation = 0 for passive
beq=spineacc(0,Stateo,Pango,U,EFo,Vars,M,E,Lo,Angs,IVD,Gl_func,Gr_func,
Mm_func,Cm_func,Gm_func,Ma_func,ML_func,MV_func,Ea_func,Rm_func,dRm_func,origin_func,dOrigin_func,ChSpace,[]);

disp('------Determining Equilibrium Conditions - Active------')
% Includes Muscle Forces
% Does NOT include intervertebral disc moments or external force

for i=1:length(length_array)
    U=zeros(90,1); % initializing a vector
    U(i)=1;
    [Neq(1,:),a,b,Cp,Od]=spineacc(i,0,Stateo,Pango,U,EFo,Vars,M,E,Lo,CSA,
Gl_func,Gr_func,Mm_func,Cm_func,Gm_func,Ma_func,ML_func,MV_func,Ea_func,
Mm_func,dM_func,origin_func,dOrigin_func,ChSpace,[]);
end

% Constructing the Jacobians
% This section is based on the "Jacobian Assembly", "Reflex" and "For Optimization (no time delay assumption)" sections
% of the optimization_script.m file developed by Timothy C. Franklin, published in Appendix A of his thesis.
%--------------------------------------------------------------------------------------
disp('------Building the Jacobians------')
tic

% Initializing Jacobian Matrices
Sp=zeros(36,36);
Sp1(1:18,19:36)=eye(18,18);
Sp2=zeros(size(M,1),36,36);
Sp3=zeros(36,36); % no passive contribution
Sp4=zeros(size(M,1),36,36);

for h=1:36
    dStateo=Stateo;
    dStateo(h)=dStateo(h)+eps;
    for i=1:90
        Uo=zeros(size(M,1),1);
        Uo(i)=1;
        [qdda,qddP,qddR]=spineacc(i,dStateo,Pango,Uo,EFo,Vars,Lo,CSA,
Lord_Angs, IVD, GI_funct, Gr_funct, Mx_funct, Cm_funct, Gm_funct, Ma_funct, ML_funct, MV_funct, Ea_funct, Rm_funct, dRm_funct, origin_funct, dorigin_funct, ChSpace, []);  
Jai(i, 19:36, h) = (qddA - Aeq(:, i))/eps;  
Jpi(19:36, h) = (qddP + bcp)/eps;  
Jar(i, 19:36, h) = (qddR)/eps;  
end  
end  

elapsedTime = toc;  
time_c = elapsedTime/60; time_c = fix(time_c);  
fprintf('Building the Jacobian took %d minutes and %3.1f seconds. \n', time_c, elapsedTime - (elapsedTime - (time_c*60)))  

%% For Optimization - No Time Delay Assumption  
Jp = Jpi + Jpi;  
Ja = Jai + Jar;  
save ( 'Jacobians.mat', 'Ja', 'Jp', 'Jai', 'Jar', 'Jpi', 'Jpr' )  
%% Perform the Constrained Optimization Procedure  

disp ('------Running fmincon to determine muscle activation------')  

% Run fmincon to determine muscle activation and metabolic power  
[exitflag_success, fval_success, Un_total_success, Un_total_initial_success] = run_fmincon =  
(A, B, Aeq, beq, CSA, Le, Stateo, Jp, Ja, N);  

%% Check the optimization output - is the system stable and in equilibrium?  
max_eig_vec = zeros(numel(exitflag_success), 1);  
for h = 1:numel(exitflag_success)  
    if exitflag_success(h) == Inf  
        Un_opt = Un_total_success(:, h);  
        Jp = Jp;  
        for i = 1:36  
            Jp(19:36, i) = Jp(19:36, i) + (Un_opt * Ja(:, 19:36, i))';  
        end  
    end  
end  

%% Check the eigenvalues  
eig_val = eig(Jp);  
max_eig_vec(h) = max(real(eig_val));  
if max_eig_vec(h) < 0  
    fval_success(h) = Inf;  
end  

%% Plot the eigenvalues  
figure(5)  
plot(real(eig_val), imag(eig_val), 'r*'); xlabel('Real'); ylabel('Imaginary');  
grid on  

%% Check Equilibrium Condition  
equilibrium_constraint = Aeq * Un_opt - beq;  
equilibrium_constraint_max = max(abs(equilibrium_constraint));
if equilibrium_constraint_max>3e-5
    fval_success(h)=1000000000;
end
end

% Determine the optimal trial
[M_best, I_best]=min(fval_success);
Un_initial_best=Un_total_initial_success(:, I_best);
Un_best=Un_total_success(:, I_best); % resulting activation
exitflag_best=exitflag_success(I_best);

%% Display Optimization Results

% For the optimal solution
Un_opt=Un_best;
J=Jp;
    for i=1:36
        J(19:36,i)=J(19:36,i)+([Un_opt'*Ja(:, 19:36, i)]);
    end

% Check the eigenvalues
    eig_val=eig(J);
    max_eig=max(real(eig_val));

% Plot the eigenvalues
    figure(5)
    plot(real(eig_val), imag(eig_val), 'r-'); xlabel('Real'); ylabel('Imaginary'); grid on

% Check Equilibrium Condition
    equilibrium_constraint=Aeq*Un_opt-b0;
    equilibrium_constraint_max=max(abs(equilibrium_constraint));

disp('Optimization Successful. System is stable and in equilibrium.')
disp(['fval=' num2str(fval_success)]);
disp(['Maximum real part of eigenvalue=' num2str(max_eig)]);
disp(['Maximum equilibrium error=' num2str(equilibrium_constraint_max)]);
disp('------End of Optimization Procedure------')
C.13.2 Angular Acceleration Functions

spineaccP.m file utilized for equilibrium constraint

function [qdd]=spineaccP(t,Stateo,Pango,U,EFO,Vars,X,E,Lord_Angs,IVD,GL_funct, 
Gr_funct,Em_funct,Cm_funct,Em_funct,Em_funct,ML_funct,IVD,Em_funct,Em_funct, 
Em_funct,origin_funct,dorieln_funct,ChSpace,verification) 
% acceleration due to passive

[A0,A1,A2,A3,A4,A5,Ap,B0,B1,B2,B3,B4,B5,Bp,DA0,DA1,DA2,DA3,DA4,DA5,DAP,DA6,DA7,DA8, 
DB3,DB4,DB5,DB6,DT6,DT7,DT8,DT9,DT10,DT11,DT12,DT13,DT14,DT15,DT16,DT17,DT18,DT19, 
ADot,ADdot,ADddot,ADdddot,Adot,Adotdot,Adotddot,T0dot,T1dot,T2dot,T3dot,T4dot,T5dot, 
B0dot, 
B1dot,B2dot,B3dot,B4dot,B5dot,Apdot,Apddot,Apdddot,Tdddot,T0dddot,T1dddot,T2dddot,T3dddot, 
T4dddot,T5dddot,B0dddot,B1dddot,B2dddot,B3dddot,B4dddot,B5dddot,A0dddot, 
A1dddot,A2dddot,A3dddot,A4dddot,A5dddot,B0dddot,B1dddot,B2dddot,B3dddot,B4dddot,B5dddot, 
P0dddot,T0dddot,B0dddot,A0dddot,Tpdddot,T0pdddot,B0pdddot,A0pdddot; 
L0=Vars(1); L1=Vars(2); L2=Vars(3); L3=Vars(4); L4=Vars(5); L5=Vars(6); 
L6=Vars(7); l1=Vars(8); g=Vars(9); 
cent1=Vars(17); cent2=Vars(18); cent3=Vars(9); 
m0=Vars(11); m1=Vars(12); m2=Vars(13); m3=Vars(14); m4=Vars(15); m5=Vars(16); 
% Recalculate due to disturbance

GL=GL_funct([B1,B2,B3,B4,B5,L1,L2,L3,L4,L5,L6,T1,T2,T3,T4,T5]); 
Gr=Gr_funct([B0,B1,B2,B3,B4,B5,T0,T1,T2,T3,T4,T5]); 
M0=M0_funct([A0,A1,A2,A3,A4,A5,B0,B1,B2,B3,B4,B5,L0,L1,L2,L3,L4,L5,L6,T0,T1,T2,T3,T4,T5]); 
Cm=Cm_funct([A0,A1,A2,A3,A4,A5,A0dot,A1dot,A2dot,A3dot,A4dot,A5dot,A0ddot,A1ddot,A2ddot,A3ddot,A4ddot,A5ddot,Apdot,Apddot,B0,B1,B2,B3,B4,B5,B0ddot,B1ddot,B2ddot]);
spineaccA.m file utilized for equilibrium constraint

% Valerie Jordon
% Department of Bioengineering, University of Kansas
% spineaccA.m
% Last Edited: 4/30/2022
% This file (spineaccA.m) is based on the AccelerationMA.m file developed by Timothy C. Franklin, Virginia Polytechnic Institute and State University.
% The AccelerationMA.m file was obtained through means of personal communication (Michael Madigan, personal communication, July 26, 2008).
% Reference:

function [qdd, q, b, Gp, Gd]=spineaccA(l, t, Stateo, Pango, J, P, Epo, Vars, M, E, Lo, CSA, Gl_func, Gd_func, Ma_func, Ca_func, Gm_func, Ma_func, NL_func, MV_func, Ea_func, Rm_func, dRm_func, origin_func, ChSpace, verification)
% acceleration due to active
% Recalculate due to the disturbance
Gl=Gl_funct(B1,B2,B3,B4,B5,L1,L2,L3,L4,L5,T1,T2,T3,T4,T5);
Gr=Gr_funct(B0,B1,B2,B3,B4,B5,T0,T1,T2,T3,T4,T5);
Ma=Ma_funct(A0,A1,A2,A3,A4,A5,B0,B1,B2,B3,B4,B5,L0,L1,L2,L3,L4,L5,T0,T1,T2,T3,T4,T5);
Mw=Mw_funct(A0,A1,A2,A3,A4,A5,B0,B1,B2,B3,B4,B5,L0,L1,L2,L3,L4,L5,T0,T1,T2,T3,T4,T5,Tp,cent1,cent2,cent3);
Ml=Ml_funct(A0,A1,A2,A3,A4,A5,B0,B1,B2,B3,B4,B5,L0,L1,L2,L3,L4,L5,T0,T1,T2,T3,T4,T5,Tp,cent1,cent2,cent3);
T5,Tp,cent1,cent2,cent3);
MV=MV_funct(A0,A1,A2,A3,A4,A5,B0,B1,B2,B3,B4,B5,L0,L1,L2,L3,L4,L5,T0,T1,T2,T3,T4,T5,Tp,cent1,cent2,cent3);
Ea=Ea_funct(A0,A1,B0,B1,B2,B3,B4,B5,L0,L1,L2,L3,L4,L5,T0,T1,T2,T3,T4,T5,Tp,cent1,cent2,cent3);
origin=origin_funct(A0,A1,B0,B1,B2,B3,B4,B5,L0,L1,L2,L3,L4,L5,T0,T1,T2,T3,T4,T5,Tp,cent1,cent2,cent3);

A=State0(37:end);

[MF,da]=musclemodel(U,p,A,ML,MV,Lo,CSA);
[MFR,da,gs,b,bp,gs]=musclemodelR(U,p,A,ML,MV,Lo,CSA,verification);

% for the verification script
if strcmp('verification',verification)
    MFR=zeros(size(MFR));
end

MF=MF+MFR;

EFO=0; % external force incorporated in spinesaccP
Q=generalizedForce(Ma,real(MF),Ea,EFO,zeros(7,3),Gl,Gr,origin);

% solve for the angular acceleration
qdd=ode45(@Q,());
end
spineacc.m file utilized for equilibrium constraint

```matlab
function [qdda, qddP, qddR] = spineacc(l, Stateo, Pango, V, P, EPho, Vars, Lo, CSA, Lord_Angs, IVD, G1_func, Gr_func, Mm_func, Gm_func, Ma_func, ML_func, Mf_func, Ra_func, Gm_func, dM_func, origin_func, dorigin_func, ChSpace, verification)

[A0, A1, A2, A3, A4, A5, Ap, B0, B1, B2, B3, B4, B5, Bp, D0, D1, D2, D3, D4, D5, Dap, DB0, DB1, DB2, DB3, DB4, DB5, DB6, DC0, DC1, DC2, DT3, DT4, LCS, DTo, T0, T1, T2, T3, T4, T5, To, ... Adot0, A1dot0, A2dot0, A3dot0, A4dot0, A5dot0, Apdot0, B0dot0, B1dot0, B2dot0, B3dot0, B4dot0, B5dot0, Bpdot0, Bpdot0, Tpdot0, Tdot0, Tpdot0, Tpdot0, Bpdpdot0, Bpdpdot0, Tpdpdot0, Tpdpdot0, Tpdpdot0]

L0 = Vars(1); L1 = Vars(2); L2 = Vars(3); L3 = Vars(4); L4 = Vars(5); L5 = Vars(6);
la = Vars(7); lb = Vars(8); gc = Vars(10);
cent1 = Vars(17); cent2 = Vars(18); cent3 = Vars(9);
m0 = Vars(11); m1 = Vars(12); m2 = Vars(13); m3 = Vars(14); m4 = Vars(15); m5 = Vars(16);

% Recalculate due to the disturbance
if i == 1
    G1 = G1_func(B1, B2, B3, B4, B5, L1, L2, L3, L4, L5, T1, T2, T3, T4, T5);
    Gr = Gr_func(B0, B1, B2, B3, B4, B5, T0, T1, T2, T3, T4, T5);
    Mm = Mm_func(A0, A1, A2, A3, A4, A5, Ap, B0, B1, B2, B3, B4, B5, L0, L1, L2, L3, L4, L5, T0, T1, T2, T3, T4, T5, la, lb, m0, m1, m2, m3, m4, m5);
    Gm = Gm_func(B0, B1, B2, B3, B4, B5, L0, L1, L2, L3, L4, L5, T0, T1, T2, T3, T4, T5, g, m0, m1, m2, m3, m4, m5);
    Ma = Ma_func(A0, A1, A2, A3, A4, A5, Ap, B0, B1, B2, B3, B4, B5, Bp, L1, L2, L3, L4, L5, T0, T1, T2, T3, T4, T5, Tp, cent1, cent2, cent3);
    ML = ML_func(A0, A1, A2, A3, A4, A5, Ap, B0, B1, B2, B3, B4, B5, Bp, L1, L2, L3, L4, L5, T0, T1, T2, T3, T4, T5, Tp, cent1, cent2, cent3);
end
```

227
MV=MV\_func(A0, A1, A2, A3, A4, A5, Ap, B0, B1, B2, B3, B4, B5, Bp, DA0, DA1, DA2, DA3, DA4, DA5, \( \varepsilon \) DAp, DB0, DB1, DB2, DB3, DB4, DB5, DP0, DP1, DP2, DP3, DP4, DP5, DTP, L1, L2, L3, L4, L5, T0, T1, T2, \( \varepsilon \) T3, T4, T5, TP, cent1, cent2, cent3); 
Ea=EA\_func(A0, Ap, B0, B1, B2, B3, B4, B5, Bp, L1, L2, L3, L4, L5, T0, T1, T2, T3, T4, T5, TP, cent1, cent2, cent3);
rotation\_matrices=RM\_func(A0, A1, A2, A3, A4, A5, Ap, B0, B1, B2, B3, B4, B5, Bp, T0, T1, T2, T3, T4, T5, TP); 
d\_rotation\_matrices=DRM\_func(A0, A1, A2, A3, A4, A5, Ap, B0, B1, B2, B3, B4, B5, Bp, DA0, DA1, \( \varepsilon \) DA2, DA3, DA4, DA5, DAp, DB0, DB1, DB2, DB3, DB4, DB5, DP0, DP1, DP2, DP3, DP4, DP5, DTP, T0, T1, T2, \( \varepsilon \) T3, T4, T5, TP); 
origin=origin\_func(Ap, B1, B2, B3, B4, B5, Bp, L1, L2, L3, L4, L5, T1, T2, T3, T4, T5, TP, cent1, cent2, cent3); 
d\_origin=dorigin\_func(Ap, B1, B2, B3, B4, B5, Bp, DAp, DB1, DB2, DB3, DB4, DB5, DP0, DP1, DP2, DP3, DP4, DP5, DTP, T0, T1, T2, \( \varepsilon \) DT3, DT4, DT5, DTP, L1, L2, L3, L4, L5, T1, T2, T3, T4, T5, TP, cent1, cent2, cent3); 
PRQ=IVO\_calc(rotation\_matrices, d\_rotation\_matrices, Stateo, Pango, Lord\_Angs, IVD); 
PRO=PRQ(1:6,:);
if strcmp('verification', verification) 
\( Cm=\text{zeros}([\text{size}(Cm)]; \) \( \% \) verification \( Cm \) 
PRQ=[1 2 3; 4 5 6; 7 8 9; 10 11 12; 1 2 3; 5 5 4; 1 2 3]; \( \% \) verification 
PRQ 
PRQ=PRQ(1:6,:); 
end
save('optimization-varsA.mat', 'rotation\_matrices', 'd\_rotation\_matrices', 'origin', 'd\_origin', 'EA', 'MA', 'ML', 'MV' \( \varepsilon \), 'MM', 'GL', 'GR', 'PRQ', 'GM', 'CM') 
else
load('optimization-varsA.mat') 
end
A=Stateo(1:end);

[Mp, dx0]=musclemodel(U, P, A, ML, MV, Lo, CSA); \( \% \) ACTIVE
Mp=\text{zeros(size(U)}; \( \% \) PASSIVE no muscle force
[Mp, dx0, \( \varepsilon \) dx0]=musclemodelR(U, P, A, ML, MV, Lo, CSA, verification); \( \% \) RFLEX
\( [Qr]=generalized\_force(Ma, real(Mp), Ea, 0, zeros(3, 3), Gl, Gr, origin); 
qddA=M \( \varepsilon \) (QR) 
[OP]=generalized\_force(Ma, MP, Ea, EPo, PRQ, GL, Gr, origin); 
qddP=M \( \varepsilon \) (GP-CM-Gr') 

[QR]=generalized\_force(Ma, real(MPr), Ea, 0, zeros(3, 3), Gl, Gr, origin); 
qddR=M \( \varepsilon \) (QR)
C.13.3 run_fmincon.mlx: Constrained Optimization Solver fmincon (MATLAB)

Valerie Jordon
Department of Biomedical Engineering, University of Kansas
run_fmincon.mlx
Last Edited: 4/30/2002

Based on the optimization script code developed by Timothy C. Franklin, Virginia Polytechnic Institute and State University.

Description: This function run_fmincon.mlx is used to run fmincon, allowing for the muscle activation and metabolic power to be determined, while ensuring the system is stable and in equilibrium.

This section is based on the “Run Optimization routine” section of the optimization script file developed by Timothy C. Franklin, published in Appendix A of his thesis.

Reference:

function [exitflag, success, err_total, success, un_total_initial, success] = run_fmincon(A, B, Aeq, beq, CSA, ia, States, Jp, Jx, M)
  num_trials = N; Number of trials

  % Initializing Variables
  un_total=zeros(N, num_trials); Matrix that will store the muscle activation for all 10 trials
eval_total=zeros(1, num_trials); Vector that will store the metabolic power for all 10 trials
  un_total_initial=zeros(99, num_trials); Matrix that will store the initial guess of muscle activation for all 10 trials
  exitflag_total=zeros(1, num_trials); Vector that will store the fmincon exitflag for all 10 trials

  % Optimization Procedure
  for k=1:num_trials % Looping for 10 trials
    u0 = randn(M, 1); % Random initial guess for muscle activation
    un_initial = u0;
    up_bound = ones(M, 1); % Muscle activation: fully activated muscle
    low_bound = zeros(M, 1); % Muscle activation: unactivated muscle

    % Setting the options for fmincon
    options = optimoptions('fmincon', 'Algorithm', 'sqp-legacy', 'MaxFunctionEvaluations', 1e6, 'MaxIterations', 400, 'StepTolerance', 1e-5, 'ToleranceFun', 1e-9, 'TolX', 1e-4, 'TolConstr', 1e-3);
    [un_n, exitflag_n, fval_n, err_total_n, exitflag_total_n] = fmincon(@(u)fn_muscles(u, A, B, Aeq, beq, CSA, ia, States, Jp, Jx, M), un_initial, up_bound, low_bound, options);
    exitflag_total(k, 1) = exitflag_n;

  end

  % Storing select inputs and outputs for this trial
  un_total(:, k) = un_n;
  un_total_initial(:, k) = un_initial;
  exitflag_total(:, k) = exitflag_n;
end

Determine the Success of the Trials

If tolerances were not met or the solver converged to an infeasible point, then the trial was not successful. Exitflags from the solver indicate its potential success or failure. The successful trial information should be stored in vectors and matrices and used as output for this function.

In optimize.m, stability and equilibrium conditions will be evaluated and the optimal solution will be utilized.

% Store optimization output from all trials
% Initialize Variables
success_vector = zeros(1, num_trials);
exitflag_success = zeros(1, num_trials);
fval_success = zeros(1, num_trials);
un_total_initial_success = zeros(99, num_trials);
un_total_success = zeros(N, num_trials);
for j=1:num_trials
  if exitflag_total(j)==1 || exitflag_total(j)==2 % Indicating that tolerances have not been met and local minima found
    exitflag_success(j) = exitflag_total(j);
fval_success(j) = fval_total(j);
    un_total_success(:, j) = un_total(:, j);
    un_total_initial_success(:, j) = un_total_initial(:, j);
  end
end
C.13.4 costfun.m: Metabolic Power Function for Constrained Optimization Procedure

% Valérie Jardon
% Department of Bioengineering, University of Kansas
% costfun.m
% Last Edited: 4/30/2022
% Based on the costfun.m code developed by Timothy C. Franklin, Virginia Polytechnic Institute and State University.

% Description: This function (costfun.m) is used with fmincon to calculate the metabolic power. This is the parameter being minimized during the constrained optimization procedure.
% This function is based on the costfun.m file developed by Timothy C. Franklin, published in Appendix A of his thesis.

% References:
% account_id=14556.
%-----------------------------------------------------------------------------------------------

function Power=costfun(Un,S,Jp,Ja,Lo,CSA)

rho=1000; % units: kg/m^3
mass=rho*(CSA/100/100).*Lo; %CSA converted to m^2; m=density*volume

slow=sin(pi/2*Un); %excitation level (Anderson 1999)
fast=1-cos(pi/2*Un); %excitation level (Anderson 1999)

%assuming 50% slow twitch fibers and 50% fast twitch fibers
percent_slowtwitch=0.5;
percent_fasttwitch=0.5;

%Total Maintenance Heat Rate (Anderson 1999 page 120)
Power=sum(mass.*((74*percent_slowtwitch*slow+111*percent_fasttwitch*fast))); % W
W=J/s
C.13.5 *nlcon.m*: Nonlinear Constraints for Constrained Optimization Procedure

Author: Timothy C. Franklin

*nlcon.m*

*nlcon.m* file provided in the appendix of his thesis (Franklin, 2006, p. 87-88)


Edits to original function made by Valene Jardon

Department of Bioengineering, University of Kansas

Last Edited: 4/30/2022

Nonlinear Constraints for Constrained Optimization

```matlab
function [c,ceq]=nlcon(Un,S,Jp,Ja,Lo,CSA)
%This function involves the nonlinear constraints utilized in the
%constrained optimization procedure
%The eigenvalues of the Jacobian are evaluated to determine stability

global PrevEig JPrevEig cequal optiter
optiter=optiter+1;

% Nonlinear Constraint for fmincon #1
ceq=0;

% Nonlinear Constraint for fmincon #2
% determine the complete Jacobian using the muscle activation
J=Jp;
for i=1:36
 J(19:36,i)=J(19:36,i)+(Un'*Ja(:,19:36,i))';
end

ap=eig(J); %eig returns a column vector containing the eigenvalues of square matrix 
B=zeros(36,1);
P=zeros(36,36);

for j=1:36
 dist=JPrevEig:np(j);
 p(j,:)=sqrt(real(dist).^2+imag(dist).^2);
end
for j=1:36
 [q,indc]=min(min(p));
 [q,indr]=min(p(:,indc));
 b(indc)=ap(indr);
 p(:,indc)=1e12;
 p(indr,:)=1e12;
end
JPrevEig=b;
c=real(b)+1e-4; %Allow 1e-4 slop
end
```
C.14 `musclemodel.mlx`: Muscle Model

Valeria Jardon

Department of Bioengineering, University of Kansas

musclemodel.mlx

Last Edited: 4/30/2022

This file (musclemodel.mlx) is based on the musclemode1.m code developed by Timothy C. Franklin, Virginia Polytechnic Institute and State University. The musclemode1.m code was obtained through means of personal communication (Michael Madigan, personal communication, July 26, 2008).

Muscle Model

Description: This function (musclemodel.mlx) is based on the Hill-type muscle model discussed in Franklin’s thesis (Franklin, 2006, p. 12-13). This Hill-type muscle model includes a force generator, a spring and a damper, with all elements in parallel.

Reflexes should not be included in this function, so the proportional and differential reflex gains should be zero. See function musclemodelR.mlx for reflexes.

References:


---

```matlab

fo=46*CSA; % units: N; maximum muscle force= maximum muscle stress * CSA (Gardner-Morse et al., 1995, pp. 883)

q=20; % Bergmark dimensionless stiffness gain (Bergmark, 1988, pp. 26-27)
b=2; % dimensionless damping gain

dx=ML-lo;

% steadystate muscle activation
U=U;

% muscle model equation
f=-f0.*u.*(dx./Lo)+b*(MV./Lo)+1;

da=zeros(size(A));
end
```
C.15 musclemodelR.mlx: Reflex Model

Valerie Jordon
Department of Bioengineering, University of Kansas

musclemodelR.mlx

Last Edited: 4/30/2022

This file (musclemodelR.mlx) is based on the musclemode.m code developed by Timothy C. Franklin, Virginia Polytechnic Institute and State University. The musclemode.m code was obtained through means of personal communication (Michael Madigan, personal communication, July 26, 2005).

Muscle Model

Description: This function (musclemodelR.mlx) incorporates the reflex activation (Franklin, 2006, p. 13). The muscle model is based on the Hill-type muscle model discussed in Franklin's thesis (Franklin, 2006, p. 12-13). This Hill-type muscle model includes a force generator, a spring and a damper, with all elements in parallel.

References:


```
function [fe,dA,q,b,Gp,Gd]=musclemodelR(U,P,A,W,L,Lo,CSA,verification)

fo=46*CSA; %units: N maximum muscle force= maximum muscle stress * CSA (Gardner-Morse et al., 1995, p. 883)
q=28; % Bergmark dimensionless stiffness gain (Bergmark, 1989, p. 26-27)
b=2; %dimensionless damping gain
dX=W-Lo;
Gp=10; % proportional reflex gain
Gd=0; % differential reflex gain

% for verification procedure
if strcmp('verification',verification)
    Gp=0;
    Gd=0;
end

% Reflex Activation
U=U.*(Gp*(dX./Lo)+Gd*(W./Lo));

% Muscle Model Equation
fe=-fo.*U.*(q*(dX./Lo)+b*(W./Lo)+1);
dA=zeros(size(A));
end
```
C.16 IVD_calcs.mlx: Intervertebral Disc Moments

Valerie Jardon
University of Kansas

IVD_calcs.mlx

Last Edited: 4/30/2002

This file IVD_calcs.mlx is based on the intervertebral disc moments code developed by Timothy C. Franklin, Virginia Polytechnic Institute and State University. These intervertebral disc moments code was obtained through means of personal communication (Michael Madigan, personal communication, July 26, 2008).

Reference:

Intervertebral Disc Moments

Description of Code:
The purpose of this code is to calculate the intervertebral disc moments.

In these calculations, two neighboring planes will be used where one plane will remain stationary and the other plane will be manipulated through rotation.

In order to determine the plane rotation angles, the difference of the bone axis angles must first be rotated out so it will not influence our measures. In this case, the difference between the bone axis angles should be determined and one of the planes should be related so that the difference in bone axis angle is no longer present between the planes.

Any remaining rotation about axis 1 is due to the plane rotation. The plane rotation angle can be calculated. Then the plane angle is rotated out for the same plane as done previously for the bone axis angle. The resulting plane should be parallel to the plane that has not been manipulated. Conceptually:

The twist rotation occurs around Axis 3. The twist angle can be determined using the parallel planes.

Lastly, the intervertebral disc moment can be calculated. Equal and opposite moments implemented.

The verification task should be performed following analysis to ensure that the plane angles in fact parallel and aligned.

```matlab
function [mom_total]=IVD_calcs(rotation_matrices,vel_rotation_matrices,Static,Force,Load_Angles,IVD)
% Initializing Variables
PP1=zeros(6,1);
PP2=zeros(6,1);
PP3=zeros(6,1);
PP4=zeros(6,1);
PP5=zeros(6,1);
PP6=zeros(6,1);
PP7=zeros(6,1);
PP8=zeros(6,1);
PP9=zeros(6,1);
PP10=zeros(6,1);
PP11=zeros(6,1);
PP12=zeros(6,1);
PP13=zeros(6,1);
PP14=zeros(6,1);
PP15=zeros(6,1);
PP16=zeros(6,1);

rot_matrix=rotation_matrices;
d_rot_matrix=vel_rotation_matrices;
for i=1:6
% Planar Calculation
% Load Rotation Matrices for Each Body
body2_Rx_rot_matrix=[1 0 0; 0 1 0; 0 0 1]; % Body 2 is the upper body (ex. thorax)
body1_Rx_rot_matrix=[1 0 0; 0 1 0; 0 0 1]; % Body 1 is the lower body (ex. L1)
body2_Ry_rot_matrix=[1 0 0; 0 1 0; 0 0 1]; % Body 2 is the upper body (ex. thorax)
body1_Ry_rot_matrix=[1 0 0; 0 1 0; 0 0 1]; % Body 1 is the lower body (ex. L1)
body2_Rz_rot_matrix=[1 0 0; 0 1 0; 0 0 1]; % Body 2 is the upper body (ex. thorax)
body1_Rz_rot_matrix=[1 0 0; 0 1 0; 0 0 1]; % Body 1 is the lower body (ex. L1)
end
```

234
% Determine the difference between the lordosis angles of the two bodies
angle_diff=(lord_Angle body(1)-lord_Angle body(2));

% This rotation matrix will be used to rotate out the lordosis angle
sys эти
RV=[1 0 0; 0 cos(q1) -sin(q1); 0 sin(q1) cos(q1)]; % Rotation axis 1
rotate_out=rvb(RV,angle_diff); rotate_out=double(rotate_out);

body(2)=body(2).rotate_out; % Move

% Vectors along axis 1 of each body needed to determine planar angle
TV1=body1_Rx{1,1}; % Basis d1
TV2=body1_Rx{1,2};
planar_ang=acos(dot(TV1,TV2)); % If the dot product of V1 & V2 equaled 1, then the angle would be zero and we would know that axis 3

if planar_ang > .00000000001
% Franklin's tolerance criteria

% Determine the perpendicular vector along axis 1 using the cross product
TV=cross(TV1,TV2);
TV_map=sort(abs( TV.*TV ));
RV=TV/TV_map;
RV=TV_planar_ang; % Scaling the vector

% Calculating the moments - equal and opposite (completed later on)
PR(1,:)=rvb(RV1)*RV;
PR(2,:)=rvb(RV2)*RV;

% This rotation matrix will be used to rotate out the planar angle using
% the lower body - planes should now be parallel
RV_map=sort(abs( RV.*RV ));
RV=RV/RV_map; % Scaling the vector

body2_Rx=transpose(body2_Rx); % So that d1n
body1_Rx=transpose(body1_Rx); % So that d1n

body2_dh=transpose(body2_dh); % So that d1n

body2_Rx=rotorious_rotation(body2_Rx,rvb(planar_ang)); % Rodrigues formula
body2_dh=rotorious_rotation(body2_dh,rvb(planar_ang));
else

body2_Rx=transpose(body2_Rx); % So that d1n
body1_Rx=transpose(body1_Rx); % So that d1n

body2_dh=transpose(body2_dh); % So that d1n

end

% Twist Calculation

clear TV1 TV2 TV TV_map RV

% Vectors along axis 1 of each body needed to determine twist angle
TV1=body1_Rx{1,1}; % Basis d1
TV2=body1_Rx{1,2};

twist_ang=acos(dot(TV1,TV2)); % If the dot product of V1 & V2 equaled 1, then the angle would be zero and we would know that axis 1

if twist_ang > .00000000001

% Franklin's criteria

% Determine the perpendicular vector along axis 3 using the cross product

% This rotation matrix will be used to rotate out the twist angle

% Calculating the moments - equal and opposite (completed later on)
PR(1,:)=rvb(RV1)*RV;
PR(2,:)=rvb(RV2)*RV;

% This rotation matrix will be used to rotate out the twist angle

else

body2_Rx=transpose(body2_Rx); % So that d1n

end
function [rotated_vectors]=sphere_to_ellipse(vectors,rotation_axis,theta)
    % Basis of rotation must be a unit vector
    rotation_axis_unit = norm(rotation_axis);
    rotation_axis_mag = norm(rotation_axis);
    rotated_vectors = zeros(1, length(vectors));
    for i=1:length(vectors)
        rotated_vectors(i) = dot(vectors(i,:),rotation_axis_unit)*rotation_axis_unit*[1-cos(theta)] +
            cross(rotation_axis_unit,vectors(i,:)*sin(theta)) + vectors(i,:)*cos(theta);
    end
end
C.17 Time Delay Analysis

The Time Delay Analysis files (tdinvest.m, tdfmax.m and tdfind.m) from the appendix of Franklin’s thesis should be utilized to perform the Time Delay Analysis (pages 94-97) \(^{24}\).
C.18 Nonlinear Verification

C.18.1 Run Nonlinear Simulation

From spine_main.m file:

```matlab
%% Simulation
% In this section, the nonlinear simulation is completed to compare the
% linear analysis results from the optimization procedure.

disp('-----Simulation Started-----')

StateDist=State0;
StateDist(16)=StateDist(16)+deg2rad(0.01); % disturbance applied to thoracic/cervical spine rigid body

Tfinal=5;
Tode=[0,Tfinal]; % defining the bounds of integration

the delay margin (Td) defines the time delay in which the system becomes
unstable, therefore the simulation should include the time delay prior to
instability
Lags=7d*0.99;

% Delay must be greater than 0.06 seconds to be physiologically
% representative of reflexes and their contribution to stability
if Lags < 0.06
    error('Simulation cannot be performed. Required lag is not physiologically possible.')
end

%----Set Simulation Options
options=odeset('OutputFcn',@plotscheme,'Events',@edcheck,'RelTol',1e-4,'AbsTol',1e-8,'Stats','on');

%-----Simulation

tic
	sol=ode23(@spinedyn,Lags,StateDist,Tode,options,Un_opt,Ph,Pang,Vars,Gl_funct, %
Ge_funct,Ma_funct,Cm_funct,Me_funct,Mb_funct,Ng_funct,MV_funct,Be_funct,Rm_funct, %
RdR_funct,Origin_funct,OrigR_funct,Lord_fangs,TVD,Lo,CSA,EFs,ChSpace);

time=sol.x;
State=sol.y';
State=State*ones(length(time),1)*State0;

%Display Magnitude of Lag Simulated & Simulation Run Time
disp(['Lag Simulated: ' num2str(Lags(1))]);

elapsedtime=toc;

elapsedtime_min=floor(elapsedtime/60);
fprintf('The simulation took %d minutes and %d seconds to run.',elapsedtime_min, %
(elapsedtime-(60*elapsedtime_min)))
```

238
C.18.2 Spinedyn.mlx: Function Handle for Delay Differential Equation Solver

239


C.18.3 Determine the Distance of the Normalized State-Space from Equilibrium
From spine_main.m file:

```matlab
%% Calculate the distance from equilibrium using the normalized state-space
% Determine if the system is approaching equilibrium

position=Stated(:,:,1:18);
mean_position=mean(position,'all'); % average of all positions
position_Dss=position./mean_position;

velocity=Stated(:,:,19:36);
mean_velocity=mean(velocity,'all');
velocity_Dss=velocity./mean_velocity;

Dss=zeros(size(Stated,1),18);
for i=1:18
    Dss(:,:,i)=(position_Dss(:,:,i)).^2+(velocity_Dss(:,:,i)).^2;
end

Dss=sqrt(sum(Dss,2)); % sum of each row - each row is a time step
figure(4); plot(Time,Dss); grid on; xlabel('Time (seconds)'); ylabel('Dss');

% ------------------------------- End of spine_main.m script
```
Appendix D: Additional MATLAB Files

D.1 replace_sym.mlx: Replace symbolic variables with numeric

Valerie Jordon
Department of Biomedical Engineering, University of Kansas
replace_sym.mlx
Last Edited: 4/30/2020

Description: The function “replace_sym” is used to replace the symbolic variables that are within our variables in the Workspace with numeric values. Once all symbolic variables have been replaced with these numeric values, the output can be converted to double data type.

Replaces model parameters variables and angles.

```matlab
function [output] = replace_sym(input, State, Pango, Vars, ChSpace)
    % Input:
    %    input: input struct containing symbolic variables
    %    State: State struct containing model parameters
    %    Pango: Pango struct containing model parameters
    %    Vars: Vars struct containing model parameters
    %    ChSpace: ChSpace struct containing model parameters
    % Output:
    %    output: output struct containing numeric variables

    % Replace the symbolic variables with numeric values
    output = struct2table(input, State, Pango, Vars, ChSpace);
```

The function `replace_sym` takes various symbolic variables and replaces them with numeric values, effectively converting symbolic variables into numerical data types. This is useful for further numerical computations and analyses in MATLAB.
D.2 *vect2ang.m*: Assign angles from input angle vectors to individual variables for function input

```matlab
%% Valerie Jardon
%% Department of Bioengineering, University of Kansas
%% vect2ang.m
%% Last Edited: 4/30/2022

%% This function takes vector inputs of angles and separates them into
%% individual angles for analysis

function [A0, A1, A2, A3, A4, A5, Ap, B0, B1, B2, B3, B4, B5, Bp, DA0, DA1, DA2, DA3, DA4, DA5, DAp, DB0, ...
    DB1, DB2, DB3, DB4, DB5, DBp, DT0, DT1, DT2, DT3, DT4, DT5, DTp, TG, T1, T2, T3, T4, T5, Tp, ...
    A0dot, A1dot, A2dot, A3dot, A4dot, A5dot, T0dot, T1dot, T2dot, T3dot, T4dot, T5dot, B0dot, ...
    B1dot, B2dot, B3dot, B4dot, B5dot, Apdot, Bpdot, Tpdot] = vect2ang(Stateo, Pango)

T5 = Stateo(1); B5 = Stateo(2); A5 = Stateo(3);
T6 = Stateo(4); B4 = Stateo(5); A4 = Stateo(6);
T3 = Stateo(7); B3 = Stateo(8); A3 = Stateo(9);
T2 = Stateo(10); B2 = Stateo(11); A2 = Stateo(12);
T1 = Stateo(13); B1 = Stateo(14); A1 = Stateo(15);
T0 = Stateo(16); B0 = Stateo(17); A0 = Stateo(18);
Tp = Pango(1); Bp = Pango(2); Ap = Pango(3);

DT5 = Stateo(19); DT6 = Stateo(20); DA5 = Stateo(21);
DT4 = Stateo(22); DT3 = Stateo(23); DA4 = Stateo(24);
DT3 = Stateo(25); DT2 = Stateo(26); DA3 = Stateo(27);
DT2 = Stateo(28); DT1 = Stateo(29); DA2 = Stateo(30);
DT1 = Stateo(31); DT0 = Stateo(32); DA1 = Stateo(33);
DT0 = Stateo(34); D0 = Stateo(35); DA0 = Stateo(36);
DTp = Pango(4); DTp = Pango(5); DAp = Pango(6);

T5dot = Stateo(19); B5dot = Stateo(20); A5dot = Stateo(21);
T4dot = Stateo(22); B4dot = Stateo(23); A4dot = Stateo(24);
T3dot = Stateo(25); B3dot = Stateo(26); A3dot = Stateo(27);
T2dot = Stateo(28); B2dot = Stateo(29); A2dot = Stateo(30);
T1dot = Stateo(31); B1dot = Stateo(32); A1dot = Stateo(33);
T0dot = Stateo(34); B0dot = Stateo(35); A0dot = Stateo(36);
Tpdot = Pango(4); Bpdot = Pango(5); Apdot = Pango(6);
end
```
D.3  \textit{sub\_timevar.mlx}: Replaces angles in variables with time-dependent angles

Valerie Jordon
Department of Bioengineering, University of Kansas

\textit{sub\_timevar.mlx}

Last Edited: 4/30/2002

This function is used to replace variables without time-dependency to variables with time-dependency.

```matlab
function [output]=sub_timevar(input)
syms T5 R5 A5 T4 R4 A4 T3 R3 A3 T2 R2 A2 T1 R1 A1 T0 R0 A0 Tp Ap dp
syms T5_val R5_val A5_val T4_val R4_val A4_val T3_val R3_val A3_val T2_val R2_val A2_val T1_val R1_val A1_val T0_val R0_val A0_val Tp_val Ap_val
evaluates([input,T5 R5 T4 R4 A5 T3 R3 A3 T2 R2 A2 T1 R1 A1 T0 R0 A0 Tp Ap dp],[T5_val R5_val A5_val T4_val R4_val A4_val T3_val R3_val A3_val T2_val R2_val A2_val T1_val R1_val A1_val T0_val R0_val A0_val Tp_val Ap_val])
evaluates(output,[T5_val R5_val A5_val T4_val R4_val A4_val T3_val R3_val A3_val T2_val R2_val A2_val T1_val R1_val A1_val T0_val R0_val A0_val Tp_val Ap_val])
evaluates([output,T5_val R5_val A5_val T4_val R4_val A4_val T3_val R3_val A3_val T2_val R2_val A2_val T1_val R1_val A1_val T0_val R0_val A0_val Tp_val Ap_val])
evaluates([output],T5_val R5_val A5_val T4_val R4_val A4_val T3_val R3_val A3_val T2_val R2_val A2_val T1_val R1_val A1_val T0_val R0_val A0_val Tp_val Ap_val)
end
```
D.4 *tofuncts.m*: Convert Symbolic Outputs to Function Handles

```matlab
%Valerie Jordon
%Department of Bioengineering, University of Kansas
%tofuncts.m
%Last Edited: 4/30/2022

%D;escription: This function (tofuncts.m) is used to convert our symbolic
%outputs to function handles using the built-in MATLAB function
%m;labFunction().

function [Gl_func,Gr_func,Mc_func,Mg_func,ML_func,NV_func, Ea_func,Rm_func,dRm_func,origin_func,dorigin_func]=tofuncts()
syms T5 T5dot B5 B5dot A5 A5dot T4 T4dot B4 B4dot A4 A4dot T3 T3dot B3 B3dot A3 A3dot T2 T2dot B2 B2dot A2 A2dot T1 T1dot B1 B1dot A1 A1dot T0 T0dot B0 B0dot A0 A0dot Tp Ap Bp Tpdot Bpdot Apdot
syms Tpdot_val Bpdot_val Apdot_val T5_val T5dot_val B5_val B5dot_val A5_val A5dot_val T4_val T4dot_val B4_val B4dot_val A4_val A4dot_val T3_val T3dot_val B3_val B3dot_val A3_val A3dot_val T2_val T2dot_val B2_val B2dot_val A2_val A2dot_val T1_val T1dot_val B1_val B1dot_val A1_val A1dot_val T0_val T0dot_val B0_val B0dot_val A0_val A0dot_val
la lb Apdot Bpdot Tpdot Bpdot Tpdot Bpdot A5dot T4dot B4dot A4dot T3dot B3dot A3dot T2dot B2dot A2dot T1dot B1dot A1dot T0dot B0dot A0dot Tpdot Bpdot Apdot T5_val T5dot_val B5_val B5dot_val A5_val A5dot_val T4_val T4dot_val B4_val B4dot_val A4_val A4dot_val T3_val T3dot_val B3_val B3dot_val A3_val A3dot_val T2_val T2dot_val B2_val B2dot_val A2_val A2dot_val T1_val T1dot_val B1_val B1dot_val A1_val A1dot_val T0_val T0dot_val B0_val B0dot_val A0_val A0dot_val

%load the symbolic output of our files
load('rotation_matrices_sym_0toP.mat') %rotation matrices
load('origin_sym_0toP.mat') %origin
load('rotate_anatomy_sym_MaMLMV.mat') %Ma, ML, MV
load('dynam_sym_output.mat') %Gl, Gr, Mm, Ch, Gm
load('Ea_sym.mat') %Ea

%rotate anatomy sym outputs
Ma_func=matlabFunction(Ma);
ML_func=matlabFunction(ML);
MV_func=matlabFunction(MV);

%rotate external force sym output
Ea_func=matlabFunction(Ea);

%rotation matrices and origins
Rm_func=matlabFunction(rotation_matrices);
dRm_func=matlabFunction(d_rotation_matrices);
origin_func=matlabFunction(origin);
dorigin_func=matlabFunction(d_origin);

%dynamics sym outputs
Gl=subs(Gl,[T5 T5dot B5 B5dot A5 A5dot T4 T4dot B4 B4dot A4 A4dot T3 T3dot B3 B3dot T2 T2dot B2 B2dot A2 A2dot T1 T1dot B1 B1dot A1 A1dot T0 T0dot B0 B0dot A0 A0dot Tp Ap Bp Tpdot Bpdot Apdot],[T5_val T5dot_val B5_val B5dot_val A5_val A5dot_val T4_val T4dot_val B4_val B4dot_val A4_val A4dot_val T3_val T3dot_val B3_val B3dot_val A3_val A3dot_val T2_val T2dot_val B2_val B2dot_val A2_val A2dot_val T1_val T1dot_val B1_val B1dot_val A1_val A1dot_val T0_val T0dot_val B0_val B0dot_val A0_val A0dot_val
```

244
B0(t) B0dot(t) A0(t) A0dot(t) Tp(t) Bp(t) Ap(t) Tp0(t) Bp0(t) Ap0(t), [T5_val \\ T5dot_val B5_val B5dot_val A5_val A5dot_val T5_val T4dot_val B4_val B4dot_val A4_val \\ A4dot_val T3_val T3dot_val B3_val B3dot_val A3_val A3dot_val T2_val T2dot_val B2_val B2dot_val \\ B2dot_val A2_val A2dot_val T1_val T1dot_val B1_val B1dot_val A1_val A1dot_val T0_val, T0dot_val \\ T0dot_val B0_val B0dot_val A0_val A0dot_val Tp_val Bp_val Ap_val Tp0_val Bp0_val Ap0_val \\ Ap0_val];

Gm=subs(Gm, [T5 T5dot B5 B5dot A5 A5dot T4 T4dot B4 B4dot A4 A4dot T3 T3dot B3 B3dot \\ A3 A3dot T2 T2dot B2 B2dot A2 A2dot T1 T1dot B1 B1dot A1 A1dot T0 T0dot B0 B0dot A0 \\ A0dot Tp Bp Ap Tp0 Bp0 Ap0], [T5_val T5dot_val B5_val B5dot_val A5_val A5dot_val \\ T4_val T4dot_val B4_val B4dot_val A4_val A4dot_val T3_val T3dot_val B3_val B3dot_val \\ A3_val A3dot_val T2_val T2dot_val B2_val B2dot_val A2_val A2dot_val T1_val T1dot_val \\ B1_val B1dot_val A1_val A1dot_val T0_val T0dot_val B0_val B0dot_val A0_val A0dot_val \\ Tp_val Bp_val Ap_val Tp0_val Bp0_val Ap0_val]);

syms T5 T5dot B5 B5dot A5 A5dot T4 T4dot B4 B4dot A4 A4dot T3 T3dot B3 B3dot A3 A3dot \\ T2 T2dot B2 B2dot A2 A2dot T1 T1dot B1 B1dot A1 A1dot T0 T0dot B0 B0dot A0 A0dot Tp \\ Bp Ap Tp0 Bp0 Ap0;


Gm_function=matlabFunction(Gm);
Gm_function=matlabFunction(Gm);
end