Optimizing road test simulation using neural network modeling techniques

Jennifer Leslie Johrendt

University of Windsor

Follow this and additional works at: https://scholar.uwindsor.ca/etd

Part of the Mechanical Engineering Commons

Recommended Citation

https://scholar.uwindsor.ca/etd/5248

This online database contains the full-text of PhD dissertations and Masters' theses of University of Windsor students from 1954 forward. These documents are made available for personal study and research purposes only, in accordance with the Canadian Copyright Act and the Creative Commons license—CC BY-NC-ND (Attribution, Non-Commercial, No Derivative Works). Under this license, works must always be attributed to the copyright holder (original author), cannot be used for any commercial purposes, and may not be altered. Any other use would require the permission of the copyright holder. Students may inquire about withdrawing their dissertation and/or thesis from this database. For additional inquiries, please contact the repository administrator via email (scholarship@uwindsor.ca) or by telephone at 519-253-3000 ext. 3208.
OPTIMIZING ROAD TEST SIMULATION USING NEURAL NETWORK MODELING TECHNIQUES

By

Jennifer Leslie Johrendt

A Dissertation
Submitted to the Faculty of Graduate Studies and Research through Mechanical Engineering in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy at the University of Windsor

Windsor, Ontario, Canada

2005

© 2005 Jennifer L. Johrendt
NOTICE: The author has granted a non-exclusive license allowing Library and Archives Canada to reproduce, publish, archive, preserve, conserve, communicate to the public by telecommunication or on the Internet, loan, distribute and sell theses worldwide, for commercial or non-commercial purposes, in microform, paper, electronic and/or any other formats.

The author retains copyright ownership and moral rights in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

In compliance with the Canadian Privacy Act some supporting forms may have been removed from this thesis.

While these forms may be included in the document page count, their removal does not represent any loss of content from the thesis.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
ABSTRACT

Growing interest in the use of virtual simulation tools as part of the automotive product development process is driven by the need for automotive manufacturers and parts suppliers to develop better quality products in shorter time at lower cost.

Component and full vehicle durability testing is one aspect of product development for which time savings can be realized. Traditionally, accelerated durability simulations have been performed using full vehicles by driving physical prototypes on specially designed road surfaces, simulating the vehicles' service life. In the last thirty years, durability testing has been accelerated in the laboratory environment where measured vehicle excitation inputs have been edited to contain only the most damaging portions. The goal of the current research is to advance the process further through the use of high-fidelity virtual prototype durability simulations, which reveal the consequences of design decisions made much earlier in the product development cycle before the first physical prototypes are built.

Virtual durability full vehicle models are computationally complex. Linearizing the individual models of nonlinear components such as shock absorbers and elastomeric bushings has been a typical method used to simplify the vehicle model. The focus of the current research is to develop a methodology to increase the fidelity of these nonlinear component models using computationally economical techniques, thus increasing the precision of the results of the full vehicle model and the speed at which the results are obtained.
Neural networks are mathematical models that possess the flexibility and computational efficiency desired for this application. These models are capable of generalizing component behaviour using training data that represents the full range of component behaviour that is to be modeled.

The current research describes the methodology required to develop and implement neural network models of nonlinear automotive components into simplified and full-vehicle virtual durability models. The data used to train the neural networks includes hysteresis effects that are not modeled with the methods currently available in the multibody dynamics software package. Correlation of the results of the virtual durability simulation with the laboratory test results is performed to show the validity of the methodology that was developed.
ACKNOWLEDGEMENTS

The author gratefully acknowledges the support, inspiration, and dedication of her academic and industrial supervisors, Dr. Peter R. Frise and Mr. Mohammed A. Malik. Together, they not only provided solid guidance during the course of this research, but also the trust and encouragement necessary to foster independent study. As people, engineers, and educators, they are wonderful mentors, role models, and friends.

Thanks to DaimlerChrysler Canada Inc. for their generosity of time and resources during this research project. Access to the equipment and personnel of the University of Windsor/DaimlerChrysler Canada Automotive Research and Development Centre made this project possible.

The author expresses her deep appreciation for the generous funding of the Natural Sciences and Engineering Research Council of Canada and DaimlerChrysler Canada Inc. through the Industrial Post-Graduate Scholarship program.

I dedicate this work to my family and friends. George’s love and encouragement has motivated me through the challenging moments and helped me celebrate each small success along the way. I am grateful that Jennifer and Jonathan have always been my biggest supporters. Throughout our years together, we have learnt that the potential of our family unit is greater than the sum of its individual parts. We manage our triumphs and tribulations better than any one of us could on their own. I can’t neglect to mention our four-legged family members, Jake, Heidi, and Bentley. I was always grateful for their loyal companionship and laughable antics. Finally, thanks to my parents for their unaltering confidence and to my sister who has been an inspiration in so many ways.
TABLE OF CONTENTS

Abstract ......................................................................................................................................... iii
Acknowledgements ....................................................................................................................... v
List of Figures ................................................................................................................................ x
List of Tables ........................................................................................................................... xxxi
List of Abbreviations ............................................................................................................ xxxiv
List of Symbols ....................................................................................................................... xxxv

1 Introduction ........................................................................................................................ 1
   1.1 Product Development ............................................................................................... 1
   1.2 Vehicle Durability: Field and Laboratory Testing ..................................................... 3
      1.2.1 Fatigue and Durability Testing ....................................................................... 3
      1.2.2 Accelerated Durability Tests ......................................................................... 4
      1.2.3 Road Test Simulation ...................................................................................... 5
   1.3 Virtual Durability Simulation .................................................................................. 6
   1.4 Neural Networks ...................................................................................................... 7
   1.5 Research Objectives of the Present Study ............................................................. 9

2 Literature Review .............................................................................................................. 11
   2.1 Product Development ............................................................................................ 11
   2.2 Durability and Road Test Simulation ................................................................... 12
   2.3 Modeling Nonlinear Components ........................................................................ 13
   2.4 Using Neural Networks to Model Automotive Components .............................. 15

3 Theory ............................................................................................................................... 17
   3.1 Vehicle Dynamics .................................................................................................... 17
      3.1.1 Ride and Handling vs. Durability ................................................................. 17
      3.1.2 The Role of Suspension Components ......................................................... 18
   3.2 Elastomers ................................................................................................................ 18
      3.2.1 Polymer Structure of Elastomers ................................................................. 18
      3.2.2 Mechanics of Viscoelastic Behaviour ......................................................... 21
      3.2.3 Traditional Models of Viscoelastic Behaviour ............................................. 25
3.3 Introduction to Neural Networks .......................................................... 28
3.4 Designing Neural Networks .................................................................... 31
  3.4.1 Methods of Construction ................................................................. 33
  3.4.2 Activation Functions ......................................................................... 34
  3.4.3 Scaling the Input and Output Data .................................................... 35
  3.4.4 Number of Hidden Layer Perceptrons .............................................. 36
  3.4.5 Training and Validation Data Sets .................................................... 37
  3.4.6 Measuring the Performance of the Network ...................................... 37
  3.4.7 Determining Appropriate Weights and Bias Values ......................... 42
3.5 Using Neural Networks to Characterize Material Behaviour .................. 49
4 Applying Neural Network Modeling Techniques ........................................ 51
  4.1 ADAMS Model Preparation ................................................................. 52
  4.2 ADAMS Model Components ............................................................... 53
    4.2.1 Location and Orientation of Markers .............................................. 53
    4.2.2 State Variables ............................................................................... 54
    4.2.3 ADAMS/View Function Builder ....................................................... 55
    4.2.4 ADAMS/Controls ........................................................................... 57
    4.2.5 Arrays and Filters ........................................................................... 58
    4.2.6 ADAMS Dampers ........................................................................... 59
    4.2.7 ADAMS Bushings ........................................................................... 63
  4.3 Modified ADAMS Components ............................................................ 67
    4.3.1 Modified Dampers .......................................................................... 67
    4.3.2 Modified Bushings .......................................................................... 71
  4.4 ADAMS/Controls Plug-in Module ......................................................... 76
  4.5 Neural Network Construction ............................................................... 78
    4.5.1 Spline Data .................................................................................... 79
    4.5.2 Hysteresis Data ............................................................................... 86
    4.5.3 Time History Data ........................................................................... 91
  4.6 Co-simulation ..................................................................................... 95
    4.6.1 Simulink ........................................................................................ 96
5 Simulation Results and Analysis ............................................................... 103
5.1 Neural Network Model of a Shock Absorber Force-Deflection Spline........... 103
5.2 Neural Network Model of a Shock Absorber Force-Deflection Hysteresis Curve ........................................................................................................................ 109
5.3 Neural Network Results: Bushing Time Series Data................................. 116
  5.3.1 Axial Force Model ...................................................................................... 120
  5.3.2 Radial Force Model ..................................................................................... 123
  5.3.3 Axial Torque Model .................................................................................... 126
  5.3.4 Conical Torque Model ................................................................................ 129
5.4 Implementation and Validation of Vehicle Model ........................................ 132
  5.4.1 Co-simulation Using a Neural Network Model of a Shock Absorber Force versus Deflection Velocity Spline Curve ......................................... 133
  5.4.2 Co-simulation Using a Neural Network Model of a Shock Absorber Force versus Deflection Velocity Hysteresis Curve .................................. 134
  5.4.3 Co-simulation Using a Neural Network Model of Bushing Forces and Torques ................................................................................................................. 140
5.5 Correlation: Laboratory and Simulation Results .......................................... 174
6 Conclusions and Recommendations ................................................................. 180
  6.1 Conclusions ...................................................................................................... 180
  6.2 Recommendations ........................................................................................... 181
    6.2.1 Neural Network Development .................................................................... 181
    6.2.2 Neural Network Implementation ................................................................. 182
Appendix A: Bushing Data Acquisition ................................................................. 184
  A.1 Data Acquisition Overview ............................................................................ 184
  A.2 Desired Data Preparation ................................................................................ 185
  A.3 Test Set-up ....................................................................................................... 194
    A.3.1 F_x, Axial Force Measurement Set-up .......................................................... 195
    A.3.2 F_z, Radial Force Measurement Set-up ....................................................... 195
    A.3.3 T_x, Axial Torque Measurement Set-up ...................................................... 197
    A.3.4 T_z, Conical Torque Measurement Set-up ................................................. 198
  A.4 Data Acquisition .............................................................................................. 199
    A.4.1 Measurement of F_x and F_z ....................................................................... 199
LIST OF FIGURES

Figure 1.1: The traditional product development cycle can be an iterative process, with revisions occurring late in the cycle. ......................................................... 1

Figure 1.2: The cost of making changes during product development cycle grows dramatically with time. ................................................................. 2

Figure 1.3: The revised product development cycle allows for design iterations earlier in the process................................................................. 2

Figure 1.4: Vehicle on test at a corporate proving grounds. (Photograph courtesy of DaimlerChrysler Corporation.) ......................................................... 4

Figure 1.5: Test vehicle on a road test simulator during the drive file development process. (Photograph courtesy of DaimlerChrysler Canada Inc.) .......... 5

Figure 1.6: Virtual durability simulation model of a road test simulator and a vehicle in ADAMS ................................................................. 6

Figure 1.7: Neurons receive and transmit signals in the brain. ......................................................... 7

Figure 1.8: Signals flow through the neural network along the weighted paths. .......... 8

Figure 1.9: Front suspension components that will be modelled with neural networks are coloured red (front lower control arm front bushing, front shock absorber). [5] ......................................................... 9

Figure 2.1: Roles of participants in traditional product development [13]. .............. 11

Figure 2.2: The front lower control arm bushing is oriented as shown with the X-axis pointing towards the front of the vehicle ................................................. 13

Figure 2.3: The curve on the left is similar to a one-to-one curve used to approximate the hysteresis normally exhibited by a damper, as shown on the right .... 16

Figure 3.1: Polysisoprene is a polymer of carbon (C) and hydrogen (H) atoms formed into long chains of repeating units or “mers”. The polysisoprene “mer” is shown here repeating 'n' times. Polysisoprene is commonly called natural rubber. ......................................................... 19

Figure 3.2: The addition of sulfur to polysisoprene forms vulcanized rubber, an elastomeric material commonly used in the manufacture of automotive bushings. ......................................................... 19

Figure 3.3: The glass transition temperature is the temperature at which the vulcanized rubber loses its viscous properties and becomes brittle. The magnitude of the change in modulus is small for the stiffest rubber materials. (Adapted from Figure 23.8 from [1]) ......................................................... 21

Figure 3.4: During viscoelastic creep, stress is held constant while strain increases.. 22
Figure 3.5: During viscoelastic relaxation, stress decreases as the strain is held constant. ................................................................. 23

Figure 3.6: This is a typical stress versus strain curve for an elastic material that exhibits yielding. Note that when the material is loaded past the yield stress, unloading occurs along a parallel path (i.e. with the same slope and thus the same elastic modulus) to a state of permanent (plastic) deformation. ........................................................................................................ 24

Figure 3.7: The strain response lags the applied dynamic stress to a viscoelastic material as illustrated by the phase lag between the two signals. ........ 24

Figure 3.8: Viscoelastic stresses exhibit a lagged strain response that develops into a hysteresis curve in a stress versus strain plot. The arrows indicate the progression of time, such that loading and unloading of the specimen are shown. ................................................................. 25

Figure 3.9: The Maxwell and Voigt-Kelvin viscoelastic models use springs to model the elastic component of the behaviour (stiffness, E) and viscous dashpots to model the viscous behaviour of the material (damping, c). ......................... 26

Figure 3.10: The Voigt element better predicts strain during creep than the Maxwell element for which strain varies linearly during constant applied stress. ... 27

Figure 3.11: The Maxwell element better predicts stress during relaxation than the Voigt element for which stress is constant during constant applied strain. .................................................................................................................. 27

Figure 3.12: The biological neuron accepts signals from other neurons, processes the combined signal at the nucleus, and sends the resulting signal along axons downstream to other neurons [20]. ................................................................. 28

Figure 3.13: For negative values of the input variable, net, the output of the function \( f(\text{net}) \) is either -1. For positive or zero values of the input variable, net, the output of the function \( f(\text{net}) \) is +1. The threshold nonlinearity function is described mathematically by \( f_{\text{threshold}}(\text{net}) = \begin{cases} 1, & \text{net} \geq 0 \\ -1, & \text{net} < 0 \end{cases} \) ................. 29

Figure 3.14: The perceptron is a mathematical representation of the biological neuron and used as the neural network building block. The \( i^{th} \) perceptron of a neural network is shown here. ................................................................. 30

Figure 3.15: The neural network shown is an example of a 2-3-1 MLP that uses two inputs \( (p_1, p_2) \) to calculate the output, \( y_1 \), using a hidden layer of three perceptrons. ........................................................................................................ 31
Figure 3.16: The threshold, hyperbolic tangent (tanh), and logistic functions are nonlinear activation functions commonly used in neural network design. The variable net is input to the activation function that acts as a trigger or firing threshold to produce an output. The continuous differentiability of the hyperbolic tangent, and logistic functions make them more desirable for neural network structures because this property will allow calculation of the back propagation of the error during the training phase of network construction.

Figure 3.17: The network input data used in the simulation is scaled using the maximum and minimum values of the training input data. The scale of the simulation outputs of the neural network is recovered using the maximum and minimum values of the training data outputs.

Figure 3.18: The single perceptron employs a linear activation function to estimate a single output value from n inputs. It will be used to illustrate the concept of optimizing the performance of the network during training.

Figure 3.19: In this example of the performance of a single layer network using a linear activation function, the value of the MSE is minimized for w*=0.75.

Figure 3.20: The use of hyperbolic activation functions causes local minima to arise in the performance minimization. The learning algorithm should be modified to ensure that the true optimal weight is found at the global minimum of the performance function.

Figure 3.21: The ith perceptron accepts n inputs and produces one output. Note that the weights, wij, and bias, bi, are the variable parameters that are adjusted during training of the neural network.

Figure 3.22: The bias of the ith perceptron in Figure 3.21 can be replaced by an additional input element, +1, and a corresponding weight, bi.

Figure 3.23: The output of the jth perceptron is weighted with wji and passed to the ith perceptron in the hidden layer, weighted with wki and passed to the kth perceptron that resides in the output layer.

Figure 3.24: The value of the weight, wi, is adjusted by a rate proportional to the gradient of the MSE curve. In this fashion, the optimal value, w*, is eventually reached after a sufficient number of iterations.

Figure 3.25: The network weights displayed in the histogram show the highest concentration of weights with small magnitude and form a bell-shape about the midpoint. Weights larger than about ten are undesirable due to the amplification effect on errors passing through that weighted path of the network.

Figure 3.26: This normalized damping force versus normalized shock deflection velocity curve shows the nature of the hysteresis exhibited by measured data. Note the separate hysteresis loops that result as the frequency and amplitude of the signal change.
Figure 4.1: Data is passed to and from the ADAMS model from the Simulink environment where the neural networks reside. .......................................................... 51

Figure 4.2: This example of a 90-90-90 Euler transformation illustrates three 90° rotations about the z-, x'-, and z''-axes, respectively, to transform the x - y - z coordinate system into x - y - z coordinate system. In the ADAMS software, x - y - z represents the global coordinate system, and the marker coordinate system, x - y - z, is said to have the orientation (90, 90, 90) relative to the global coordinate system......................................................... 54

Figure 4.3: A control system monitors the output variables of a plant process. Adjustments to the plant input variables are made by a controller to achieve output variables that are closer to desired values [8]......................... 55

Figure 4.4: The transfer function, TFSISO_lp_filter, represents the low pass filter used to filter the incoming damping force data has a cutoff frequency of 100Hz.............................................................. 59

Figure 4.5: The damper exerts equal and opposite forces that act along the line-of-sight between Sprung_mass.MARKER_3 and Unsprung_mass.MARKER_4. The dialog box is used to define the bodies and the damping coefficient constant expressed in units of N-s/mm or a spline curve that defines the damping force with respect to the velocity of the shock absorber deflection......................................................... 61

Figure 4.6: The spline curve represents a nonlinear approximation of the damping force with respect to the rate of deflection of the damper. Both force and velocity data have been normalized with respect to their maximum respective values to protect proprietary data. Rebound or extension of the damper is produces a positive deflection velocity and damping force, while the jounce condition or compression of the damper produces negative deflection rate and damping force ................................................. 63

Figure 4.7: The bushing exerts three translational and three rotational forces between shaft.MARKER_4 and sleeve.MARKER_3. Note that the ADAMS bushing definition aligns the z-axis along the bushing’s axial direction. If the bushing is symmetrical, the x-axis and y-axis coefficients are identical. The dialog box is used to define the bodies and the constant stiffness and damping coefficients................................................................. 65

Figure 4.8: The dialog boxes for creation of state variables are shown. The damping force variable (left) acts as a placeholder for incoming data and must show a value of zero (0). The deflection velocity (right) is an example of a variable that whose value will be calculated and output to MATLAB for processing. Note that it is a measurement of the relative velocity along the line-of-sight between the markers.......................................................... 68

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
Figure 4.9: *ARRAY_damping_input* (left) is an input array (U) that used to filter *damping_force_variable* (input to ADAMS from MATLAB) using the transfer function, *TFSISO_lp_filter*. *ARRAY_filtered_damping* is the output array (Y) from *TFSISO_lp_filter*, the value of which is then used as the damping force.

Figure 4.10: The modified damping force definition calls upon the filtered value of the damping force that was input from MATLAB.

Figure 4.11: The modified damping force acts along the line of sight between the unsprung and sprung masses. It is defined to accept data from MATLAB input through the Controls Plug-in module.

Figure 4.12: This figure shows the orientation of a bushing that is defined to produce three forces and three torques that act between an action and a reaction body (this example shows a shaft acting as the action body and a sleeve acting as the reaction body). Note the distinction between the axial direction (z-axis) and the two radial directions (x-axis and y-axis), indicating that the largest rotation occurs about the z-axis. Asymmetric bushings possess different bushing stiffness and damping properties in each of the two radial directions.

Figure 4.13: The dialog boxes for creation of state variables are shown. The bushing axial force variable (left) acts as a placeholder for incoming data and must show a value of zero (0). The x-axis deflection (right) is an example of a variable that whose value will be calculated and output to MATLAB for processing. Note that it is a measurement of the x-component of the relative translation vector between the markers.

Figure 4.14: *ARRAY_axial_force_input* (left) is used to pass *Axial_force_variable* (input to ADAMS from MATLAB) as input to the low pass filter (e.g., *TFSISO_lp_filter*). *ARRAY_filtered_axial_force* contains the output of the filter, which is then passed onto the bushing force element.

Figure 4.15: The modified bushing force definition is defined by the filtered values of the forces and torques that were input from MATLAB.

Figure 4.16: The modified bushing force acting between MARKER_5 and MARKER_6 is now ready to accept data from MATLAB input through the ADAMS Controls Plug-in module.

Figure 4.17: The plant input and output define which state variables will feed the information into and out of ADAMS to MATLAB.

Figure 4.18: Exporting the ADAMS subsystem representation to external control packages, like MATLAB.

Figure 4.19: A neural network developed for the training data input and output pairs shown here will produce a model that will allow interpolation within the limits of the data.
Figure 4.20: The Network/Data Manager is the first graphic user interface that opens when using the Neural Network Toolbox. The user can create or import data and neural networks. ................................................................. 80

Figure 4.21: The 1-1-1 feed forward network defined in the graphic user interface accepts input data ranging from -1 to 1, trains using the gradient descent with momentum method (traingdm), and monitors performance using the MSE. The activation function chosen for use in the hidden layer is the hyperbolic tangent while the output layer employs a linear function.... 81

Figure 4.22: The Network graphic user interface allows processing of the network, including viewing (shown), training (shown), simulating, and display of the weights and bias values................................................................. 82

Figure 4.23: The spline data points were separated into training and validation data sets as shown in the plot................................................................. 84

Figure 4.24: The hysteresis curve that is based upon the ADAMS spline curve will be modeled using a time delay neural network structure. .................. 87

Figure 4.25: The damping hysteresis data points were separated into training and validation data sets as shown in the plot. The validation points were randomly selected from the complete set of thirty points on the hysteresis curve. Their locations on the hysteresis curve are shown here.......... 90

Figure 4.26: The Simulink model shows the flow of data during the co-simulation. The adams_sub block represents the ADAMS component of the model. In the example shown, the neural network estimated damping force is passed to the adams_sub block where the ADAMS solvers are employed to calculate the state of the system for the current time step. The input or excitation force, damper deflection, and damper deflection velocity are output and plotted together with the damping force. The damper deflection velocity is fed back and input to the neural network for estimation of the damper force for use in the next time step calculation. 95

Figure 4.27: The neural network block produced by the "gensim" command is core to the neural network subsystem shown here. The incoming data is scaled using the maximum and minimum training data values, passed through the neural network, and then the resulting output is scaled back to the original range of the training target data. .................................................. 96

Figure 4.28: The use of time delay data as input to the neural network not only requires an additional "Transport Delay" block be added to the main model, but the neural network subsystem must be modified to accommodate the pre- and post-processing of more than one input........................................ 98

Figure 4.29: The parameters of the ADAMS plant include the name of output files, communication interval, mode of simulation, and mode of animation. Co-simulation is achieved by selecting the discrete mode of simulation, whereby the ADAMS solver is used to evaluate the multi-body dynamics of the system........................................ 100
The Simulink library contains many sets of predefined blocks including function generators, plotters, transport delay, gains, summing, and signal multiplexers.

The spline curve is fitted to the measured shock absorber force versus deflection velocity data points. Note the differences in slope of the curve between jounce and rebound conditions. This curve will be modeled by a neural network.

The 1-3-1 neural network was developed using all fifteen spline points for training and did not use early stopping. The training yielded an $MSE$ of 0.0010.

The 1-5-1 neural network was also developed using all fifteen spline points and did not use early stopping. In this case, the training also yielded an $MSE$ of 0.0010. Since the slightly more complicated network structure yielded the same performance as the simpler network, the simpler network structure of the 1-3-1 neural network is preferable to the 1-5-1 network without early stopping.

The 1-5-1 neural network that was developed with early stopping could only utilize thirteen spline points for training and two data points for validation. The network performance during training was better than that of the networks developed without early stopping ($MSE = 0.0006$). Further examination of this network’s accuracy compared to the 1-3-1 network of Figure 5.2 is required before a model can be selected.

The results of the 1-3-1 and 1-5-1 neural networks are shown with the ADAMS spline data. In general, the 1-3-1 network that was trained without early stopping models the ADAMS spline curve better, especially in the region of higher rebound velocity, where there are fewer training points available during training.

The weights of the 1-3-1 neural network that was trained without validation data, shows that the weights range from about -4 to 4. The compact range of the weights is desirable, but the low quantity of weights in the network prevents the ability of the histogram to take on a more distinctive bell-shape that is desirable of such a distribution.

The weights of the 1-5-1 neural network that was trained with validation data, shows the weights range from about -7 to 7. The increased range of the values of the weights makes it less desirable than the histogram of the 1-3-1 network weights shown in Figure 5.6. This histogram does not exhibit the desired bell-shape.
Figure 5.8: The hysteresis curve differs from the spline curve in that the upper and lower branches correspond to different loading of the shock absorber. The upper branch shows that a larger force is produced in the damper than the lower branch. The branches correspond to shock absorber motion between maximum jounce velocity to maximum rebound velocity. A spline curve of shock absorber damping approximates the force produced regardless of the direction of loading. The hysteresis curve shown here was generated using the spline curve data used to model the shock absorber in ADAMS.

Figure 5.9: The 1-3-1 neural network was developed without early stopping using all thirty data points for training. The $MSE$ achieved during training was 0.0010. Note that the hysteresis was not modeled by this network structure. Instead, the model produces shock absorber forces between the upper and lower branches of the hysteresis curve.

Figure 5.10: The 2-7-1 neural network was trained with all thirty data point and did not use early stopping. The $MSE$ for the training data was 0.0010. The hysteresis is evident in the model, but does not follow the curve very well for jounce conditions (negative velocity).

Figure 5.11: The 3-7-1 neural network also used all thirty data points for training and no early stopping. Again, the training yielded an $MSE$ of 0.0010. The network shows better estimation of the hysteresis curve than the 2-7-1 model shown in Figure 5.10.

Figure 5.12: The 2-5-1 neural network was developed with early stopping, using twenty-seven points for training and three data points for validation. An $MSE$ value of 0.0041 was achieved. The hysteresis is evident in the model, but does not follow the data points to adequately generalize the target data as is also shown in the linear regression plot on the right by the spread of data about the line of regression between the target and network output.

Figure 5.13: The 3-7-1 neural network was trained using early stopping. Twenty-seven points were used for training and three data points for validation, yielding an $MSE$ of 0.0031. As in Figure 5.12, the hysteresis is evident in the model, but does not follow the data points to adequately estimate the target data as supported by the spread of data about the linear regression line between the target and network output.

Figure 5.14: The weights of the 2-7-1 neural network that was trained without early stopping range from about $-3$ to $3$. The higher number of weights at the boundaries makes the 2-7-1 network structure less desirable than the 2-5-1 network shown in Figure 5.15 because of the tendency of higher weights to amplify errors as they propagate through the network.
Figure 5.15: The weights of the 2-5-1 neural network that was trained with early stopping range from about -3 to 3. The distribution more closely resembles a bell-shaped curve compared to the 2-7-1 network shown in Figure 5.14.  

Figure 5.16: The results of the 2-7-1 and 2-5-1 neural networks are shown with the original hysteresis data as the velocity is swept through from maximum jounce to maximum rebound. Both networks model the hysteresis curve well, but the 2-5-1 model was selected due to its better distribution of weights per the histogram shown in Figure 5.15. The discrepancy of both curves to model the hysteresis data at the far left of the plot is due to the use of zero initial time delay values for the networks' input.  

Figure 5.17: Bushing force and translational deflection data was measured in the axial and radial directions. Bushing torque and rotation deflection data was measured in the axial and radial directions. Radial torque is also referred to as conical torque. Since the bushing was symmetric, the models of the radial force and torque can be applied in two perpendicular radial directions during co-simulation.  

Figure 5.18: This plot illustrates typical results of the calculation of the sensitivity of the axial force with respect to the ten inputs. In general, it was determined that both the axial and radial bushing forces showed higher sensitivity to zero time-step delay displacement and velocity inputs compared to the acceleration and the one- and two-time delay inputs of the displacement, velocity, and acceleration.  

Figure 5.19: This plot illustrates typical results of the calculation of the sensitivity of the conical torque with respect to the ten inputs. In general, it was determined that both the axial and conical bushing torques showed higher sensitivity to zero and one time-step delay rotation and angular velocity inputs compared to the remaining inputs.  

Figure 5.20: The weights of the 2-3-1 axial force neural network model exhibit a bell-shape concentrated about zero ranging from -2 to 2. Note the compact range of the weights and the large concentration weights near zero. The lack of large weight values indicates greater model stability as discussed in Section 3.4.7.  

Figure 5.21: The ability of the 2-3-1 neural network to model the bushing axial force is shown here in the time domain. The good correlation for the training (top) and validation data (middle) illustrates the success of the network to generalize both the training and validation data sets during the training process. The success of the network to estimate the test data (bottom) shows the network's ability to model the axial force produced during axial translational bushing deflection that was not used during the training phase of network development.
Figure 5.22: The ability of the 2-3-1 neural network to estimate training, validation, and test data of the axial bushing force is shown here. Note the concentration of the cluster of test data points within the training and validation data point clusters. This illustrates the network’s ability to model data to which it was not exposed during the training phase and thus shows the efficacy of the neural network approach. ................................. 123

Figure 5.23: The weights of the 2-3-1 radial force neural network model exhibit a bell-shape concentrated about zero ranging from −5 to 4 ........................................ 124

Figure 5.24: The ability of the 2-3-1 neural network to model the bushing radial force is shown here in the time domain. The good correlation for the training (top) and validation data (middle) illustrates the success of the network to generalize the training and validation data sets during the training process. The success of the network to estimate the test data (bottom) shows the network’s ability to model the radial force produced during radial translational bushing deflection that was not used during the training phase of network development. ................................................................. 125

Figure 5.25: The ability of the 2-3-1 neural network to estimate training, validation, and test data of the radial bushing force is shown here. Note the concentration of the cluster of test data points within the training and validation data point clusters. This illustrates the network’s ability to model data to which it was not exposed during the training phase. ....... 126

Figure 5.26: The weights of the 4-7-1 axial torque neural network model exhibit a bell-shape concentrated about zero ranging from −4 to 4 ......................... 127

Figure 5.27: The time histories of the normalized axial torque from the training (top), validation (middle), and test data sets (bottom) are shown for the 4-7-1 neural network model. The time delay network structure successfully generalized the axial torque response to bushing axial rotation during training. Once the network was successfully trained, it was then able to correctly estimate the axial torque from previously unseen input data. 128

Figure 5.28: The ability of the 4-7-1 neural network to estimate training, validation, and test data of the axial bushing torque is shown here. Note the concentration of the cluster of test data points within the training and validation data point clusters. This illustrates the network’s ability to model data to which it was not exposed during the training phase. ....... 129

Figure 5.29: The weights of the 4-7-1 conical torque neural network model exhibit a bell-shape concentrated about zero ranging from −4 to 7 .................... 130

Figure 5.30: The time histories of the normalized conical torque from the training (top), validation (middle), and test data sets (bottom) are shown for the 4-7-1 neural network model. The time delay network structure successfully generalized the conical torque response to bushing conical rotation during training. Once the network was successfully trained, it was then able to correctly estimate the torque generated from previously unseen input data. ................................................................. 131

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
The ability of the 4-7-1 neural network to estimate training, validation, and test data of the conical bushing torque is shown here. Note the concentration of the cluster of test data points within the training and validation data point clusters. This illustrates the network's ability to model data to which it was not exposed during the training phase. The width of the clusters is a result of the small signal range relative to the axial force, axial torque, and radial force data.

The Simulink model shows the method used to run the co-simulation using the spline look-up table and the neural network model of the damping force. For the Simulink model shown, the neural network-estimated damping force is input to the "adams_sub" block.

The random input shown was one of the signals used to excite the single degree of freedom model during the co-simulation. A baseline response to the excitation was measured in ADAMS and then co-simulations using the spline and hysteresis neural network models were performed. The results are shown in Figure 5.34 to Figure 5.37.

This portion of the time history plot of the shock absorber force generated in the co-simulation of the single degree of freedom model for both the spline (blue) and hysteresis (green) neural network models are shown in comparison with ADAMS model baseline (black). The neural network model estimates higher positive and negative peaks in the time history. This corresponds to the position of the hysteresis forces relative to the spline curve shown in Figure 5.8.

The frequency plots of the shock absorber force generated in the co-simulation of the single degree of freedom model for both the spline (blue) and hysteresis (green) neural network models are shown in comparison with ADAMS model baseline (black) from 0 – 45 Hz. The full time history of the response to the random input signal is used to calculate this plot. In general, the hysteresis model estimates higher amplitude signals for most frequencies (e.g., $\sqrt{\frac{0.175}{0.105}} = 1.29$ at 1 Hz). This agrees with the position of the hysteresis forces relative to the spline curve shown in Figure 5.8. Note that all signal amplitudes are very small at frequencies above 10 Hz.

This portion of the time history plot of the shock absorber deflection generated in the co-simulation of the single degree of freedom model for both the spline (blue) and hysteresis (green) neural network models are shown in comparison with ADAMS model baseline (black). The neural network model estimates slightly higher peaks for the higher frequency component of the response, but the low frequency content the motion shows good correlation for the three cases.
This frequency plot of the shock absorber deflection generated in the co-
simulation of the single degree of freedom model for both the spline (blue) 
and hysteresis (green) neural network models are shown in comparison 
with ADAMS model baseline (black) from 0 – 30 Hz. The full time 
history for the response to the random input signal is used to calculate this 
plot. The amplitudes of the spline and hysteresis models correlate well 
with the ADAMS model. At about 1 Hz, the amplitude of the hysteresis 
model is $\sqrt{0.107/0.118} = 0.95$ of the amplitude of the ADAMS baseline. 

The Simulink model shows that the axial displacement and velocity output 
from the ADAMS subsystem were used to calculate the axial bushing 
force and the axial rotation and angular velocity were used to calculate the 
bushing axial torque using the neural network models developed in 
Section 5.3. The axial force and torque were then input to the ADAMS 
subsystem. Subsystems are expanded and shown in Figure 5.39 to Figure 
5.41.

The calculation of the axial bushing force and torque uses the neural 
network models developed in Section 5.3. The inclusion of the transport 
delay of a single time step is included here to resolve the algebraic loop 
that results from using the axial displacement and velocity feedback to 
calculate the input force to the ADAMS subsystem. The gain values are 
used to ensure the correct polarity of the force as it acts between the lower 
control arm and the vehicle body in the ADAMS model.

The bushing axial force neural network subsystem contains the pre- and 
post-processing functions along with the neural network model that was 
developed in Section 5.3. The data defining the maximum and minimum 
avial displacements, velocities, and forces from the training data is used to 
scale the network inputs and outputs.

The bushing axial torque neural network subsystem contains the pre- and 
post-processing functions along with the neural network model that was 
developed in Section 5.3. The data defining the maximum and minimum 
avial rotations, angular velocities, and torques from the training data is 
used to scale the network inputs and outputs. Note the additional time 
delay block just previous to the 4-7-1 Torsional Neural Network block. 
This delay provides the delayed rotation and angular velocity inputs 
required for the neural network model.

The results of the co-simulation using neural network models of both the 
front and rear axial bushing forces show that the neural network model 
correlates extremely well with the ADAMS simulation with regard to the 
normalized front normalized bushing axial displacements. This portion of 
the results from 30 to 35 seconds contains the global peaks for the full 50 
second simulation showing that the method developed in the present 
research provides a valid simulation over the entire range of the physical 
phenomenon.
The results of the co-simulation using neural network models of both the front and rear axial bushing forces show that the neural network model correlates extremely well with the ADAMS simulation with regard to the normalized rear normalized bushing axial displacements. This portion of the results from 30 to 35 seconds contains the global peaks for the full 50 second simulation showing that the method developed in the present research provides a valid simulation over the entire range of the physical phenomenon.

The frequency plots for the front normalized axial displacements were calculated using the full 50 seconds of road simulation data. The plots show good correlation between the ADAMS simulation and the co-simulation using the axial force neural network model. The peak located at 0 Hz shows the results from the co-simulation results are about \[ \sqrt{\frac{0.04}{0.03}} = 1.15 \] times (about 15% larger than) the ADAMS simulation results, which corresponds to a small difference in the mean value of the signal through the full simulation.

The frequency plots for the rear normalized axial displacements were calculated using the full 50 seconds of road simulation data. The plots show good correlation between the ADAMS simulation and the co-simulation using the axial force neural network model. The peak located at 0 Hz shows the results from the co-simulation results are about \[ \sqrt{\frac{0.04}{0.03}} = 1.15 \] times (about 15% larger than) the ADAMS simulation results, which corresponds to a small difference in the mean value of the signal through the full simulation.
The frequency plot for the front normalized axial bushing force was calculated using the full 50 seconds of road simulation data. The plot shows good correlation for all frequencies between the ADAMS simulation and the co-simulation using the neural network models of both front and rear axial forces.

The frequency plot for the rear normalized axial bushing force was calculated using the full 50 seconds of road simulation data. The plot shows good correlation for all frequencies between the ADAMS simulation and the co-simulation using the neural network models of both front and rear axial forces.

The frequency plot for the front bushing normalized axial rotation was calculated using the full 50 seconds of road simulation data. The plot shows good correlation between the ADAMS simulation and the co-simulation using the axial force neural network model for all frequencies.

The frequency plot for the rear bushing normalized axial rotation was calculated using the full 50 seconds of road simulation data. The plot shows good correlation between the ADAMS simulation and the co-simulation using the axial force neural network model for all frequencies.

The results of the co-simulation using neural network models of both front and rear axial bushing torques show that the neural network model correlates well with the ADAMS simulation with regard to the front bushing normalized axial rotation. This portion of the results from 30 to 35 seconds contains the global peaks for the 50 second simulation showing that the method developed in the present research provides a valid simulation over the entire range of the physical phenomenon.

The results of the co-simulation using neural network models of both the front and rear axial bushing torques show that the neural network model correlates well with the ADAMS simulation with regard to the rear bushing normalized axial rotation. This portion of the results from 30 to 35 seconds contains the global peaks for the 50 second simulation showing that the method developed in the present research provides a valid simulation over the entire range of the physical phenomenon.

The results of the co-simulation using neural network models of both front and rear axial bushing torques show that the neural network model predicts that the front bushing reacts with a smaller negative torque than the constant coefficient stiffness and damping of the ADAMS bushing model. Despite this difference between the models, the rotation of the front bushing correlates well (see Figure 5.50). This portion of the results from 30 to 35 seconds contains the global peaks for the simulation.
Figure 5.55: The results of the co-simulation using neural network models of both front and rear axial bushing torques show that the neural network model predicts that the rear bushing reacts with a smaller negative torque than the constant coefficient stiffness and damping of the ADAMS bushing model. Despite this difference between the models, the rotation of the rear bushing correlates well (see Figure 5.51). This portion of the results from 30 to 35 seconds contains the global peaks for the simulation.

Figure 5.56: The frequency plot for the front normalized bushing axial torque was calculated using the full 50 seconds of road simulation data. The plot shows that at 0 Hz the results from the co-simulation are approximately $\sqrt{\frac{3.2}{4.8}} = 0.82$ times the ADAMS simulation results. This result corresponds to the shift in mean value of the full 50 second signal compared to the ADAMS simulation.

Figure 5.57: The frequency plot for the rear normalized bushing axial torque was calculated using the full 50 seconds of road simulation data. The plot shows that at 0 Hz the results from the co-simulation are approximately $\sqrt{\frac{3.2}{4.8}} = 0.82$ times the ADAMS simulation results. This result corresponds to the shift in mean value of the full 50 second signal compared to the ADAMS simulation.

Figure 5.58: The results of the co-simulation using neural network models of the radial bushing forces in both the X- and Y-directions shows that the neural network model correlates extremely well with the ADAMS simulation with regard to the normalized front bushing radial displacements in the X-direction. This portion of the results from 30 to 35 seconds contains the global peaks for the 50 second simulation showing that the method developed in the present research provides a valid simulation over the entire range of the physical phenomenon.

Figure 5.59: The results of the co-simulation using neural network models of the radial bushing forces in both the X- and Y-directions shows that the neural network model correlates extremely well with the ADAMS simulation with regard to the normalized front bushing radial displacements in the Y-direction. This portion of the results from 30 to 35 seconds contains the global peaks for the 50 second simulation showing that the method developed in the present research provides a valid simulation over the entire range of the physical phenomenon.

Figure 5.60: The frequency plot for the front bushing normalized radial displacement in the X-direction was calculated using the full 50 seconds of road simulation data. The plot shows good correlation between the ADAMS simulation and the co-simulation using the radial force neural network model for all frequencies.
Figure 5.61: The frequency plot for the front bushing normalized radial displacement in the Y-direction was calculated using the full 50 seconds of road simulation data. The plot shows good correlation between the ADAMS simulation and the co-simulation using the radial force neural network model for all frequencies. ........................................................................................................ 165

Figure 5.62: The results of the co-simulation using neural network models of the conical bushing torques in both the X- and Y-directions show that the neural network model correlates extremely well with the ADAMS simulation with regard to the normalized conical rotation in the X-direction. This portion of the results from 30 to 35 seconds contains the global peaks for the 50 second simulation showing that the method developed in the present research provides a valid simulation over the entire range of the physical phenomenon. ..................................................... 166

Figure 5.63: The results of the co-simulation using neural network models of the conical bushing torques in both the X- and Y-directions show that the neural network model correlates extremely well with the ADAMS simulation with regard to the normalized conical rotation in the Y-direction. This portion of the results from 30 to 35 seconds contains the global peaks for the 50 second simulation showing that the method developed in the present research provides a valid simulation over the entire range of the physical phenomenon. ..................................................... 167

Figure 5.64: The frequency plot for the front bushing normalized conical rotation in the X-direction was calculated using the full 50 seconds of road simulation data. The plot shows good correlation between the ADAMS simulation and the co-simulation using the conical torque neural network model for all frequencies. ........................................................................................................ 168

Figure 5.65: The frequency plot for the front bushing normalized conical rotation in the Y-direction was calculated using the full 50 seconds of road simulation data. The plot shows good correlation between the ADAMS simulation and the co-simulation using the conical torque neural network model for all frequencies. ........................................................................................................ 169

Figure 5.66: The results of the co-simulation using neural network models of the conical bushing torques in both the X- and Y-directions show that the neural network model correlates well with the ADAMS simulation with regard to the normalized conical torque produced in the X-direction. This portion of the results from 30 to 35 seconds contains the global peaks for the 50 second simulation showing that the method developed in the present research provides a valid simulation over the entire range of the physical phenomenon................................................................. 170

xxv

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
Figure 5.67: The results of the co-simulation using neural network models of the conical bushing torques in both the X- and Y-directions show that the neural network model correlates well with the ADAMS simulation with regard to the normalized conical torque produced in the Y-direction. This portion of the results from 30 to 35 seconds contains the global peaks for the 50 second simulation showing that the method developed in the present research provides a valid simulation over the entire range of the physical phenomenon.............................................................. 171

Figure 5.68: The frequency plot for the front bushing normalized conical torque in the X-direction was calculated using the full 50 seconds of road simulation data. The plot shows excellent correlation between the ADAMS simulation and the co-simulation using the conical torque neural network model for all frequencies................................................................. 172

Figure 5.69: The frequency plot for the front bushing normalized conical torque in the Y-direction was calculated using the full 50 seconds of road simulation data. The plot shows excellent correlation between the ADAMS simulation and the co-simulation using the conical torque neural network model for all frequencies................................................................. 173

Figure 5.70: The ADAMS simulation and neural network co-simulation results correlate well with the RTS laboratory data with regard to the normalized lower control arm angle. This portion of the results from 30 to 35 seconds contains the global peaks for the 50 second simulation........................................ 175

Figure 5.71: The frequency plot for the normalized lower control arm angle were calculated using 50 seconds of road simulation data. The plots show that the ADAMS simulation and all neural network co-simulations correlate well with the RTS laboratory results for all frequencies. ......................... 176

Figure 5.72: The ADAMS simulation and neural network co-simulation results for the damping force show higher positive peak forces and a slight upward shift in the average load towards zero compared to the RTS laboratory data. This portion of the results from 30 to 35 seconds contains the global peaks for the 50 second simulation...................................................... 177

Figure 5.73: The frequency plot for the normalized shock force was calculated using the full 50 seconds of road simulation data. The plot shows that at 0 Hz the results of all virtual simulations are approximately $\sqrt{\frac{.53}{1.56}} = 0.58$ times the RTS laboratory results, corresponding the mean shift towards zero shown in Figure 5.72. The higher peaks in the time history (Figure 5.72) are evident in the frequency plot at frequencies above 0.3 Hz. ............ 178

xxvi

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
Figure A.1: Orientation of the lower control arm pivot bushing with respect to the vehicle coordinate system: X - vehicle fore/aft, Y - vehicle lateral, Z - vehicle vertical ................................................................. 184

Figure A.2: The desired data for the axial displacement and force acquisition was created so that the peaks (blue) lie outside the range of the simulation data (black) as shown here. ...................................................................................... 187

Figure A.3: The desired data (blue) for the axial displacement and force acquisition was created so that the frequency spectrum of the data follows the general shape of the simulation data (black). ........................................................................ 188

Figure A.4: The desired data (green) for the radial displacement and force acquisition was created so that the peaks lie outside the range of both radial displacements of the simulation data (black, blue) as shown here. ...... 189

Figure A.5: The desired data (green) for the radial displacement and force acquisition was created so that the frequency spectrum of the data follows the general shape of the radial displacement simulation data (black, blue). .......... 190

Figure A.6: The desired data for the axial rotation and torque acquisition was created so that the peaks (blue) lie outside the range of the simulation data (black) as shown here. ....................................................................................... 191

Figure A.7: The desired data (blue) for the axial rotation and torque acquisition was created so that the frequency spectrum of the data follows the general shape of the simulation data (black). ............................................................. 192

Figure A.8: The desired data (green) for the radial rotation and torque acquisition was created so that the peaks lie outside the range of both radial displacements of the simulation data (black, blue) as shown here. .................. 193

Figure A.9: The desired data (green) for the radial rotation and torque acquisition was created so that the frequency spectrum of the data follows the general shape of the radial displacement simulation data (black, blue). .......... 194

Figure A.10: The bushing was mounted vertically inside the outer metal housing with the lip of the bushing sleeve clamped by a metal ring. The actuator is secured to the shaft going through the bushing’s inner sleeve. Polarity of the axial displacement and force is shown with the arrow pointing up. 195

Figure A.11: The linear rig is set up to measure the radial displacement (z) and force (Fz). Polarity of the displacement and force is shown with the arrow pointing up. ................................................................. 196

Figure A.12: The bushing, sleeve, bolt and fixture pieces for the radial measurements are shown disassembled. ........................................................................................................ 197

Figure A.13: The torsional test set-up was achieved by installing the bushing on a shaft that rotated with the vertical motion of the actuator. The assembly steps are shown in the bottom figures with the final assembly shown in the large photograph with the polarity of the angular displacement and torque measurements .............................................................................. 198

xxvii
Figure A.14: The bushing was assembled in the conical rotation fixture as shown in the three photographs. Per the description, the conical angle was measured positive and the torque negative when the outer sleeve rotated clockwise relative to the inner sleeve. ................................................................. 199

Figure A.15: Frequency domain iteration results for replicating the axial displacement signal for bushing 01. The desired, achieved and error signal are shown. Note the good agreement between the desired and achieved signals and the resulting small error. The peak in the error signal occurs at the filter cut-off frequency of 40 Hz. .............................................................................. 205

Figure A.16: Frequency domain iteration results for replicating the axial displacement signal for bushing 02. The desired, achieved and error signal are shown. Once again, the error between the desired and achieved signals is small over the entire frequency range ................................................................. 206

Figure A.17: Frequency domain iteration results for replicating the radial displacement signal for bushing 01. The desired, achieved and error signal are shown. Once again, the error between the desired and achieved signals is small over the entire frequency range ................................................................. 207

Figure A.18: Frequency domain iteration results for replicating the radial displacement signal for bushing 02. The desired, achieved and error signal are shown. Once again, the error between the desired and achieved signals is small over the entire frequency range ................................................................. 208

Figure A.19: Frequency domain iteration results for replicating the axial rotation signal for bushing 01. The desired, achieved and error signal are shown. Once again, the error between the desired and achieved signals is small over the entire frequency range ................................................................. 209

Figure A.20: Frequency domain iteration results for replicating the axial rotation signal for bushing 02. The desired, achieved and error signal are shown. Once again, the error between the desired and achieved signals is small over the entire frequency range ................................................................. 210

Figure A.21: Frequency domain iteration results for replicating the conical rotation signal for bushing 01. The desired, achieved and error signal are shown. Note the low signal level above about 10 Hz where the amplitude of the error is of the same order of magnitude as the desired and achieved signal. The desired motion of the bushing for the conical rotation data was designed to possess a similar mean value to the ADAMS simulation data and peaks beyond the range of the ADAMS simulation data without being so large as to damage the bushing. Thus, as the magnitude of the frequency plot of the desired conical motion reduces with frequency, it meets the measurement limits of the RVIT measuring the motion. ....... 211
Figure A.22: Frequency domain iteration results for replicating the conical rotation signal for bushing 02. The desired, achieved and error signal are shown. Again, as exhibited by bushing 01, the signal level is low above about 10Hz where the amplitude of the error is of the same order of magnitude as the desired and achieved signal. The limits of the RVIT measurement abilities were met.

Figure A.23: The normalized axial displacement and measured normalized force data will be used to create a neural network that will model the axial behaviour of the bushing.

Figure A.24: The normalized axial rotation and measured normalized torque data will be used to create a neural network that will model the torsional behaviour of the bushing.

Figure A.25: The normalized radial displacement and measured normalized force data will be used to create a neural network that will model the radial behaviour of the bushing.

Figure A.26: The normalized conical rotation and measured normalized torque data will be used to create a neural network that will model the conical behaviour of the bushing.

Figure A.27: Time history iteration results for replicating the axial displacement signal for bushing 01. The desired, achieved and error signal are shown. Note the small magnitude of the error signal.

Figure A.28: Time history iteration results for replicating the axial displacement signal for bushing 02. The desired, achieved and error signal are shown. Note the small magnitude of the error signal.

Figure A.29: Time history iteration results for replicating the radial displacement signal for bushing 01. The desired, achieved and error signal are shown. Note the small magnitude of the error signal.

Figure A.30: Time history iteration results for replicating the radial displacement signal for bushing 02. The desired, achieved and error signal are shown. Note the small magnitude of the error signal.

Figure A.31: Time history iteration results for replicating the axial rotation signal for bushing 01. The desired, achieved and error signal are shown. Note the small magnitude of the error signal.

Figure A.32: Time history iteration results for replicating the axial rotation signal for bushing 02. The desired, achieved and error signal are shown. Note the small magnitude of the error signal.

Figure A.33: Time history iteration results for replicating the conical rotation signal for bushing 01. The desired, achieved and error signal are shown. Note the small magnitude of the error signal.

Figure A.34: Time history iteration results for replicating the conical rotation signal for bushing 02. The desired, achieved and error signal are shown.
Figure B.1: A neural network structure with one input, a single hidden layer with a hyperbolic tangent activation function, and one output with a linear activation function. ................................................................. 228

Figure B.2: The 1-1-1 structure will be used to illustrate the process of modeling nonlinear data. ..................................................................................................................... 229

Figure B.3: The training and test data lie along the curve \( d = -\tanh(1.4x) \). .......... 229

Figure B.4: The value of the training MSE appeared to level off after about ten epochs. ........................................................................................................................................... 232

Figure B.5: The convergence of the weights is shown here as the global minimum of the MSE surface. ........................................................................................................ 232

Figure B.6: The values of the weights continued converging until about seventy epochs had been completed. ............................................................................................. 233

Figure B.7: The linear regression analysis of the test data and network estimated values in Table B.3 is shown here with the trendline equation of best linear fit. The linear regression analysis of the validation data shows excellent correlation of the network results with the known validation target values. ........................................................................................................ 235

Figure B.8: The training, validation, and test data sets have an added noise component. ....................................................................................................................................... 236

Figure B.9: A 1-5-1 feedforward neural network was used to model the data in Table B.5. ........................................................................................................................................ 237

Figure B.10: The 1-1 feed forward network generalizes the data (left) with the ability to estimate the test data within 2% of the target data with an \( R^2 \) value of 99.5% as shown by the linear regression (right). ......................................................... 239

Figure B.11: The 5-1 feedforward network generalizes the data (left) with the ability to estimate the test data within 2% of the target data with an \( R^2 \) value of 99.5% as shown by the linear regression (right). ......................................................... 239

Figure B.12: The plot illustrates the over fitting of the training data using a network with too many hidden layer neurons and no early stopping. ............... 240
The spline data input and output pairs will be used to develop a neural network model.  ........................................................................................................ 79

The parameters used during training of the neural networks to model the spline data are listed here. ......................................................................................... 83

The spline data was separated into training and validation data sets as shown in the table. .............................................................................................. 85

The thirty points that make up the hysteresis curve were normalized by the maximum and minimum values and listed. ........................................... 88

The hysteresis data was separated into training and validation data sets as shown in the table. .............................................................................................. 91

The additional parameters required by the Levenberg-Marquardt training method are listed here. ......................................................................................... 94

The statistics are summarized for each of the chosen network structures with and without early stopping using a validation set. The performance (MSE), linear regression slope, intercept, and $R^2$ values are listed for each network. ....................................................................................... 105

The statistics are summarized for each of the chosen network structures with and without early stopping using a validation set. The performance (MSE), linear regression slope, intercept, and $R^2$ values are listed for each network. Those networks with the highest training accuracy are highlighted for each method. ................................................................. 110

All axial force neural networks had similar performance and linear regression statistics for both the training and testing phases. Upon examination of each network’s weight histogram, the highlighted 2-3-1 neural network model was selected for implementation in the ADAMS model........................................................................................................ 120

All radial force neural networks had similar performance and linear regression statistics for both the training and testing phases. Upon examination of each network’s weight histogram, the highlighted 2-3-1 neural network model was selected for implementation in the ADAMS model........................................................................................................ 124

All axial torque neural networks had similar performance and linear regression statistics for both the training and testing phases. Upon examination of each network’s weight histogram, the highlighted 4-7-1 neural network model was selected for implementation in the ADAMS model........................................................................................................ 127
All conical torque neural networks had similar performance and linear regression statistics for both the training and testing phases. Upon examination of each network’s weight histogram, the highlighted 4-7-1 neural network model was selected for implementation in the ADAMS model.

Table A.1: Iterations results for the axial translation and rotation of bushings 01 and 02 show good correlation for the peak and mean values compared to the desired data. Error time histories also exhibit low peak and mean values.

Table A.2: Iterations results for the radial translation and rotation of bushings 01 and 02 show good correlation for the peak and mean values compared to the desired data. Error time histories also exhibit low peak and mean values.

Table A.3: The range of the bushing data acquisition signals compared to the 10 Volt full scale of the transducers are listed. Note that the conical rotation signal occupied less than 1% of the RVIT full scale. This may limit the accuracy of the results due to the lack of signal resolution.

Table A.4: Variation in peak and mean signal values was minimal between the two bushings while attempting to replicate the same linear and rotational desired data.

Table A.5: The training data peaks must lie beyond the range of motion of the bushing experienced during the ADAMS simulation. The values in red indicate failure of the dataset to meet this criterion.

Table A.6: The full statistics files for the time histories shown in Figure A.27 to Figure A.34 are listed here.

Table B.1: The training and validation data points plotted in Figure B.3 are shown here.

Table B.2: Calculations for the first two epochs for the neural network of Figure B.2 using a learning rate of 0.2 and a momentum rate of 0.5.

Table B.3: Test data is used to check for accuracy of the neural network model with the weights calculated after 100 epochs, $w_1(100) = -1.3985$ and $w_2(100) = 1.0006$.

Table B.4: Four different initial values for $w_1(1)$ and $w_2(1)$ were used to determine the optimal values $(w^*_1, w^*_2) = \{(-1.4, 1.0), (1.4, -1.0)\}$. The results show the neural network of Figure B.2 estimated the coefficients within 0.2% accuracy.

Table B.5: Training, validation, and test data that is shown plotted in Figure B.8.
Table B.6: The results for network development for the data with noise, shows the benefits of early stopping using a validation dataset. 239
LIST OF ABBREVIATIONS

ADAMS: Automatic Dynamic Analysis of Mechanical Systems
ARDC: Automotive Research and Development Centre
CRPC: Component Remote Parameter Control
deg: degrees
E: modulus of elasticity
EDM: Empirical Dynamics Modeling
Hz: Hertz
kg: kilogram
LCA: lower control arm
LVDT: Linear Variable Differential Transducer
MLP: multi-layer perceptron
mm: millimeter
MTS: Mechanical Testing and Simulation Inc.
N: Newton
rad: radian
RPC: Remote Parameter Control
RTS: Road Test Simulation
RVIT: Rotational Variable Inductance Transducer
s: second
LIST OF SYMBOLS

\(a\): acceleration
\(b_i\): bias value for the \(i^{th}\) perceptron
\(c\): damping coefficient

\([C_{ij}]\): diagonal matrix of translational and rotational damping coefficients

\(d\): vector of neural network output target values

\(d_{kp}\): \(k^{th}\) training target value of the \(p^{th}\) training pattern

\(e_{kp}\): error of the network’s estimate of the \(k^{th}\) training target and \(p^{th}\) training pattern

\(f(\cdot)\): function

\(f'(\cdot)\): first derivative of function, \(f(\cdot)\)

\(f_{\text{threshold}}(\cdot), f_{\text{hyperbolic tangent}}(\cdot), f_{\text{logistic}}(\cdot)\): activation functions

\(F_x, F_y, F_z\): \(X-, Y-, Z\)-components of the bushing force, \(F\)

\(F_{\text{damping}}\): damping force

\(G(s)\): transfer function

\(H\): Hessian matrix

\(J\): Jacobian matrix

\(k\): stiffness coefficient

\(K\): number of network outputs

\([K_{ij}]\): diagonal matrix of translational and rotational stiffness coefficients

\(m\): epoch

\(MSE\): mean square error, network performance

\(net_i\): input to \(i^{th}\) perceptron, sum of weighted inputs and bias

\(N\): number of training patterns

\(p\): vector of neural network input values

\(p_{\text{Max}}\): maximum input value of the neural network training data set

\(p_{\text{Min}}\): minimum input value of the neural network training data set

\(r\): relative translational deflection along a straight line between two points

\(R\): correlation coefficient

\(R^2\): coefficient of determination, square of correlation coefficient
$\mathbb{R}$: set of real numbers

$t_{\text{Max}}$: maximum target value of the neural network training data set

t_{\text{Min}}$: minimum target value of the neural network training data set

$T_x, T_y, T_z$: rotational bushing torques

$u(s)$: Laplace transform of the function, $U(t)$

$v_x, v_y, v_z$: translational bushing deflection velocity

$x, y, z$: translational bushing deflection along the X-, Y-, and Z-axes

$x$: vector of perceptron input values

X: longitudinal bushing axis

$w_{ij}$: network weight value from the $j^{\text{th}}$ perceptron to the $i^{\text{th}}$ perceptron

$w_{ij}^*$: optimal network weight value from the $j^{\text{th}}$ perceptron to the $i^{\text{th}}$ perceptron

$y$: vector of neural network output values

$y_{kp}$: $k^{\text{th}}$ network-calculated output of the $p^{\text{th}}$ training pattern, network’s estimate of $d_{kp}$

$y(s)$: Laplace transform of function $Y(t)$

Y: lateral bushing axis

Z: vertical bushing axis

$\alpha$: momentum rate

$\alpha_x, \alpha_y, \alpha_z$: angular bushing rotation deflection about the X-, Y-, and Z-axes

$\delta_i$: local error associated with the $i^{\text{th}}$ perceptron

$\varepsilon$: strain

$\eta$: learning rate

$\eta_{\text{eff}}$: effective learning rate

$\mu$: variable scalar value

$\sigma$: stress

$\omega_x, \omega_y, \omega_z$: rotational bushing deflection velocity
1 INTRODUCTION

1.1 Product Development

Historically, product development has been a cyclic process such as that shown in Figure 1.1. Following the initial design and preliminary analysis of a product concept, the testing of the initial virtual prototype is likely followed by design revisions, corresponding analyses, and reconstruction and testing of revised physical prototype designs. It is often found that further revisions are required upon physical testing due to the lack of fidelity of the virtual models that were used to guide the construction of the first physical prototypes.

Figure 1.1: The traditional product development cycle can be an iterative process, with revisions occurring late in the cycle.

The development cycle for prototypes can prove to be costly, especially when working in the world of automotive design and manufacturing. The cost of implementing changes increases exponentially as the project progresses in both time and in the commitment of decisions related to serial production (Figure 1.2). The process of making design changes thus becomes increasingly difficult and costly as time progresses through the development cycle. Revalidating the affected systems and tooling the assembly line are major sources of cost resulting from changes that occur late in the development cycle.
Figure 1.2: The cost of making changes during product development cycle grows dramatically with time.

Competition on a global scale has driven the automotive industry to strive to reduce development costs by investing funds in research activities that would reduce the time for new product development. This reduction can be further enhanced by performing more virtual prototype analysis early in the product development cycle, thus producing a design that is much closer to production intent before a physical prototype is produced (Figure 1.3).

Figure 1.3: The revised product development cycle allows for design iterations earlier in the process.
1.2 Vehicle Durability: Field and Laboratory Testing

The development of automobiles that can withstand severe use in customer service requires extensive testing. Automotive companies develop standardized durability tests to ensure reliability and predictable service lives based on many factors including: vehicle type, geographic market, target component life, and intended vehicle life.

Field durability tests are constructed from, and correlated with, customer in-service data. Corporate proving grounds are controlled environments where road and driving conditions are monitored and maintained. Still, consistency can be an issue with respect to factors such as road degradation, weather, and driver variability. For this reason, when component performance comparisons are being made some accelerated durability tests are simulated in the laboratory environment where most test conditions can be controlled.

1.2.1 Fatigue and Durability Testing

Fatigue tests are performed to determine the expected life of a component in service. Cyclic loading is applied to the component in the mode of interest until it cracks or experiences some other failure mode. The component failure mode normally depends upon its functionality within the assembly. Structural fatigue testing of a vehicle is similar. Loading is applied at the wheels or spindles until vehicle components crack or deflect beyond their specified limits. Defining part failure for this type of test may mean that a crack of a specific length has formed and propagated in a suspension part or body panel, for example.
1.2.2 Accelerated Durability Tests

Accelerated durability tests cycle the full vehicle through a variety of cyclic loading and frequency inputs. Automaker corporate proving grounds are equipped with roads that provide equivalent damage to that which would normally accumulate over the full life of a customer-driven vehicle. The accumulation of years of customer-equivalent damage can usually be achieved in a period of months on a proving grounds course. Durability schedules are constructed based on the type of vehicle undergoing the specific test: rough pavement for passenger vehicles and off-road terrain for sport utility vehicles and trucks, for example (see Figure 1.4).

![Vehicle on test at a corporate proving grounds. (Photograph courtesy of DaimlerChrysler Corporation.)](image)

Proving ground test roads provide the necessary cyclic loading on a somewhat repeatable basis, but variation in factors such as road conditions, weather, and driver variability may compromise the validity of comparisons of test results. Test variability can also adversely affect the reliability of test results if a suspension component is changed to study its influence on ride and handling.
1.2.3 Road Test Simulation

Road test simulation laboratories provide repeatable simulated durability test results. Initially, an instrumented representative vehicle is driven on the test roads at the proving ground site to record transducer signals that capture the vehicle's dynamic response as it negotiates the various road surfaces. The same vehicle is then mounted on a servo-hydraulic road test simulator that is controlled by special computer data files known as drive files (Figure 1.5). The drive files transmit signals to the road test simulator to move the vehicle. After performing iterations, during which adjustments are made to the signals, the final drive files will replicate the behaviour of the same vehicle on the proving ground test roads within a specified error value. Subsequent test vehicles that incorporate desired or potential component modifications are installed on the simulator and cycled continuously using the drive files. In this manner, test results can be compared in a controlled laboratory environment with a limited number of uncertainties as compared to traditional road test methods.

Figure 1.5: Test vehicle on a road test simulator during the drive file development process. (Photograph courtesy of DaimlerChrysler Canada Inc.)
1.3 Virtual Durability Simulation

While physical testing is necessary in automotive product development, huge financial savings can be realized if more virtual testing can be performed prior to costly prototype production. More accurate computer simulations will bring about a higher degree of confidence in the design and lay a more solid foundation to the production of the first physical prototypes.

Virtual durability tests can be performed using computer models of road test simulators as well as virtual vehicle models in a multi-body dynamics simulation software package such as ADAMS (Figure 1.6). Force and displacement signals, measured from the load cells and linear variable differential transducers (LVDT's) of the road test simulator laboratory, can be input into the virtual model to drive the virtual simulation. Subsequently, transmitted forces can be measured at various locations on the vehicle components for analysis and durability studies.

Figure 1.6: Virtual durability simulation model of a road test simulator and a vehicle in ADAMS.
The overall goal of all of these tasks is to increase the fidelity of the design process so that when the vehicle enters service as a consumer product it performs as expected and desired. Additionally, the design and prototyping process can be accomplished in a minimal amount of time.

1.4 Neural Networks

The human brain is an exemplary model of adaptive computing, with an ability to learn how to react to new inputs based on previous experiences. It contains many billions of neurons that are joined together to form a network. Electric signals pass through the brain, travelling from neuron to neuron by way of the axons that connect them (see Figure 1.7). At each neuron, all of the incoming signals are combined and the magnitude is compared to a threshold firing value. If the neuron's threshold value is exceeded, it fires a signal to all neurons to which it is connected for further processing of the signal. These biological neurons are the inspiration for mathematical neural network models.

![Neurons receive and transmit signals in the brain.](image)

Mimicking the biological neuron, the mathematical model also receives weighted inputs that are combined and compared with a bias value before the result is sent on to all
downstream neurons (see Figure 1.8). Determining the values of the weights and biases of the network adapts the neural network to model the behaviour of interest.

![Image](image_url)

**Figure 1.8:** Signals flow through the neural network along the weighted paths.

Exposing an untrained neural network to known input and output datasets determines the network parameters. Once the learning process is complete and the parameters are set, the network is validated using a known input and output dataset that is previously unseen by the neural network during training. If the desired output is estimated within pre-set error specifications, the network is now ready to accept the input test data, for which the output values are unknown.

In general, data that is input to the network can be numeric, binary, or qualitative. An important property of the training data set is that it exhibits the full range of possible values. If the input data does not cover the full range of values, the network will be forced to extrapolate during the test phase and the output may exhibit errors out of the originally specified range.
1.5 Research Objectives of the Present Study

The object of the current research is to refine the virtual vehicle durability process by using neural networks to help create more accurate efficient representations of nonlinear components within the multi-body dynamics model. To accomplish this goal, two types of components have been identified for modification within an ADAMS virtual durability model: a damper and a bushing (Figure 1.9).

![Figure 1.9: Front suspension components that will be modelled with neural networks are coloured red (front lower control arm front bushing, front shock absorber). [5]](image)

Shock absorbers are components that exhibit nonlinear damping behaviour. In other words, the force does not vary linearly with deflection velocity. The reaction force from these components is unidirectional, acting only along the line between the mounting points. The second type of component to be examined in this dissertation is the lower control arm suspension bushing. The complexity of the bushing is greater due to the number of degrees of freedom inherent in this type of component. Three forces and three moments are reacted through the nonlinear elastomeric part between the lower control arm and the vehicle body, in this case.
For each of these elements, the appropriate neural network structure will be identified, constructed, implemented within the software, and validated with known vehicle data.

Chapter 2 of this dissertation contains a review of literature in the areas of product development, durability and road test simulation, nonlinear component modeling techniques, and the use of neural networks in automotive applications. The theoretical background material related to vehicle dynamics, the behaviour of elastomeric materials, and neural network development is included in Chapter 3. Chapter 4 describes the application of neural network modeling techniques with the results of the corresponding simulations and analyses discussed in Chapter 5. Conclusions and recommendations follow in Chapter 6.
2 LITERATURE REVIEW

2.1 Product Development

Optimizing the automotive product development process is a critical objective of automakers. To improve their business in this area allows them to compete successfully in the aggressive global automotive industry of the twenty-first century. Major phases in the development process include: design, analysis/verification, prototype development, physical testing, and production. In general, the participation of the product engineer, designer, analysis specialist, and manufacturing engineer as identified by Frise, et al. [13] contributes to the complex and time consuming nature of the traditional product development process (Figure 2.1).

![Figure 2.1: Roles of participants in traditional product development [13].](image)

The emergence of the design engineer in combination with multi-faceted software suites streamlines the design process significantly with significant time savings. Specifically, work has been done by test equipment and software companies to consolidate the development process by prescribing methods of streamlining the process.
One of the areas targeted for improvement is the virtual test methods used to evaluate performance and durability of nonlinear components, where fatigue life of the component is drastically affected by the peak loads experienced by the parts [17].

Software vendors are recognizing the need for system level as well as component level virtual prototyping tools [34]. The functionality of the emerging virtual prototype tools pushes back testing requirements to later in the product development cycle where it serves as more of a validation tool than a design study tool.

2.2 Durability and Road Test Simulation

A portion of the system design analysis and verification is performed during Road Test Simulation (RTS), which is an evaluation of the vehicle’s durability under standardized conditions. Automotive companies perform standardized tests on physical prototypes at their corporate proving grounds. The test schedule is composed of percentages of varying road surfaces. The test protocols mimic in-service damage that would occur during one or more vehicle life cycles when driven in a specified manner by the customer. Typically, these types of tests take months to complete. Factors such as weather, road conditions, and available manpower limit the efficiency of the proving ground durability tests.

In a laboratory environment, controlling factors such as those previously mentioned condenses test times and provides consistent test inputs required for comparison studies. In addition to providing consistent environmental conditions and repeatable load inputs, structural durability simulations can be performed using road surfaces that have been edited to include only the most damaging portions of the surfaces. Fatigue analysis is a process that assesses component damage based upon the magnitude
and number of loading cycles that the component experiences. This concept forms the basis of tools available in software programs such as nCode® by nSoft and RPCPro® by MTS that are used to create condensed road profiles. The condensed data files maintain a specified percentage of damage of the original files but with a substantial reduction in time. These time savings can be significant, generally reducing simulation test times by about 75% while still maintaining at least 98% of the load amplitudes, reaching completion in weeks instead of months [25].

2.3 Modeling Nonlinear Components

Elastomeric components exhibit amplitude and frequency dependent behaviour. Traditionally, a linear model can approximate nonlinear behaviour if the assumption of small deflections is made. Generally, during automotive structural durability testing, the assumption of small bushing deflections is invalid [22]. Since durability tests incorporate high amplitude excursions and motion covering a wide range of frequencies, there is a need to develop more accurate models of bushings and other nonlinear components for use in virtual durability tests.

Figure 2.2: The front lower control arm bushing is oriented as shown with the X-axis pointing towards the front of the vehicle.
Sleeve construction automotive suspension bushings experience radial, axial, torsional, and conical loading (Figure 2.2). The six degrees of freedom that the bushings experience increase the complexity of the bushing model, in general. It is common practice to make simplifications to reduce the degrees of freedom or linearize the behaviour.

Work has been done to develop limited models of automotive bushing behaviour, but there has been no progress in a comprehensive modeling that can be used to estimate the full six degrees of freedom of a sleeve construction bushing for use with a multi-body dynamics model. Previous studies model limited motion of the bushings and require extensive testing to gather appropriate data for use in the model development. For example, the axial elastomeric vibration of sleeve construction bushings has been modeled by Mallik et al. [28] and by Lee and Wineman [24], producing constitutive parameter-dependent equations. A model that estimates both radial and axial behaviour was developed by Dzierzek [10], also using parameter identification methods.

During vehicle loading, the interaction of the axial, radial, torsional, and conical loads is compounded with the amplitude, frequency, and temperature dependency of the bushing behaviour. Thus, the most accurate model would account for all of these factors simultaneously. In the past, such comprehensive models have rarely been incorporated into a virtual durability simulation with ADAMS in a computationally efficient way. Given the number of bushings contained in a full-vehicle model, work to develop an efficient yet complete model is required.

While much attention has been focused on refining models of the nonlinear components, there is also research being performed that utilizes the simplified linear
models while making adjustments to other model parameters. Specifically, Sohn, et al. [36] developed a method to improve virtual testing of multi-body models that compensates for tire nonlinearity by using a linear tire model and adjusting the road inputs to the model. The road input adjustments are made by minimizing the error between the target wheel accelerations and those measured after making the appropriate the road profile adjustments.

2.4 Using Neural Networks to Model Automotive Components

Neural networks are flexible tools that have already been recognized for their ability to model complicated vehicle behaviour in a computationally efficient manner. Le Riche et al. [26] showed that acceleration signals were viable network inputs for estimating vehicle mass and dynamic loading. Neural networks have also been used to estimate drive-by noise test results based on vehicle design parameters [14], and to develop a model that estimates side-slip angle [35] based on yaw rate and lateral accelerations. Neural networks have also been used to estimate roll, yaw rate, and lateral acceleration from steering input by creating a model based on data collected from an ADAMS model [9]. In this particular study, Durali and Kassaiezadeh found that the time required to perform the neural computation with a less powerful processor at an acceptable accuracy level was less than 1% of the time to simulate the same model in ADAMS.

Finally, previous work has been done to develop neural networks to model the stiffness and damping forces of shock absorbers using displacement, velocity, and acceleration data [11, 27]. Currently, there is a commercially available software tool
(Empirical Dynamics Modeling or EDM\textsuperscript{1}) that claims to have the ability to model shock absorbers and bushings for use with fatigue life prediction tools [3]. To date, there has been a lack of published data to support the success of using this tool.

Moving road test simulation onto a virtual platform that can provide good correlation with laboratory and proving grounds tests requires the use of multi-body dynamics software containing precise individual component models. In the past, the accuracy of each component model has been sacrificed for computational efficiency using linear assumptions about the nonlinear behaviour. Linear, piecewise linear, or one-dimension functional approximations were used to simplify the models of the stiffness and damping forces without consideration of the hysteretic behaviour of the material (Figure 2.3). The task at hand is to develop a method by which neural networks can be used to accurately estimate shock absorber loads as well as axial, radial, torsional, and conical bushing loads within a full-vehicle virtual durability model without sacrificing computational speed.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure2.3.png}
\caption{The curve on the left is similar to a one-to-one curve used to approximate the hysteresis normally exhibited by a damper, as shown on the right.}
\end{figure}

\textsuperscript{1} Empirical Dynamics and EDM are trademarks of MTS Systems Corporation

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
3 Theory

3.1 Vehicle Dynamics

Within the context of the present research, vehicle dynamics refers to the manner in which the wheels, suspension, and body of a vehicle react to road load inputs or steering and braking inputs. The suspension plays an important role in the dynamics of a vehicle in that it isolates the passenger compartment from the vibrations that would otherwise be transmitted directly through the vehicle structure to the occupants. Essentially, the suspension isolates the vehicle body (sprung mass) from the unsprung mass. The unsprung mass is the mass of the tires and wheels along with the mass of that portion of the suspension system that moves with them [15].

3.1.1 Ride and Handling vs. Durability

According to Gillespie [15], the ride quality of a vehicle is a subjective perception made by occupants of the vehicle. Vibration caused by tire and wheel, engine, and driveline components (i.e. on-board sources) are transmitted through to the passenger compartment along with the road roughness inputs. Frequencies of the on-board sources generally fall within the range of 0-25 Hertz. The accumulation of these sources of vibration produces a perception of ride quality to the passenger compartment occupants.

Unlike ride, durability of a vehicle is not a quality that is perceivable by vehicle occupants. Vibration that contributes to the durability of a vehicle has frequency content beyond 25 Hertz. The durability of each individual vehicle component affects the durability of the entire assembly. The integrity of the vehicle is compromised when
individual parts are no longer able to perform their function as part of the vehicular system.

3.1.2 The Role of Suspension Components

The suspension components isolate the sprung mass from the unsprung mass by damping vibration through the dissipation of the energy that is transmitted from the road through the wheels to the rest of the car. Rubber bushings serve as flexible joints that join parts of the suspension together and in turn, fasten the suspension to the body. The viscoelastic properties of elastomeric components make them difficult to model using constitutive equations.

3.2 Elastomers

Automotive suspension bushings are fabricated from elastomeric materials because they provide the necessary stiffness and damping characteristics for comfortable and safe vehicle ride and handling. These components transfer load between vehicle components and isolate the body of the vehicle from numerous sources of vibration: the engine, exhaust, and road load input via the wheels and suspension components.

3.2.1 Polymer Structure of Elastomers

Natural rubber, which has the chemical name polyisoprene, is used in the fabrication of bushings (Figure 3.1). Polyisoprene is a polymer, which means it is composed of long chains of atoms possessing a backbone of carbon atoms. In this state, the consistency of natural rubber is tacky and viscous [31].
Figure 3.1: Polyisoprene is a polymer of carbon (C) and hydrogen (H) atoms formed into long chains of repeating units or "mers". The polyisoprene "mer" is shown here repeating 'n' times. Polyisoprene is commonly called natural rubber.

First discovered by Goodyear and Hancock in the mid-nineteenth century, vulcanization was used to eliminate the tackiness of the substance by introducing sulfur to the compound. During vulcanization, sulfur is introduced to polyisoprene with heat, which breaks the double carbon bond and allows chains of sulfur atoms to join adjacent polymer chains together (Figure 3.2). This structure is referred to as cross-linking and essentially links all the long chains together at various sites, restricting their relative movement and as a result, increases the stiffness of the rubber [31].

Figure 3.2: The addition of sulfur to polyisoprene forms vulcanized rubber, an elastomeric material commonly used in the manufacture of automotive bushings.

When a strain is applied to the material, the sulfur bonds stretch along with the other bonds in the chain, but the tendency is for the material to return to its original state.
after the load is removed. This is the physical manifestation of the stiffness or elasticity of the material. The stiffness is proportional to the degree of vulcanization, or number of sulfur cross-links [31]. More pliable bushings have less cross-linking. The location and function of the bushing dictates the appropriate amount of vulcanization and thus stiffness.

Additives are also used to customize the mechanical properties of the components to suit their function. For example, carbon black is an additive used in dynamic bushings made from natural rubber to increase damping, modulus of elasticity, and tensile strength. Plasticizers are added to reduce material stiffness near the glass transition temperature, at which the material is cold enough that it no longer exhibits viscous properties and becomes brittle [4]. This is important in a vehicle application where operating temperatures cover a wide range.

Vehicle suspension fasteners are generally torqued to specification with the weight of the vehicle on the suspension (i.e. at curb weight) [5]. Curb weight of a vehicle is defined as the weight of the vehicle with fluids filled to operating levels and no passengers or cargo inside [5]. This procedure is used because tightening the fasteners with the weight of the vehicle supported on its frame or body and the suspension hanging without a load causes a preload on the bushings once the weight of the vehicle is applied and this reduces the available elastic strain of the elastomeric components. Preloading the bushings in this manner causes stress relaxation and creep to occur, during which the polymer chains slide by each other over time. This altering of the material structure on a molecular level reduces the life of the part.
3.2.2 Mechanics of Viscoelastic Behaviour

Elastomers exhibit properties of both elastic solids and viscous fluids and are often referred to as viscoelastic materials. The response of these materials to an applied stress or strain is dependent on the material temperature and time and frequency of the loading.

The modulus of elasticity of elastomeric materials increases as temperature decreases. The rate at which the modulus increases depends upon the material structure, including degree of vulcanization. As shown in Figure 3.3, the stiffest rubber material does not experience much variation in elastic modulus with temperature. The glass transition temperature is reached when the modulus sharply increases and the material becomes brittle, losing all viscous properties. At the glass transition temperature, the carbon backbone of Figure 3.2 is completely frozen [31].

![Glass Transition Temperature Diagram](image)

**Figure 3.3:** The glass transition temperature is the temperature at which the vulcanized rubber loses its viscous properties and becomes brittle. The magnitude of the change in modulus is small for the stiffest rubber materials. (Adapted from Figure 23.8 from [1]).
Creep and relaxation illustrate the concept of time dependency during loading of a viscoelastic material. The frequency-dependent nature of the material is revealed during dynamic loading. Creep occurs when a constant stress is applied to the material and the strain changes with time (Figure 3.4). On a microscopic level, the polymer chains slide past each other to maintain the applied stress until the molecular branches extending from the backbone entangle and eventually prevent further elongation of the material.

**Viscoelastic Creep**

![Viscoelastic Creep Diagram](image)

**Figure 3.4:** During viscoelastic creep, stress is held constant while strain increases.

Relaxation is a phenomenon during which strain is held constant and the stress experienced by the material decreases over time (Figure 3.5). Again, the polymer chains move past each other as the bonds are stretched during application of the strain. This movement alleviates the stress in the material.
Viscoelastic Relaxation

Figure 3.5: During viscoelastic relaxation, stress decreases as the strain is held constant.

Viscoelastic material behaviour differs from the stress-strain response of ductile elastoplastic materials in that hysteresis experienced by elastic material is indicative of permanent plastic deformation once the stress is removed. Hooke’s Law states that as an elastic material is loaded, it first exhibits a strain response proportional to the applied stress until the yield stress is surpassed. The proportionality constant is the modulus of elasticity, E. Once the load is removed, the material returns to an unloaded state at the same rate as loading and a plastic deformation results (Figure 3.6).
The dynamic behaviour of a viscoelastic material shows marked differences from the behaviour of solid elastic or plastic materials especially in terms of the frequency dependent response to loading. The viscous properties of the material cause the strain response to lag the applied stress during cyclic loading (Figure 3.7).

Figure 3.6: This is a typical stress versus strain curve for an elastic material that exhibits yielding. Note that when the material is loaded past the yield stress, unloading occurs along a parallel path (i.e. with the same slope and thus the same elastic modulus) to a state of permanent (plastic) deformation.

Dynamic Loading of Viscoelastic Material

Figure 3.7: The strain response lags the applied dynamic stress to a viscoelastic material as illustrated by the phase lag between the two signals.
The corresponding stress-strain curve of the data in Figure 3.7 yields a hysteresis loop (Figure 3.8). For each instant in time, stress is plotted as a function of the strain. The difference of the curve during loading (top portion of the curve) and unloading (bottom portion of the curve) produces a hysteresis loop. The arrows indicate the direction of increasing time.

**Figure 3.8:** Viscoelastic stresses exhibit a lagged strain response that develops into a hysteresis curve in a stress versus strain plot. The arrows indicate the progression of time, such that loading and unloading of the specimen are shown.

### 3.2.3 Traditional Models of Viscoelastic Behaviour

Modeling viscoelastic behaviour is a challenging task. Traditional models combine the stiffness of spring elements with the damping of viscous dashpot (i.e. damping) elements in series and in parallel to model the creep and relaxation that is typical of viscoelastic material.
Figure 3.9: The Maxwell and Voigt-Kelvin viscoelastic models use springs to model the elastic component of the behaviour (stiffness, E) and viscous dashpots to model the viscous behaviour of the material (damping, c).

The Maxwell and Voigt-Kelvin models shown in Figure 3.9 each assume linearity of the viscoelastic material in that the stress in the spring and damper are assumed to exhibit the following relationship to the strain and strain rate in the corresponding element:

\[
\sigma_{spring} = E\varepsilon_{spring} \quad \text{Equation 3.1}
\]

\[
\sigma_{dashpot} = c\frac{d\varepsilon_{dashpot}}{dt} \quad \text{Equation 3.2}
\]

where,

\(\sigma\) = stress

\(E\) = modulus of elasticity

\(\varepsilon\) = strain

\(c\) = constant damping coefficient

Unfortunately, these two models each lack the ability to simultaneously replicate both creep and relaxation behaviour. The Voigt-Kelvin element models the mechanics of
creep better than the Maxwell model which better represents the behaviour of stress relaxation [2] (see Figures 3.10 and 3.11).

**Creep: Maxwell vs. Voigt**

![Creep: Maxwell vs. Voigt](image)

*Figure 3.10:* The Voigt element better predicts strain during creep than the Maxwell element for which strain varies linearly during constant applied stress.

**Relaxation: Maxwell vs. Voigt**

![Relaxation: Maxwell vs. Voigt](image)

*Figure 3.11:* The Maxwell element better predicts stress during relaxation than the Voigt element for which stress is constant during constant applied strain.

Assembly of the Maxwell and Voigt elements in series or parallel with other spring and dashpot elements is a method of refining this type of model, but the process of doing so for each variation of the elastomeric material is cumbersome and impractical for
the present research. Our focus is to use neural network tools to more accurately estimate the nonlinear behaviour of elastomeric materials.

3.3 Introduction to Neural Networks

Neural networks are adaptive models that can generalize linear and nonlinear behaviour and categorize inputs through pattern recognition techniques. These mathematical tools originated from the concept of biological neurons (Figure 3.12), which are the means by which the brains of living creatures function.

![Biological Neuron Diagram](image)

**Figure 3.12:** The biological neuron accepts signals from other neurons, processes the combined signal at the nucleus, and sends the resulting signal along axons downstream to other neurons [20].

Similar to neurons, the perceptron (the building block of a mathematical neural network) accepts input signals and calculates a weighted sum, which will be referred to as the variable net. The inputs to a perceptron may be network inputs or outputs from perceptrons. The value of net is then assessed against a firing or trigger threshold function (Figure 3.13). The threshold function is a step function that dictates what value the neuron will transmit downstream to subsequent neurons.

28

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
Threshold Nonlinearity

Figure 3.13: For negative values of the input variable, net, the output of the function \( f(\text{net}) \) is either \(-1\). For positive or zero values of the input variable, net, the output of the function \( f(\text{net}) \) is \(+1\). The threshold nonlinearity function is described mathematically by

\[
f_{\text{threshold}}(\text{net}) = \begin{cases} 
1, & \text{net} \geq 0 \\
-1, & \text{net} < 0 
\end{cases}
\]

The threshold function is one type of activation function used by perceptrons to process the variable net. Other types of activation functions that can be used to process data through the perceptron are sigmoid or linear functions. A sigmoid function is a "smoothed" version of the threshold function (i.e. differentiable). Sigmoid functions can range from \(-1\) to \(1\) or from \(0\) to \(1\).
The perceptron is a mathematical representation of the biological neuron and used as the neural network building block. The $i^{th}$ perceptron of a neural network is shown here.

The $i^{th}$ perceptron of a neural network shown in Figure 3.14 receives $n$ inputs ($x_1$, $x_2$, $x_3$, ..., $x_n$) and weights them using constants $w_{i1}$, $w_{i2}$, $w_{i3}$, ..., $w_{in}$. The sum of weighted inputs is shifted by adding a bias value, $b_i$, to produce the value of the variable $net_i$. The value of $net_i$ is passed through the activation function to produce the value of $y_i$.

The perceptrons can be assembled in layers to form what is referred to as a multilayer perceptron (MLP). The input layer receives the inputs to the network and the output layer produces the final network output. Any layers in between the input and output layers are referred to as hidden layers. The nomenclature for describing a network’s structure includes the number of inputs, the number of perceptrons in each hidden layer, and the number of network outputs. The neural network shown in Figure 3.15 is referred to as a 2-3-1 MLP. Two inputs are processed through a single hidden layer of three perceptrons to produce one network output. The resulting neural network maps the input pair ($p_1$, $p_2$) to the output $y_1$. 

Figure 3.14: The perceptron is a mathematical representation of the biological neuron and used as the neural network building block. The $i^{th}$ perceptron of a neural network is shown here.
Neural networks can be used to generalize the nonlinear relationship between a set of known input and target data. In other words, the neural network provides an estimate of the nonlinear function, $f$, that describes the relationship between the network input, $p$, and the target values, $d = f(p)$. The variable parameters of a neural network are the values of the weights and biases for each perceptron unit. By adjusting the values of these parameters, the network output, $y$, becomes a good estimate for the target data $d$. The optimal values of the weights and biases are determined by the number of layers, number of perceptrons, and the type of activation functions that are chosen for the network structure. Once the structure is finalized by minimizing the error between the output and target data, the network can be used to calculate the output for previously unseen input data.

3.4 Designing Neural Networks

Neural networks are computational black-box models. They are used to model nonlinear behaviour through generalization. There are three stages of neural network development: training, testing, and simulation. Separate data sets are associated with
each of the three phases: training data (with or without validation data), test data, and simulation data.

During the training phase, known input/output data sets are used to determine the network parameters (i.e. the network weights and bias values) for a particular network structure. The network training is an iterative procedure that can be performed with or without the use of a validation data set. The difference in the two data sets is that the error resulting from estimation of the training data is used in calculating corrections to the parameters, while the error in estimating the validation data is used to monitor the generalization capability of the network. Without the validation data set to monitor this characteristic, the network will learn to memorize the training data, rather than generalize its behaviour. As the network parameters converge to optimal values that minimize the error of the training data set, the error in estimating the validation data is checked to ensure that the corresponding error does not begin to increase. The method of utilizing a validation data during training is referred to as early stopping [33].

After the parameters are optimized, testing is performed using another known input/output dataset that is different from the training and validation data sets. The proposed network structure is used to ensure that the errors of estimation for the test data are within specified criteria. The chosen criteria might correspond to the sensitivity of the transducer (e.g. 1% of the transducer’s full measurement range) used to obtain the original data or the repeatability of acquiring the data (e.g. 5% of the signal’s range), or perhaps the largest of all applicable factors.

Once the criteria are met for the training, validation, and test data, the proposed network structure is accepted and used for simulation, where inputs are presented to the
network for which the output values are yet unknown. Given the training, validation, and test phase results, it can be assumed that the network calculated outputs from the simulation are also within the same error criteria from the actual target values.

3.4.1 Methods of Construction

Neural network construction is performed using iterative methods. Sufficient generalization of training data can be achieved with both simple and more complex network geometries, so the directive of Occam’s razor should be considered [33]:

"Any learning machine should be sufficiently large to solve the problem, but not larger."

For this reason, the simplest network geometries are evaluated first, followed by networks of increasing complexity.

Neural networks employing nonlinear activation functions in a single hidden layer are considered universal approximators, capable of modeling any nonlinear continuous functions so long as a sufficient number of hidden layer perceptrons are included [33]. A sufficient number of hidden layer perceptrons is the minimum needed to produce the desired error of estimation (e.g. 1%).

Factors such as scaling of the input and output data, choice of activation function, minimum number of hidden layer perceptrons, and size of the training and validation data sets will be discussed in the following sections. An illustrative example of neural network development is included in Appendix B.
3.4.2 Activation Functions

Modeling nonlinear behaviour requires the use of nonlinear activation functions in the neural network structure. Three common nonlinear activation functions are the threshold, hyperbolic tangent, and logistic functions:

\[
\begin{align*}
    f_{\text{threshold}}(net) & = \begin{cases} 
        -1, & net < 0 \\
        +1, & net \geq 0 
    \end{cases} \\
    f_{\text{hyperbolic tangent}} & = \tanh(net) \\
    f_{\text{logistic}} & = \frac{1}{1 + e^{-net}} 
\end{align*}
\]

Equation 3.3

Each of the functions is shown in (Figure 3.16).

![Nonlinear Activation Functions](image)

Figure 3.16: The threshold, hyperbolic tangent (tanh), and logistic functions are nonlinear activation functions commonly used in neural network design. The variable *net* is input to the activation function that acts as a trigger or firing threshold to produce an output. The continuous differentiability of the hyperbolic tangent, and logistic functions make them more desirable for neural network structures because this property will allow calculation of the back propagation of the error during the training phase of network construction.
The domain of all three functions is the set of real numbers, $\mathbb{R}$, however unlike the threshold and hyperbolic tangent functions that map the variable $\text{net}$ to the range $[-1,1]$, the logistic function has a range of $[0,1]$. Either the hyperbolic tangent or the logistic function can be used in the network construction with equal success. The values of the weights and biases will change to reflect the selection.

3.4.3 Scaling the Input and Output Data

Using input data that contains a variety of data types that operate on significantly different scales (e.g. displacement, velocity, and acceleration) will yield a collection of network weights and biases that may differ by several orders of magnitude. As will be shown below, larger weights have a greater effect on the error associated with the network outputs. To prevent this from happening, all input data should be scaled to the same range before being used as input for the network.

For this reason, the training data used as network inputs and outputs will be linearly scaled to the range $[-1,1]$ with regard to their respective maximum and minimum values. Validation and simulation test data will also be linearly scaled using the training data extremes prior to processing. After the network calculations are made, the output data is returned to the original scale as shown by the process diagram in Figure 3.17.
3.4.4 Number of Hidden Layer Perceptrons

The general consensus of the neural network research community is that a neural network with one and at most two hidden layers can be used to approximate any continuous function \([19, 33, 37]\). The exact number of perceptrons required in each of the hidden layers cannot be analytically determined but the literature agrees on the idea that there should at most approximately twice as many hidden layer perceptrons as there are inputs to the network. Kolmogorov’s theorem that stated that any continuous function can be represented by the superposition of \((2n+1)\) univariate functions, where \(n\) is the number of independent variables \([37]\). Kůrková [23] reiterated the work of Kolmogorov to include the use of sigmoidal functions.
3.4.5 Training and Validation Data Sets

Scaling the training and validation data sets increases the efficiency of the neural network training process [6, 37]. It is difficult for the training algorithms to compensate for errors that may differ by several orders of magnitude stemming from the difference in ranges of different input data elements. Another advantage to scaling the data is exhibited during the sensitivity analysis, where sensitivity of the output with respect to each input can be directly compared for scaled data.

The training data set should also encompass the full range of possible input and output values. As is common with nonlinear functions, extrapolation outside the range of known behaviour is bound to provide less accurate results than within the training region, where the estimation error is known after the network has been trained and tested.

Principe [33] states that the number of training patterns required is approximately

\[
N > \frac{\text{total number of weights}}{\text{error in network accuracy}}
\]

Equation 3.4

So, to estimate test data with 90% accuracy, the total number of training samples should be at least as large as ten times the total number of weights in the network.

For relatively small networks structures that are to be used with dynamic data, the training data set will have to be significantly larger than that specified in Equation 3.4 to capture all of the necessary frequency and amplitude content.

3.4.6 Measuring the Performance of the Network

The mean square error (MSE) is a common statistic used for evaluating the accuracy of the network's output [33, 37]. Minimizing the MSE for a specific network
structure will yield an optimal neural network model. For a set of \( N \) training patterns and \( K \) network outputs, let

\[
MSE = \frac{1}{2N} \sum_{p=1}^{N} \sum_{k=1}^{K} e_{kp}^2 = \frac{1}{2N} \sum_{p=1}^{N} \sum_{k=1}^{K} (d_{kp} - y_{kp})^2 \quad \text{Equation 3.5}
\]

where,

- \( N \) = number of training patterns
- \( K \) = number of network outputs
- \( d_{kp} \) = \( k \)th training target value of the \( p \)th training pattern
- \( y_{kp} \) = \( k \)th network-calculated output of the \( p \)th training pattern, also the network estimate of \( d_{kp} \)
- \( e_{kp} \) = network's error of estimating the \( k \)th training target value of the \( p \)th training pattern

While this performance measure is used in the example included in Appendix B, similar performance measures that are also used include the root square error [37] and the average of the sum of the squared error terms [6]. The latter is utilized in the MATLAB neural network development of this research.

While the network output, \( y_{kp} \), is a nested function of the weights and the network inputs, the function depends upon the number of layers and choice of activation functions of the network.

The goal of the network construction algorithm is reduced to finding optimal weight values, \( w_{ij}^* \), for which the value of the \( MSE \) is minimized. The optimal solution yields a model that estimates outputs that lie within the aforementioned criteria (see section 3.4.1) to the target outputs of the training and validation data sets.
Mathematically, the optimal weights, \( w_{ij}^* \), are the values at which the performance function reaches a minimum value. These values occur when the slope of the performance function reaches a zero value:

\[
\frac{d(MSE)}{dw_{ij}} \bigg|_{w_{ij} = w_{ij}^*} = 0
\]

Equation 3.6

Graphically, the values of \( w_{ij}^* \) occur at the global minimum of the \( MSE \) function in the space of the weights.

To illustrate the concept of minimizing the \( MSE \), consider the \( n \)-1-1 network shown in Figure 3.18. An \( n \)-dimensional input vector, \( x = (x_1, x_2, ..., x_n) \) is input to a single linear perceptron to estimate the value of one target, \( d \), for each training pattern. The network's output is denoted \( y \), and is the estimate of \( d \) for the target, \( d \).

![Diagram of a single perceptron](image)

Figure 3.18: The single perceptron employs a linear activation function to estimate a single output value from \( n \) inputs. It will be used to illustrate the concept of optimizing the performance of the network during training.
For $N$ training patterns, the $MSE$ for the network in Figure 3.18 is calculated as:

$$MSE = \frac{1}{2N} \sum_{p=1}^{N} (e_p)^2$$

$$= \frac{1}{2N} \sum_{p=1}^{N} (d_p - y_p)^2$$

Equation 3.7

$$= \frac{1}{2N} \sum_{p=1}^{N} \left( d_p - f \left( \sum_{i=1}^{n} w_i x_{pi} \right) \right)^2$$

For a linear activation function, $f(\cdot)$, the $MSE$ is a quadratic function in $w_i$. Thus, for each training pattern, the $MSE$ function is an upward-facing parabolic surface for which a global minimum exists. If the input vector has one element (and consequently the network has one weight), the $MSE$ function is a two-dimensional parabola as shown in Figure 3.19. Similarly, for an input vector with two inputs (and thus the network has two weights), the $MSE$ surface is a three-dimensional parabolic surface.

**Mean Square Error vs Weight**

![Mean Square Error vs Weight](image)

**Figure 3.19:** In this example of the performance of a single layer network using a linear activation function, the value of the $MSE$ is minimized for $w^*=0.75$. 

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
If the activation function is sigmoidal in nature and the network incorporates multiple layers in its structure, the minimization problem becomes more complex. The nesting of sigmoidal activation functions gives rise to the existence of local minima in the $MSE$ surface as shown in Figure 3.20.

**Mean Square Error vs. Weight**

![Mean Square Error vs. Weight](image)

Figure 3.20: The use of hyperbolic activation functions causes local minima to arise in the performance minimization. The learning algorithm should be modified to ensure that the true optimal weight is found at the global minimum of the performance function.

During the training process, convergence to a local minimum is avoided by modifying the learning algorithm to include a momentum correction term that continues weight adjustment until the true global minimum of the $MSE$ is determined. The network training algorithms are described in the following section.
3.4.7 Determining Appropriate Weights and Bias Values

The weights, \( w_{ij} \), and biases, \( b_i \), for each perceptron (Figure 3.21) of a neural network are used to calculate \( net_i \), the input to the activation function. Weights determine the relative influence of each input as it progresses through the layers of the network, from perceptron to perceptron. Biases provide a vertical shift of the sum of the weighted values so that the resulting value of \( net_i \) passes through the correct portion of the activation function. The weights and biases are the variable parameters that are adjusted during network training until the network's estimation of the training data target is determined within the previously discussed error criteria. Determination of the weight and bias values for a chosen network structure will be performed using a gradient descent method. First, a single perceptron with multiple inputs will be used to explain the process and then it will be generalized to a neural network with a single hidden layer.

The perceptron shown in Figure 3.21 accepts the \( n \)-dimensional input vector, passes the weighted and biased sum of the \( n \) elements through an activation function and produces a single output.
The $i^{th}$ perceptron accepts $n$ inputs and produces one output. Note that the weights, $w_{ij}$, and bias, $b_i$, are the variable parameters that are adjusted during training of the neural network.

Without loss of generality, the bias input at any perceptron can be eliminated and replaced by an additional input element, $x_{i(n+1)} = +1$ and an additional weight, $w_{i(n+1)} = b_i$, as shown in Figure 3.22. With this simplification, future reference to the bias value will be eliminated in the explanation of the gradient descent method below.

**Figure 3.21:** The $i^{th}$ perceptron accepts $n$ inputs and produces one output. Note that the weights, $w_{ij}$, and bias, $b_i$, are the variable parameters that are adjusted during training of the neural network.

**Figure 3.22:** The bias of the $i^{th}$ perceptron in Figure 3.21 can be replaced by an additional input element, $+1$, and a corresponding weight, $b_i$. 

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
Consider a multilayer perceptron (MLP) with multiple inputs and multiple outputs. During the training phase, known input and target data sets are presented to the neural network structure. The training data input and target data sets are denoted by \((x, d)_p\), where \(p\) denotes the training pattern, \(x\) is the input vector, and \(d\) is the target vector. The training pattern index will be dropped here for simplicity as each pattern is presented individually to the network during the training process.

Initial values for the weights are chosen at random. Using the first training pattern, the input data, \(x\), is passed through the network and the first estimate of the target vector is calculated, \(y\). The error between the target and the network output is computed for each training pattern, \(e = d - y\), and is used by an error propagation method to determine weight adjustments for each weight in the network. The error, \(e\), will be referred to as the global error to distinguish it from the local error, which is the portion of the global error that can be attributed to each perceptron. Error propagation refers to the process by which the local error is calculated by tracing the global error backwards (from the output towards the input) through the network. In doing so, the effect of each weight variable on the local error can be calculated. This effect is then used to determine the weight adjustment required to reduce the global error, \(e\).

To visualize the propagation of the error backwards through the network, consider the processing path through a network with a hidden layer that starts at the \(j^{th}\) perceptron and goes to the \(i^{th}\) perceptron and then \(k^{th}\) perceptron (Figure 3.23). The calculation of \(w_{ij}\), the weight of the hidden layer perceptron, will be derived using the notation shown.
Each pass of the training data through a network during training is called an “epoch”. The method used to calculate the change in the weights for each epoch is called the gradient descent method. The chain rule is used to calculate the contribution of each weight to the \(MSE\) by calculating the gradient of the \(MSE\) with respect to the weight for each training pattern, \(\frac{\partial MSE}{\partial w_{ij}(m)}\), where \(m\) denotes the epoch. The gradients for all training patterns for the current epoch are summed and the weight is adjusted by a value proportional to the total gradient. The portion of the gradient used to calculate the weight correction is called the learning rate, \(\eta\).

\[
w_{ij}(m+1) = w_{ij}(m) - \eta \sum_p \left( \frac{\partial MSE}{\partial w_{ij}(m)} \right)_p
\]

Equation 3.8

where,

\(m\) = the epoch

\(w_{ij}\) = the weight from the \(j\)\(^{th}\) perceptron to the \(i\)\(^{th}\) perceptron

\(\eta\) = learning rate
and for each epoch, $m$, and training pattern, $p$,

$$\frac{\partial \text{MSE}}{\partial w_{ij}} = \left( \frac{\partial \text{MSE}}{\partial y_j} \right) \left( \frac{\partial y_j}{\partial (\text{net}_j)} \right) \left( \frac{\partial (\text{net}_j)}{\partial w_{ij}} \right)$$

$$= \left( \sum_{k \text{ outputs}} \frac{\partial \text{MSE}}{\partial y_k} \frac{\partial y_k}{\partial (\text{net}_k)} \frac{\partial (\text{net}_k)}{\partial y_j} \frac{\partial y_j}{\partial (\text{net}_j)} \frac{\partial (\text{net}_j)}{\partial w_{ij}} \right)$$

$$= \left( \sum_{k \text{ outputs}} - e_k f'(\text{net}_k) w_{ki} \right) f'(\text{net}_i)(y_j)$$

$$= \left( \sum_{k \text{ outputs}} e_k f'(\text{net}_k) w_{ki} \right) f'(\text{net}_i)(-y_j)$$

Equation 3.9

Recall that

$$y_i = \text{output if the } i^{th} \text{ perceptron}$$

$$\text{net}_i = \text{input to } i^{th} \text{ perceptron activation function, } f(\cdot)$$

$$f'(\cdot) = \text{derivative of the activation with respect to its input, } \text{net}_i$$

The summation in Equation 3.9 is used because the MSE is a function of all network outputs. The result illustrates that the gradient of the MSE with respect to $w_{ij}$ is the product of the global error for each output, $e_k$, propagated back through the output layer activation function and connecting weights with the derivative of the $i^{th}$ perceptron’s activation function and the input to the $i^{th}$ perceptron. This expression can be simplified by denoting the local error of the $i^{th}$ perceptron by $\delta_i$.

$$\delta_i = f'(\text{net}_i) \left( \sum_{k \text{ outputs}} e_k f'(\text{net}_k) w_{ki} \right)$$

Equation 3.10

46
Thus,

\[ w_{ij}(m+1) = w_{ij}(m) + \eta \delta_i(m) y_j \]  \hspace{1cm} \text{Equation 3.11}

It is apparent that the weight adjustment for \( w_{ij} \) for each epoch is dependent only on the local error, \( \delta_i \), and the input to the \( i^{th} \) perceptron, \( y_j \).

Regarding the value of the learning rate that should be used, small values of \( \eta \) provide stable but slow convergence. Larger values of the learning rate can cause divergence of the solution. For nonlinear, multilayer perceptron networks, there are algorithms that calculate different learning rates for use in different layers of the network that evolve as iterations progress, but an iterative approach of determining a uniform value of \( \eta \) is also deemed acceptable due to the complexity of the calculation for these intricate network structures [28].

An improvement in the convergence speed of the learning algorithm can be achieved by including a momentum term whereby a portion of the previous weight change is also added to the current weight change. The momentum term, \( \alpha \), is included in the gradient descent with momentum equation:

\[ w_{ij}(m+1) = w_{ij}(m) + \eta \delta_i(m) y_j + \alpha(w_{ij}(m) - w_{ij}(m-1)) \]  \hspace{1cm} \text{Equation 3.12}

The use of the momentum term in addition to the original learning rate produces an effective learning rate, \( \eta_{\text{eff}} \), defined by Hertz et al. [19] as:

\[ \eta_{\text{eff}} = \frac{\eta}{1 - \alpha} \]  \hspace{1cm} \text{Equation 3.13}
where,

\[ \eta = \text{learning rate} \]

\[ \alpha = \text{momentum rate} \]

Typical values for \( \eta \) and \( \alpha \) when used together in the gradient descent with momentum learning algorithm are \( \eta = 0.05 \) and \( \alpha = 0.9 \) [6], producing an effective learning rate of \( \eta_{\text{eff}} = 0.5 \). An example of the determination of a single weight parameter is shown in Figure 3.24.

**Mean Square Error vs. Weight**

![Graph showing Mean Square Error (MSE) vs. Weight](image)

Figure 3.24: The value of the weight, \( w \), is adjusted by a rate proportional to the gradient of the MSE curve. In this fashion, the optimal value, \( w^* \), is eventually reached after a sufficient number of iterations.

It is customary to examine the final values of the network weights to assess the sufficiency of the network's abilities. Using a histogram to view the distribution of the weights will highlight any outlying weights that could cause instability in the network.
calculations by amplifying errors passing through the relevant portion of the network. The appropriate shape of the histogram is a bell shape centred about zero with all values less than or equal to ten [37]. Refer to the example shown in Figure 3.25.

![Network Weights Histogram](image)

**Figure 3.25:** The network weights displayed in the histogram show the highest concentration of weights with small magnitude and form a bell-shape about the midpoint. Weights larger than about ten are undesirable due to the amplification effect on errors passing through that weighted path of the network [37].

3.5 Using Neural Networks to Characterize Material Behaviour

Modeling nonlinear component behaviour (e.g. damper or elastomeric bushing) using neural networks requires an understanding about the properties that depict the component’s behaviour. For example, the damping force of a shock absorber is roughly proportional to the deflection velocity of the component. The measured force versus deflection velocity of a shock absorber shows the true hysteresis of the behaviour as illustrated in Figure 3.26.
Figure 3.26: This normalized damping force versus normalized shock deflection velocity curve shows the nature of the hysteresis exhibited by measured data. Note the separate hysteresis loops that result as the frequency and amplitude of the signal change.

The location of the current damper force on the hysteresis curve can then be more accurately estimated if the history of deflection velocity is known. A series of two deflection points will yield the instantaneous deflection velocity and three points will provide instantaneous deflection acceleration data. Using these values as the input to a neural network to estimate the corresponding damping force is a recommended starting point for the design of a suitable network structure that will model the nonlinear behaviour of a damper.

Similarly, a rubber bushing under load exhibits stiffness and damping as a result of being deformed. The present research will use the deformation, deformation velocity, and deformation acceleration as input to design a suitable neural network to model the nonlinear behaviour of the bushing.
4 APPLYING NEURAL NETWORK MODELING TECHNIQUES

The current ADAMS multibody dynamics software offers constant stiffness and damping coefficients and spline curve models to estimate damper and bushing forces with respect to their deflection. The package does not currently offer the flexibility that neural networks do to model the hysteresis inherent in the force and torque components. Incorporating a neural network representation of a force or torque element into an ADAMS multi-body model requires that modifications be made to the original model.

To understand the changes that are required of the original ADAMS model, the global process will be outlined with regard to the flow of data between MATLAB and ADAMS, where the neural network and multibody models reside, respectively. As shown in Figure 4.1, the ADAMS model calculates the multi-body dynamics based on the excitation forces (i.e. load and displacement inputs from the road test simulator).

![Diagram of data flow between MATLAB, neural network, and ADAMS models.](image)

**Figure 4.1:** Data is passed to and from the ADAMS model from the Simulink environment where the neural networks reside.

The state variables of a dynamic system describe the current state of the system in such a way that with the knowledge of the system dynamics and input functions, the future behaviour of the system is known [8].
In the current research, the state variables used to describe the behaviour of the damper or bushing dynamics are the deflection and deflection velocity of the component. These state variables are exported from the model using ADAMS/Controls, a software module that resides within ADAMS and can be accessed from ADAMS/View (see Section 4.2.4). These state variables are used in Simulink as inputs to the neural networks that estimate the reactive force or torque of the damper or bushing. The resulting forces are passed back into ADAMS for the next time step solution of the motion of the body.

Once the ADAMS model has been modified, the neural networks are created using training data described above in Section 3.4.5. Links are established between ADAMS and MATLAB for the information to flow from one software package to another during the simulation.

This chapter describes the required modifications of the ADAMS model, creation of neural networks, and the integration of the neural networks and the ADAMS model.

4.1 ADAMS Model Preparation

The ADAMS model was prepared to accept data from neural networks residing in MATLAB. The nature of the global process shown in Figure 4.1 compels consideration of the following issues, described in more detail in Section 4.2:

1. The force component (damper or bushing) must be remodelled as a generic force element that is defined using ADAMS state variables that feed data into the model.

2. State variables must be defined to store the necessary data that will be passed out of the ADAMS model to MATLAB.

3. The ADAMS Controls Plug-in module must be utilized to define the model's input and output variables as defined above.
4. The Controls Plug-in module must be used to export the nonlinear ADAMS model for use in the Simulink model.

5. A low pass filter should be applied to the data coming into the ADAMS model from Simulink to improve stability of the model. The low pass filter will remove signal content possessing frequencies that are greater than the highest frequency of interest.

4.2 ADAMS Model Components

The traditional method of simulating dampers and bushings in an ADAMS model is to utilize the translational spring-damper and bushing flexible connector tools that are available in the software. Both tools require the user to specify the action and reaction bodies between which the forces react. Standard ADAMS components will be reviewed before the topic is broached regarding modifications to the defined forces.

4.2.1 Location and Orientation of Markers

Markers are local to the coordinate systems that are attached to the ground or to parts in the model [32]. ADAMS automatically generates markers at the centre of mass of all solid parts as well as at specific anchor points of the parts. Every marker has a location and an orientation with which it is associated.

Marker locations are defined as their coordinates relative to the global origin of the model and the orientation is defined by the Euler angle of the coordinates relative to the global origin [12, 38]. Euler angles are three consecutive rotation transformations from the global $x$-$y$-$z$ coordinate system to the $x'$-$y'$-$z'$, $x''$-$y''$-$z''$, and finally to the $\hat{x}$-$\hat{y}$-$\hat{z}$ coordinate system via consecutive rotations about the $z$, $x'$, and $z''$ axes. The transformation is also referred as a 3-1-3 rotation transformation due to the three consecutive rotations about the third axis of the original coordinate system, the first axis of the first transformed coordinate system, and the third axis of the second transformed
coordinate system (Figure 4.2). Positive rotations are implemented using the standard right hand rule.

![Coordinate System Diagram]

**Figure 4.2:** This example of a 90-90-90 Euler transformation illustrates three 90° rotations about the z-, x', and z''-axes, respectively, to transform the x - y - z coordinate system into x - y - z coordinate system. In the ADAMS software, x - y - z represents the global coordinate system, and the marker coordinate system, x - y - z, is said to have the orientation (90, 90, 90) relative to the global coordinate system.

### 4.2.2 State Variables

State variables can be used in ADAMS “to define scalar algebraic equations for independent use or as part of the plant input, plant output, or array elements” [32]. State variables will be used in this research to define the measured dynamics of the multibody system, including displacements, rotations, velocities, and forces.

In classic control theory, the model of a process is called a plant. The plant possesses input and output variables. The output of the plant is measured against a desired output signal, and the controller corrects the plant input to minimize the difference between the actual and desired output signals (Figure 4.3).
State variables are used in the ADAMS Controls Plug-in module to define the plant input and output variables. These are the variables entering and leaving the ADAMS subsystem during co-simulation in the MATLAB package.

4.2.3 ADAMS/View Function Builder

In the context of the present research, the Function Builder module is a tool used to create algebraic expressions using simulation states of the model that are evaluated during simulation. This is referred to as the run-time functionality of the Function Builder [32].

In general, the run-time functions that were required for this work were used to measure the deflection and rate of deflection of the damper or bushing or to access current values of state variables. The following list describes some of the functions utilized in the models with examples of each type of function.
DX, DY, DZ: Distance Along X, Y, and Z-axes

\[ DX(to\_marker, \ from\_marker) \] calculates the x-component of the relative translation vector from one marker to another (and similarly for DY and DZ).

VX, VY, VZ: Velocity Along X, Y, and Z-axes

\[ VX(to\_marker, \ from\_marker) \] calculates the x-component of the difference in velocity vectors of two markers (and similarly for VY and VZ). The velocity of the \textit{from} marker is subtracted from the velocity of the \textit{to} marker.

VR, Velocity Along Line-of-Sight

\[ VR(to\_marker, \ from\_marker) \] measures the relative velocity from one marker to another. Marker separation yields a positive value, while markers approaching each other yield a negative value.

ACCX, ACCY, ACCZ: Acceleration Along X, Y, and Z-axes

\[ ACCX(to\_marker, \ from\_marker) \] calculates the x-component of the difference in acceleration vectors of two markers (and similarly for ACCY and ACCZ). The acceleration of the \textit{from} marker is subtracted from the acceleration of the \textit{to} marker.

AX, AY, AZ: Angle About X, Y, and Z-axes

\[ AX(to\_marker, \ from\_marker) \] measures the rotation of one marker coordinate system relative to another about the X-axis in radians. The command

\[ AX(to\_marker, \ from\_marker) \ast RTOD \]

yields the value in degrees (RTOD converts from radians to degrees by multiplying by the factor \(180/\pi\) degrees/radian).

WX, WY, WZ: Angular Velocity About X, Y, and Z-axes

\[ WX(to\_marker, \ from\_marker) \] calculates the x-component of the difference in angular velocity vectors of two markers in radians per second (and similarly for WY and WZ). The angular velocity of the \textit{from} marker is subtracted from the angular velocity of the \textit{to} marker. Using the command

\[ WX(to\_marker, \ from\_marker) \ast RTOD \]

yields the value in degrees per second.

WDTX, WDTY, WDTZ: Angular Acceleration About X, Y, and Z-axes

\[ WDTX(to\_marker, \ from\_marker) \] calculates the x-component of the difference in angular acceleration vectors of two markers in radians per
second squared (and similarly for WDTY and WDTZ). The angular acceleration of the from_marker is subtracted from the angular acceleration of the to_marker. Using the command

\[
\text{WDX}(\text{to\_marker}, \text{from\_marker}) \times \text{RTOD}
\]

yields the value in degrees per second squared (RTOD: radians to degrees).

**VARVAL**: Algebraic Variable Value

VARVAL(State_variable) returns the present value of State_variable at the current time step of the simulation.

**ARYVAL**: Array Element Value

ARYVAL(Array_variable, element_number) supplies the present value of the element_numberth component of Array_variable at the current time step of the simulation.

### 4.2.4 ADAMS/Controls

The Controls Plug-in module is activated from ADAMS/View. It is used to establish a connection between the ADAMS model and Simulink models in MATLAB where controllers can be designed and utilized with the model. Input and output variables, \(PINPUT\) and \(POUTPUT\), are defined in ADAMS as well as the degree of linearity of the ADAMS model to be exported (i.e. linear or nonlinear). Selection of the nonlinear model will create conditions for a co-simulation to occur. During co-simulation, the ADAMS solver is utilized to solve the system dynamics while the neural computations are performed in Simulink with the MATLAB solver. Selecting the linear model representation, the ADAMS model is linearized about the equilibrium point and the state-space matrices are exported to MATLAB where the calculations are performed using the MATLAB solver. Since the goal of the current research is to implement the neural network-estimated forces in the ADAMS model and not a linear representation of the ADAMS model, the nonlinear model was selected.
4.2.5 Arrays and Filters

The vehicle data entering into ADAMS passes through a single input single output transfer function, $G(s)$, which functions as a low-pass filter. To utilize the filter, the input and output data must first be stored as elements of an array. The input array element, $U(t)$, is input to the transfer function. The output of the transfer function, $Y(t)$, contains the filtered data that will be applied to the multibody model for the next time-step computation.

\[ G(s) = \frac{\sum_{i=0}^{n} b_i s^i}{\sum_{i=0}^{m} a_i s^i}, \quad m \leq k \]  

Equation 4.1

where

\[ u(s) = \text{Laplace transform of the input, } U(t) \]

\[ y(s) = \text{Laplace transform of the output, } Y(t) \]

To determine an appropriate format of the transfer function, some background regarding Road Test Simulation (RTS) data processing is warranted. The standard RTS data sample rate of 409.6 samples per second allows for fatigue analysis of data up to one quarter of the sample rate, or 102.4 Hz [7]. The sample rate then guarantees that the peak values of the data will be recorded for signals up to 102.4Hz. In general, the frequency for fatigue analysis is limited to one half of the Nyquist frequency, which is equal to one half of the data sampling rate (204.8Hz). For a data sampling rate of 409.6 samples per second, this sets the upper limit of frequency for fatigue analysis of 102.4 Hz. Given that the road input data generally has negligible amplitude for signals above about 50Hz, a low pass filter of 100Hz should sufficiently filter out high frequency instability with little effect on the data in the range of 0-50 Hz.
A low pass filter corresponding to a cut-off frequency of 100Hz (628 rad/s) is described by Equation 4.2 and Figure 4.4:

\[
TFSISO_{lp\_filter} = \frac{628}{628 + s}
\]

Equation 4.2

Figure 4.4: The transfer function, TFSISO_{lp\_filter}, represents the low pass filter used to filter the incoming damping force data has a cutoff frequency of 100Hz.

4.2.6 ADAMS Dampers

In the case of the translational spring-damper element available in ADAMS, stiffness and damping coefficients or spline curves are specified in the definition and are used to calculate the force that results from the combined deflection of the spring and damper. The spring-damper element can be used to represent a sole damper by setting the stiffness coefficient equal to zero. Similarly, a spring can be represented by equating the damping to zero.

The original durability model being used for this study utilized separate connector elements for the spring and the damper. Since the focus of this analysis is the nonlinearity of the damper component, references to the spring will not be made, although the method can be applied to the spring in the same manner with the appropriate
data. An ADAMS representation of a damper is shown in Figure 4.5 with the corresponding dialog box.
Figure 4.5: The damper exerts equal and opposite forces that act along the line-of-sight between Sprung_mass.MARKER_3 and Unsprung_mass.MARKER_4. The dialog box is used to define the bodies and the damping coefficient constant expressed in units of N-s/mm or a spline curve that defines the damping force with respect to the velocity of the shock absorber deflection.
Equation 4.3 describes the force generated by the deflection of a linear damper with damping approximated by a constant damping coefficient.

\[ F_{\text{damping}} = -c \left( \frac{dr}{dt} \right) \]  

Equation 4.3

where,

- \( F_{\text{damping}} \) = damping portion of the translational spring-damper force
- \( c \) = constant damping coefficient
- \( r \) = relative deflection between the action and reaction bodies, the locations of which are indicated by the location of two markers.

Alternatively, damping can be defined using a spline function that describes the force as a function of velocity.

\[ (F_{\text{damping}})_{\text{spline}} = f \left( \frac{dr}{dt} \right) \]  

Equation 4.4

where,

- \( f \) = nonlinear function of the damper deflection velocity.

Spline curves are defined by specifying measured force and deflection velocity data points. The curve fit method used for interpolation can be a traditional cubic curve fit or an Akima cubic curve fit [32].

The spline curve used in the original ADAMS model is shown in Figure 4.6. At the current time, spline curve representation is the only nonlinear damping available in ADAMS. While the relationship is nonlinear, it is still a one-to-one relationship, providing a single force value for each value of the deflection velocity, unlike the behaviour exhibited during hysteresis. The data has been normalized by the maximum values to protect proprietary data.
Figure 4.6: The spline curve represents a nonlinear approximation of the damping force with respect to the rate of deflection of the damper. Both force and velocity data have been normalized with respect to their maximum respective values to protect proprietary data. Rebound or extension of the damper produces a positive deflection velocity and damping force, while the jounce condition or compression of the damper produces negative deflection rate and damping force.

Since the spline function is an approximation of the hysteresis loops that the damper experiences in actuality, the objective is to find a way to include the hysteresis effects in the damping force for more accurate loading estimation through the damper to the vehicle. Currently, ADAMS does not possess the ability to include this aspect of the damping in the model and this represents a serious compromise in the fidelity of the present generation of vehicle models which the current research will alleviate.

4.2.7 ADAMS Bushings

The ADAMS bushing element is a linear model with six degrees of freedom. Three translational and three rotational constant stiffness and damping coefficients are
used to calculate the forces and torques acting in three dimensions between the action and reaction bodies. The ADAMS representation of a bushing component and corresponding dialog box are shown in Figure 4.7.
The bushing exerts three translational and three rotational forces between shaft.MARKER_4 and sleeve.MARKER_3. Note that the ADAMS bushing definition aligns the z-axis along the bushing’s axial direction. If the bushing is symmetrical, the x-axis and y-axis coefficients are identical. The dialog box is used to define the bodies and the constant stiffness and damping coefficients.

<table>
<thead>
<tr>
<th>Name</th>
<th>BUSHING</th>
</tr>
</thead>
<tbody>
<tr>
<td>Action Body</td>
<td>sleeve</td>
</tr>
<tr>
<td>Reaction Body</td>
<td>shaft</td>
</tr>
</tbody>
</table>

Translational Properties (x,y,z components):
- Stiffness: 600.0, 300.0, 600.0
- Damping: 10.0, 5.0, 10.0
- Preload: 0.0, 0.0, 0.0

Rotational Properties (x,y,z components):
- Stiffness: 15.0, 15.0, 15.0
- Damping: 4.0, 4.0, 4.0
- Preload: 0.0, 0.0, 0.0

Force Display: On Action Body
The following equation describes the force generated by the deflection of a linear bushing.

\[
\begin{bmatrix}
F_x \\
F_y \\
F_z \\
T_x \\
T_y \\
T_z \\
\end{bmatrix} =
\begin{bmatrix}
K_{11} & 0 & 0 & 0 & 0 & x \\
0 & K_{22} & 0 & 0 & 0 & y \\
0 & 0 & K_{33} & 0 & 0 & z \\
0 & 0 & K_{44} & 0 & 0 & \alpha_x \\
0 & 0 & 0 & K_{55} & 0 & \alpha_y \\
0 & 0 & 0 & 0 & K_{66} & \alpha_z \\
\end{bmatrix}
\begin{bmatrix}
x \\
y \\
z \\
\alpha_x \\
\alpha_y \\
\alpha_z \\
\end{bmatrix}
- 
\begin{bmatrix}
C_{11} & 0 & 0 & 0 & 0 & v_x \\
0 & C_{22} & 0 & 0 & 0 & v_y \\
0 & 0 & C_{33} & 0 & 0 & v_z \\
0 & 0 & 0 & C_{44} & 0 & \omega_x \\
0 & 0 & 0 & 0 & C_{55} & \omega_y \\
0 & 0 & 0 & 0 & 0 & C_{66} & \omega_z \\
\end{bmatrix}
\begin{bmatrix}
v_x \\
v_y \\
v_z \\
\omega_x \\
\omega_y \\
\omega_z \\
\end{bmatrix}
+ 
\begin{bmatrix}
F_1 \\
F_2 \\
F_3 \\
T_1 \\
T_2 \\
T_3 \\
\end{bmatrix}
\]

Equation 4.5

Where,

\(F_x, F_y, F_z\) = translational bushing forces

\(T_x, T_y, T_z\) = rotational bushing torques

\([K_{ii}]\) = diagonal matrix of translational and rotational stiffness coefficients, \(i = 1\) to \(6\)

\(x, y, z\) = translational bushing deflection

\(\alpha_x, \alpha_y, \alpha_z\) = small angle bushing rotation deflection

\([C_{ii}]\) = diagonal matrix of translational and rotational damping coefficients, \(i = 1\) to \(6\)

\(v_x, v_y, v_z\) = translational bushing deflection velocity

\(\omega_x, \omega_y, \omega_z\) = rotational bushing deflection velocity

\(F_1, F_2, F_3\) = constant preload forces

\(T_1, T_2, T_3\) = constant preload torques.

\(^2\) According to ADAMS [32], two of the three rotations must remain less than 10° because the largest rotation must occur about an axis that does not rotate too far from its original position. If two or more large rotational deflections occurred simultaneously, the measured stiffness and damping coefficients would be invalid because they are measured for rotation about each of the three axes separately. In addition, \(\alpha_z\) should be chosen such that it is the axial rotation of the bushing, permitting rotation greater than 90°; a typical application might be a lower control arm bushing which might experience a rotation of 20° to 40° in service.
Typically, static and dynamic bushing stiffness and damping coefficients are determined during characterization tests performed by vendors. Generally, not all of the dynamic coefficients are provided, so assumptions are required to provide complete information to the model. A lack of dynamic rates for use in durability models requires assumptions based upon static measured values as well as “typical values used by the automotive industry” [12]. Developing neural network models of bushing behaviour eliminates the need for these assumptions because the data used is physically measured from the bushing.

4.3 Modified ADAMS Components

In general, the flexible connector components that are the focus of this research must be replaced with generic force components that can read in data through a variable placeholder. Both the damper and bushing generate forces that act between two bodies as a result of relative motion between the two bodies.

4.3.1 Modified Dampers

To create an appropriate representation of the damper for use with the neural networks, a number of tasks were performed. First, a state variable was created to contain the incoming damping force value from MATLAB, \textit{Damping-force-variable}. To support its role as a placeholder for incoming data through the Controls Plug-in module, it is given a value of zero (0). State variables storing data to be output to MATLAB were created measuring damper deflection and deflection velocity, \textit{Deformation} and \textit{Deformation_vel}. See Figure 4.8 for examples of these definitions.
Figure 4.8: The dialog boxes for creation of state variables are shown. The damping force variable (left) acts as a placeholder for incoming data and must show a value of zero (0). The deflection velocity (right) is an example of a variable that whose value will be calculated and output to MATLAB for processing. Note that it is a measurement of the relative velocity along the line-of-sight between the markers.

Array elements were defined to hold the incoming damping force variable, 

\textit{ARRAY\_damping\_input}, and the filtered damping force, \textit{ARRAY\_filtered\_damping} (see Figure 4.9).

Figure 4.9: \textit{ARRAY\_damping\_input} (left) is an input array (U) that used to filter \textit{damping\_force\_variable} (input to ADAMS from MATLAB) using the transfer function, \textit{TFSISO\_lp\_filter}. \textit{ARRAY\_filtered\_damping} is the output array (Y) from \textit{TFSISO\_lp\_filter}, the value of which is then used as the damping force.

Once the incoming damping force is filtered and stored in the output array, the value is used to define the damping force. The force is applied and the dynamics at that time step are calculated (Figure 4.10).
The modified damping force definition calls upon the filtered value of the damping force that was input from MATLAB.

The original damper force was defined to act between two markers, *MARKER_3* and *MARKER_4*, that were attached to the unsprung mass (action body) which represents the vehicle chassis, and the sprung mass (reaction body) which represents the vehicle body, respectively. These two markers were used to define a single component applied force, acting along the straight line of sight between the markers (Figure 4.11). Once the new force was defined, the original damper element was deactivated.
Figure 4.11: The modified damping force acts along the line of sight between the unsprung and sprung masses. It is defined to accept data from MATLAB input through the Controls Plug-in module.
4.3.2 Modified Bushings

A similar method was used to define the multi-dimensional force element that replaces the standard bushing flexible connector. Unlike the one-dimensional damping force, the generalized force used to replace the bushing component is composed of three forces and three torques. As illustrated in the example shown in Figure 4.12, the forces and torques are defined to act along and about the orientation of the original bushing markers. Two markers are required to define a bushing; one marker is attached to the action body and the second is attached to the reaction body. The axes defining the markers’ orientation coincide with the axial and radial orientation of the original bushing component. The axial force and torque act along and about the bushing’s axial axis (z), and the radial force and conical torque act along and about two perpendicular radial axes (x, y). Asymmetric bushings require definitions that distinguish between the x- and y-directions. The bushing used in the context of this research was symmetric and did not require this consideration.
Figure 4.12: This figure shows the orientation of a bushing that is defined to produce three forces and three torques that act between an action and a reaction body (this example shows a shaft acting as the action body and a sleeve acting as the reaction body). Note the distinction between the axial direction (z-axis) and the two radial directions (x-axis and y-axis), indicating that the largest rotation occurs about the z-axis. Asymmetric bushings possess different bushing stiffness and damping properties in each of the two radial directions.

Another distinction between the modified bushing and the modified damper is the coincidence of the two markers that are used to define the point of action and reaction on the two bodies. The two markers share the same location and orientation but are attached to two different bodies. As shown in Figure 4.7, the bushing acts between \textit{MARKER_4} on the shaft (action body) and \textit{MARKER_3} on the sleeve (reaction body).
For each of the three force and three torque components, a state variable was required to accommodate the incoming force or torque data from MATLAB. Again, since they act as placeholders, all state variables for incoming data were given values of zero (0). State variables measuring the relative deflection and rotation along and about the three marker axes were also defined to use as output to MATLAB. Examples are shown in Figure 4.13.

![Figure 4.13](image)

**Figure 4.13:** The dialog boxes for creation of state variables are shown. The bushing axial force variable (left) acts as a placeholder for incoming data and must show a value of zero (0). The x-axis deflection (right) is an example of a variable that whose value will be calculated and output to MATLAB for processing. Note that it is a measurement of the x-component of the relative translation vector between the markers.

Array elements were created for all of the incoming forces and torques and filtered forces and torques. Examples are shown for `ARRAY_axial_force_input` and `ARRAY_filtered_axial_force` in Figure 4.14.
Once the incoming bushing forces and torques are filtered and stored in the output array, the values are read by the modified bushing forces and torques. These filtered values are used to calculate the dynamics at that time step (Figure 4.15).
To create the general force component in the model, new markers must be defined as part of the action and reaction bodies with the same location and orientation as the original bushing component. This is vital to ensuring that the axial and radial directions of the forces and torques correspond to the original bushing definition.

The original bushing connector was defined to act between two markers that were attached to the unsprung mass (action body) and the sprung mass (reaction body), MARKER_3 and MARKER_4. Two new markers with the same location and orientation as MARKER_3 and MARKER_4 were used to define a general force component, acting between the two markers (Figure 4.16). Once the new force was defined, the original bushing element was deactivated.
4.4 ADAMS/Controls Plug-in Module

The ADAMS Controls Plug-in module builds a plant for use in a controls system. Generically, the plant contains the model processes or calculations of the system that is being modeled and controlled. Sensors, feedback, and control features are added to the plant to create a controls system. With regard to the current research, the plant contains the differential equations describing the motion of the multi-body model. Position and velocity information regarding the motion is measured by sensors, fed into a model that

Figure 4.16: The modified bushing force acting between MARKER_5 and MARKER_6 is now ready to accept data from MATLAB input through the ADAMS Controls Plug-in module.
calculates the reactive forces generated by the motion, and then the force information is passed back into the plant for processing of the motion at the next time step.

The ADAMS Controls Plug-in module requires that the ADAMS input and output variables be defined. In the case of the damper, the input variable, \textit{Damping	extunderscore force	extunderscore variable}, is listed in the plant input definition, PINPUT. The variables output to MATLAB, \textit{Deformation} and \textit{Deformation	extunderscore vel}, are listed in the plant output definition. See Figure 4.17 for an example of the plant input and output dialog boxes.

![Plant Input and Output Dialog Boxes](image)

Figure 4.17: The plant input and output define which state variables will feed the information into and out of ADAMS to MATLAB.

Once the inputs and outputs have been defined, the ADAMS subsystem is exported for use in the Simulink model. As shown in Figure 4.18, the system is named, the input and output are specified, the package which will use the exported system, the type of system to be exported and whether or not an initial static analysis should be included.
Figure 4.18: Exporting the ADAMS subsystem representation to external control packages, like MATLAB.

4.5 Neural Network Construction

Neural networks are used to model two types of data in the current study: damping force spline data and time series bushing force-deflection data.

The damper spline data is made up of data pairs representing damper deflection velocity and resulting damping force. These data pairs are not time dependent. The most suitable neural network model will create an appropriate curve fit through the measured data points and will allow interpolation within the limits of the data.

The bushing characterization data is time-series data. The corresponding force and deflection data points are measured as a function of time and will be used as training and validation data during the network development. When the force and deflection data at each point in time is plotted, a hysteresis curve is formed.
4.5.1 Spline Data

The damping force spline that was used in ADAMS to represent the damping force of the shock absorber in the ADAMS model was normalized to the range of [-1, 1] and is shown in Figure 4.19 and listed in Table 4.1. The data was measured and provided by the supplier of the shock absorbers.

Figure 4.19: A neural network developed for the training data input and output pairs shown here will produce a model that will allow interpolation within the limits of the data.

<table>
<thead>
<tr>
<th>Normalized Velocity (input)</th>
<th>Normalized Force (output)</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1.000</td>
<td>-1.000</td>
</tr>
<tr>
<td>-0.750</td>
<td>-0.901</td>
</tr>
<tr>
<td>-0.500</td>
<td>-0.827</td>
</tr>
<tr>
<td>-0.375</td>
<td>-0.796</td>
</tr>
<tr>
<td>-0.250</td>
<td>-0.772</td>
</tr>
<tr>
<td>-0.125</td>
<td>-0.748</td>
</tr>
<tr>
<td>-0.050</td>
<td>-0.730</td>
</tr>
<tr>
<td>0.000</td>
<td>-0.641</td>
</tr>
<tr>
<td>0.050</td>
<td>-0.469</td>
</tr>
<tr>
<td>0.125</td>
<td>-0.326</td>
</tr>
<tr>
<td>0.251</td>
<td>-0.175</td>
</tr>
<tr>
<td>0.375</td>
<td>-0.036</td>
</tr>
<tr>
<td>0.499</td>
<td>0.113</td>
</tr>
<tr>
<td>0.750</td>
<td>0.517</td>
</tr>
<tr>
<td>1.000</td>
<td>1.000</td>
</tr>
</tbody>
</table>

Table 4.1: The spline data input and output pairs will be used to develop a neural network model.
MATLAB offers the use of command line programming or a graphic user interface Neural Network Toolbox that can be opened by typing the command “nntool” at the MATLAB prompt. The Toolbox is a means of getting acquainted with the available network structures and parameters, but the final network generation was done using command line programming.

The user interface of the Neural Network Toolbox is a Network/Data Manager, shown in Figure 4.20. Selection of the “New Network…” button opens the window in Figure 4.21.

Figure 4.20: The Network/Data Manager is the first graphic user interface that opens when using the Neural Network Toolbox. The user can create or import data and neural networks.
Figure 4.21: The 1-1-1 feed forward network defined in the graphic user interface accepts input data ranging from -1 to 1, trains using the gradient descent with momentum method (`traingdm`), and monitors performance using the `MSE`. The activation function chosen for use in the hidden layer is the hyperbolic tangent while the output layer employs a linear function.

Once created, the network can be viewed and processed with the graphic user interface shown in Figure 4.22. The tabs along the top of the window show the processing and viewing options that can be used with the selected network.
Figure 4.22: The Network graphic user interface allows processing of the network, including viewing (shown), training (shown), simulating, and display of the weights and bias values.

The same 1-1-1 feed forward neural network shown in Figure 4.22 can be created and trained by typing the following at the MATLAB command lines:

```matlab
ff11 = newff([-1 1],[1 1],{'tansig' 'purelin'},'trainlm','learngdm','mse');
[ff11,ff11_tr,ff11_outputs] = train(ff11,pn,tn);
```

The training parameters used are listed in Table 4.2.
The first command line creates a new feed forward neural network, $ff11$.

Arguments of the command “newff” include the range of the input data, the number of neurons in the hidden and output layers, the functions used in each layer, the training algorithm, adaptive learning algorithm, and chosen performance function. The number of inputs to be used to create the network is inferred from the input variable (each row of the input variable corresponds to one input variable). The training data pairs, $(p_n, t_n)$, are used to train the $ff11$ network structure, yielding the final network, $ff11$, it’s associated epoch count and performance values (stored in $ff11_tr$) and the network outputs, $ff11_outputs$.

As a network is trained with a training data set, the error will continue to decrease as the network refines the model of the data. As the error continues to decrease for the training set, the ability of the network to generalize the behaviour of data outside the training set disappears. Early stopping is a process used during neural network training to halt the training process before the neural network model over-fits to the training data and loses its ability to generalize outside the training set. Principe [33] recommends that at least 10% of the total number of training points be taken as validation data to be used for early stopping during network training. After each pass of the training data through the network to determine the weight and bias adjustments, the error associated with the
network’s estimation of the validation data set is calculated. Training of the neural network is halted once the error in estimating the validation data ceases to decrease for a specified number of consecutive passes. This practice of early stopping ensures that the network that is developed adequately generalizes the behaviour of both the training and validation data sets without over-fitting the model to the training set alone.

For the set of fifteen data points that define the damping spline, thirteen points can be used for training and two points for validation. By measuring the neural network’s ability to generalize the two validation data points, training can cease before the mean square error ($MSE$) of the validation data becomes too large. The spline data was divided into two sets. The data sets are shown in Figure 4.23 and listed in Table 4.3.

![Damping Force Spline Data](image)

Figure 4.23: The spline data points were separated into training and validation data sets as shown in the plot.
Table 4.3: The spline data was separated into training and validation data sets as shown in the table.

In the same way as described above, the network was created and trained with the following command:

\[
[\text{ff11}, \text{ff11}_\text{tr}, \text{ff11}_\text{outputs}] = \text{train(}\text{ff11}, \text{pn}, \text{tn}, [], [], VV)\]

The addition of the last three arguments in the train command triggers the use of validation data (stored in \(VV.P\) and \(VV.T\)). The two bracket placeholder arguments are used for time-delay structures to input the initial delay conditions for the arrays of input and target data. The default values used here are zeros.

Feed forward neural networks with one hidden layer of various dimensions were employed to model the spline data. The previous chapter describes two guidelines for selection of the number of hidden neurons for the network with regard to the number of inputs to the network and the size of the training data.

Inputs: Given that there is one input \((n = 1)\), the maximum number of hidden layer neurons, \(h = 2n + 1\), was three.

Training Data: Given that there are fifteen training data sets \((N = 15)\), the maximum number of weights, \(w < N \times error\), required to achieve approximately 10% error was two.

85
The resulting neural network development is included the following chapter.

4.5.2 Hysteresis Data

Windowing the input data to include previous data points is one way that neural networks can be used to model hysteresis or time series data. Networks developed using this method are called time delay neural networks. By introducing each input data point with preceding values, the network can learn to recognize patterns in the data.

The procedure for developing these types of networks differs from the procedure outlined in Section 4.5.1 in the format of the input data. If one time delay is implemented, then the network has two inputs: the current data point and the data point from the previous time step, \((x_n, x_{n-1})\). If two time delays are implemented, then the network uses three inputs, \((x_n, x_{n-1}, x_{n-2})\). So, in general, for \(z\) time delays, there will be \((z+1)\) inputs to the network.

Consider the damping force versus velocity curves shown in Figure 4.24. The hysteresis curve is based upon the original spline curve that was modeled in Section 4.5.1.
Figure 4.24: The hysteresis curve that is based upon the ADAMS spline curve will be modeled using a time delay neural network structure.

Network inputs composed of zero, one, and two time delays were used to construct various neural network models of the data. Again, the procedure was performed both with and without early stopping. The complete normalized data set is listed in Table 4.4.
Table 4.4: The thirty points that make up the hysteresis curve were normalized by the maximum and minimum values and listed.

For network construction without early stopping, the full data set was used for training. To construct the time delay input vectors, each row of the input vector is staggered by one position. The zero-time delay input \((pn)\), one-time delay input \((pn1td)\), two-time delay input \((pn2td)\), and the corresponding target vector \((tn)\) are shown in Equation 4.6.
\[ \begin{align*}
    p_n &= \begin{bmatrix}
        p_n(1) & p_n(2) & p_n(3) & p_n(4) & \cdots & p_n(n)
    \end{bmatrix} \\
    p_{n\mid td} &= \begin{bmatrix}
        p_n(1) & p_n(2) & p_n(3) & p_n(4) & \cdots & p_n(n) \\
        p_n(n) & p_n(1) & p_n(2) & p_n(3) & \cdots & p_n(n-1)
    \end{bmatrix} \\
    p_{n2td} &= \begin{bmatrix}
        p_n(1) & p_n(2) & p_n(3) & p_n(4) & \cdots & p_n(n) \\
        p_n(n) & p_n(1) & p_n(2) & p_n(3) & \cdots & p_n(n-1) \\
        p_n(n-1) & p_n(n) & p_n(1) & p_n(2) & \cdots & p_n(n-2)
    \end{bmatrix} \\
    t_n &= \begin{bmatrix}
        t_n(1) & t_n(2) & t_n(3) & t_n(4) & \cdots & p_n(n)
    \end{bmatrix}
\end{align*} \]

**Equation 4.6**

The input vectors are read such that each column is considered an input point to the network during training. The network output is compared to the corresponding column entry in the target vector. Neural networks with one, three, five, and seven hidden layer neurons were constructed for zero, one, and two time delays for the full thirty-point data set.

The process was repeated using twenty-seven of the data points for training and three data points for validation (10% of the data set). The training and validation data sets are plotted in Figure 4.25 and listed in Table 4.5. Essentially, when the validation point is chosen from the full data set, the corresponding time delay points are also selected. For example, one of the validation points is the entry \( p_n(5) \), so the complete input for the two-time delay input that is used to estimate the value of \( t_n(5) \) is \([p_n(5), p_n(4), p_n(3)]\). The resulting neural network development is included the following chapter.
Normalized Damping Force vs. Normalized Velocity

Normalized Deflection Velocity
- Training ▲ Validation

Figure 4.25: The damping hysteresis data points were separated into training and validation data sets as shown in the plot. The validation points were randomly selected from the complete set of thirty points on the hysteresis curve. Their locations on the hysteresis curve are shown here.
Table 4.5: The hysteresis data was separated into training and validation data sets as shown in the table.

4.5.3 Time History Data

The use of time history data to create a neural network is similar to the use of hysteresis data. The benefit of using time history data is that the time delay structure of input data can be replaced by the use of displacement, velocity, and acceleration data. The use of these three states as input to a neural network is roughly equivalent to using a two-time delay input structure because the approximate velocity can be calculated with two consecutive displacement points and the approximate acceleration can be calculated with three consecutive displacement points. The use of the displacement, velocity, and
acceleration reduces the effect of the sample rate, which might become an issue if the neural network is developed with data sampled at a rate different from the simulation.

\[
v(t + 1) = \frac{d(t + 1) - d(t)}{\Delta t}
\]

\[
a(t + 2) = \frac{v(t + 2) - v(t + 1)}{\Delta t}
\]

\[
= \frac{d(t + 2) - d(t + 1) - d(t + 1) - d(t)}{\Delta t} \quad \text{Equation 4.7}
\]

\[
= \frac{\Delta t}{\Delta t}
\]

\[
= \frac{d(t + 2) - 2d(t + 1) + d(t)}{\Delta t^2}
\]

where,

\( v \) = relative translational velocity between two points

\( r \) = relative deflection along a straight line between two points

\( a \) = relative translational acceleration between two points

As described in Appendix A, force-displacement and torque-rotation data sets were collected for the lower control arm pivot bushing for a randomly generated signal containing frequencies and amplitude that encompassed the simulation signals at the location of the chosen bushing on the vehicle.

The data was fifty seconds in duration, thus at least five seconds could be used for validation during training or for testing following the training phase. The training data was chosen such that the maximum and minimum values of the signal resided in the training set. Due to the large data sets used for this portion of the research, the training algorithm was changed from the standard gradient descent with momentum method to increase speed of the training operation.

Goldman [16] and Demuth [6] both recommend using the Levenberg-Marquardt training algorithm to increase the speed of the training process for small to medium-sized
networks that utilize a performance metric based on the square of the error. For larger network structures, memory requirements increase with the square of the number of network weights and the Levenberg-Marquardt algorithm may not prove beneficial [18].

The Levenberg-Marquardt method is a variation of the back propagation methods discussed in Chapter 3. It is based upon a variation called Newton's method, where the weights are adjusted by an amount \(-H^T J^T\), where \(H\) is the Hessian, or matrix of second partial derivatives of the \(MSE\) and \(J\) is the Jacobian, or matrix of first partial derivatives of the \(MSE\). For sum of squares-type performance measures such as the \(MSE\),

\[ H = J^T J + \sum_p e_p \nabla^2 e_p = J^T J \]

because Newton's method assumes that \(\sum_p e_p \nabla^2 e_p\) is close to zero. Applying this result to Newton's method yields the Marquardt-Levenberg training method [18].

\[
w_{k+1} = w_k - \left[ J^T J + \mu I \right]^{-1} J^T e
\]

where,

\(w = \) weight

\(k = \) epoch, (pass of the training data through the network during training)

\(J = \) Jacobian of first derivatives of \(MSE\) with respect to weights and biases

\(J^T = \) transpose of the Jacobian matrix

\(\mu = \) variable scalar value (\(\mu\) large yields the gradient descent with momentum method, \(\mu = 0\) yields Newton's method)

Implementing the Levenberg-Marquardt training algorithm requires additional parameters shown in Table 4.6. The variable scalar, \(\mu\), is increased until the \(MSE\) is reduced, after which time it is decreased. An additional stopping condition results when \(\mu\) exceeds the stated maximum value that is set to place an upper bound on the value of \(\mu\).
<table>
<thead>
<tr>
<th>initial, $\mu$</th>
<th>0.001</th>
</tr>
</thead>
<tbody>
<tr>
<td>factor, $\mu_{\text{dec}}$</td>
<td>0.1</td>
</tr>
<tr>
<td>factor, $\mu_{\text{inc}}$</td>
<td>10</td>
</tr>
<tr>
<td>maximum, $\mu_{\text{max}}$</td>
<td>$1 \times 10^{10}$</td>
</tr>
</tbody>
</table>

Table 4.6: The additional parameters required by the Levenberg-Marquardt training method are listed here.

Determining which data to use as input for the neural network can be accomplished by examining the sensitivity of the network output with respect to the chosen inputs. In order to check the sensitivity of the network output to changes in the input data, the method developed by Goldman [16] was used.

A network using all available inputs and an additional randomly generated input was constructed and, for each element of the data set, the sensitivity was calculated. Note that the following equation is only valid for networks with $n$ inputs, one output, and a single hidden layer of $h$ neurons employing the hyperbolic tangent activation function and a linear output layer.

\[
\frac{\partial y}{\partial x_j} = 4 \sum_{k=1}^{h} w_{i,k} w_{j,k} e^{-2M_k} \left(1 + e^{2M_k}\right)^{-2}, \quad M_k = \sum_{c=1}^{n} w_{k,c} x_c + b_k
\]

Equation 4.9

The resulting sensitivities are then plotted for each data element and compared to the result of the sensitivity to the randomly generated input, which, by design, will be close to zero. The inputs with the most effect on the network output (i.e. highest sensitivity values) were chosen for the neural network design. Once the network inputs have been identified, the network construction proceeds as above. The resulting neural network development is included the following chapter.
4.6 Co-simulation

The process by which an ADAMS simulation runs in conjunction with controllers residing in the Simulink package is referred to as co-simulation. Simulink offers a platform whereby block diagrams illustrate the flow of data during the co-simulation. Calculations are performed in discrete time steps at the sample rate indicated in the ADAMS subsystem parameters. For each discrete time step, data is passed to the ADAMS subsystem where the ADAMS solver is employed to calculate the dynamics of the model. Once the system state has been calculated, the ADAMS output data is passed from the subsystem back to Simulink where the Simulink solver is used to perform the remaining calculations outside the ADAMS model. A simple illustrative example of a Simulink model is shown in Figure 4.26.

Figure 4.26: The Simulink model shows the flow of data during the co-simulation. The adams_sub block represents the ADAMS component of the model. In the example shown, the neural network estimated damping force is passed to the adams_sub block where the ADAMS solvers are employed to calculate the state of the system for the current time step. The input or excitation force, damper deflection, and damper deflection velocity are output and plotted together with the damping force. The damper deflection velocity is fed back and input to the neural network for estimation of the damper force for use in the next time step calculation.
4.6.1 Simulink

To obtain a block containing the neural network structure, the following MATLAB command specifies the name of the neural network and the desired sampling interval for all blocks that perform sampling during the simulation:

```
gensim(nnet_model_name, 0.00244)
```

The sampling interval, 0.00244 seconds, is equal to the reciprocal of the data sampling rate, 409.6 samples per second.

The resulting neural network subsystem block can be used to create a block that encompasses all necessary pre- and post-processing of the data (Figure 4.27).

![Diagram](image)

**Figure 4.27:** The neural network block produced by the “gensim” command is core to the neural network subsystem shown here. The incoming data is scaled using the maximum and minimum training data values, passed through the neural network, and then the resulting output is scaled back to the original range of the training target data.

The existence of the necessary velocity feedback loop (see Figure 4.26) creates what is referred to in the MATLAB literature as an “algebraic loop” [30]. The condition occurs when the output of a direct feed through block is required to compute the input to the same block. In this case, the velocity output from the “adams_sub” block is used to
calculate the damping force that is input to the “adams_sub” block. During the execution of a Simulink model, a simultaneous solution for all of the model’s elements is attempted. The presence of an algebraic loop will increase the time necessary to compute the solution, if it can be found at all. Due to the increased probability of computational errors with the presence of algebraic loops, the issue is resolved by including a single time step delay with the “Transport Delay” block. The delay of the feedback data flow into the “adams_sub” block by a single simulation time step resolves the issues associated with the algebraic loop. To implement time delays to the input data of a neural network, additional “Transport Delay” blocks are included as shown in Figure 4.28.
Figure 4.28: The use of time delay data as input to the neural network not only requires an additional “Transport Delay” block be added to the main model, but the neural network subsystem must be modified to accommodate the pre- and post-processing of more than one input.
As a result of exporting the nonlinear ADAMS model using the Controls Plug-In module, a MATLAB m-file is produced with the model name given in the Controls module with the *.m extension. This file is run in MATLAB and the appropriate variables are generated in the MATLAB workspace for use in the co-simulation. Typing “adams_sys” at the MATLAB command line will produce a Simulink “adams_sub” block for use in the Simulink model with all appropriate references to the ADAMS variables, ADAMS_*, generated from the execution of the m-file. The contents of the “adams_sub” are shown in Figure 4.29.
Figure 4.29: The parameters of the ADAMS plant include the name of output files, communication interval, mode of simulation, and mode of animation. Co-simulation is achieved by selecting the discrete mode of simulation, whereby the ADAMS solver is used to evaluate the multi-body dynamics of the system.
The communication interval between MATLAB and ADAMS during the co-simulation is set in the “ADAMS Plant” block along with the output file names and simulation modes (Figure 4.29).

Generating a Simulink model that is driven by data sampled at a specific rate (e.g. the standard RTS sample rate of 409.6 samples/second corresponds to a time interval between data points of 0.00244 seconds), it is important that the same time step values are inherited in the neural network input blocks as well as the “adams_sub” block. We have chosen that the transport delay blocks withhold data for the period of one time step (0.00244 seconds), which is the same as the communication interval with ADAMS. In this manner, the data calculated at each output step will have the same step size as the original durability data being used for correlation.

As mentioned previously, the discrete simulation method is selected to execute the co-simulation. For each time step, the ADAMS solver evaluates the system dynamics using its own solvers. The results are then passed back to Simulink for evaluation by MATLAB.

The animation mode can be set to either interactive or batch mode. Using interactive mode, ADAMS/View is launched and the model is opened to show a visualization of the simulation. During batch mode, the calculations are done using the ADAMS solver in the background without the ADAMS model visualization. Operating the co-simulation in batch mode offers significant time savings over the interactive mode setting (e.g. fifteen minutes versus four hours).

Simulink offers a library of block functions that can be used in the Simulink models (see Figure 4.30). The blocks used in the models are described as the details
require explanation. Results of the co-simulations are included and discussed in the following chapter.

Figure 4.30: The Simulink library contains many sets of predefined blocks including function generators, plotters, transport delay, gains, summing, and signal multiplexers.
5 SIMULATION RESULTS AND ANALYSIS

The development of the methodology for the creation of the neural network models and the procedure of integrating them with the ADAMS virtual durability model required a number of steps. The results of the neural network modeling task will be presented and discussed. Subsequently, the results of the dynamic simulations using these networks will be presented.

5.1 Neural Network Model of a Shock Absorber Force-Deflection Spline

The function of a shock absorber is to dampen vibrations that are transmitted from the tire-wheel assembly through the suspension to the vehicle body structure as the vehicle passes over a road surface. Damping of the resulting vibration prevents exposure of the passengers inside the vehicle to the road roughness. The shock absorber forces are, on average, proportional to the rate of shock absorber deflection. The constant of proportionality differs between jounce (shock compression) and rebound (shock extension) conditions. The shock force can be estimated as a nonlinear function of the deflection velocity with the use of a spline curve. The spline curve is defined by a collection of measured force and deflection velocity data from the shock absorber. Recall the spline curve (Figure 5.1) from the previous chapter that will be modeled in the current research.
Figure 5.1: The spline curve is fitted to the measured shock absorber force versus deflection velocity data points. Note the differences in slope of the curve between jounce and rebound conditions. This curve will be modeled by a neural network.

Networks with one, two, three, and five nonlinear hidden layer neurons and one linear output layer (known as 1-1-1, 1-2-1, 1-3-1, and 1-5-1 networks, respectively) were trained with the spline data with and without the use of the early stopping method that was introduced in Section 3.4. Results were evaluated in terms of the performance and a linear regression of the target output values versus the network output values. Recall from Section 3.4.6 that the network performance is measured by calculating the mean square error, $MSE$, between the target and network output data. With regard to the linear regression component of the results, the desired values for the slope, intercept, and coefficient of determination, $R^2$, would be one, zero, and one, respectively. (See Appendix B for the definition of $R^2$.) These values of the parameters would indicate that a linear regression of the network’s output versus the target values would produce a line.
with slope one and intercept zero along which all the plotted points would lay. In other words, the network was able to exactly predict the target data.

To appreciate the relative magnitude of the $MSE$ to the range of data, consider that normalization of the training data to the range of -1 to 1 implies that a performance target of 0.001 is equivalent to 0.05% of the training data’s full range. For the networks that met the performance target of 0.001 (see the shaded cells in Table 5.1), plots of training data as compared to network output are provided in Figures 5.2 to 5.4.

<table>
<thead>
<tr>
<th>No Early Stopping</th>
<th>With Early Stopping</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>MSE</strong></td>
<td><strong>m</strong></td>
</tr>
<tr>
<td>1-1-1</td>
<td>0.0038</td>
</tr>
<tr>
<td>1-2-1</td>
<td>0.0038</td>
</tr>
<tr>
<td>1-3-1</td>
<td>0.0010</td>
</tr>
<tr>
<td>1-5-1</td>
<td>0.0010</td>
</tr>
</tbody>
</table>

Table 5.1: The statistics are summarized for each of the chosen network structures with and without early stopping using a validation set. The performance ($MSE$), linear regression slope, intercept, and $R^2$ values are listed for each network.

Figure 5.2: The 1-3-1 neural network was developed using all fifteen spline points for training and did not use early stopping. The training yielded an $MSE$ of 0.0010.
The 1-5-1 neural network was also developed using all fifteen spline points and did not use early stopping. In this case, the training also yielded an $MSE$ of 0.0010. Since the slightly more complicated network structure yielded the same performance as the simpler network, the simpler network structure of the 1-3-1 neural network is preferable to the 1-5-1 network without early stopping.

The 1-5-1 neural network that was developed with early stopping could only utilize thirteen spline points for training and two data points for validation. The network performance during training was better than that of the networks developed without early stopping ($MSE = 0.0006$). Further examination of this network's accuracy compared to the 1-3-1 network of Figure 5.2 is required before a model can be selected.

Without early stopping, the 1-3-1 network produces satisfactory results for approximating the spline function with fifteen data points. With the use of early stopping, a slightly more complex structure is required to achieve a comparable
performance value using a 1-5-1 network. The comparison of the two neural networks with the original cubic spline that represents the shock absorber force versus deflection velocity is shown in Figure 5.5.

![Spline Damping Force: ADAMS and Neural Network Models](image)

Figure 5.5: The results of the 1-3-1 and 1-5-1 neural networks are shown with the ADAMS spline data. In general, the 1-3-1 network that was trained without early stopping models the ADAMS spline curve better, especially in the region of higher rebound velocity, where there are fewer training points available during training.

As recommended by Swingler [37], histograms showing the distribution of the weights for each of the three networks being considered are shown in Figures 5.6 and 5.7. Due to the more compact range (-4 to 5) of the weights of the 1-3-1 neural network compared to the 1-5-1 network without validation (-7 to 7), the 1-3-1 neural network will be used to model the damping curve in the co-simulation with the ADAMS model.
Figure 5.6: The weights of the 1-3-1 neural network that was trained without validation data, shows that the weights range from about $-4$ to $4$. The compact range of the weights is desirable, but the low quantity of weights in the network prevents the ability of the histogram to take on a more distinctive bell-shape that is desirable of such a distribution.

Figure 5.7: The weights of the 1-5-1 neural network that was trained with validation data, shows the weights range from about $-7$ to $7$. The increased range of the values of the weights makes it less desirable than the histogram of the 1-3-1 network weights shown in Figure 5.6. This histogram does not exhibit the desired bell-shape.
5.2 Neural Network Model of a Shock Absorber Force-Deflection Hysteresis Curve

The shock absorber force versus deflection velocity spline curve of the previous section serves to approximate the behaviour of the shock absorber during jounce and rebound events. In actuality, the curve is composed of an upper branch and a lower branch. The force generated in the shock absorber as it travels from maximum jounce velocity to maximum rebound velocity is different from the force resulting from maximum rebound velocity to maximum jounce velocity.

![Damper Force vs. Velocity Diagram]

Figure 5.8: The hysteresis curve differs from the spline curve in that the upper and lower branches correspond to different loading of the shock absorber. The upper branch shows that a larger force is produced in the damper than the lower branch. The branches correspond to shock absorber motion between maximum jounce velocity to maximum rebound velocity. A spline curve of shock absorber damping approximates the force produced regardless of the direction of loading. The hysteresis curve shown here was generated using the spline curve data used to model the shock absorber in ADAMS.
The hysteresis curve of Figure 5.8 will be modeled using a time delay neural network structure. The results of the development of various networks with and without early stopping are shown in Table 5.2. The lowest achieved performance values (MSE) were 0.001 (see the shaded cells in Table 5.2). Plots of the training results for the highlighted network structures are shown in Figures 5.9 to 5.13.

<table>
<thead>
<tr>
<th>No time delay</th>
<th>No Early Stopping</th>
<th>With Early Stopping</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MSE</td>
<td>m</td>
</tr>
<tr>
<td>1-1-1</td>
<td>0.0038</td>
<td>0.9881</td>
</tr>
<tr>
<td>1-3-1</td>
<td>0.0010</td>
<td>0.9967</td>
</tr>
<tr>
<td>1-5-1</td>
<td>0.0010</td>
<td>0.9948</td>
</tr>
<tr>
<td>1-7-1</td>
<td>0.0010</td>
<td>0.9968</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>One time delay</th>
<th>No Early Stopping</th>
<th>With Early Stopping</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MSE</td>
<td>m</td>
</tr>
<tr>
<td>2-1-1</td>
<td>0.0035</td>
<td>0.9890</td>
</tr>
<tr>
<td>2-3-1</td>
<td>0.0031</td>
<td>0.9946</td>
</tr>
<tr>
<td>2-5-1</td>
<td>0.0013</td>
<td>0.9961</td>
</tr>
<tr>
<td>2-7-1</td>
<td>0.0010</td>
<td>0.9969</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Two time delays</th>
<th>No Early Stopping</th>
<th>With Early Stopping</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MSE</td>
<td>m</td>
</tr>
<tr>
<td>3-1-1</td>
<td>0.0035</td>
<td>0.9890</td>
</tr>
<tr>
<td>3-3-1</td>
<td>0.0022</td>
<td>0.9929</td>
</tr>
<tr>
<td>3-5-1</td>
<td>0.0010</td>
<td>0.9967</td>
</tr>
<tr>
<td>3-7-1</td>
<td>0.0010</td>
<td>0.9967</td>
</tr>
<tr>
<td>3-9-1</td>
<td>0.0012</td>
<td>0.9960</td>
</tr>
</tbody>
</table>

Table 5.2: The statistics are summarized for each of the chosen network structures with and without early stopping using a validation set. The performance (MSE), linear regression slope, intercept, and R^2 values are listed for each network. Those networks with the highest training accuracy are highlighted for each method.
Figure 5.9: The 1-3-1 neural network was developed without early stopping using all thirty data points for training. The $MSE$ achieved during training was 0.0010. Note that the hysteresis was not modeled by this network structure. Instead, the model produces shock absorber forces between the upper and lower branches of the hysteresis curve.

Figure 5.10: The 2-7-1 neural network was trained with all thirty data point and did not use early stopping. The $MSE$ for the training data was 0.0010. The hysteresis is evident in the model, but does not follow the curve very well for jounce conditions (negative velocity).
Figure 5.11: The 3-7-1 neural network also used all thirty data points for training and no early stopping. Again, the training yielded an $MSE$ of 0.0010. The network shows better estimation of the hysteresis curve than the 2-7-1 model shown in Figure 5.10.

Figure 5.12: The 2-5-1 neural network was developed with early stopping, using twenty-seven points for training and three data points for validation. An $MSE$ value of 0.0041 was achieved. The hysteresis is evident in the model, but does not follow the data points to adequately generalize the target data as is also shown in the linear regression plot on the right by the spread of data about the line of regression between the target and network output.
Figure 5.13: The 3-7-1 neural network was trained using early stopping. Twenty-seven points were used for training and three data points for validation, yielding an $MSE$ of 0.0031. As in Figure 5.12, the hysteresis is evident in the model, but does not follow the data points to adequately estimate the target data as supported by the spread of data about the linear regression line between the target and network output.

Without early stopping, the 2-7-1 network produces slightly better results for approximating the hysteresis function with thirty data points. Note that the use of two inputs indicates one time-step delay in the input data. With the use of early stopping, the 2-5-1 network (one time-step delay) gives a better curve representation at the negative velocities than the 3-7-1 structure (two time-step delays).

The distribution of the network weights for the 2-7-1 and 2-5-1 neural network models will be examined using histogram plots to assist in selecting the best model of the hysteresis curve (Figures 5.14 and 5.15).
Figure 5.14: The weights of the 2-7-1 neural network that was trained without early stopping range from about -3 to 3. The higher number of weights at the boundaries makes the 2-7-1 network structure less desirable than the 2-5-1 network shown in Figure 5.15 because of the tendency of higher weights to amplify errors as they propagate through the network.

Figure 5.15: The weights of the 2-5-1 neural network that was trained with early stopping range from about -3 to 3. The distribution more closely resembles a bell-shaped curve compared to the 2-7-1 network shown in Figure 5.14.
With regard to the distribution of the weights for each of the two networks, the histogram of the 2-5-1 network weights (Figure 5.15) exhibits a more desirable bell-shaped distribution than the histogram of the 2-7-1 network weights (Figure 5.14). The output of each network is shown in Figure 5.16 with the original hysteresis curve. Both networks exhibit good estimation of the damping force, but due to the comparison of weights histograms, the 2-5-1 network developed with early stopping will be selected for co-simulation with the ADAMS modified damper.

**Figure 5.16:** The results of the 2-7-1 and 2-5-1 neural networks are shown with the original hysteresis data as the velocity is swept through from maximum jounce to maximum rebound. Both networks model the hysteresis curve well, but the 2-5-1 model was selected due to its better distribution of weights per the histogram shown in Figure 5.15. The discrepancy of both curves to model the hysteresis data at the far left of the plot is due to the use of zero initial time delay values for the networks' input.
5.3 Neural Network Results: Bushing Time Series Data

Radial and axial force-displacement data and axial and conical torque-rotation data were measured from a lower control arm bushing as described in Appendix A. The bushing orientation is shown in Figure 5.17.

![Figure 5.17: Bushing force and translational deflection data was measured in the axial and radial directions. Bushing torque and rotation deflection data was measured in the axial and radial directions. Radial torque is also referred to as conical torque. Since the bushing was symmetric, the models of the radial force and torque can be applied in two perpendicular radial directions during co-simulation.](image)

Each data set was fifty seconds long and was divided into training (thirty seconds), validation (ten seconds), and test data sets (ten seconds) for the network development. The training data set was selected so that the global maximum and minimum displacement and force values occurring in the fifty-second set were included in the training data set.

The first step to creating the neural network models of the bushing forces and torques was to determine which inputs have the greatest effect on the network output. Using Goldman's method for determining the relative sensitivity of the output to various inputs [16], the most influential input variables can be used to create a neural network with the most computationally efficient structures. Using the displacement data, the
velocity and acceleration were calculated using the difference quotient described
previously in Equation 4.7. The zero, one and two time-step delay data for the three
input variables were assembled with a random input variable to form a total of ten inputs.
These ten inputs were used to generate a variety of network structures with varying
numbers of hidden layer perceptrons. By examining trends in relative sensitivity
amongst the various network structures, conclusions were made about the most
appropriate input variable set to use for each of the four network models: axial force,
radial force, axial torque, and conical torque. Typical sensitivity plots for the force and
torque network models are shown in Figures 5.18 and 5.19 to illustrate the selection
process.
This plot illustrates typical results of the calculation of the sensitivity of the axial force with respect to the ten inputs. In general, it was determined that both the axial and radial bushing forces showed higher sensitivity to zero time-step delay displacement and velocity inputs compared to the acceleration and the one- and two-time delay inputs of the displacement, velocity, and acceleration.
Figure 5.19: This plot illustrates typical results of the calculation of the sensitivity of the conical torque with respect to the ten inputs. In general, it was determined that both the axial and conical bushing torques showed higher sensitivity to zero and one time-step delay rotation and angular velocity inputs compared to the remaining inputs.

As a result of the sensitivity analyses of the axial and radial force and the torsional and conical torque neural networks of various structures, it was found that, in general, the forces showed highest sensitivity to the displacement and velocity inputs without time delay (i.e. two network inputs) while the axial and conical torques were most sensitive to the displacement and velocity using one time-delay as network inputs (i.e. four network inputs).

After the input variables were determined, networks with various numbers of hidden layer perceptrons were generated for each of the force and torque components (axial force, radial force, axial torque, and conical torque). The performance of the networks with respect to their ability to estimate both training and test data was used to
determine the optimal structures for modeling the force and torque data from the bushing deformation data.

5.3.1 Axial Force Model

Using the translational axial displacement and velocity of the bushing deformation as inputs, neural networks with one, three, five, and seven hidden layer perceptrons were developed using the thirty second training data set and ten second validation data set.

The data in Table 5.3 contains the training and test performance values ($MSE$) and linear regression parameters for each structure. The results for the chosen model are shown in Figures 5.20 to 5.22.

<table>
<thead>
<tr>
<th>Training</th>
<th>With Early Stopping</th>
<th>Test Data</th>
<th>With Early Stopping</th>
</tr>
</thead>
<tbody>
<tr>
<td>No time delay</td>
<td>MSE</td>
<td>m</td>
<td>b</td>
</tr>
<tr>
<td>2-1-1</td>
<td>0.0011</td>
<td>0.9908</td>
<td>0.0004</td>
</tr>
<tr>
<td>2-3-1</td>
<td>0.0011</td>
<td>0.9908</td>
<td>0.0003</td>
</tr>
<tr>
<td>2-5-1</td>
<td>0.0011</td>
<td>0.9908</td>
<td>0.0004</td>
</tr>
<tr>
<td>2-7-1</td>
<td>0.0010</td>
<td>0.9907</td>
<td>0.0003</td>
</tr>
</tbody>
</table>

Table 5.3: All axial force neural networks had similar performance and linear regression statistics for both the training and testing phases. Upon examination of each network's weight histogram, the highlighted 2-3-1 neural network model was selected for implementation in the ADAMS model.
Figure 5.20: The weights of the 2-3-1 axial force neural network model exhibit a bell-shape concentrated about zero ranging from -2 to 2. Note the compact range of the weights and the large concentration weights near zero. The lack of large weight values indicates greater model stability as discussed in Section 3.4.7.
Figure 5.21: The ability of the 2-3-1 neural network to model the bushing axial force is shown here in the time domain. The good correlation for the training (top) and validation data (middle) illustrates the success of the network to generalize both the training and validation data sets during the training process. The success of the network to estimate the test data (bottom) shows the network's ability to model the axial force produced during axial translational bushing deflection that was not used during the training phase of network development.
Figure 5.22: The ability of the 2-3-1 neural network to estimate training, validation, and test data of the axial bushing force is shown here. Note the concentration of the cluster of test data points within the training and validation data point clusters. This illustrates the network's ability to model data to which it was not exposed during the training phase and thus shows the efficacy of the neural network approach.

5.3.2 Radial Force Model

Using the translational radial displacement and velocity of the bushing deflection as inputs, neural networks with one, three, five, and seven hidden layer perceptrons were developed using the thirty second training and ten second validation data sets.

The data in Table 5.4 contains the training and test performance and linear regression parameters for each structure. The results for the chosen model are shown in Figures 5.23 to 5.25.
Table 5.4: All radial force neural networks had similar performance and linear regression statistics for both the training and testing phases. Upon examination of each network’s weight histogram, the highlighted 2-3-1 neural network model was selected for implementation in the ADAMS model.

![Weights Histogram - 2-3-1 Neural Network With Validation](image)

Figure 5.23: The weights of the 2-3-1 radial force neural network model exhibit a bell-shape concentrated about zero ranging from -5 to 4.
Figure 5.24: The ability of the 2-3-1 neural network to model the bushing radial force is shown here in the time domain. The good correlation for the training (top) and validation data (middle) illustrates the success of the network to generalize the training and validation data sets during the training process. The success of the network to estimate the test data (bottom) shows the network's ability to model the radial force produced during radial translational bushing deflection that was not used during the training phase of network development.
Figure 5.25: The ability of the 2-3-1 neural network to estimate training, validation, and test data of the radial bushing force is shown here. Note the concentration of the cluster of test data points within the training and validation data point clusters. This illustrates the network’s ability to model data to which it was not exposed during the training phase.

5.3.3 Axial Torque Model

Using the axial rotation and angular velocity of the bushing deformation as inputs, neural networks with one, three, five, seven, and nine hidden layer perceptrons were developed using the thirty second training and ten second validation data sets.

The data in Table 5.5 contains the training and test performance and linear regression parameters for each structure. Results are shown in Figures 5.26 to 5.28.
Training With Early Stopping

<table>
<thead>
<tr>
<th></th>
<th>MSE</th>
<th>m</th>
<th>b</th>
<th>R²</th>
</tr>
</thead>
<tbody>
<tr>
<td>4-1-1</td>
<td>0.0017</td>
<td>0.9855</td>
<td>0.0016</td>
<td>0.9933</td>
</tr>
<tr>
<td>4-3-1</td>
<td>0.0017</td>
<td>0.9856</td>
<td>0.0016</td>
<td>0.9933</td>
</tr>
<tr>
<td>4-5-1</td>
<td>0.0016</td>
<td>0.9868</td>
<td>0.0003</td>
<td>0.9938</td>
</tr>
<tr>
<td>4-7-1</td>
<td>0.0016</td>
<td>0.9877</td>
<td>0.0014</td>
<td>0.9939</td>
</tr>
<tr>
<td>4-9-1</td>
<td>0.0016</td>
<td>0.9878</td>
<td>0.0013</td>
<td>0.9939</td>
</tr>
</tbody>
</table>

Test Data With Early Stopping

<table>
<thead>
<tr>
<th></th>
<th>MSE</th>
<th>m</th>
<th>b</th>
<th>R²</th>
</tr>
</thead>
<tbody>
<tr>
<td>4-1-1</td>
<td>0.0019</td>
<td>0.9520</td>
<td>0.0107</td>
<td>0.9927</td>
</tr>
<tr>
<td>4-3-1</td>
<td>0.0019</td>
<td>0.9540</td>
<td>0.0106</td>
<td>0.9927</td>
</tr>
<tr>
<td>4-5-1</td>
<td>0.0017</td>
<td>0.9794</td>
<td>0.0121</td>
<td>0.9933</td>
</tr>
<tr>
<td>4-7-1</td>
<td>0.0017</td>
<td>0.9777</td>
<td>0.0087</td>
<td>0.9933</td>
</tr>
<tr>
<td>4-9-1</td>
<td>0.0017</td>
<td>0.9727</td>
<td>0.0088</td>
<td>0.9933</td>
</tr>
</tbody>
</table>

Table 5.5: All axial torque neural networks had similar performance and linear regression statistics for both the training and testing phases. Upon examination of each network's weight histogram, the highlighted 4-7-1 neural network model was selected for implementation in the ADAMS model.

Axial Torque Neural Network
Weights Histogram - 4-7-1 Neural Network With Validation

![Weights Histogram](image)

Figure 5.26: The weights of the 4-7-1 axial torque neural network model exhibit a bell-shape concentrated about zero ranging from -4 to 4.

127

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
Figure 5.27: The time histories of the normalized axial torque from the training (top), validation (middle), and test data sets (bottom) are shown for the 4-7-1 neural network model. The time delay network structure successfully generalized the axial torque response to bushing axial rotation during training. Once the network was successfully trained, it was then able to correctly estimate the axial torque from previously unseen input data.
5.3.4 Conical Torque Model

Using the conical rotation and angular velocity of the bushing deformation as inputs, neural networks with one, three, five, seven, and nine hidden layer perceptrons were developed using the thirty second training and ten second validation data sets.

The data in Table 5.6 contains the training and test performance and linear regression parameters for each structure. Results are shown in Figures 5.29 to 5.31.
Training With Early Stopping

<table>
<thead>
<tr>
<th></th>
<th>MSE</th>
<th>m</th>
<th>b</th>
<th>R²</th>
</tr>
</thead>
<tbody>
<tr>
<td>One time</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>delay</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4-1-1</td>
<td>0.0024</td>
<td>0.9759</td>
<td>0.0006</td>
<td>0.9885</td>
</tr>
<tr>
<td>4-3-1</td>
<td>0.0024</td>
<td>0.9776</td>
<td>0.0006</td>
<td>0.9888</td>
</tr>
<tr>
<td>4-5-1</td>
<td>0.0024</td>
<td>0.9760</td>
<td>0.0014</td>
<td>0.9887</td>
</tr>
<tr>
<td>4-7-1</td>
<td>0.0024</td>
<td>0.9777</td>
<td>0.0006</td>
<td>0.9888</td>
</tr>
<tr>
<td>4-9-1</td>
<td>0.0024</td>
<td>0.9780</td>
<td>0.0006</td>
<td>0.9889</td>
</tr>
</tbody>
</table>

Test Data With Early Stopping

<table>
<thead>
<tr>
<th></th>
<th>MSE</th>
<th>m</th>
<th>b</th>
<th>R²</th>
</tr>
</thead>
<tbody>
<tr>
<td>One time</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>delay</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4-1-1</td>
<td>0.0027</td>
<td>0.9564</td>
<td>0.0100</td>
<td>0.9873</td>
</tr>
<tr>
<td>4-3-1</td>
<td>0.0026</td>
<td>0.9574</td>
<td>0.0005</td>
<td>0.9872</td>
</tr>
<tr>
<td>4-5-1</td>
<td>0.0026</td>
<td>0.9737</td>
<td>0.0070</td>
<td>0.9871</td>
</tr>
<tr>
<td>4-7-1</td>
<td>0.0027</td>
<td>0.9776</td>
<td>-0.0112</td>
<td>0.9870</td>
</tr>
<tr>
<td>4-9-1</td>
<td>0.0026</td>
<td>0.9777</td>
<td>0.0090</td>
<td>0.9872</td>
</tr>
</tbody>
</table>

Table 5.6: All conical torque neural networks had similar performance and linear regression statistics for both the training and testing phases. Upon examination of each network's weight histogram, the highlighted 4-7-1 neural network model was selected for implementation in the ADAMS model.

Figure 5.29: The weights of the 4-7-1 conical torque neural network model exhibit a bell-shape concentrated about zero ranging from -4 to 7.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
Figure 5.30: The time histories of the normalized conical torque from the training (top), validation (middle), and test data sets (bottom) are shown for the 4-7-1 neural network model. The time delay network structure successfully generalized the conical torque response to bushing conical rotation during training. Once the network was successfully trained, it was then able to correctly estimate the torque generated from previously unseen input data.
The ability of the 4-7-1 neural network to estimate training, validation, and test data of the conical bushing torque is shown here. Note the concentration of the cluster of test data points within the training and validation data point clusters. This illustrates the network's ability to model data to which it was not exposed during the training phase. The width of the clusters is a result of the small signal range relative to the axial force, axial torque, and radial force data.

5.4 Implementation and Validation of Vehicle Model

The methodology of implementing neural network models in a full vehicle co-simulation was developed using models of increasing complexity.

To ensure that the data was flowing properly through the neural network and ADAMS models during co-simulation, the single degree of freedom model was first used to implement the neural network model of the shock absorber force versus deflection velocity spline curve. A baseline was established by running a simulation of the ADAMS model using the ADAMS spline representation of the damper. Following this simulation, the ADAMS damper was replaced by an equivalent force set up to receive
data input from Simulink in the form of a look-up table and a neural network model of the spline curve. Following correlation of the spline network model, a neural network of the damping hysteresis based upon the original spline was used to generate the damping force.

Once the method of neural network integration was validated for the simplified model, the method was applied to a full vehicle durability model.

5.4.1 Co-simulation Using a Neural Network Model of a Shock Absorber Force versus Deflection Velocity Spline Curve

The single degree of freedom co-simulation ran using two representations of the damping force in the Simulink model (see Figure 5.32). The “adams_sub” block was defined in ADAMS using the $P_{\text{OUTPUT}}$ variable to have as its outputs the input or excitation force, the damper deflection, and the damper deflection velocity. The $P_{\text{INPUT}}$ variable was defined such that the damper force was input to the “adams_sub” block. Both a spline table look-up and the 1-3-1 neural network model of the spline curve were included in the Simulink model for comparison, although the neural network model was used to calculate the damping force resulting from the damper deflection velocity (Figure 5.32).

The co-simulation ran using step, sinusoidal, and random input forces to the unsprung mass. The results of the co-simulation using each damper force model were compared to the ADAMS simulation. The results showed excellent correlation and are included in the following section with the results of the hysteresis neural network damping model.
5.4.2 Co-simulation Using a Neural Network Model of a Shock Absorber Force versus Deflection Velocity Hysteresis Curve

The co-simulation of the single degree of freedom model was repeated using the neural network model of the hysteresis damping shown above in Figure 5.8. The results of the co-simulation using the random input signal are shown here in Figures 5.33 to 5.37.
The random input shown was one of the signals used to excite the single degree of freedom model during the co-simulation. A baseline response to the excitation was measured in ADAMS and then co-simulations using the spline and hysteresis neural network models were performed. The results are shown in Figures 5.34 to 5.37.
Figure 5.34: This portion of the time history plot of the shock absorber force generated in the co-simulation of the single degree of freedom model for both the spline (blue) and hysteresis (green) neural network models are shown in comparison with ADAMS model baseline (black). The neural network model estimates higher positive and negative peaks in the time history. This corresponds to the position of the hysteresis forces relative to the spline curve shown in Figure 5.8.
Figure 5.35: The frequency plots of the shock absorber force generated in the co-simulation of the single degree of freedom model for both the spline (blue) and hysteresis (green) neural network models are shown in comparison with ADAMS model baseline (black) from 0 – 45 Hz. The full time history of the response to the random input signal is used to calculate this plot. In general, the hysteresis model estimates higher amplitude signals for most frequencies (e.g., $\frac{0.175}{0.105} = 1.69$ at 1 Hz). This agrees with the position of the hysteresis forces relative to the spline curve shown in Figure 5.8. Note that all signal amplitudes are very small at frequencies above 10 Hz.
Figure 5.36: This portion of the time history plot of the shock absorber deflection generated in the co-simulation of the single degree of freedom model for both the spline (blue) and hysteresis (green) neural network models are shown in comparison with ADAMS model baseline (black). The neural network model estimates slightly higher peaks for the higher frequency component of the response, but the low frequency content the motion shows good correlation for the three cases.
Figure 5.37: This frequency plot of the shock absorber deflection generated in the co-simulation of the single degree of freedom model for both the spline (blue) and hysteresis (green) neural network models are shown in comparison with ADAMS model baseline (black) from 0 – 30 Hz. The full time history for the response to the random input signal is used to calculate this plot. The amplitudes of the spline and hysteresis models correlate well with the ADAMS model. At about 1 Hz, the amplitude of the hysteresis model is \[
\frac{0.107}{0.118} = 0.95
\]
of the amplitude of the ADAMS baseline.
5.4.3 Co-simulation Using a Neural Network Model of Bushing Forces and Torques

The first task in establishing an ADAMS model baseline was to ensure that the bushing stiffness and damping coefficients in the existing full vehicle ADAMS model were the same as the measured coefficients of the physical specimen. Values obtained from bushing measurements performed during the data acquisition (see Appendix A) and by the actual automotive parts manufacturer were used to modify the stiffness and damping coefficients of the lower control arm pivot bushings in the left and right lower control arms of the front suspension system. A fifty-second virtual durability simulation was run using the same virtual road surface chosen by Wood [38], so that the results could be compared with the previous research and used as a baseline for the neural network results.

After establishing the baseline vehicle response, the process was initiated to replace the bushing forces and torques with neural network-estimated forces and torques. Once again, steps were taken to ensure proper flow of the data in Simulink as well as correct polarity of the calculated force and torque. An example of the Simulink models used to implement the neural network models of the bushing forces and torques are shown in Figures 5.38 to 5.41.
The Simulink model shows that the axial displacement and velocity output from the ADAMS subsystem were used to calculate the axial bushing force and the axial rotation and angular velocity were used to calculate the bushing axial torque using the neural network models developed in Section 5.3. The axial force and torque were then input to the ADAMS subsystem. Subsystems are expanded and shown in Figures 5.39 to 5.41.
Figure 5.39: The calculation of the axial bushing force and torque uses the neural network models developed in Section 5.3. The inclusion of the transport delay of a single time step is included here to resolve the algebraic loop that results from using the axial displacement and velocity feedback to calculate the input force to the ADAMS subsystem. The gain values are used to ensure the correct polarity of the force as it acts between the lower control arm and the vehicle body in the ADAMS model.
Figure 5.40: The bushing axial force neural network subsystem contains the pre- and post-processing functions along with the neural network model that was developed in Section 5.3. The data defining the maximum and minimum axial displacements, velocities, and forces from the training data is used to scale the network inputs and outputs.
Figure 5.41: The bushing axial torque neural network subsystem contains the pre- and post-processing functions along with the neural network model that was developed in Section 5.3. The data defining the maximum and minimum axial rotations, angular velocities, and torques from the training data is used to scale the network inputs and outputs. Note the additional time delay block just previous to the 4-7-1 Torsional Neural Network block. This delay provides the delayed rotation and angular velocity inputs required for the neural network model.

To ensure the correct polarity of the forces and torques, the ADAMS constant stiffness and damping coefficients were used in Simulink to calculate the bushing forces and torques. The signals were compared with the neural network outputs to verify the
polarity of the forces and torques before they were input to the ADAMS subsystem.

Once the polarity was determined, the co-simulation was repeated with the forces from
the neural network models fed into the ADAMS subsystem.

Due to memory limitations of the software, the neural network implementation
required that some concessions be made to implement the models by parts. To reduce the
number of variables and processing in the ADAMS model, the unfiltered force and torque
data was applied in the ADAMS model for the fifty-second segment of road simulation
data. Also, the neural networks were implemented in four sets of pairs:

1. Front and rear bushing axial force
2. Front and rear bushing axial torque
3. Front bushing radial forces, X and Y directions
4. Front bushing conical torques, X and Y directions

The results of the ADAMS baseline and neural network co-simulations are
presented in Figures 5.42 to 5.69. For each result set, the motions and resulting force or
torque are compared in the time and frequency domains.
Figure 5.42: The results of the co-simulation using neural network models of both the front and rear axial bushing forces show that the neural network model correlates extremely well with the ADAMS simulation with regard to the normalized front normalized bushing axial displacements. This portion of the results from 30 to 35 seconds contains the global peaks for the full 50 second simulation showing that the method developed in the present research provides a valid simulation over the entire range of the physical phenomenon.
Figure 5.43: The results of the co-simulation using neural network models of both the front and rear axial bushing forces show that the neural network model correlates extremely well with the ADAMS simulation with regard to the normalized rear normalized bushing axial displacements. This portion of the results from 30 to 35 seconds contains the global peaks for the full 50 second simulation showing that the method developed in the present research provides a valid simulation over the entire range of the physical phenomenon.
Figure 5.44: The frequency plots for the front normalized axial displacements were calculated using the full 50 seconds of road simulation data. The plots show good correlation between the ADAMS simulation and the co-simulation using the axial force neural network model. The peak located at 0 Hz shows the results from the co-simulation results are about $\sqrt{\frac{0.04}{0.03}} = 1.15$ times (about 15% larger than) the ADAMS simulation results, which corresponds to a small difference in the mean value of the signal through the full simulation.
Figure 5.45: The frequency plots for the rear normalized axial displacements were calculated using the full 50 seconds of road simulation data. The plots show good correlation between the ADAMS simulation and the co-simulation using the axial force neural network model. The peak located at 0 Hz shows the results from the co-simulation results are about \( \frac{0.04}{\sqrt{0.03}} = 1.15 \) times (about 15% larger than) the ADAMS simulation results, which corresponds to a small difference in the mean value of the signal through the full simulation.
Figure 5.46: The results of the co-simulation using neural network models of both the front and rear axial bushing forces show that the neural network model correlates well with the ADAMS simulation with regard to the normalized axial force produced in the front bushing. This portion of the results from 30 to 35 seconds contains the global peaks for the 50 second simulation showing that the method developed in the present research provides a valid simulation over the entire range of the physical phenomenon.
The results of the co-simulation using neural network models of both the front and rear axial bushing forces show that the neural network model correlates well with the ADAMS simulation with regard to the normalized axial force produced in the rear bushing. This portion of the results from 30 to 35 seconds contains the global peaks for the 50 second simulation showing that the method developed in the present research provides a valid simulation over the entire range of the physical phenomenon.
Figure 5.48: The frequency plot for the front normalized axial bushing force was calculated using the full 50 seconds of road simulation data. The plot shows good correlation for all frequencies between the ADAMS simulation and the co-simulation using the neural network models of both front and rear axial forces.
Figure 5.49: The frequency plot for the rear normalized axial bushing force was calculated using the full 50 seconds of road simulation data. The plot shows good correlation for all frequencies between the ADAMS simulation and the co-simulation using the neural network models of both front and rear axial forces.
The results of the co-simulation using neural network models of both the front and rear axial bushing torques show that the neural network model correlates well with the ADAMS simulation with regard to the front bushing normalized axial rotation. This portion of the results from 30 to 35 seconds contains the global peaks for the 50 second simulation showing that the method developed in the present research provides a valid simulation over the entire range of the physical phenomenon.
The results of the co-simulation using neural network models of both the front and rear axial bushing torques show that the neural network model correlates well with the ADAMS simulation with regard to the rear bushing normalized axial rotation. This portion of the results from 30 to 35 seconds contains the global peaks for the 50 second simulation showing that the method developed in the present research provides a valid simulation over the entire range of the physical phenomenon.
Figure 5.52: The frequency plot for the front bushing normalized axial rotation was calculated using the full 50 seconds of road simulation data. The plot shows good correlation between the ADAMS simulation and the co-simulation using the axial force neural network model for all frequencies.
Figure 5.53: The frequency plot for the rear bushing normalized axial rotation was calculated using the full 50 seconds of road simulation data. The plot shows good correlation between the ADAMS simulation and the co-simulation using the axial force neural network model for all frequencies.
The results of the co-simulation using neural network models of both front and rear axial bushing torques show that the neural network model predicts that the front bushing reacts with a smaller negative torque than the constant coefficient stiffness and damping of the ADAMS bushing model. Despite this difference between the models, the rotation of the front bushing correlates well (see Figure 5.50). This portion of the results from 30 to 35 seconds contains the global peaks for the simulation.
The results of the co-simulation using neural network models of both front and rear axial bushing torques show that the neural network model predicts that the rear bushing reacts with a smaller negative torque than the constant coefficient stiffness and damping of the ADAMS bushing model. Despite this difference between the models, the rotation of the rear bushing correlates well (see Figure 5.51). This portion of the results from 30 to 35 seconds contains the global peaks for the simulation.
Figure 5.56: The frequency plot for the front normalized bushing axial torque was calculated using the full 50 seconds of road simulation data. The plot shows that at 0 Hz the results from the co-simulation are approximately $\sqrt{\frac{3.2}{4.8}} = 0.82$ times the ADAMS simulation results. This result corresponds to the shift in mean value of the full 50 second signal compared to the ADAMS simulation.
Figure 5.57: The frequency plot for the rear normalized bushing axial torque was calculated using the full 50 seconds of road simulation data. The plot shows that at 0 Hz the results from the co-simulation are approximately $\sqrt{\frac{3.2}{4.8}} = 0.82$ times the ADAMS simulation results. This result corresponds to the shift in mean value of the full 50 second signal compared to the ADAMS simulation.
The results of the co-simulation using neural network models of the radial bushing forces in both the X- and Y-directions shows that the neural network model correlates extremely well with the ADAMS simulation with regard to the normalized front bushing radial displacements in the X-direction. This portion of the results from 30 to 35 seconds contains the global peaks for the 50 second simulation showing that the method developed in the present research provides a valid simulation over the entire range of the physical phenomenon.
Figure 5.59: The results of the co-simulation using neural network models of the radial bushing forces in both the X- and Y-directions shows that the neural network model correlates extremely well with the ADAMS simulation with regard to the normalized front bushing radial displacements in the Y-direction. This portion of the results from 30 to 35 seconds contains the global peaks for the 50 second simulation showing that the method developed in the present research provides a valid simulation over the entire range of the physical phenomenon.
Figure 5.60: The frequency plot for the front bushing normalized radial displacement in the X-direction was calculated using the full 50 seconds of road simulation data. The plot shows good correlation between the ADAMS simulation and the co-simulation using the radial force neural network model for all frequencies.
Figure 5.61: The frequency plot for the front bushing normalized radial displacement in the Y-direction was calculated using the full 50 seconds of road simulation data. The plot shows good correlation between the ADAMS simulation and the co-simulation using the radial force neural network model for all frequencies.
The results of the co-simulation using neural network models of the conical bushing torques in both the X- and Y-directions show that the neural network model correlates extremely well with the ADAMS simulation with regard to the normalized conical rotation in the X-direction. This portion of the results from 30 to 35 seconds contains the global peaks for the 50 second simulation showing that the method developed in the present research provides a valid simulation over the entire range of the physical phenomenon.
Figure 5.63: The results of the co-simulation using neural network models of the conical bushing torques in both the X- and Y-directions show that the neural network model correlates extremely well with the ADAMS simulation with regard to the normalized conical rotation in the Y-direction. This portion of the results from 30 to 35 seconds contains the global peaks for the 50 second simulation showing that the method developed in the present research provides a valid simulation over the entire range of the physical phenomenon.
Figure 5.64: The frequency plot for the front bushing normalized conical rotation in the X-direction was calculated using the full 50 seconds of road simulation data. The plot shows good correlation between the ADAMS simulation and the co-simulation using the conical torque neural network model for all frequencies.
Figure 5.65: The frequency plot for the front bushing normalized conical rotation in the Y-direction was calculated using the full 50 seconds of road simulation data. The plot shows good correlation between the ADAMS simulation and the co-simulation using the conical torque neural network model for all frequencies.
The results of the co-simulation using neural network models of the conical bushing torques in both the X- and Y-directions show that the neural network model correlates well with the ADAMS simulation with regard to the normalized conical torque produced in the X-direction. This portion of the results from 30 to 35 seconds contains the global peaks for the 50 second simulation showing that the method developed in the present research provides a valid simulation over the entire range of the physical phenomenon.
Figure 5.67: The results of the co-simulation using neural network models of the conical bushing torques in both the X- and Y-directions show that the neural network model correlates well with the ADAMS simulation with regard to the normalized conical torque produced in the Y-direction. This portion of the results from 30 to 35 seconds contains the global peaks for the 50 second simulation showing that the method developed in the present research provides a valid simulation over the entire range of the physical phenomenon.
Figure 5.68: The frequency plot for the front bushing normalized conical torque in the X-direction was calculated using the full 50 seconds of road simulation data. The plot shows excellent correlation between the ADAMS simulation and the co-simulation using the conical torque neural network model for all frequencies.
Figure 5.69: The frequency plot for the front bushing normalized conical torque in the Y-direction was calculated using the full 50 seconds of road simulation data. The plot shows excellent correlation between the ADAMS simulation and the co-simulation using the conical torque neural network model for all frequencies.
5.5 Correlation: Laboratory and Simulation Results

To compare the results of the suspension motion and loads transmitted through the suspension to the vehicle structure, the lower control arm angle and shock force were identified for comparison of the results for the RTS laboratory simulation described above in Section 1.2.3, ADAMS model virtual simulation, and co-simulation data acquired for the four cases of neural network model implementation described in the previous section. The data is presented in Figures 5.70 to 5.73 in the time and frequency domains.
Comparison of Normalized Lower Control Arm Angle

Figure 5.70: The ADAMS simulation and neural network co-simulation results correlate well with the RTS laboratory data with regard to the normalized lower control arm angle. This portion of the results from 30 to 35 seconds contains the global peaks for the 50 second simulation.
Figure 5.71: The frequency plot for the normalized lower control arm angle were calculated using 50 seconds of road simulation data. The plots show that the ADAMS simulation and all neural network co-simulations correlate well with the RTS laboratory results for all frequencies.
Figure 5.72: The ADAMS simulation and neural network co-simulation results for the damping force show higher positive peak forces and a slight upward shift in the average load towards zero compared to the RTS laboratory data. This portion of the results from 30 to 35 seconds contains the global peaks for the 50 second simulation.
Figure 5.73: The frequency plot for the normalized shock force was calculated using the full 50 seconds of road simulation data. The plot shows that at 0 Hz the results of all virtual simulations are approximately $\frac{0.53}{1.56} = 0.58$ times the RTS laboratory results, corresponding the mean shift towards zero shown in Figure 5.72. The higher peaks in the time history (Figure 5.72) are evident in the frequency plot at frequencies above 0.3 Hz.
The results indicate that the neural network models of the bushing forces and torques all produce similar dynamic motion of the lower control arm, but transmit loading that exhibits higher peak values to the shock absorber. The higher peak loads transmitted to the components would play a crucial role in the fatigue analysis of the components, possibly leading to a reduction in life of the parts.
6 CONCLUSIONS AND RECOMMENDATIONS

This chapter discusses the results and implications of the current research as they relate to the objectives of the present research program. Recommendations for further work are also made to assist in guiding the work of future researchers.

6.1 Conclusions
1. Neural network models of automotive components that exhibit nonlinear force-deflection behaviour, including hysteresis, were created using empirical data that was acquired for this purpose. In doing so, a comprehensive model of elastomeric bushing behaviour was developed for use in full-vehicle virtual durability simulations.

2. Integration of neural network models into simplified and complex multibody dynamics models resulted in the development of a feasible tool, capable of increasing the fidelity of the results of virtual durability simulations. This development lays a solid foundation for future implementation in consideration of the recommendations listed in Section 6.2.

3. Virtual prototype test results are used to set the parameters of physical prototypes that are as close to production intent as possible. As a result, the relevance of each physical test that is performed is increased. The need for fewer physical prototypes and less physical testing provides significant time and cost savings, although physical test results are still necessary for the purpose of correlating the virtual test results.
4. On a broader scale, the benefit of obtaining more accurate virtual prototype design feedback at an early stage of the product development cycle provides an important competitive advantage for companies competing globally, for example in the automotive industry. The production design will be more robust and achieved with significant time and cost savings in comparison to traditional methods.

6.2 Recommendations

Recommendations for future work will be discussed with regard to the neural network development and implementation stages of the research.

6.2.1 Neural Network Development

1. The present research has provided a means of directly addressing the lack of computationally economical, high fidelity models of either vehicle dampers (i.e. shock absorbers) or elastomeric bushings, but it should be noted that some physical testing of components is still required to provide training data for the neural network development process and the process of acquiring such data needs to be put in place.

2. Opportunities exist in exploring alternative architectures for the neural network models of the different force components. The inclusion of additional parameters such as temperature, physical dimensions, and elastomeric compound composition can also be included to further develop more comprehensive models offering a wider range of application.
3. The current models were developed with training data that was acquired from single axis excitation on servo-hydraulic test equipment. Models that account for interaction of simultaneous forces and torques could be created using training data acquired during multi-axis excitation of the bushing specimen.

4. Further opportunity exists in examining the degradation of bushing reactive forces and other properties after a period of service. Including bushing degradation as an input to the neural network models would expand the range of application of the resulting models to include all possible dynamic conditions short of catastrophic bushing failure. Bushing wear is an important issue in both vehicle durability and for the durability of the road test simulator rigs themselves. Test equipment degradation can lead to equipment failure, inaccurate loading of the test specimen, or test specimen damage. Costly repairs or replacement of equipment and specimens could be minimized if bushing degradation were also accounted for in the models. This would also assist in quantifying the expected lifetime of bushings and dampers and thus enable automotive engineers to develop more durable components.

6.2.2 Neural Network Implementation

The flexibility and ease of use of MATLAB makes it a particularly useful tool for creating and integrating the neural network models with multibody dynamics models. Limitations of the ADAMS software prevented full implementation with regard to the following three aspects:

1. All six neural networks (three forces and three torques) could not be implemented simultaneously without incurring dynamic memory allocation errors.
2. The force data that was input to the ADAMS model could not be filtered prior to application to the model due to the increased number of variables and processing required and the effects on the software memory requirements.

3. Co-simulations longer than fifty seconds could be performed, but the size of the results files was so large that the files could not all be analyzed in concert.

The above limitations have been presented to the software company and are currently being investigated, but as a result, it is recommended that alternative multi-body dynamics solvers be investigated for future development. The possibility of performing the co-simulations using more powerful hardware tools is also an area that requires further study.
APPENDIX A: BUSHING DATA ACQUISITION

A.1 DATA ACQUISITION OVERVIEW

The task of acquiring data for use in training the neural network and validation was performed at Cooper Standard Automotive in Mitchell, Ontario, Canada. Using single axis testing machines described in Sections A.3.1 to A.3.4, four separate measurements were acquired from the front lower control arm pivot bushing per the orientation shown in Figure A.1.

1. Radial displacement and corresponding translation force \((z, F_z)\).
2. Axial displacement and corresponding translation force \((x, F_x)\).
3. Axial rotation and corresponding rotation torque \((\alpha_x, T_x)\).
4. Conical rotation and corresponding rotation torque \((\alpha_z, T_z)\).

Figure A.1: Orientation of the lower control arm pivot bushing with respect to the vehicle coordinate system: \(X\) – vehicle fore/aft, \(Y\) – vehicle lateral, \(Z\) – vehicle vertical.
Two separate machines were used to acquire the data: a linear actuator test machine for the translational measurements and a linear actuator machine set up with a torque arm for the rotational measurements.

Using the same approach employed in developing drive files using similar servo-hydraulic test rigs [12], four separate desired signals were replicated (axial translation and rotation and radial translation and rotation) and the resulting corresponding translational force or torque were measured. The test specimen was installed in the fixture and the following steps were performed to obtain the required data:

1. The system frequency response function was measured.
2. The initial drive signal was prepared based on the desired displacement or rotation and the frequency response function.
3. Iterations were performed until a drive file was calculated that generated a displacement or rotation of the bushing that matched that of the desired file within acceptable error criteria.
4. The corresponding force or torque was measured using the final drive file.

A.2 Desired Data Preparation

To prepare for data acquisition, bushing displacement and rotation desired data in both the radial and axial directions was required. This data would serve as the desired data for the iterations to develop suitable drive files.

The RPCPro software package was used to create time history desired displacement and rotation data for motion in the axial, radial, torsional, and conical directions. The peak amplitudes, mean signal values, and frequency ranges were chosen to ensure that the resulting neural network model would estimate bushing behaviour beyond the range of motion of the ADAMS durability model.
The rotation and translation of the bushing were measured from the current ADAMS simulation model using the same road inputs as the related work by Ferry [12] and Wood [38]. Once the magnitude and frequency content of the signals were examined, a white noise file for the bushing data acquisition was generated with peak values and a frequency range that encompassed the range of motion of the simulation data. By replicating the white noise displacements and rotations, forces and torques were collected for use as training data for the neural networks models.

Figures A.2 and A.3 are time history and frequency plots that show the ADAMS simulation data with the desired data, which was created for use in neural network development.
Figure A.2: The desired data for the axial displacement and force acquisition was created so that the peaks (blue) lie outside the range of the simulation data (black) as shown here.
Axial Displacement: Setting Criteria for Training Data

There were two available data sets for consideration from the ADAMS simulation for the radial deformation of the bushing. They correspond to the lateral (Y) and vertical (Z) bushing directions shown in Figure A.1. Due to the greater range of the motion in the lateral direction, this data set will be used to specify the range of motion of the desired data to be used in the iterations. Both data sets are shown in Figures A.4 and A.5 for illustrative purposes.

Figure A.3: The desired data (blue) for the axial displacement and force acquisition was created so that the frequency spectrum of the data follows the general shape of the simulation data (black).
Figure A.4: The desired data (green) for the radial displacement and force acquisition was created so that the peaks lie outside the range of both radial displacements of the simulation data (black, blue) as shown here.
Figure A.5: The desired data (green) for the radial displacement and force acquisition was created so that the frequency spectrum of the data follows the general shape of the radial displacement simulation data (black, blue).

Similarly, axial and conical rotational desired data sets were created using the corresponding ADAMS simulation data for the bushing as shown in Figures A.6 to A.9.
The desired data for the axial rotation and torque acquisition was created so that the peaks (blue) lie outside the range of the simulation data (black) as shown here.
Axial Rotation: Setting Criteria for Training Data

The desired data (blue) for the axial rotation and torque acquisition was created so that the frequency spectrum of the data follows the general shape of the simulation data (black).

Figure A.7: The desired data (blue) for the axial rotation and torque acquisition was created so that the frequency spectrum of the data follows the general shape of the simulation data (black).
Figure A.8: The desired data (green) for the radial rotation and torque acquisition was created so that the peaks lie outside the range of both radial displacements of the simulation data (black, blue) as shown here.
Conical Rotation: Setting Criteria for Training Data

Figure A.9: The desired data (green) for the radial rotation and torque acquisition was created so that the frequency spectrum of the data follows the general shape of the radial displacement simulation data (black, blue).

A.3 TEST SET-UP

Two separate machines were used to acquire the motion and force bushing data: a linear actuator test machine was used for the axial and radial translation measurements ($F_x, F_z$) and a linear actuator machine set up with a torque arm was used for the torsional and conical rotational measurements ($T_x, T_z$). The following descriptions and photographs describe the test set-up for each of the four configurations.
A.3.1 $F_x$, Axial Force Measurement Set-up

The bushing was installed in the fixture for measurement of the axial displacement and corresponding force as shown in Figure A.10.

Figure A.10: The bushing was mounted vertically inside the outer metal housing with the lip of the bushing sleeve clamped by a metal ring. The actuator is secured to the shaft going through the bushing's inner sleeve. Polarity of the axial displacement and force is shown with the arrow pointing up.

The LVDT full scale was 60 mm for the maximum of 10 Volts excitation. The load cell measured 15000 N at the maximum 10 Volts excitation.

A.3.2 $F_z$, Radial Force Measurement Set-up

Figure A.11 shows the lower control arm pivot bushing installed in the linear fixture for radial testing.
Figure A.11: The linear rig is set up to measure the radial displacement ($z$) and force ($F_z$). Polarity of the displacement and force is shown with the arrow pointing up.

The bushing was enclosed in the transversely mounted block fixture that was mounted on top of the load cell. The pivot bolt was inserted through the inner bushing sleeve and connected to the actuator by way of the outer brackets. The individual parts are shown disassembled in Figure A.12.
Figure A.12: The bushing, sleeve, bolt and fixture pieces for the radial measurements are shown disassembled.

The LVDT full scale was 60 mm for the maximum of 10 Volts excitation. The load cell measured 15000 N at the maximum 10 Volts excitation.

### A.3.3 $T_x$, Axial Torque Measurement Set-up

To rotate the bushing in the axial direction, it was mounted on a shaft that rotated due to the vertical movement of the actuator and a torque arm as shown in Figure A.13. The vertical downward motion of the arm corresponded to a positive rotation angle and a negative torque measurement. In this configuration, the sleeve of the bushing rotates counter clockwise relative to the shaft when viewed from the flange-end of the bushing. The angle of rotation was measured directly from an RVIT (rotational variable inductance transducer). A torsional load cell was mounted on the end of the bushing to measure the axial torque.
Figure A.13: The torsional test set-up was achieved by installing the bushing on a shaft that rotated with the vertical motion of the actuator. The assembly steps are shown in the bottom figures with the final assembly shown in the large photograph with the polarity of the angular displacement and torque measurements.

The rotational variable inductance transducer (RVIT) full scale was 204.06° for the maximum of 10 Volts excitation. The torque cell measured 2270 lb-in (256482.17 N-mm) at the maximum 10 Volts excitation.

A.3.4 $T_z$, Conical Torque Measurement Set-up

The conical angle about the lateral (y) axis and the corresponding torque were measured using the set-up shown in Figure A.14. The bushing was installed into a transversely mounted block fixture that rotated with the vertical motion of the actuator and torque arm. The inner bushing sleeve was held fixed. Both the conical angle and the torque were measured with the polarity shown in the photographs.
Figure A.14: The bushing was assembled in the conical rotation fixture as shown in the three photographs. Per the description, the conical angle was measured positive and the torque negative when the outer sleeve rotated clockwise relative to the inner sleeve.

The RVIT full scale was 204.06° for the maximum of 10 Volts excitation. The torque cell measured 2270 lb-in (256482.17 N-mm) at the maximum 10 Volts excitation.

A.4 DATA ACQUISITION

A.4.1 Measurement of $F_x$ and $F_z$

Iterations were performed for each of two lower control arm pivot bushings to replicate the desired white noise signals described in section A.2. Using Component Remote Parameter Control (CRPC) software, data was measured at a sample rate of 256 samples per second and then up-sampled to 409.6 samples per second using nCode®.
software. The sample rate of 409.6 samples per second was not available in CRPC due to limitations of the software, but was required to correspond to the existing ADAMS and Remote Parameter Control (RPC) files that were used in the virtual simulation as well as current RTS laboratory standards.

The standard RTS data sample rate of 409.6 samples per second allows for fatigue analysis of data up to one quarter of the sample rate, or 102.4Hz [7]. The sample rate then guarantees that the peak values of the data will be recorded for signals up to about 100Hz.

A.4.2 Measurement of $T_x$ and $T_z$

Iterations were performed for each of two lower control arm pivot bushings to replicate the desired white noise signals described in section A.2. Using Servotest software, Cooper Standard Automotive measured data at a sample rate of 512 samples per second and then translated the files to RPC format before re-sampling the data set at 409.6 samples per second. Again, the sample rate of 409.6 samples per second was not available in Servotest due to limitations of the software, but was required to correspond to the existing ADAMS and Remote Parameter Control (RPC) files that were used in the virtual simulation as well as current RTS laboratory standards, where fatigue analysis requires that the data be sampled at a rate at least four times the highest frequency of interest [7].

A.4.3 Iterations Results

Two bushings were used for the data acquisition. Table A.1 shows the desired data statistics and the iterations results for each of the two bushings. The accuracy of iterations results can be measured by calculating the difference in peak and mean values.
of the achieved response with respect to the desired response. Another indication of accuracy in the time domain is the magnitude of the peak and mean values of the error time history (the point-by-point difference between the desired and achieved time history data). Tables A.1 and A.2 summarize these statistics for both bushing specimens.

<table>
<thead>
<tr>
<th>AXIAL (mm)</th>
<th>MAX</th>
<th>MIN</th>
<th>MEAN</th>
</tr>
</thead>
<tbody>
<tr>
<td>DESIRED</td>
<td>2.734</td>
<td>-3.090</td>
<td>-0.075</td>
</tr>
<tr>
<td>BUSHING 01 - ACHIEVED</td>
<td>2.724</td>
<td>-3.076</td>
<td>-0.076</td>
</tr>
<tr>
<td>Error Time History (Desired-Achieved)</td>
<td>0.055</td>
<td>-0.040</td>
<td>0.001</td>
</tr>
<tr>
<td>Absolute Error of Achieved Relative to Desired</td>
<td>0.010</td>
<td>0.014</td>
<td>0.001</td>
</tr>
<tr>
<td>% Error of Achieved Relative to Desired</td>
<td>-0.4%</td>
<td>-0.5%</td>
<td>1.2%</td>
</tr>
<tr>
<td>BUSHING 02 - ACHIEVED</td>
<td>2.724</td>
<td>-3.106</td>
<td>-0.075</td>
</tr>
<tr>
<td>Error Time History (Desired-Achieved)</td>
<td>0.043</td>
<td>-0.047</td>
<td>-0.001</td>
</tr>
<tr>
<td>Absolute Error of Achieved Relative to Desired</td>
<td>0.010</td>
<td>0.015</td>
<td>0.001</td>
</tr>
<tr>
<td>% Error of Achieved Relative to Desired</td>
<td>-0.4%</td>
<td>0.5%</td>
<td>-0.9%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TORSION (deg)</th>
<th>MAX</th>
<th>MIN</th>
<th>MEAN</th>
</tr>
</thead>
<tbody>
<tr>
<td>DESIRED</td>
<td>3.842</td>
<td>-17.712</td>
<td>-8.015</td>
</tr>
<tr>
<td>BUSHING 01 - ACHIEVED</td>
<td>3.756</td>
<td>-17.774</td>
<td>-8.051</td>
</tr>
<tr>
<td>Error Time History (Desired-Achieved)</td>
<td>0.418</td>
<td>-0.443</td>
<td>0.035</td>
</tr>
<tr>
<td>Absolute Error of Achieved Relative to Desired</td>
<td>0.086</td>
<td>0.062</td>
<td>0.035</td>
</tr>
<tr>
<td>% Error of Achieved Relative to Desired</td>
<td>-2.2%</td>
<td>0.3%</td>
<td>0.4%</td>
</tr>
<tr>
<td>BUSHING 02 - ACHIEVED</td>
<td>4.025</td>
<td>-17.474</td>
<td>-7.782</td>
</tr>
<tr>
<td>Error Time History (Desired-Achieved)</td>
<td>0.197</td>
<td>-0.760</td>
<td>-0.233</td>
</tr>
<tr>
<td>Absolute Error of Achieved Relative to Desired</td>
<td>0.182</td>
<td>0.238</td>
<td>0.233</td>
</tr>
<tr>
<td>% Error of Achieved Relative to Desired</td>
<td>4.7%</td>
<td>-1.3%</td>
<td>-2.9%</td>
</tr>
</tbody>
</table>

Table A.1: Iterations results for the axial translation and rotation of bushings 01 and 02 show good correlation for the peak and mean values compared to the desired data. Error time histories also exhibit low peak and mean values.
### Table A.2

<table>
<thead>
<tr>
<th></th>
<th>MAX</th>
<th>MIN</th>
<th>MEAN</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>RADIAL (mm)</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>DESIRED</strong></td>
<td>0.539</td>
<td>-1.248</td>
<td>-0.310</td>
</tr>
<tr>
<td>BUSHING 01 - ACHIEVED</td>
<td>0.555</td>
<td>-1.260</td>
<td>-0.310</td>
</tr>
<tr>
<td>Error Time History (Desired-Achieved)</td>
<td>0.049</td>
<td>-0.052</td>
<td>0.000</td>
</tr>
<tr>
<td>Absolute Error of Achieved Relative to Desired</td>
<td>0.016</td>
<td>0.012</td>
<td>0.000</td>
</tr>
<tr>
<td>% Error of Achieved Relative to Desired</td>
<td>3.0%</td>
<td>0.9%</td>
<td>0.0%</td>
</tr>
<tr>
<td>BUSHING 02 - ACHIEVED</td>
<td>0.551</td>
<td>-1.260</td>
<td>-0.311</td>
</tr>
<tr>
<td>Error Time History (Desired-Achieved)</td>
<td>0.049</td>
<td>-0.040</td>
<td>0.001</td>
</tr>
<tr>
<td>Absolute Error of Achieved Relative to Desired</td>
<td>0.012</td>
<td>0.012</td>
<td>0.001</td>
</tr>
<tr>
<td>% Error of Achieved Relative to Desired</td>
<td>2.3%</td>
<td>0.9%</td>
<td>0.5%</td>
</tr>
</tbody>
</table>

| **CONICAL (deg)**  |      |      |      |
|**DESIRED**         | 0.928| -0.100| 0.389|
| BUSHING 01 - ACHIEVED | 0.933| -0.136| 0.385|
| Error Time History (Desired-Achieved) | 0.059| -0.052| 0.004 |
| Absolute Error of Achieved Relative to Desired | 0.005| 0.036| 0.004 |
| % Error of Achieved Relative to Desired | 0.5%| 36.3%| -1.1% |
| BUSHING 02 - ACHIEVED | 0.825| -0.219| 0.287|
| Error Time History (Desired-Achieved) | 0.156| 0.043| 0.102 |
| Absolute Error of Achieved Relative to Desired | 0.103| 0.119| 0.102 |
| % Error of Achieved Relative to Desired | -11.1%| 119.6%| -26.2% |

Iterations results for the radial translation and rotation of bushings 01 and 02 show good correlation for the peak and mean values compared to the desired data. Error time histories also exhibit low peak and mean values.

Note that while the percentage error for the conical rotation iterations for bushing 02 seems high, it is exaggerated due to the small magnitude of the desired data that was used to normalize the error. The absolute values of the peak and mean value errors are approximately 0.1°, which is a reasonable value.

The resolution of the displacement, rotation, load, and torque signals are summarized in Table A.3. Essentially, the conical rotation of the bushing achieved a maximum of 0.825°, a value less than 1% of the RVIT transducer’s full range of 204.06°. This fact may incur deficiencies in the ability to generalize the conical rotational behaviour of the bushing in the neural network modeling phase of the research due to a low signal to noise ratio for the measurement.
<table>
<thead>
<tr>
<th></th>
<th>Maximum Displacement (mm)</th>
<th>Transducer Full Scale (mm)</th>
<th>% Full Scale Utilized</th>
</tr>
</thead>
<tbody>
<tr>
<td>Axial</td>
<td>3.08</td>
<td>60</td>
<td>5.1%</td>
</tr>
<tr>
<td>Radial</td>
<td>1.26</td>
<td>60</td>
<td>2.1%</td>
</tr>
<tr>
<td>Maximum Force (N)</td>
<td>Transducer Full Scale (N)</td>
<td>% Full Scale Utilized</td>
<td></td>
</tr>
<tr>
<td>Axial</td>
<td>898.13</td>
<td>15000</td>
<td>6.0%</td>
</tr>
<tr>
<td>Radial</td>
<td>6218.3</td>
<td>15000</td>
<td>41.5%</td>
</tr>
<tr>
<td>Maximum Rotation (degrees)</td>
<td>Transducer Full Scale (degrees)</td>
<td>% Full Scale Utilized</td>
<td></td>
</tr>
<tr>
<td>Torsion</td>
<td>17.5</td>
<td>204.06</td>
<td>8.6%</td>
</tr>
<tr>
<td>Conical</td>
<td>0.825</td>
<td>204.06</td>
<td>0.4%</td>
</tr>
<tr>
<td>Maximum Torque (N-mm)</td>
<td>Transducer Full Scale (N-mm)</td>
<td>% Full Scale Utilized</td>
<td></td>
</tr>
<tr>
<td>Torsion</td>
<td>28105</td>
<td>256480</td>
<td>11.0%</td>
</tr>
<tr>
<td>Conical</td>
<td>10550</td>
<td>256480</td>
<td>4.1%</td>
</tr>
</tbody>
</table>

Table A.3: The range of the bushing data acquisition signals compared to the 10 Volt full scale of the transducers are listed. Note that the conical rotation signal occupied less than 1% of the RVIT full scale. This may limit the accuracy of the results due to the lack of signal resolution.

It was also observed that the two bushings showed little variation relative to each other with regard to replicating the same desired signals as shown in Table A.4.

Generally, peaks of the linear motion were within 0.03mm and peaks of the rotational motion were within 0.3°. Mean values came within 0.002mm and 0.3° for the linear and rotational motions respectively. From this we conclude that part-to-part repeatability is good.
<table>
<thead>
<tr>
<th>AXIAL (mm)</th>
<th>MAX</th>
<th>MIN</th>
<th>MEAN</th>
</tr>
</thead>
<tbody>
<tr>
<td>BUSHING 01</td>
<td>2.724</td>
<td>-3.076</td>
<td>-0.076</td>
</tr>
<tr>
<td>BUSHING 02</td>
<td>2.724</td>
<td>-3.106</td>
<td>-0.075</td>
</tr>
<tr>
<td>Absolute error</td>
<td>0.000</td>
<td>0.029</td>
<td>0.002</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RADIAL (mm)</th>
<th>MAX</th>
<th>MIN</th>
<th>MEAN</th>
</tr>
</thead>
<tbody>
<tr>
<td>BUSHING 01</td>
<td>0.555</td>
<td>-1.260</td>
<td>-0.310</td>
</tr>
<tr>
<td>BUSHING 02</td>
<td>0.551</td>
<td>-1.260</td>
<td>-0.311</td>
</tr>
<tr>
<td>Absolute error</td>
<td>0.004</td>
<td>0.000</td>
<td>0.001</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TORSION (deg)</th>
<th>MAX</th>
<th>MIN</th>
<th>MEAN</th>
</tr>
</thead>
<tbody>
<tr>
<td>BUSHING 01</td>
<td>3.756</td>
<td>-17.774</td>
<td>-6.051</td>
</tr>
<tr>
<td>BUSHING 02</td>
<td>4.025</td>
<td>-17.474</td>
<td>-7.782</td>
</tr>
<tr>
<td>Absolute error</td>
<td>0.268</td>
<td>0.300</td>
<td>0.268</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CONICAL (deg)</th>
<th>MAX</th>
<th>MIN</th>
<th>MEAN</th>
</tr>
</thead>
<tbody>
<tr>
<td>BUSHING 01</td>
<td>0.933</td>
<td>-0.136</td>
<td>0.385</td>
</tr>
<tr>
<td>BUSHING 02</td>
<td>0.825</td>
<td>-0.219</td>
<td>0.287</td>
</tr>
<tr>
<td>Absolute error</td>
<td>0.107</td>
<td>0.083</td>
<td>0.097</td>
</tr>
</tbody>
</table>

Table A.4: Variation in peak and mean signal values was minimal between the two bushings while attempting to replicate the same linear and rotational desired data.

The accuracy of the iterations in the frequency domain can be examined by looking at the frequency plot of the desired, achieved, and error signals in Figures A.15 to A.22.
Figure A.15: Frequency domain iteration results for replicating the axial displacement signal for bushing 01. The desired, achieved and error signal are shown. Note the good agreement between the desired and achieved signals and the resulting small error. The peak in the error signal occurs at the filter cut-off frequency of 40Hz.
Figure A.16: Frequency domain iteration results for replicating the axial displacement signal for bushing 02. The desired, achieved and error signal are shown. Once again, the error between the desired and achieved signals is small over the entire frequency range.
Figure A.17: Frequency domain iteration results for replicating the radial displacement signal for bushing 01. The desired, achieved and error signal are shown. Once again, the error between the desired and achieved signals is small over the entire frequency range.
Figure A.18: Frequency domain iteration results for replicating the radial displacement signal for bushing 02. The desired, achieved and error signal are shown. Once again, the error between the desired and achieved signals is small over the entire frequency range.
Axial Rotation Iterations Results, Bushing 01

Figure A.19: Frequency domain iteration results for replicating the axial rotation signal for bushing 01. The desired, achieved and error signal are shown. Once again, the error between the desired and achieved signals is small over the entire frequency range.
Axial Rotation Iterations Results, Bushing 02

Figure A.20: Frequency domain iteration results for replicating the axial rotation signal for bushing 02. The desired, achieved and error signal are shown. Once again, the error between the desired and achieved signals is small over the entire frequency range.
Figure A.21: Frequency domain iteration results for replicating the conical rotation signal for bushing 01. The desired, achieved and error signal are shown. Note the low signal level above about 10Hz where the amplitude of the error is of the same order of magnitude as the desired and achieved signal. The desired motion of the bushing for the conical rotation data was designed to possess a similar mean value to the ADAMS simulation data and peaks beyond the range of the ADAMS simulation data without being so large as to damage the bushing. Thus, as the magnitude of the frequency plot of the desired conical motion reduces with frequency, it meets the measurement limits of the RVIT measuring the motion.
Figure A.22: Frequency domain iteration results for replicating the conical rotation signal for bushing 02. The desired, achieved and error signal are shown. Again, as exhibited by bushing 01, the signal level is low above about 10Hz where the amplitude of the error is of the same order of magnitude as the desired and achieved signal. The limits of the RVIT measurement abilities were met.

The iterations performed at Cooper-Standard Automotive produced two datasets for each motion that required consideration for use in neural network construction. The relative magnitude of the achieved signal with respect to the ADAMS simulation data was examined as a final step to assist in the dataset selection.
A.5 **DATASET SELECTION**

The selection of the datasets for training the neural network models of the bushing forces and torques involved not only the iteration results, but also the requirement that the training data cover the full range of motion of the bushing as experienced in the ADAMS durability road simulation. Table A.5 shows the values of the achieved maximum and minimum displacement or rotation with respect to the ADAMS simulation bushing motion.

<table>
<thead>
<tr>
<th>Source</th>
<th>Name</th>
<th>Units</th>
<th>Maximum</th>
<th>Minimum</th>
<th>Use for Training?</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ADAMS</strong></td>
<td>Axial Deformation</td>
<td>mm</td>
<td>2.527</td>
<td>-2.625</td>
<td></td>
</tr>
<tr>
<td>Iterations</td>
<td>BUSHING 01</td>
<td></td>
<td>2.724</td>
<td>-3.076</td>
<td>Y</td>
</tr>
<tr>
<td>Iterations</td>
<td>BUSHING 02</td>
<td></td>
<td>2.724</td>
<td>-3.106</td>
<td>N</td>
</tr>
<tr>
<td><strong>ADAMS</strong></td>
<td>Radial Deformation</td>
<td>mm</td>
<td>0.224</td>
<td>-1.127</td>
<td></td>
</tr>
<tr>
<td>Iterations</td>
<td>BUSHING 01</td>
<td></td>
<td>0.555</td>
<td>-1.260</td>
<td>Y</td>
</tr>
<tr>
<td>Iterations</td>
<td>BUSHING 02</td>
<td></td>
<td>0.551</td>
<td>-1.260</td>
<td>N</td>
</tr>
<tr>
<td><strong>ADAMS</strong></td>
<td>Torsional Rotation</td>
<td>deg</td>
<td>3.775</td>
<td>-15.265</td>
<td></td>
</tr>
<tr>
<td>Iterations</td>
<td>BUSHING 01</td>
<td></td>
<td>3.756</td>
<td>-17.774</td>
<td>N</td>
</tr>
<tr>
<td>Iterations</td>
<td>BUSHING 02</td>
<td></td>
<td>4.025</td>
<td>-17.474</td>
<td>Y</td>
</tr>
<tr>
<td><strong>ADAMS</strong></td>
<td>Conical Rotation</td>
<td>deg</td>
<td>0.718</td>
<td>-0.158</td>
<td></td>
</tr>
<tr>
<td>Iterations</td>
<td>BUSHING 01</td>
<td></td>
<td>0.933</td>
<td>-0.136</td>
<td>N</td>
</tr>
<tr>
<td>Iterations</td>
<td>BUSHING 02</td>
<td></td>
<td>0.825</td>
<td>-0.219</td>
<td>Y</td>
</tr>
</tbody>
</table>

*Table A.5: The training data peaks must lie beyond the range of motion of the bushing experienced during the ADAMS simulation. The values in red indicate failure of the dataset to meet this criterion.*

Due to the fact that the range of the training data is of vital importance, the data for bushing sample 01 will be chosen to model the behaviour of $F_x$ and $F_z$ and the data for bushing sample 02 will be used to develop the models of $T_x$ and $T_z$. The chosen datasets are shown in Figures A.23 to A.26.
Figure A.23: The normalized axial displacement and measured normalized force data will be used to create a neural network that will model the axial behaviour of the bushing.
Figure A.24: The normalized axial rotation and measured normalized torque data will be used to create a neural network that will model the torsional behaviour of the bushing.
Figure A.25: The normalized radial displacement and measured normalized force data will be used to create a neural network that will model the radial behaviour of the bushing.
Figure A.26: The normalized conical rotation and measured normalized torque data will be used to create a neural network that will model the conical behaviour of the bushing.

A.6 DATA SUMMARY

Included in Figures A.27 to A.34 are the time history plots for the desired, achieved and error files resulting from the iterations. The error signal shown is calculated as the point-by-point difference between the desired and achieved time history signals. Following the plots, Table A.6 summarizes the complete statistics calculated from the files used for the previous analysis and neural network development.
Figure A.27: Time history iteration results for replicating the axial displacement signal for bushing 01. The desired, achieved and error signal are shown. Note the small magnitude of the error signal.
Figure A.28: Time history iteration results for replicating the axial displacement signal for bushing 02. The desired, achieved and error signal are shown. Note the small magnitude of the error signal.
Figure A.29: Time history iteration results for replicating the radial displacement signal for bushing 01. The desired, achieved and error signal are shown. Note the small magnitude of the error signal.
Figure A.30: Time history iteration results for replicating the radial displacement signal for bushing 02. The desired, achieved and error signal are shown. Note the small magnitude of the error signal.
Figure A.31: Time history iteration results for replicating the axial rotation signal for bushing 01. The desired, achieved and error signal are shown. Note the small magnitude of the error signal.
Figure A.32: Time history iteration results for replicating the axial rotation signal for bushing 02. The desired, achieved and error signal are shown. Note the small magnitude of the error signal.
Conical Rotation Iterations Results, Bushing 01

Figure A.33: Time history iteration results for replicating the conical rotation signal for bushing 01. The desired, achieved and error signal are shown. Note the small magnitude of the error signal.
Figure A.34: Time history iteration results for replicating the conical rotation signal for bushing 02. The desired, achieved and error signal are shown.
### Axial Displacement

<table>
<thead>
<tr>
<th></th>
<th>Full Scale</th>
<th>Units</th>
<th>Maximum</th>
<th>Minimum</th>
<th>Mean</th>
<th>Variance</th>
<th>Std Dev</th>
<th>RMS</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Desired</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>3.40 mm</td>
<td></td>
<td>2.7344</td>
<td>-3.0901</td>
<td>-0.0754</td>
<td>0.9142</td>
<td>0.9562</td>
<td>0.9591</td>
</tr>
<tr>
<td><strong>Bushing 01</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Achieved</td>
<td>3.08 mm</td>
<td></td>
<td>2.7242</td>
<td>-3.0762</td>
<td>-0.0763</td>
<td>0.9128</td>
<td>0.9554</td>
<td>0.9584</td>
</tr>
<tr>
<td>Error</td>
<td>0.06 mm</td>
<td></td>
<td>0.0554</td>
<td>-0.0397</td>
<td>0.0009</td>
<td>0.0001</td>
<td>0.0103</td>
<td>0.0103</td>
</tr>
<tr>
<td><strong>Bushing 02</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Achieved</td>
<td>3.11 mm</td>
<td></td>
<td>2.7241</td>
<td>-3.1055</td>
<td>-0.0748</td>
<td>0.9127</td>
<td>0.9553</td>
<td>0.9582</td>
</tr>
<tr>
<td>Error</td>
<td>0.05 mm</td>
<td></td>
<td>0.0429</td>
<td>-0.0475</td>
<td>-0.0007</td>
<td>0.0001</td>
<td>0.0103</td>
<td>0.0103</td>
</tr>
</tbody>
</table>

### Radial Displacement

<table>
<thead>
<tr>
<th></th>
<th>Full Scale</th>
<th>Units</th>
<th>Maximum</th>
<th>Minimum</th>
<th>Mean</th>
<th>Variance</th>
<th>Std Dev</th>
<th>RMS</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Desired</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.37 mm</td>
<td></td>
<td>0.5391</td>
<td>-1.2482</td>
<td>-0.3097</td>
<td>0.0893</td>
<td>0.2989</td>
<td>0.4304</td>
</tr>
<tr>
<td><strong>Bushing 01</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Achieved</td>
<td>1.26 mm</td>
<td></td>
<td>0.5551</td>
<td>-1.2598</td>
<td>-0.3097</td>
<td>0.0893</td>
<td>0.2989</td>
<td>0.4304</td>
</tr>
<tr>
<td>Error</td>
<td>0.06 mm</td>
<td></td>
<td>0.0488</td>
<td>-0.0523</td>
<td>0.0000</td>
<td>0.0001</td>
<td>0.0098</td>
<td>0.0098</td>
</tr>
<tr>
<td><strong>Bushing 02</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Achieved</td>
<td>1.26 mm</td>
<td></td>
<td>0.5514</td>
<td>-1.2598</td>
<td>-0.3111</td>
<td>0.0892</td>
<td>0.2986</td>
<td>0.4313</td>
</tr>
<tr>
<td>Error</td>
<td>0.05 mm</td>
<td></td>
<td>0.0487</td>
<td>-0.0399</td>
<td>0.0014</td>
<td>0.0001</td>
<td>0.0104</td>
<td>0.0105</td>
</tr>
</tbody>
</table>

### Axial Rotation

<table>
<thead>
<tr>
<th></th>
<th>Full Scale</th>
<th>Units</th>
<th>Maximum</th>
<th>Minimum</th>
<th>Mean</th>
<th>Variance</th>
<th>Std Dev</th>
<th>RMS</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Desired</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Bushing 01</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Error</td>
<td>0.49 Deg</td>
<td></td>
<td>0.4183</td>
<td>-0.4480</td>
<td>0.0354</td>
<td>0.0047</td>
<td>0.0886</td>
<td>0.0772</td>
</tr>
<tr>
<td><strong>Bushing 02</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Achieved</td>
<td>17.48 Deg</td>
<td></td>
<td>4.0246</td>
<td>-17.4745</td>
<td>-7.7823</td>
<td>16.5905</td>
<td>4.0731</td>
<td>8.7837</td>
</tr>
<tr>
<td>Error</td>
<td>0.84 Deg</td>
<td></td>
<td>0.1974</td>
<td>-0.7599</td>
<td>-0.2330</td>
<td>0.0050</td>
<td>0.0707</td>
<td>0.2435</td>
</tr>
</tbody>
</table>

### Conical Rotation

<table>
<thead>
<tr>
<th></th>
<th>Full Scale</th>
<th>Units</th>
<th>Maximum</th>
<th>Minimum</th>
<th>Mean</th>
<th>Variance</th>
<th>Std Dev</th>
<th>RMS</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Desired</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.02 deg</td>
<td></td>
<td>0.9281</td>
<td>-0.0998</td>
<td>0.3891</td>
<td>0.0284</td>
<td>0.1685</td>
<td>0.4240</td>
</tr>
<tr>
<td><strong>Bushing 01</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Achieved</td>
<td>0.93 deg</td>
<td></td>
<td>0.9326</td>
<td>-0.1360</td>
<td>0.3846</td>
<td>0.0282</td>
<td>0.1679</td>
<td>0.4197</td>
</tr>
<tr>
<td>Error</td>
<td>0.06 Deg</td>
<td></td>
<td>0.0587</td>
<td>-0.0520</td>
<td>0.0045</td>
<td>0.0002</td>
<td>0.0144</td>
<td>0.0151</td>
</tr>
<tr>
<td><strong>Bushing 02</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Achieved</td>
<td>0.83 deg</td>
<td></td>
<td>0.8254</td>
<td>-0.2191</td>
<td>0.2872</td>
<td>0.0275</td>
<td>0.1859</td>
<td>0.3317</td>
</tr>
<tr>
<td>Error</td>
<td>0.17 Deg</td>
<td></td>
<td>0.1555</td>
<td>0.0433</td>
<td>0.1018</td>
<td>0.0002</td>
<td>0.0148</td>
<td>0.1029</td>
</tr>
</tbody>
</table>

Table A.6: The full statistics files for the time histories shown in Figures A.27 to A.34 are listed here.
APPENDIX B: NEURAL NETWORK CONSTRUCTION EXAMPLE

To illustrate the method of constructing a suitable neural network model, a simple illustrative example is shown here for which the answer is easily verified. Data generated from a nonlinear function is modeled with a feed forward network with one hidden layer containing a single neuron. The process is repeated for the same data with an added noise component. To illustrate the concept of over fitting the data that may occur with a network employing too many hidden layer neurons, two different network structures are used to model the noisy data: one with a single hidden layer neuron, and one with five hidden layer neurons. Training with early stopping is also discussed.

B.1 NONLINEAR FUNCTION APPROXIMATION

B.1.1 Training and Test Data

The training and test data was generated from a known function so that optimal weights were known. Test data was used to check the accuracy of the network’s output.

The function used to generate the data is shown in Equation B.1.

\[ d = -\tanh(1.4x) \]  

Equation B.1
Consider the network structure shown in Figure B.1. From the network shown, the relationship between the calculated output, $y_2$, and the input, $x_1$, can be calculated as:

$$y_2 = w_2 (\tanh(w_1 x_1 + b_1)) + b_2$$  Equation B.2

Comparing Equations B.1 and B.2, it can be shown that modeling the data generated from Equation B.1 with a network of the structure shown in Figure B.1, one set of optimal weight values are $w_1^* = 1.4$ and $w_2^* = -1$. Since the hyperbolic tangent function is an odd function (symmetric through the origin), it is also possible that the calculation could converge on another set of optimal weights, $w_1^* = -1.4$ and $w_2^* = 1$. The solution depends upon the random initial values of the weights.

### B.1.2 Network Structure

The data was modeled using a one input feed forward neural network with one hidden nonlinear layer unit (hyperbolic tangent) and one linear output unit (this is known as a 1-1-1 neural network) as shown in Figure B.2.
Figure B.2: The 1-1-1 structure will be used to illustrate the process of modeling nonlinear data. The training and test data sets are plotted in Figure B.3 and listed in Table B.1.

Training and Test Data

Figure B.3: The training and test data lie along the curve $d = -\tanh(1.4x)$. 

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
### Table B.1: The training and validation data points plotted in Figure B.3 are shown here.

<table>
<thead>
<tr>
<th>$x_i$</th>
<th>$d_i$</th>
<th>$x_i$</th>
<th>$d_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1.00</td>
<td>0.8854</td>
<td>-0.95</td>
<td>0.8692</td>
</tr>
<tr>
<td>-0.90</td>
<td>0.8511</td>
<td>-0.65</td>
<td>0.7211</td>
</tr>
<tr>
<td>-0.80</td>
<td>0.8076</td>
<td>-0.35</td>
<td>0.4542</td>
</tr>
<tr>
<td>-0.70</td>
<td>0.7531</td>
<td>-0.05</td>
<td>0.0699</td>
</tr>
<tr>
<td>-0.60</td>
<td>0.6858</td>
<td>0.25</td>
<td>-0.3364</td>
</tr>
<tr>
<td>-0.50</td>
<td>0.6044</td>
<td>0.55</td>
<td>-0.6469</td>
</tr>
<tr>
<td>-0.40</td>
<td>0.5080</td>
<td>0.85</td>
<td>-0.8306</td>
</tr>
<tr>
<td>-0.30</td>
<td>0.3969</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-0.20</td>
<td>0.2729</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-0.10</td>
<td>0.1391</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.00</td>
<td>0.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.10</td>
<td>-0.1391</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.20</td>
<td>-0.2729</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.30</td>
<td>-0.3969</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.40</td>
<td>-0.5080</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.50</td>
<td>-0.6044</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.60</td>
<td>-0.6858</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.70</td>
<td>-0.7531</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.80</td>
<td>-0.8076</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.90</td>
<td>-0.8511</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.00</td>
<td>-0.8854</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### B.1.3 Neural Network Development

A spreadsheet was used to perform the iterative weight adjustments based on the gradient descent method with momentum described in Chapter 3. The initial values of the weights were randomly selected from the range of (-2.4, 2.4) per the guideline suggested in [33] for a single input hyperbolic tangent neuron. The term “epoch” refers to each time that the complete set of the training data is passed through the network and the performance measure is calculated. The calculations of the first two epochs are shown in Table B.2.
<table>
<thead>
<tr>
<th>Epoch</th>
<th>w1</th>
<th>y1</th>
<th>xi</th>
<th>di</th>
<th>net1</th>
<th>y1</th>
<th>net2</th>
<th>y2</th>
<th>e</th>
<th>delta2</th>
<th>deltaw2</th>
<th>delta1</th>
<th>total deltaw2</th>
<th>deltaw1</th>
<th>total deltaw1</th>
<th>Training MSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.5752</td>
<td>1.7572</td>
<td>-3.00</td>
<td>1.9864</td>
<td>-0.7753</td>
<td>-0.6900</td>
<td>-0.7659</td>
<td>0.7659</td>
<td>1.6513</td>
<td>0.6513</td>
<td>-0.2147</td>
<td>1.1237</td>
<td>-1.95E+00</td>
<td>-2.36E-00</td>
<td>6.21E-01</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1.5752</td>
<td>1.7572</td>
<td>-3.00</td>
<td>1.9864</td>
<td>-0.7753</td>
<td>-0.6900</td>
<td>-0.7659</td>
<td>0.7659</td>
<td>1.6513</td>
<td>0.6513</td>
<td>-0.2147</td>
<td>1.1237</td>
<td>-1.95E+00</td>
<td>-2.36E-00</td>
<td>6.21E-01</td>
<td></td>
</tr>
</tbody>
</table>

Calculations for the first two epochs for the neural network of Figure B.2 using a learning rate of 0.2 and a momentum rate of 0.5.
The learning and momentum rates were adjusted to optimize the accuracy and speed of learning without inducing system instability. Values of \( \eta = 0.2 \) and \( \alpha = 0.5 \), yielding \( \eta_{\text{eff}} = 0.4 \) were used to achieve acceptable results for numerous sets of initial weights. The calculations were repeated for one hundred epochs. The performance values, performance surfaces, and weight tracks are shown in Figures B.4 to B.6, respectively.

![Performance](image)

Figure B.4: The value of the training MSE appeared to level off after about ten epochs.

![MSE Surface](image)

Figure B.5: The convergence of the weights is shown here as the global minimum of the MSE surface.
Figure B.6: The values of the weights continued converging until about seventy epochs had been completed.

B.1.4 Results

For each set of initial weights, $w_1(I)$ and $w_2(I)$, the test set was used to calculate the network’s estimation of the output and the actual versus estimated values were plotted and a linear regression was performed. If a network accurately estimates target output values, the linear regression between the network output and target values will yield a linear regression line with a slope of one, a y-intercept of zero, and a correlation coefficient of one. The correlation coefficient, $R$, is an indication of the spread of the data points about the line of regression. The coefficient of determination, $R^2$, produced by the trendline function in Microsoft® Excel is the square of the correlation coefficient. The correlation coefficient, $R$, for the linear regression between data sets $d$ and $y$ is defined by the ratio of the covariance between the two sets of data and the product of the standard deviations of the two data sets:
\[
R = \frac{\sum_{i} (d_i - \bar{d})(y_i - \bar{y})}{\sqrt{\frac{\sum_{i} (d_i - \bar{d})^2}{N}} \sqrt{\frac{\sum_{i} (y_i - \bar{y})^2}{N}}}
\]
Equation B.3

where,

\(N\) = number of data points
\(d_i\) = target network data values
\(\bar{d}\) = mean of the target data
\(y_i\) = network output data values
\(\bar{y}\) = mean of the output data

The value of \(R^2\) varies from zero to one. A value close to one indicates that the \((d, y)\) data points lie on or very close to the line of regression.

The final values of the weights after one hundred epochs are denoted \(w_1(100)\) and \(w_2(100)\). One set of results is shown in Table B.3 and Figure B.7 for illustrative purposes.

<table>
<thead>
<tr>
<th>(x_i)</th>
<th>(d_i)</th>
<th>Predicted (y_i)</th>
<th>%error</th>
</tr>
</thead>
<tbody>
<tr>
<td>-0.95</td>
<td>0.8692</td>
<td>0.8694</td>
<td>0.02%</td>
</tr>
<tr>
<td>-0.65</td>
<td>0.7211</td>
<td>0.7211</td>
<td>-0.01%</td>
</tr>
<tr>
<td>-0.35</td>
<td>0.4542</td>
<td>0.4541</td>
<td>-0.03%</td>
</tr>
<tr>
<td>-0.05</td>
<td>0.0699</td>
<td>0.0699</td>
<td>-0.05%</td>
</tr>
<tr>
<td>0.25</td>
<td>-0.3364</td>
<td>-0.3362</td>
<td>-0.04%</td>
</tr>
<tr>
<td>0.55</td>
<td>-0.6469</td>
<td>-0.6468</td>
<td>-0.02%</td>
</tr>
<tr>
<td>0.85</td>
<td>-0.8306</td>
<td>-0.8307</td>
<td>0.01%</td>
</tr>
</tbody>
</table>

Table B.3: Test data is used to check for accuracy of the neural network model with the weights calculated after 100 epochs, \(w_1(100) = -1.3985\) and \(w_2(100) = 1.0006\).
Figure B.7: The linear regression analysis of the test data and network estimated values in Table B.3 is shown here with the trendline equation of best linear fit. The linear regression analysis of the validation data shows excellent correlation of the network results with the known validation target values.

The same procedure was repeated for three other sets of initial weight values and the results summarized in Table B.4.

<table>
<thead>
<tr>
<th>Initial Weights</th>
<th>Final Weights &amp; % Error</th>
<th>Slope</th>
<th>Intercept</th>
<th>R-Squared</th>
</tr>
</thead>
<tbody>
<tr>
<td>$w_1^{(1)}$</td>
<td>$w_2^{(1)}$</td>
<td>$w_1^{(100)}$</td>
<td>$w_2^{(100)}$</td>
<td>Error</td>
</tr>
<tr>
<td>0.7753</td>
<td>1.1783</td>
<td>-1.3985</td>
<td>0.1%</td>
<td>1.0006</td>
</tr>
<tr>
<td>-1.3003</td>
<td>0.3341</td>
<td>-1.3946</td>
<td>0.4%</td>
<td>1.0021</td>
</tr>
<tr>
<td>1.5299</td>
<td>-0.1285</td>
<td>1.3938</td>
<td>0.4%</td>
<td>-1.0025</td>
</tr>
<tr>
<td>1.6794</td>
<td>-1.7951</td>
<td>1.3956</td>
<td>0.3%</td>
<td>-1.0017</td>
</tr>
</tbody>
</table>

Table B.4: Four different initial values for $w_1^{(1)}$ and $w_2^{(1)}$ were used to determine the optimal values $(w_1^{*}, w_2^{*}) = (-1.4, 1.0), (1.4, -1.0))$. The results show the neural network of Figure B.2 estimated the coefficients within 0.2% accuracy.

In summary, the ability of a single hidden layer nonlinear neural network to model Equation B.1 shows excellent correlation.

The procedure will now be repeated for the same training data with the addition of noise.
B.2 APPROXIMATING A NONLINEAR FUNCTION WITH NOISE

B.2.1 Training, Validation, and Test Data

The function used to generate the data with the added noise component was

\[ d = -\tanh(1.4x) + \text{random}(-0.1, 0.1) \]  

Equation B.4

Training, validation, and test data is shown in Figure B.8 and listed in Table B.5.

![Training and Validation Data](image)

Figure B.8: The training, validation, and test data sets have an added noise component.
Table B.5: Training, validation, and test data that is shown plotted in Figure B.8.

B.2.2 Network Structure

The 1-1-1 neural network structure was employed to model the data shown in Figure B.8. For illustrative purposes, a 1-5-1 network was also generated (Figure B.9).

![Diagram of a neural network](image)

Figure B.9: A 1-5-1 feedforward neural network was used to model the data in Table B.5.
B.2.3 Neural Network Development

Using the neural network modeling capabilities of MATLAB, the same gradient descent with momentum training algorithm used in Section B.1 was employed to model the data with noise. Two methods were used to develop the models of the 1-1-1 and 1-5-1 feed forward neural networks. The first algorithm halted training when any one of the following three events occurred:

1. The performance goal of 0.001 for the training data was reached.
2. The minimum gradient of 1E-10 was reached, at which point the adjustments made to the weights become negligible.
3. The maximum number of epochs was reached (50,000).

The second method used early stopping to halt network training. After each epoch, the performance associated with a validation data set is calculated. The following fourth condition added to stop the training:

4. The validation performance fails to decrease for five consecutive epochs.

Following network training, test data was used to check the ability of the network to generalize the data.

B.2.4 Results

The results in Table B.6 show that the number of epochs required to obtain an equivalent level of accuracy for the 1-1-1 network fell from 8356 to 290 with the use of early stopping. Similarly, the 1-5-1 feed forward network never did reach the convergence criteria without early stopping. Including early stopping halted the training at 385 epochs.
No Early Stopping
(performance and minimum gradient only)

Early Stopping
(maximum of five non-decrease of validation)

<table>
<thead>
<tr>
<th>1 hidden-layer neuron</th>
<th>5 hidden-layer neurons</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSE</td>
<td>epochs</td>
</tr>
<tr>
<td>Training</td>
<td>0.003</td>
</tr>
<tr>
<td>Test</td>
<td>0.002</td>
</tr>
</tbody>
</table>

Table B.6: The results for network development for the data with noise, shows the benefits of early stopping using a validation dataset.

The results of the network models are shown in Figures B.10 and B.11.

Figure B.10: The 1-1 feed forward network generalizes the data (left) with the ability to estimate the test data within 2% of the target data with an $R^2$ value of 99.5% as shown by the linear regression (right).

Figure B.11: The 5-1 feedforward network generalizes the data (left) with the ability to estimate the test data within 2% of the target data with an $R^2$ value of 99.5% as shown by the linear regression (right).
As shown in the plots of the network results, the increase in number of hidden layer neurons provides a model with a higher order, or more degrees of freedom. In the case of added noise, the extra degrees of freedom of the larger neural network may introduce over fitting, or modeling of the noise component. Since the original function from which the data is generated is not generally known, the graphic results must be carefully considered along with the statistical results when selecting an appropriate network structure.

Unless early stopping is used with larger network structures, the higher degree of freedom models will approach a point-to-point fit rather than a generalization (Figure B.12). As a result, the network is no longer able to generalize the behaviour.

Figure B.12: The plot illustrates the over fitting of the training data using a network with too many hidden layer neurons and no early stopping.
REFERENCES


VITA AUCTORIS

Jennifer Johrendt was born in 1970 in Barrie, Ontario. She attended Cairine Wilson Secondary School in Orleans, Ontario, and graduated in 1988. From there, she went on to study at Queen’s University in Kingston, Ontario where she obtained a Bachelor of Science in Mathematics and Engineering Applied Mechanics option with honours in 1992. She continued studying at Queen’s University and obtained her Master’s of Science in Mathematics and Engineering in 1994. She is currently a candidate for a Doctoral degree in Mechanical Engineering at the University of Windsor and hopes to graduate in Fall 2005.