2-9-2010

Optimal passive nonlinear damper design methodology for road race application

Ryan Robinson

Ryan W. Robinson
Candidate

Mechanical Engineering
Department

This thesis is approved, and it is acceptable in quality and form for publication:

Approved by the Thesis Committee:

[illegible], Chairperson

[illegible]

[illegible]

OPTIMAL PASSIVE NONLINEAR DAMPER DESIGN METHODOLOGY FOR ROAD RACE APPLICATION

BY

Ryan W. Robinson

B.S. Mechanical Engineering, University of New Mexico, 2007

THESIS

Submitted in Partial Fulfillment of the
Requirements for the Degree of

Master of Science

Mechanical Engineering

The University of New Mexico
Albuquerque, New Mexico

December, 2009

OPTIMAL PASSIVE NONLINEAR DAMPER DESIGN METHODOLOGY FOR ROAD RACE APPLICATION

BY

Ryan W. Robinson

B.S. Mechanical Engineering, University of New Mexico, 2007

ABSTRACT OF THESIS

Submitted in Partial Fulfillment of the
Requirements for the Degree of

Master of Science

Mechanical Engineering

The University of New Mexico
Albuquerque, New Mexico

December, 2009

OPTIMAL PASSIVE NONLINEAR DAMPER DESIGN METHODOLOGY FOR ROAD RACE APPLICATION

by

Ryan W. Robinson

B.S. Mechanical Engineering, University of New Mexico, 2007
M.S., Mechanical Engineering, University of New Mexico, 2009

ABSTRACT

Methodology to develop a baseline passive nonlinear damper and inerter for road race application to maximize vehicle lateral acceleration is developed here using optimization techniques. The method includes use of equations-of-motion for suspension models, assembly of equations in a computer model for simulation, identification of the objective function which maximizes lateral acceleration based on tire data, optimization of the objective function by varying damping and inertance, identification of mode shapes and root locus analysis.

No closed form solution exists for an optimal linear or nonlinear damper which maximizes lateral acceleration. Consequently, numerical analysis is required to solve the problem. Several suspension models ranging from a quarter-suspension to a full-car suspension are examined to determine whether the simpler models are reasonable substitutes for higher order models. Fixed parameters required for the Simulink simulation, and thus the optimal damper, are from the 2007 UNM FSAE vehicle. Inertial forces from a data acquisition system are used to provide maneuver/handling input to the models. Tire data is analyzed using optimization to find the optimal vertical loading which maximizes lateral tire force. This optimization shows lateral force is maximized when inner and outer tire loads are equal, thus the objective function to be minimized is load transfer. Matlab Optimization Toolbox is then used to optimize the objective function by varying the linear/nonlinear damper rates followed by a separate optimization of the inerter. Root locus and mode shape identification are used to understand the results of the optimized system.

The analysis indicates that a nonlinear damper attenuates lateral load transfer better than a linear design. It was also observed that the simpler models' optimizations do not agree well with the full-car model due to over-simplified inputs and assumptions. The optimal inerter further increased lateral acceleration by decreasing load transfer.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS AND NOMENCLATURE

$\|\|\Delta W\|\|$	Weight Transfer Norm
α	Slip Angle
α_f	Front Slip Angle
α_r	Rear Slip Angle
b	Inertance
b_{f0}	Initial Front Inertance
b_{fOpt}	Optimized Front Inertance
b_f	Front Inertance
b_{r0}	Initial Rear Inertance
b_{rOpt}	Optimized Rear Inertance
b_r	Rear Inertance
C	Damping Coefficient
$\mathbf{C}$	Damping Matrix
C_0	Initial Damping Coefficient
C_1	Low-Speed damping
C_{1f0}	Initial Low-Speed Front Damping
C_{1fOpt}	Optimized Low-Speed Front Damping
C_{1r0}	Initial Low-Speed Rear Damping
C_{1rOpt}	Optimized Low-Speed Rear Damping
C_2	High-Speed Damping
C_{2f0}	Initial High-Speed Front Damping

C_{2fOpt}	Optimized High-Speed Front Damping
C_{2r0}	Initial High-Speed Rear Damping
C_{2rOpt}	Optimized High-Speed Rear Damping
CAD	Computer Aided Design
C_{f0}	Initial Front Damping Coefficient
CG	Center of Gravity
C_f	Front Damping Rate
C_{opt}	Optimized Damping Coefficient
C_{r0}	Initial Rear Damping Coefficient
C_r	Rear Damping Rate
C_{wfOpt}	Optimized Front Installed Damping Coefficient
C_{wf0}	Installed Initial Front Damping Coefficient
C_{wrOpt}	Optimized Rear Installed Damping Coefficient
C_{wr0}	Installed Initial Rear Damping Coefficient
DOF	Degrees-of-Freedom
ΔW_f	Front Vehicle Weight Transfer
ΔW_r	Rear Vehicle Weight Transfer
EOM	Equations of motion
$\mathbf{f}$	System Forcing vector
FSAE	Formula Society of Automotive Engineers
F_y	Lateral Load
F_{yfi}	Front-Inner Lateral Tire Load
F_{yfo}	Front-Outer Lateral Tire Load

F_{yl}	Combined Lateral Left Tires Load
F_{yr}	Combined Lateral Right Tires Load
F_{yri}	Rear-Inner Lateral Tire Load
F_{yro}	Rear-Outer Lateral Tire Load
F_z	Vertical Load
g	Gravity
γ	Camber Angle
g's	Acceleration as Related to Gravity
I_{xx}	Roll Moment of Inertia
I_{yy}	Pitch Moment of Inertia
$\mathbf{K}$	Stiffness Matrix
k_1	Quarter-Suspension Spring Stiffness
k_2	Quarter-Suspension Tire Stiffness
k_f	Front Wheel Rate
k_r	Rear Wheel Rate
L	Half-track
L_f	Front Lever arm to CG
L_r	Rear Lever arm to CG
$\mathbf{M}$	Mass Matrix
m_1	Quarter-Suspension Sprung Mass
m_2	Quarter-Suspension Unsprung Mass
M_b	Sprung Mass
m_f	Front Unsprung Corner Mass

m_r	Rear Unsprung Corner Mass
M_x	Roll Torque
M_y	Pitch Torque
n	Iteration
n_y	Lateral Acceleration (g's)
θ	Chassis Pitch
Ψ	Chassis Roll
RMS	Root Mean Square
t_f	Front Track Width
t_i	Time Index
t_r	Rear Track Width
UNM	University of New Mexico
V_t	Transition Velocity
V_{tf}	Front Transition Velocity
V_{tfOpt}	Optimized Front Transition Velocity
V_{tr}	Rear Transition Velocity
V_{trOpt}	Optimized Rear Transition Velocity
W	Vehicle Weight
ω	Natural Frequency
$\mathbf{x}$	State-Vector
x_1	Quarter-Suspension Sprung Mass Coordinate
x_2	Quarter-Suspension Unsprung Mass Coordinate
Z	Chassis Heave

z_1	4DOF Half-Car Unsprung Mass Coordinate
z_2	4DOF Half-Car Unsprung Mass Coordinate
z_{lf}	Left Front Wheel Vertical Motion
z_{lr}	Left Rear Wheel Vertical Motion
z_{rf}	Right Front Wheel Vertical Motion
z_{rr}	Right Rear Wheel Vertical Motion

1 INTRODUCTION

Shock absorbers by nature are nonlinear transient devices used to increase vehicle handling performance. The force from the damper is a function of the damper shaft velocity and when plotted, is known as a force-velocity curve. The nonlinearity of the passive damper should be engineered in such a way that it maximizes performance of a road race vehicle in terms of lateral acceleration capabilities. After extensive literature research, methodology necessary to achieve this nonlinear curve for a concept car is nowhere to be found. Milliken [1] explains that damper choice, sizing and tuning have been historically determined by reputation, adjustability, packaging requirements, etc. Further, "the subject of transient wheel loads is one which has yet to be fully explored." The methodology to build a baseline nonlinear damper curve to maximize lateral-load carrying capabilities for a theoretical car is proposed here; from this point a damper can be fine tuned on the vehicle during testing.

A tradeoff exists between vehicle responsiveness (handling) and driver comfort. A soft suspension with a low natural frequency is more compliant and transfers less force from the ground to the driver. However, a soft suspension allows excessive variation in the tire contact patch, reducing mechanical grip. A firm suspension increases responsiveness by increasing the natural frequency and maintaining the tire contact patch. However, this means that more force from the ground is transferred to the driver which results in a less comfortable ride.

1.1 Tire Behavior

Tire lateral force is modeled as a function of vertical load, slip angle and camber for a given surface and operating at nominal temperature. The three specific parameters mentioned will be discussed throughout the document. Different tire brands and models have different lateral force capabilities but are still a function of vertical load, slip angle and camber. Due to the fact that tires behave like nonlinear softening springs, transferring vertical load from one tire to another results in a net loss of lateral force. A lateral load transfer example is shown in Figure 1.1. The vehicle has a center-of-gravity (CG) which is above ground, thus turning causes lateral acceleration which in turn causes the tire load to be transferred from the inside tires to the outside tires.

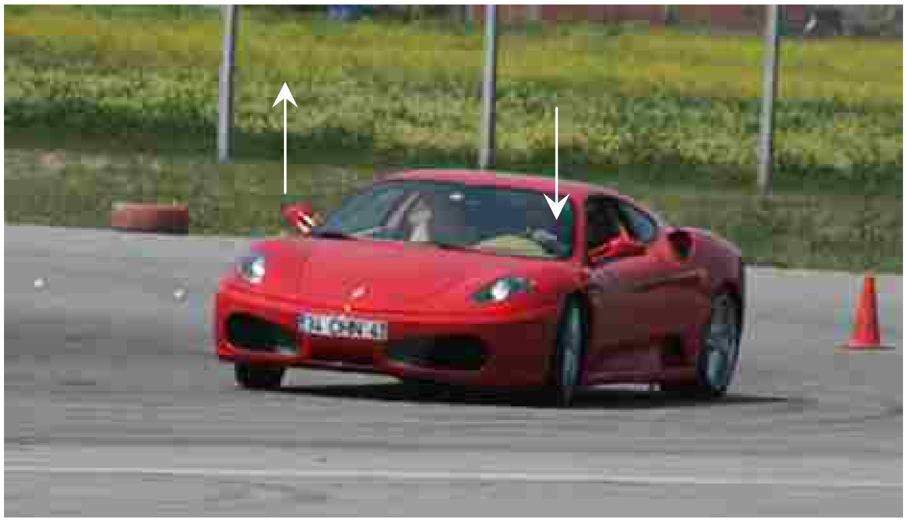
Figure 1.1 – Example of Lateral Load Transfer

Due to the nonlinearity of the tires, it is generally accepted that equally loaded tires create the largest lateral acceleration for a vehicle, or when normalized by gravity, g-force or simply "g's." The equations which are used to model tire forces are known as Pacejka formulas [1]. For the lateral case, the variables required to find tire force are vertical load, slip angle and camber angle.

The vertical load is simply the vertical reaction force from the ground. This is generally found from a vehicle data acquisition system or a simulation, the latter is used here.

When a rolling tire is influenced with lateral force an apparent path is made with an angle relative to the wheel plane as depicted in Figure 1.2. This angle is referred to as the slip angle. Conversely, the slip angle can be used to determine the lateral force in conjunction with the aforementioned camber angle and the vertical load. The slip angle can be measured by specific sensors or back-calculated from a data acquisition system which measures vehicle lateral, longitudinal and vertical accelerations as well as vehicle yaw and pitch.

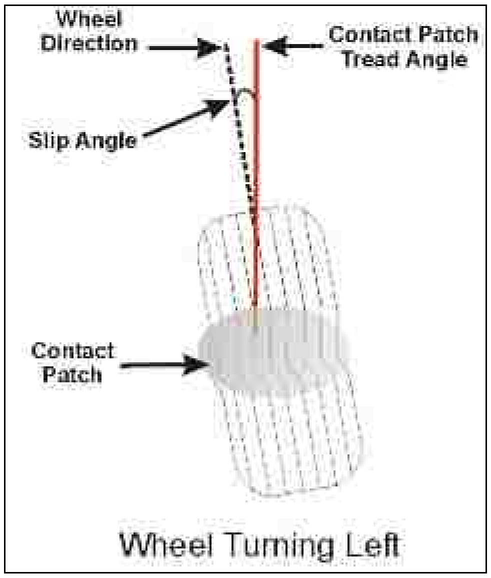

Figure 1.2 – Tire Slip Angle

Camber angle is the angle at which a tire leans relative to the vertical as shown in Figure 1.3. The camber angle depicted is positive as defined by SAE Tire Axis System [1]. As the vehicle rolls when entering a corner, the tire with negative camber rolls to a vertical orientation which increases the tire contact patch and therefore, increases the tire lateral load.

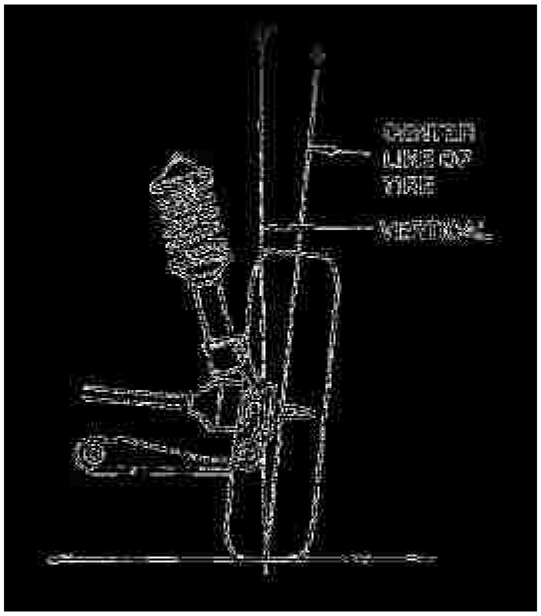

Figure 1.3 – Tire Camber Angle Defined

To show how weight transfer adversely affects lateral force, consider two tires with equal vertical loads of 1,350 lbs. The lateral load carrying capability of these tires can be found from Figure 1.4 [1], and using a reasonable approximation that the inner and outer slip angles are the same. The maximum combined lateral force these tires would produce is the sum of the two tires lateral force at the slip angle. From the figure, each tire would produce lateral force of 1,462 lbs, or a total of 2,924 lbs. If 450 lbs of load transfer were to be taken into account as though the vehicle were in a steady-state corner, the vertical load on the inner tire would be reduced to 900lbs and the vertical load on the outer tire would increase to 1800lbs. From Figure 1.1, the maximum combined lateral force at these loads would be 1,771 lbs from the outer tire, plus 1,012 lbs from the inner tire for a total of 2,783 lbs. Comparing this value to the equally loaded tires case, 141 lbs of lateral load carrying capability is lost simply due to the nonlinearity of the tires.

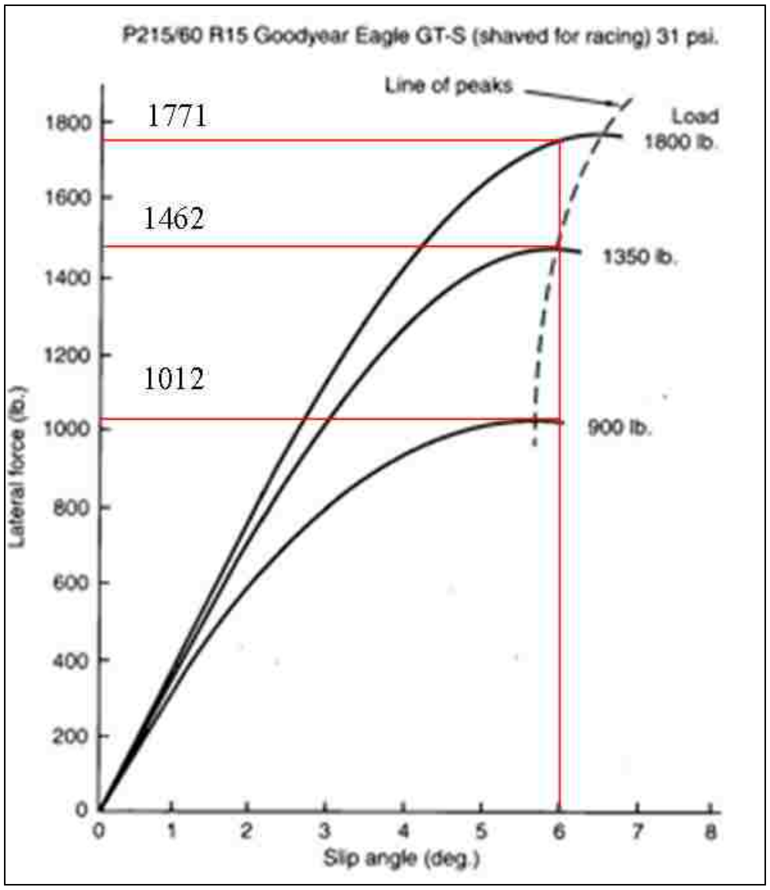

Figure 1.4 – Lateral Tire Force vs. Slip Angle (typical)

1.2 Passive Nonlinear Damper Parameters

The purpose of this paper is to show that the damper can be engineered to reduce lateral load transfer since road racing is comprised mainly of transient cornering maneuvers.

The method to find the nonlinear damper force-velocity curve which maximizes lateral acceleration is developed here, specifically the low-speed damping (C_1), high-speed damping (C_2) and transition velocity (V_t).

The compression (positive velocity) or rebound (negative velocity) components of the nonlinear force-velocity curve are generally broken down into two parts, low-speed damping (C_1) and high-speed damping (C_2). The transition from low-speed to high-speed occurs at the transition velocity (V_t); an example of a typical nonlinear damper force-velocity curve can be seen in Figure 1.5. The force-velocity curve developed in this document will be strictly odd-symmetric about the origin (i.e. C_1 applies to the low-speed rebound damping, V_t applies to the rebound transition velocity and C_2 applies to the high-speed rebound damping).

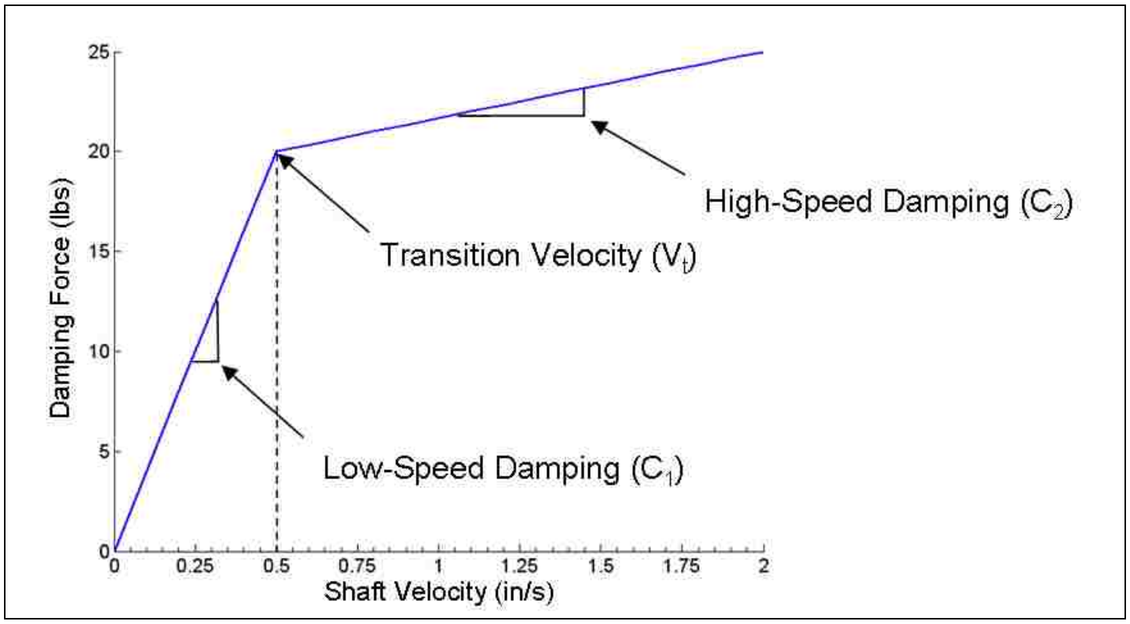

Figure 1.5 – Nonlinear Damper Force-Velocity Curve

It can be shown that the body motions of a vehicle are much slower velocities (or lower frequencies) than the road input. Thus, low-speed damping (C_1) is designed for controlling body motion while the high-speed damping (C_2) is designed for controlling road input. The low-speed segment (C_1) generally uses a high damping rate in order to make the chassis take a set quickly when entering a corner, thus increasing the responsiveness of the vehicle at turn-in; the chassis may not even enter steady-state in a typical road race situation. Conversely, if a low rate is used, the chassis response is slow meaning the vehicle responds slowly to the steering input. The high-speed segment (C_2) generally uses a low damping rate so that the road input has little effect on the car as a whole and allows the tire to maintain its contact patch. If the damping is too low, the tire cannot follow the contour of the road properly as the chassis would maintain attitude and the wheel would bounce on the road, limiting the mechanical grip the tire can provide. Conversely, if damping is too high, a bumpy surface transfers excessive force to the chassis disrupting the chassis attitude, launching the wheel and providing no tire traction at all. Finally, determining the transition velocity where low-speed ends and high-speed begins is typically found by examining race car data. However, the transition velocity was determined by optimization in this methodology.

1.3 Inerter Formulation

Recent developments have introduced a device known as an inerter (also known as a mass-damper), which adds additional passive control to a suspension [2]. Just as a spring's force is proportional to its displacement and a damper's force is proportional to velocity, the inerter's output force is proportional to acceleration. The proportionality, "b," is known as the inertance and can be adjusted by changing the internal mechanics of the inerter. The methodology used to optimize the passive nonlinear damper was extended to optimize the inerter as well. Optimization of an inerter and damper together is also a new phenomenon.

1.4 Methodology and Design

This passive damper and inerter design was accomplished by incorporating Matlab/Simulink to develop a usable vehicle suspension model and employ control system techniques for analysis. Suspension parameters were obtained from the University of New Mexico Formula Society of Automotive Engineers (UNM FSAE) team as the methodology developed here will hopefully be used as a tool for future designs.

Key components of the study include:

- System formulation
- Objective function identification – Tire load optimization
- Damper compression curve optimization
- Inerter optimization
- Mode shape identification and root locus

While natural frequency determines the responsiveness of the total vehicle, weight transfer is a key component in determining lateral acceleration as mentioned previously and demonstrated in section 4.5.1. Damper rebound can also increase performance by lowering the center of gravity, and thus decreasing weight transfer further. Rebound is the region of negative velocities on a force-velocity curve.

However, a compression curve was analyzed alone as designing the curve was sufficiently difficult. By analyzing the compression curve alone, the rebound curve is forced to have the same slope as the compression curve. In other words, the force velocity-curve constructed here is odd-symmetric.

1.5 Outline

Section 2 of this document summarizes previous work from various sources showing the methodology developed here is original. Several methods for analyzing damper parameters on vehicle behavior has been previously performed, however, none of the literature describes the methodology to design a damper for a road race vehicle.

Section 3 describes the formulation of the methodology to design an optimal passive nonlinear damper for a road race vehicle based on vehicle parameters, inertial input and road noise. The methodology also extends to optimizing a linear inerter. The parameters to be optimized are the low-speed damping, high-speed damping and transition velocity for the nonlinear damper and inertance for the inerter. The procedure for this methodology is also developed in section 3.

The analysis of increasingly complex vehicle suspension models is accomplished in section 4. This includes assumptions necessary for these models and is listed in section 4.1 which include necessary simplifications or approximations. The equations of motion which define the models are provided in section 4.2 so that a simulation or computer model could be constructed as shown in section 4.4. Empirical data from a vehicle data acquisition system was used in the analysis to provide inertial input to the suspension models as described in section 4.3. The data acquisition system collected lateral/longitudinal/vertical vehicle accelerations as a function of time which created a true course on which to “drive” the virtual car. The optimization algorithm used is explained in section 4.5. Finally, mode identification is performed in section 4.6 along with a method for creating a root locus plot for each of the models. These in conjunction are used to determine how the optimization has changed the suspension behavior.

Section 5 contains the optimization of a linear damper for the suspension models as well as the creation of the root locus figures. The results of the nonlinear damper optimization for the full-car model are presented separately from the linear damper and root locus analysis. The inerter was optimized separately from the damper and the results of the optimization are presented separately.

1.6 Introduction Chapter Recapitulation

Tire lateral force behaves nonlinearly and will produce maximum lateral force when the inner tire of a vehicle has equal vertical loading to the outer tire. This implies that weight transferred from the inner tire to the outer tire should be minimized. A road race car damper should be engineered to minimize the lateral load transfer of a vehicle to maximize lateral acceleration. The nonlinear damper is composed of low-speed and high-speed segments which control chassis motions and wheel motions, respectively.

The inerter is a passive mechanical device which produces force proportional to the relative acceleration applied to it. The inerter can also be optimized to minimize lateral load transfer of a vehicle to increase vehicle lateral acceleration.

The methodology to develop a nonlinear damper is nowhere to be found or not available to the public which the background section addresses by examining previous work on constructing damper force-velocity curves to increase vehicle performance.

2 BACKGROUND

2.1 Previous Work

As discussed previously, a suspension damper should be designed to increase lateral load capabilities of a road race vehicle, however the methodology to design an optimal nonlinear damper for a full-car either does not exist or is not available to the public.

A damper design proposed by Lacroix, et al. attempts to minimize tire load fluctuations [3]. In the design, a linear damper and quarter suspension is first modeled in Matlab/Simulink. Parametric studies are performed on the model including tire spring stiffness, unsprung mass, tire stiffness, and damping ratio to reduce the tire load fluctuation. Plots of tire load fluctuation versus exciting frequencies are used to choose the “best” parameter studied. No attempt was made at minimizing the objective function here. Finally, a linear damper, 7 Degrees-of-Freedom (DOF) car was modeled which used experimental lateral/longitudinal acceleration data from an FSAE vehicle as input. Another parametric study was made by choosing four damping ratios, plotting them as load fluctuation versus frequency and then selecting the “best” result again. Based on the results of the parametric damping ratios curves, a nonlinear force-velocity curve was created and shown to improve tire load fluctuations.

Lacroix’s design does not have any evidence to suggest that minimal tire load fluctuation maximizes lateral acceleration of the vehicle. There is also no formal optimization which shows the objective function was minimized in the design, only an “educated guess” method to create a damper force-velocity curve. A damping ratio was referred to in the 7DOF case, although it was never shown how it was defined. Section 5.1.3 of this document shows that the damping ratios being referred to by Lacroix are only applicable to a quarter suspension model as several modes exist for the 7DOF model. It was never shown, but assumed, that the same damping would be used at each suspension corner. Furthermore, no methodology is presented for determining damping for a generic car, only for the specific car studied in the paper.

Fukushima [4] of Nissan's Vehicle Research Laboratory presents empirical data on spring and damper forces for a variety of characteristics, including handling. As the requirements for the various conditions were different, the range of the damper shaft velocity and range of piston stroke for each condition were plotted. It was found that piston velocities for a rough road or a large bump were identical. However, little damping is required on the single bump to increase driver comfort, while significant damping is required for a rough road to increase lateral load capabilities. Since the stroke lengths for these conditions varied considerably, it was proposed that a stroke-sensitive damping feature be used to reduce harshness.

While Fukushima's paper is not directly related to racing, it is another example of maintaining tire-ground contact in the sense of a bumpy road. This has little implication on racing where a transient corner may be bumpy but also has weight transfer which alters the vertical load on the tires. However, the strategy of isolating different road conditions to construct a condition dependent damper is similar to that used here.

Yet another example of minimizing tire load fluctuations is presented by Sugasawa [15] which reports that another group at Nissan performed a linear analysis to determine an estimate of an ideal damping ratio for various conditions. The measure used for road-holding is the root-mean-square (RMS) average of the wheel motion relative to the road. It was decided that this RMS average should be minimized, such that the wheel would follow the road and would thus provide optimal road-holding. It is shown that if the wheel motion matches the road then the force between the wheel and road is theoretically the static load.

The idea of minimizing tire load fluctuations would typically be the job of the high-speed damping. Low-speed damping is neglected for controlling chassis attitude to maintain the vertical load on the tire. When a vehicle enters a corner, the vertical load on the inner tires must be transferred to the outer tires. This is also ignored and is contradictory to the RMS average idea that static loads are maintained if tire load fluctuation is minimized since the vertical loads on the tires would not be the same as static if the vehicle is cornering.

To verify that rough roads do, in fact, require consideration, Rill [6] of Daimler-Benz explores the matter. The analysis used a comprehensive nonlinear vehicle model to simulate a steady-state, constant radius skid pad test in which varying surface roughnesses were simulated. Plots of tire lateral load loss with vertical load variation (due to road roughness) are developed from tire data which shows that the road surface greatly affects steering wheel angle, as the model controller tries to keep the vehicle on the circle, and lateral acceleration. The effect of reducing damping ratio during the skid-pad event was also studied and shows increased steering and decreased lateral acceleration.

Els, et. al., "evaluate the feasibility of using gradient based approximation methods for the optimization of spring and damper characteristics of an off-road vehicle, for both ride comfort and handling" [7]. The vehicle in this evaluation uses a semi-active suspension modeled in ADAMS. Here the attempt was to optimize handling based on a simple simulated double lane change by minimizing body roll. The model used lateral, but not longitudinal, damper rate symmetry.

It was not shown that body roll is principal to maximizing lateral acceleration for either off-road or on-road vehicles. Further, the goal of the analysis is not to provide methodology for designing an optimal damper for a generic car, so no quantitative values for the optimized damper are presented.

Penske Technology Group develops optimal damping characteristics for vehicles on its test rig to improve their performance [8]. Basic car models are generated using software until the simulations reflect the test data. Software is then used to optimize parameters for best track performance. The use of these tools allows for reduced development time and increased offline testing which makes better use of test time on the rig and streamlines development.

This is an example of a company using proprietary methodology to optimize shock absorbers using state-of-the-art technology. However, because this is proprietary, the methodology was not made available to the public.

Smith and Wang propose a preliminary investigation to determine an inerter's benefit on a passive suspension system using various suspension configurations [9]. The inerter and linear damper were optimized for RMS body vertical acceleration, RMS dynamic tire load and dynamic load carrying while parametrically changing the suspension stiffness for each configuration studied. Performance improvements were shown in a quarter-car suspension as well as a full-car in the areas of ride quality, tire loads and dynamic load carrying. The inputs for the models were simple road disturbances modeled in the simulation. This analysis was not directed at road racing and does not provide a method for designing a suspension for a generic car. The objective of the paper was to show that the inerter concept would provide improvements to the three vehicle responses studied but not necessarily to lateral load carrying capabilities of the vehicle.

2.2 Summary of Previous Work

The previous work in damper design has largely used the idea that tire load fluctuations are the objective function to be minimized which increases vehicle handling performance. There was no proof, however, to show that lateral acceleration increases with minimal tire load fluctuation. The methodology presented in this document shows that lateral acceleration increases by decreasing lateral load transfer. The previous work focuses on either road input or inertial input and not the combination of the two. Since the purpose of a nonlinear damper is to control both inertial and road input, it was decided that the methodology to design the optimal force-velocity curve should use both of these inputs in an attempt to simulate a more realistic vehicle driving a road course.

The methodology necessary to develop an optimal passive nonlinear damper to maximize vehicle lateral acceleration is not developed in any previous work unless it is unavailable to the public. The methodology to maximize lateral acceleration by optimizing a nonlinear damper is presented in the next chapter.

3 METHODOLOGY

The previous section has shown that methodology for optimizing a nonlinear damper is not available public use. A method for first optimizing generic system parameters is presented here as a means for maximizing or minimizing the output of a plant. Secondly, the vehicle suspension system optimization procedure is presented representing a specific system using the same optimization method of the generic system.

3.1 Generic System Optimization

Given a system with several inputs and/or outputs, an optimization routine takes the desired output to be minimized or maximized (the objective function) and adjusts the chosen plant parameters accordingly. Typically only a small number of parameters are optimized at one time due to the large complexity on the optimizer. Figure 3.1 is a generic example of how the process progresses.

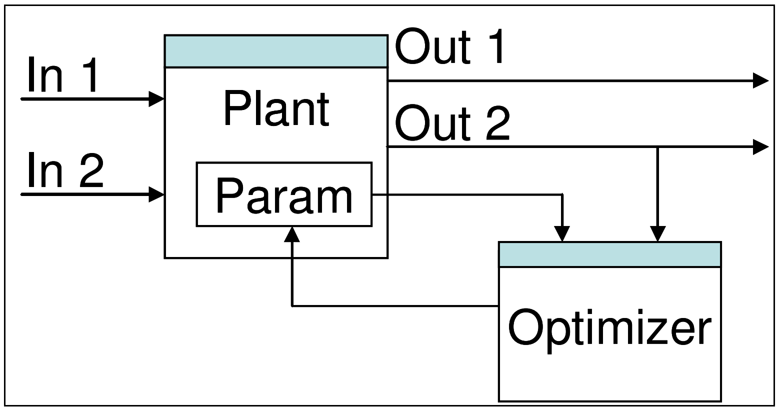

Figure 3.1 – Generic System with Optimization

It is not implied, however, that all plant outputs improve or become optimal themselves. They may in fact worsen to a point where further consideration is required for implementing constraints or using multi-objective criteria to solve the problem. The vertical motion of the optimized 4DOF system in section 5.1.2 would be unacceptable for an actual road race car. This multi-objective problem can be solved by the weighted sum method amongst a variety of different algorithms. Attempting to optimize several outputs is a multi-objective problem which is beyond the scope of this study.

The equations-of-motion (EOMs) of the system are solved independently of the optimization and the optimization routine itself iterates on the system by modifying the chosen parameters until convergence is achieved. This iterative process is depicted in Figure 3.2.

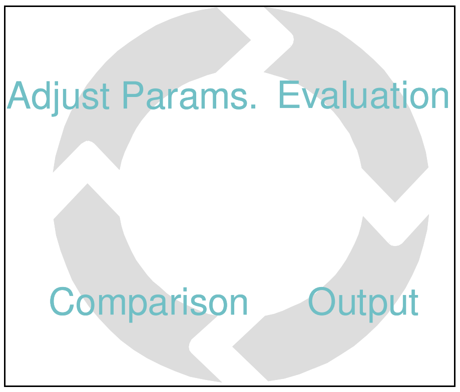

Figure 3.2 – Optimization Iterations

3.2 Vehicle Suspension Optimization Procedure

The following general steps were taken to build a model of a suspension system and optimize its lateral load carrying capabilities:

1. Four suspension system configurations were analyzed by developing computer models for each vehicle suspension from the EOMs. The systems include heave (the vertical motion of the chassis) for a 2DOF quarter-suspension model, heave

and roll for a 2DOF roll model, heave, roll and wheel motion for a 4DOF model, and heave, pitch, roll and wheel motion for a full 7DOF model.

2. A block diagram for each system was constructed in Simulink to solve the EOMs. The inertial forces used for input were adapted from a vehicle data acquisition system which recorded lateral/longitudinal/vertical acceleration as a function of time. All systems include road input at the wheel/tire and inertial forces. Additionally, the model contains a block which only allowed the load on the tire to be positive (i.e. compression only) as physically, the wheel would simply lift off of the ground if force was less than zero.
3. The system parameters were those of the 2007 UNM FSAE. Successive UNM FSAE vehicles used similar designs.
4. From the Pacejka formulas, tire data was examined to determine whether minimizing lateral load transfer would be a relevant objective function for optimizing lateral acceleration. The lateral acceleration of the vehicle was first written as a function of the four tires lateral forces. The function was then optimized to determine the load transfer which would maximize lateral acceleration. Attempting to directly optimize lateral acceleration by varying damping is not possible due to the fact that the inertial inputs are the same each iteration of the optimization. The pre-recorded vehicle test data used for input does not have feedback from the simulation to update the vehicle inertial forces.
5. Matlab contains an optimization toolbox which was used in the optimization of the inerter and, initially, a linear damper. The number of variables can become overwhelming for optimization and must be taken in steps. By first modeling a

linear damper, an educated guess was made for initial damping value in the vicinity of the optimal nonlinear damper. After the nonlinear damper was optimized, an inerter was optimized. The damper was preferentially optimized first before the inerter since information is available for an FSAE damper and an inerter is not available for FSAE vehicles.

6. Mode identification was developed to understand the suspension system behavior and in understanding the optimized results. This is further developed in section 4.6. Using a root locus plot verifies that the mode identification was done correctly as well as gain further understanding of the vehicle behavior and how the optimization has arrived at its result.

Using the above procedure will maximize the lateral acceleration of a vehicle by optimizing a nonlinear damper to minimize lateral load transfer. It will also provide a method to understand how the optimization has changed the suspension system to accomplish its objective. Section 4 is the implementation of this procedure.

3.3 Methodology Chapter Recapitulation

To optimize a nonlinear damper, the EOMs must first be developed to create a computer simulation since a closed form solution is not possible. The vehicle parameters and inputs to drive the model should come from the vehicle being modeled. An objective function was examined to show that it maximizes lateral acceleration, in this case lateral load transfer was used. An appropriate optimization routine was chosen to optimize the damper. Mode identification is used in conjunction with root locus to understand the results of the optimization. Section 4 is the implementation of this procedure.

4 ANALYSIS

Using the procedure developed in section 3, this section presents the derivation and setup of the suspension system, obtainment of the objective function which maximizes vehicle lateral acceleration and the identification of mode shapes for the root locus analysis which provided information on the relative motion of the optimized suspension model. The algorithm used by the optimization is also presented here.

A closed form solution for an optimal damper is not possible, thus numerical analyses must be employed to find an optimal force-velocity curve which maximizes lateral acceleration. Matlab with Simulink and the Optimization Toolbox was used to calculate the optimal damper force-velocity curve parameters: low-speed damping (C_1), high-speed damping (C_2) and the transition velocity (V_t).

The models leading up to the full-car are strictly linear since they are mostly used as step-by-step learning as to how the analysis must progress. Assumptions listed in section 4.1 are made to simplify the models for simulation. The various models also show whether a simplified version is acceptable for designing an optimal damper. However, before any optimization to maximize lateral acceleration is invoked, an objective function must be determined to obtain an optimum. Using tire data, a preliminary optimization was completed to create an objective function for the suspension models.

In order to construct a computer simulation of the systems, EOMs are required which mode the behavior of a system. The vehicle coordinate system is shown in Figure 4.1. The EOMs for the various models are shown in section 4.2. The identification of mode shapes provides the relative motions of the suspension system at each natural frequency of the various models and is developed in section 4.6.

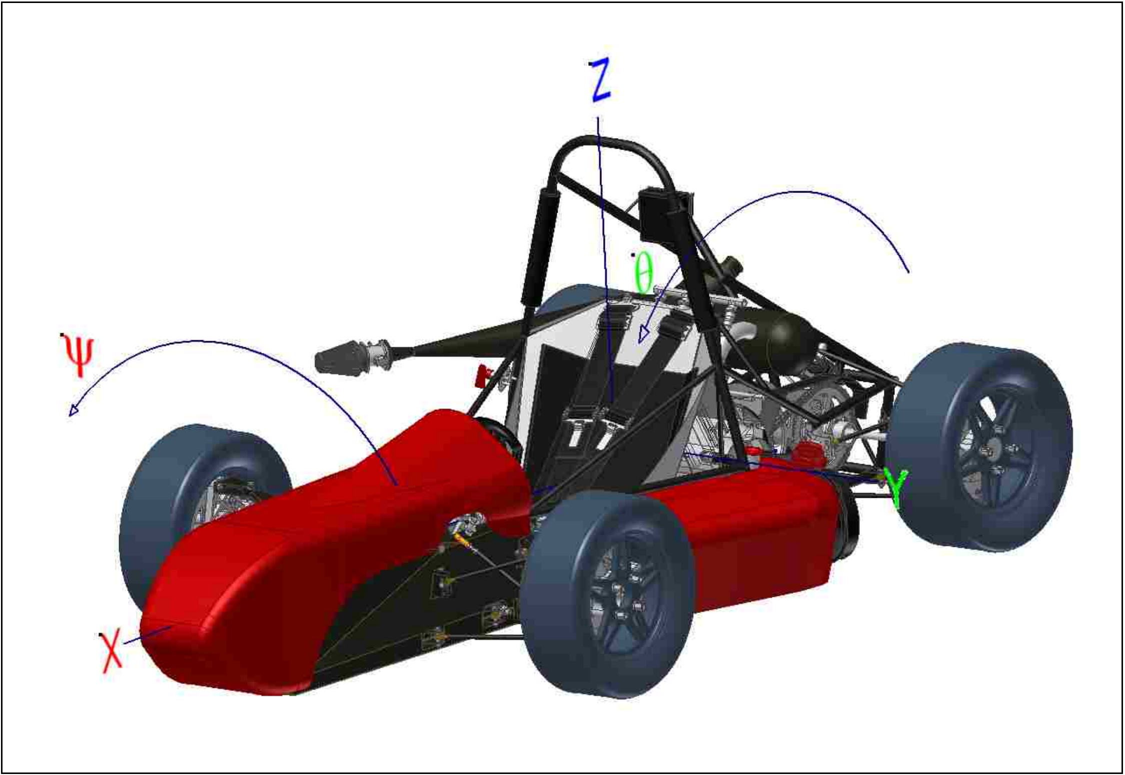

Figure 4.1 – Vehicle Axis System

4.1 Assumptions

Major assumptions made in the modeling, design and analysis are:

1. In order to simplify the Simulink model an assumption was made that the anti-roll bar was unattached. This approximation is acceptable due to the fact that the car is initially designed without the anti-roll bar and added mainly to fine-tune race car handling for a race. The anti-roll bar is a device which can change the weight transfer at the front or rear of the vehicle. The model, however, can be easily modified to include chassis roll stiffness from the anti-roll bar.

2. The suspension geometry was not modeled as this would overcomplicate the EOMs and problem at hand. This simplification is conservative as other lateral

load improvers, such as camber, can be added to a vehicle suspension design on top of the proposed damper optimization to increase the total lateral acceleration of the car.

3. It was assumed that the vehicle would be laterally symmetric in order to simplify computational efforts and also due to the fact that this is typically the case. FSAE cars are designed symmetrically as the road course at competition is undisclosed until the event occurs.
4. Several configurations of the spring, damper and inerter can be used at one corner of the suspension [9]. The most straightforward design was used for the analysis: components in parallel. This is typical of FSAE vehicle designs.
5. Aerodynamics are not evaluated in the design as the UNM FSAE program does not use an aero-kit and additional parameters would be required to optimize performance including but not limited to, pitch and ride height.

4.2 Equations of Motion

In order to build a model to minimize lateral load transfer, the EOMs for each of the suspension models were constructed to allow simulation and damper optimization. The equations can be developed by means of Newton's or Lagrange's equations however, the details of the formulation have been left out and simply the respective EOMs have been presented. All the models (except 2DOF roll model) examined have a "sprung" and "unsprung" mass. The sprung mass, or chassis, is the mass which sits on top of the suspension components. The unsprung mass lies below the suspension components.

Typically, the unsprung mass consists of the tire, wheel, brake system, etc. The 2DOF Roll model uses some assumptions for its construction.

The EOMs are represented in mass/damping/stiffness matrix form as shown in Eq. 4.1 where **M** is the mass matrix, **C** the damping matrix, **K** the stiffness matrix and **x** is the state-vector which describes the coordinates of each system. The forces on the system are described by **f**. These equations are only applicable to the linear models, however, the nonlinear damper simply replaces the damping coefficient used in the Simulink model to represent the full-car nonlinear damper model.

$$\{\mathbf{M}\}\{\ddot{\mathbf{x}}\}+\{\mathbf{C}\}\{\dot{\mathbf{x}}\}+\{\mathbf{K}\}\{\mathbf{x}\}=\{\mathbf{f}\} \qquad \text{Eq. 4.1}$$

4.2.1 2DOF Quarter Suspension Model

A quarter-suspension model was initially developed for verification and validation of the proposed methodology based on methodology from Mechanical Vibrations [10]. This analysis with results is performed in section 4.6.1. The model used in Figure 4.2 has 2DOFs including sprung (m_1) and unsprung mass (m_2), linear spring stiffness (k_1), linear damper (c) and tire stiffness (k_2). Using Newton's equations, the EOMs for the two-mass suspension model were derived in Mechanical Vibrations and are shown again here [10].

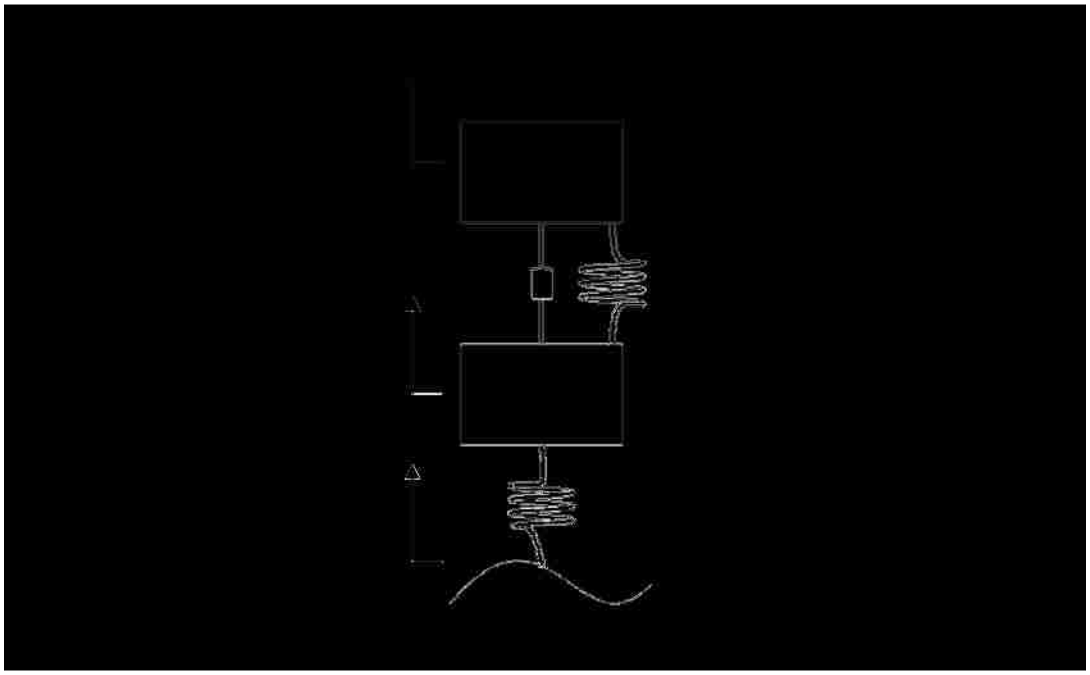
Figure 4.2 – 2DOF Suspension Model

The inerter has not been included as this model is mainly used for validation purposes.

The equations of motion for the 2DOF model are:

$$\mathbf{M} = \begin{pmatrix} m_1 & 0 \\ 0 & m_2 \end{pmatrix}$$

$$\mathbf{C} = \begin{pmatrix} c & -c \\ -c & c \end{pmatrix}$$

$$\mathbf{K} = \begin{pmatrix} k_1 & -k_1 \\ -k_1 & k_1 + k_2 \end{pmatrix}$$

$$\mathbf{f} = \begin{pmatrix} 0 \\ k_2 y \end{pmatrix}$$

Eq. 4.2

The state-vector for the 2DOF quarter-suspension model is given in Eq. 4.3.

$$\mathbf{x} = \{x_1 \quad x_2\}^{\mathrm{T}} \qquad \text{Eq. 4.3}$$

4.2.2 2DOF Roll Model

Based upon Mathworks Simulink-Stateflow Technical Examples [11], a Simulink model was created for studies of a 2DOF roll model. The model [11] was for vehicle pitch and was modified to be used as for vehicle roll. The model included CG position, body mass, mass moment of inertia, front/rear suspension stiffness and front/rear damping. It was later enhanced for the 4DOF and 7DOF models. Figure 4.3 depicts the half-car suspension roll model. The wheel rate and tire rate were modeled as springs in series and then the front springs and rear springs were combined in parallel. Due to the fact that the initial roll model does not have unsprung mass, it was assumed that the total mass of the car would be modeled as sprung mass. The model has uncoupled motion meaning the heave motion has no bearing on the roll motion and vice versa.

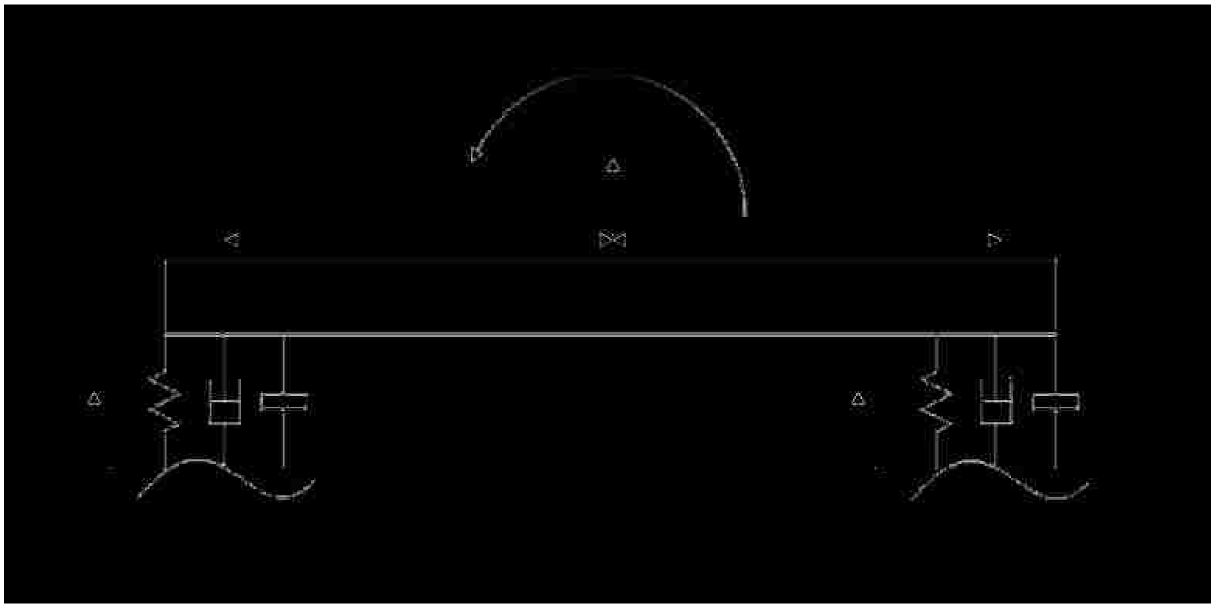
Figure 4.3 – 2DOF Half Car (roll) Suspension Model

The equations of motion for the 2DOF half-car roll model are given in Eq. 4.4.

$$\mathbf{M} = \begin{pmatrix} M_b + 4b & 0 \\ 0 & 4bL^2 + I_{xx} \end{pmatrix}$$

$$\mathbf{C} = \begin{pmatrix} 4c & 0 \\ 0 & 4cL^2 \end{pmatrix}$$

Eq. 4.4

$$\mathbf{K} = \begin{pmatrix} 4k & 0 \\ 0 & 4kL^2 \end{pmatrix}$$

$$\mathbf{f} = \begin{pmatrix} 0 \\ T(t) \end{pmatrix}$$

The state-vector for the 2DOF roll model is given in Eq. 4.5.

$$\mathbf{x} = \{Z \quad \psi\}^{\mathrm{T}}$$ Eq. 4.5

4.2.3 4DOF Half-Car Model

The EOMs for the sprung and unsprung-mass, half-car suspension roll model were derived and are shown in Eq. 4.6. Figure 4.4 depicts the half-car suspension roll model. The FSAE car uses unequal front and rear track widths as shown in Table 4.1. The roll-model is defined by having the front and rear wheels inline so an approximation had to be made; the average track width was used for the distance between each suspension attach point with the CG mid-track.

The front and rear tire stiffness was modeled as springs in parallel as was the front and rear suspension springs; similarly, the dampers and inerters are modeled as being in parallel.

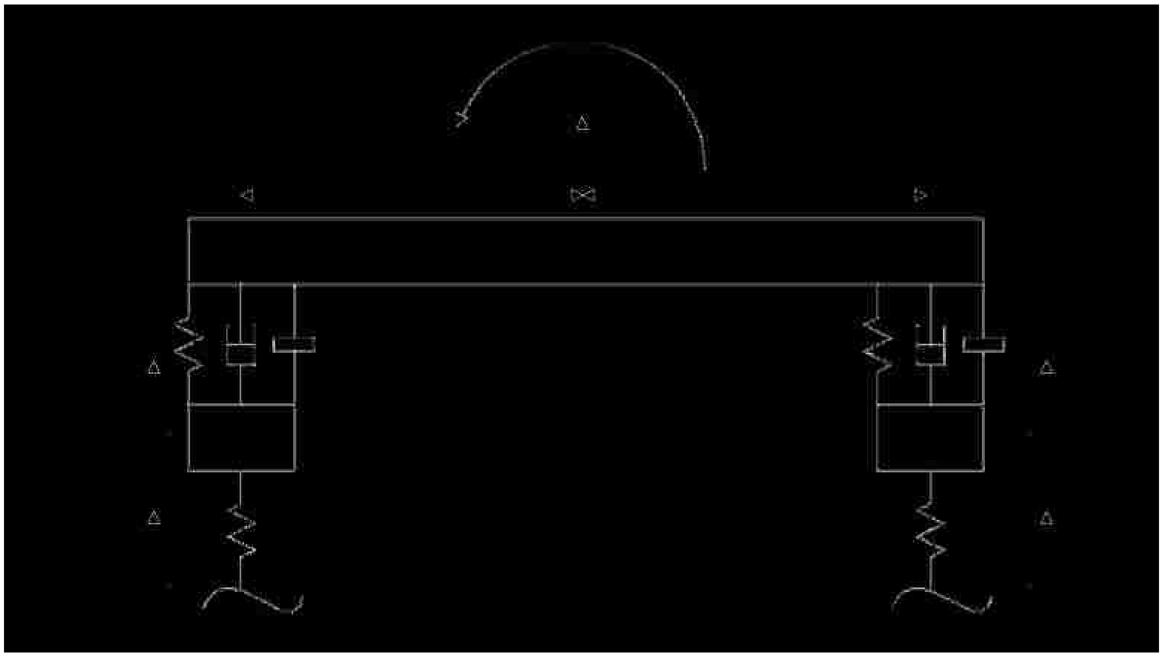
Figure 4.4 – 4DOF Half-Car Suspension Model

The equations of motion for the 4DOF half-car model are given in Eq. 4.6.

$$\mathbf{M} = \begin{pmatrix} 4b + M_b & -2b & -2b & 0 \\ -2b & 2b + 2m_1 & 0 & 2bL \\ -2b & 0 & 2b + 2m_2 & -2bL \\ 0 & 2bL & -2bL & 4bL^2 + I_{xx} \end{pmatrix}$$

$$\mathbf{C} = \begin{pmatrix} 4c & -2c & -2c & 0 \\ -2c & 2c & 0 & 2cL \\ -2c & 0 & 2c & -2cL \\ 0 & 2cL & -2cL & 4cL^2 \end{pmatrix} \qquad \text{Eq. 4.6}$$

$$\mathbf{K} = \begin{pmatrix} 4k & -2k & -2k & 0 \\ -2k & 2k + 2k_t & 0 & 2kL \\ -2k & 0 & 2k + 2k_t & -2kL \\ 0 & 2kL & -2kL & 4k_r L_r^{\ 2} \end{pmatrix}$$

$$\mathbf{f} = \begin{pmatrix} 0 & 2k_t q_1(t) & 2k_t q_2(t) & T(t) \end{pmatrix}^{\mathrm{T}}$$

The state-vector for the 4DOF model is given in Eq. 4.7.

$$\mathbf{x} = \{Z \quad z_1 \quad z_2 \quad \psi\}^{\mathrm{T}} \qquad \text{Eq. 4.7}$$

4.2.4 7DOF Full-Car Model

The EOMs for the full-car model include coordinates for the vertical motion of the unsprung masses, chassis heave (Z), roll (ψ) and pitch (θ). The model uses unequal front (t_f) and rear (t_r) trackwidths, accurately representing the UNM FSAE vehicle. The front/rear vehicle weight distribution was also included in the model which requires a front (L_f) and rear (L_r) moment arm. Figure 4.5 depicts the full-car suspension model.

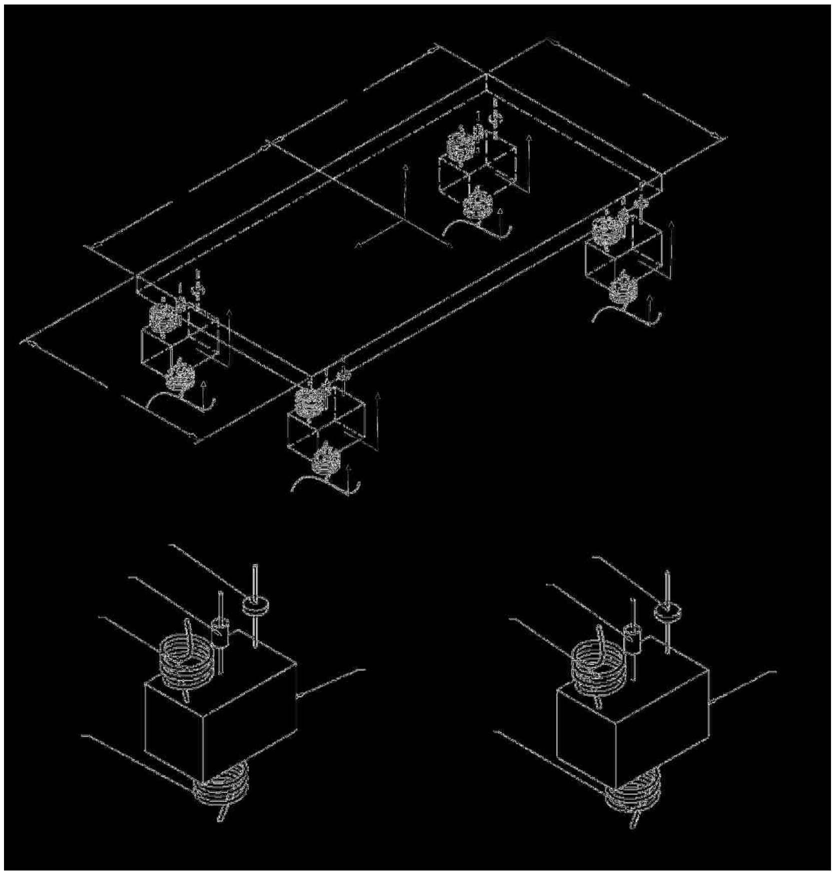
Figure 4.5 – 7DOF Full-Car Suspension Model

The equations of motion for the full-car roll model are given in Eq. 4.8.

[illegible]

[illegible]

[illegible]

$$\mathbf{f} = \left(k_t q_{lf}(t) - m_f g \quad k_t q_{lr}(t) - m_r g \quad k_t q_{rr}(t) - m_r g \quad k_t q_{rf}(t) - m_f g \quad -M_b g \quad M_x(t) \quad M_y(t)\right)^T$$

Eq. 4.8

The state-vector for the 7DOF model is given in Eq. 4.9.

$$\mathbf{x} = \{ z_{lf} \quad z_{lr} \quad z_{rr} \quad z_{rf} \quad Z \quad \psi \quad \theta \}^{\mathrm{T}} \qquad \text{Eq. 4.9}$$

4.3 Design Input

This section lists the inputs used in the methodology for designing the nonlinear damper force-velocity curve. These include the car specifications as well as the inertial and road inputs used in the simulation.

4.3.1 Generic Car Parameters

Vehicle data from a data acquisition system was used to simulate a virtual car driving on a track. This data includes lateral/longitudinal/vertical chassis acceleration which was recorded as a function of time for one lap. The data is from a larger car which is capable of higher lateral and longitudinal acceleration than the FSAE car. Thus, the data has been scaled to represent UNM FSAE vehicle handling characteristics. This approximately models a small car on a big road course. A layout of the track driven is shown in Figure 4.6. The blue vertical line indicates the starting point of the lap and the track was traversed in a clockwise direction.

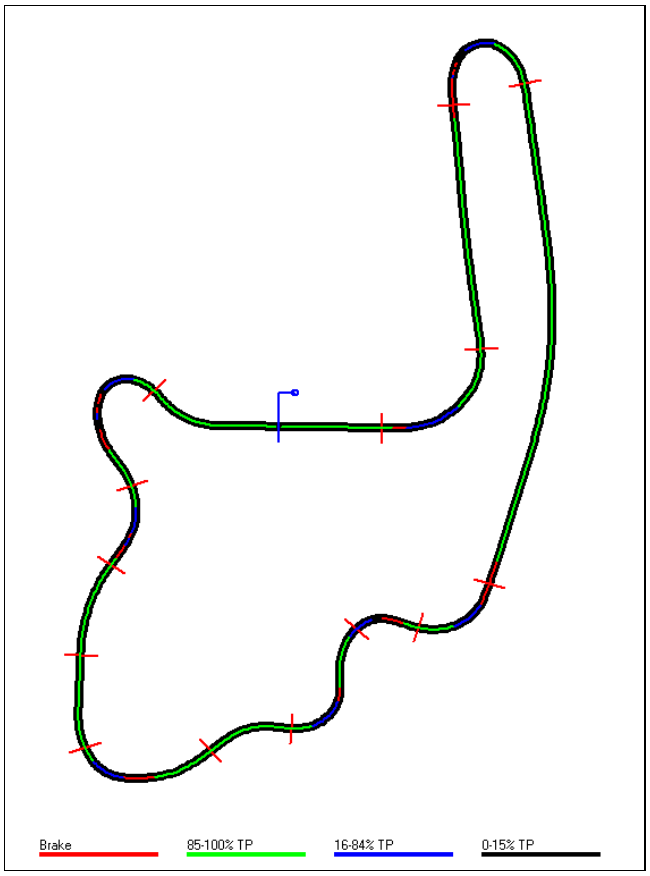

Figure 4.6 – Mid-Ohio Race Track

The simulation input requires inertial moments so the acceleration data was multiplied by the vehicle weight and CG height. Figure 4.7 - Figure 4.9 illustrate the scaled data used for the input into the various simulations. The vertical acceleration did not require scaling for the simulations.

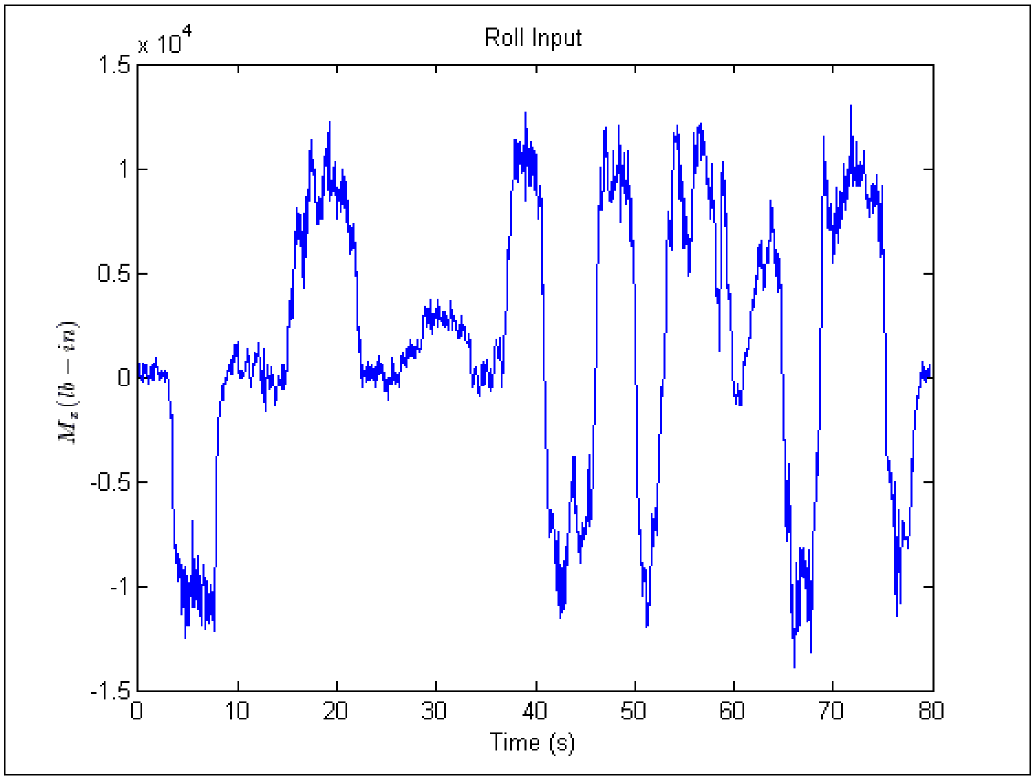

Figure 4.7 – Roll Torque Input

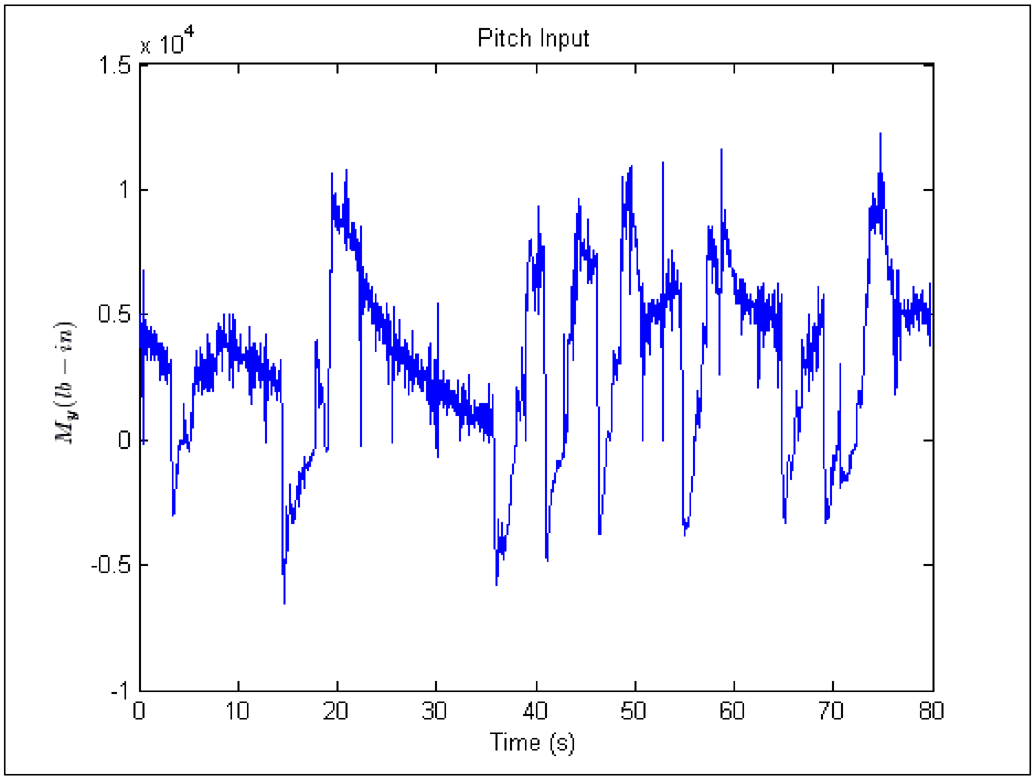

Figure 4.8 – Pitch Torque Input

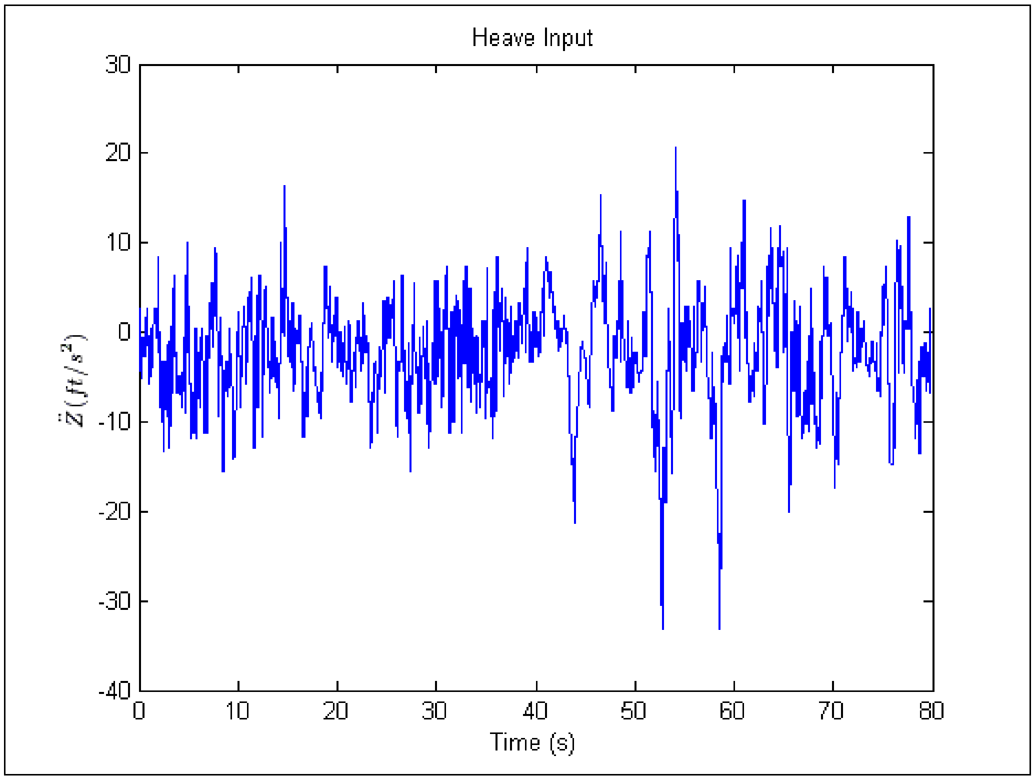

Figure 4.9 – Heave Acceleration Input

4.3.2 UNM FSAE Race Car Parameters

The 2007 UNM FSAE car parameters were used in the design of the linear and nonlinear damper and linear inerter. These parameters include CG position, body mass, unsprung mass, front/rear suspension stiffness and front/rear installation ratios which were found from the 2007 FSAE Design Specification Sheet (see Appendix C). The relevant parameters are listed in Table 4.1.

Table 4.1 – 2007 UNM FSAE Car Parameters

Parameters	**Front**	**Rear**
Wheelbase	60 inches	
Track	50 inches	48 inches
Weight with 150lb Driver	310 lbs	340 lbs
Unsprung Weight	62 lbs	70 lbs
Inertia (Ixx)	2.056×10^5 lb-in^2	
Inertia (Iyy)	2.460×10^6 lb-in^2	
Center of Gravity Height	12 in	
Wheel Rate (chassis to wheel center)	148 lb/in	126 lb/in
Motion Ratio	0.77	0.53
Tire Rate	550 lb/in	550 lb/in

The front/rear damping rate is required for the optimization to form an initial guess and to provide verification that the magnitude of the optimized damping rates are practical. The low-speed damping was found by fitting a trendline to the initial slope of the 2007 UNM FSAE damper force-velocity curve. The high-speed damping was found by fitting a trendline to the final slope of the force-velocity curve, and the transition velocity was found at the minimum velocity of the high-speed damping. This curve was generated by a damper dynamometer at the bicycle shock absorber manufacturer Manitou and provided to the 2007 UNM FSAE team. The curve fit is shown in Figure 4.10 below.

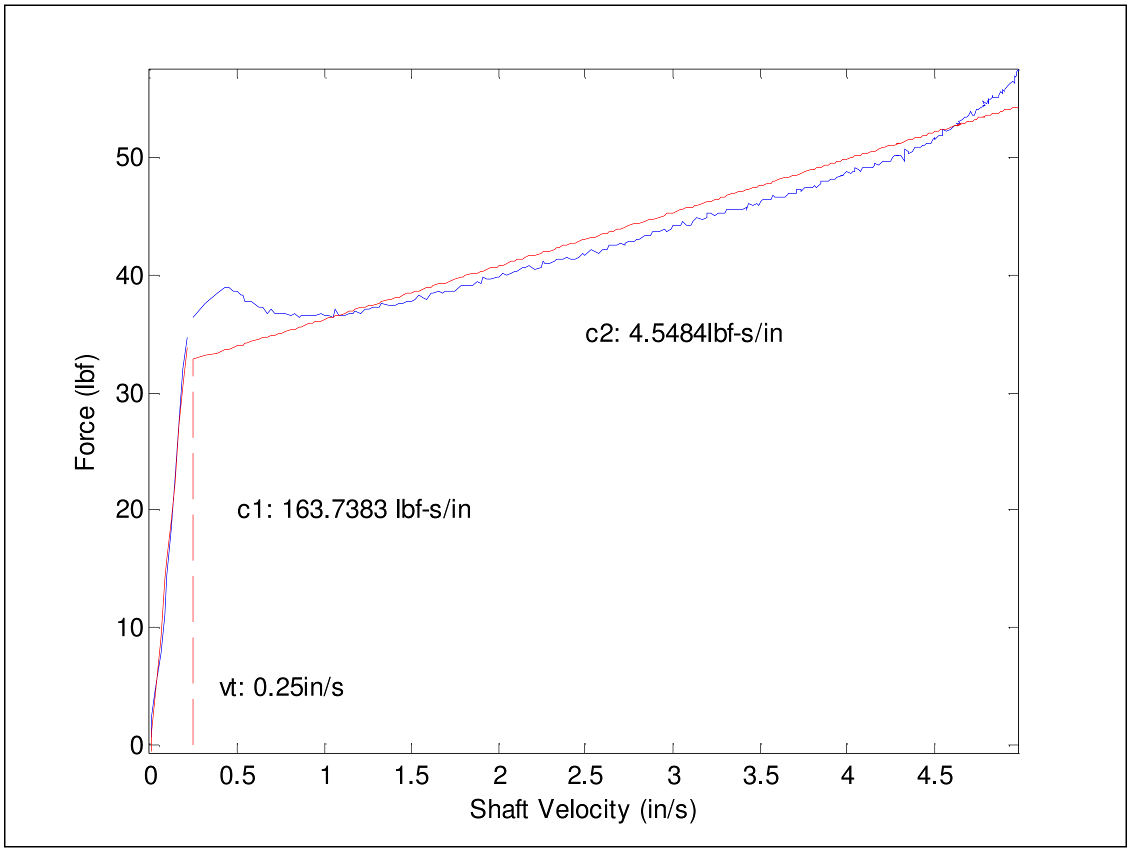

Figure 4.10 – 2007 UNM FSAE Nonlinear Damper Curve Fit

Tire data is required to lay the foundation for the methodology to create a proper optimization goal or objective function. This data was purchased by UNM FSAE in 2008 but is still applicable to the 2007 car. The analysis of this data is performed in section 4.5.

4.4 Model Construction

Simulink models were then constructed to solve the EOMs developed in Section 4.2, allow for damper optimization and perform other functions as needed including root locus analysis. The block diagrams are included in Appendix B. Using the average damping coefficient of the front and rear dampers as the gain, the root locus was created to determined the behavior of the system and facilitate comprehension of the optimization results. This analysis is performed in section 4.6.

A drawback to the root locus technique is that the system must be single-in single-out (SISO). The 2DOF quarter suspension model is a SISO system and Matlab is capable of producing a root locus figure for it [10]. The other systems, however, are multiple-input multiple-output (MIMO) and require special attention which is addressed in section 4.6.

4.4.1 2DOF Half-Car Roll Model

The 2DOF roll model used simple harmonic motion for the lateral load input as it was used for a preliminary model. The sinusoid resembles a slalom that the vehicle would undergo in a typical road race. The 4.33 rad/s spatial frequency of the slalom was based on lateral acceleration through the slalom amongst other parameters. The amplitude of the sinusoid was found by using the 1.6 lateral g's from section 4.5.1 multiplied by the weight of the car, 680 lb, multiplied by the 12 in CG height from section 3. This produces a moment of 13,056 lb-in. Since there was no data on road noise collected, the road input was modeled as random noise ranging ± 0.005 inches.

4.4.2 4DOF Half-Car Model

The model used the roll torque data seen in Figure 4.7 in the simulation to represent a simulated track. The road input was again modeled as random noise in the range ± 0.005 inches.

4.4.3 7DOF Full-Car Model

The full-car benefits from not having the simplifications required in the previous models. This implies that the full-car model is the benchmark for the other models. The model used roll, pitch and heave input from section 4.3.1 as well the random noise road input used in the previous models. Lateral symmetry was enforced in the full-car model, but not longitudinal symmetry.

4.5 Optimization

The objective function to be minimized which maximizes vehicle lateral acceleration is identified here as well as the algorithm used by the optimization. The ultimate goal of the optimization is to find a nonlinear damping curve which best reduces the weight transfer at each time step. The optimization was accomplished numerically using Matlab. The algorithm used by the Matlab tool solves nonlinear data-fitting problems. The optimization code is included in Appendix A.

4.5.1 Objective Function Identification

To verify the postulate that equally loaded tires provide the most lateral force, Goodyear tire data was used to find an optimal tire loading which maximized lateral g's for a four-tire vehicle based on the Pacejka formula (see Appendix A). In order to find the lateral force from the tire, the Pacejka formulas require vertical load (F_z), slip angle (α) and camber angle (γ) for each tire (i.e. $F_y = F_y(Fz, \alpha, \gamma)$). This implies that there are a total of

12 variables which need to be established. From assumption 2, the camber angle can be ignored as it requires knowledge of suspension geometry to be determined. This reduces the number of variables from 12 to eight. A reasonable approximation is that the inner and outer slip angles are the same. This reduces the number of variables from eight to six. Rather than define the load on each tire, it was more convenient to define the weight transfer for the front and rear of the car. Consider a vehicle making a turn; the weight transfer at the front of the car (ΔW_f) is the load that is transferred from the front-inner tire to the front-outer tire. If $\Delta W_f = 0$, then no weight is transferred and the load on the tire is the static load. If $\Delta W_f > 0$ the load on the front-inner tire is reduced by ΔW_f and the load on the front-outer tire increases by ΔW_f. The same logic for the front weight transfer applies to the rear weight transfer (ΔW_r). This further reduced the number of variables from six to four.

The Pacejka formulas were then used to optimize the vehicle lateral acceleration. The vehicle lateral acceleration is then given as:

$$n_y = \frac{Fy_{fo}(\Delta W_f, \alpha_f) + Fy_{fi}(\Delta W_f, \alpha_f) + Fy_{ro}(\Delta W_r, \alpha_r) + Fy_{ri}(\Delta W_r, \alpha_r)}{W \cdot g} \qquad \text{Eq. 4.10}$$

where,

Fy_{fo} = front outer tire lateral force

Fy_{fi} = front inner tire lateral force

Fy_{ro} = rear outer tire lateral force

Fy_{ri} = rear inner tire lateral force

W = vehicle weight

g = acceleration of gravity

ΔW_f = front lateral weight transfer

ΔW_r = rear lateral weight transfer

α_f = front slip angle

α_r = rear slip angle

The result of maximizing n_y as a function of the four variables (ΔW_f, ΔW_r, α_f, α_r) is shown in Figure 4.11. The bottom subplot shows the function (n_y) at each iteration of the optimization as it reaches the maximum value at the final point on the right (~iteration 210). The last iteration point on this subplot is the maximum lateral acceleration of the vehicle in g's which the four tires can produce. The value of the last (and therefore optimal) point is displayed as the "Current Function Value" shown in the middle of the figure in g's. The top subplot shows the optimal variables at the last iteration which produce the maximum lateral acceleration. From left to right the variables are α_f, α_r, ΔW_f, and ΔW_r.

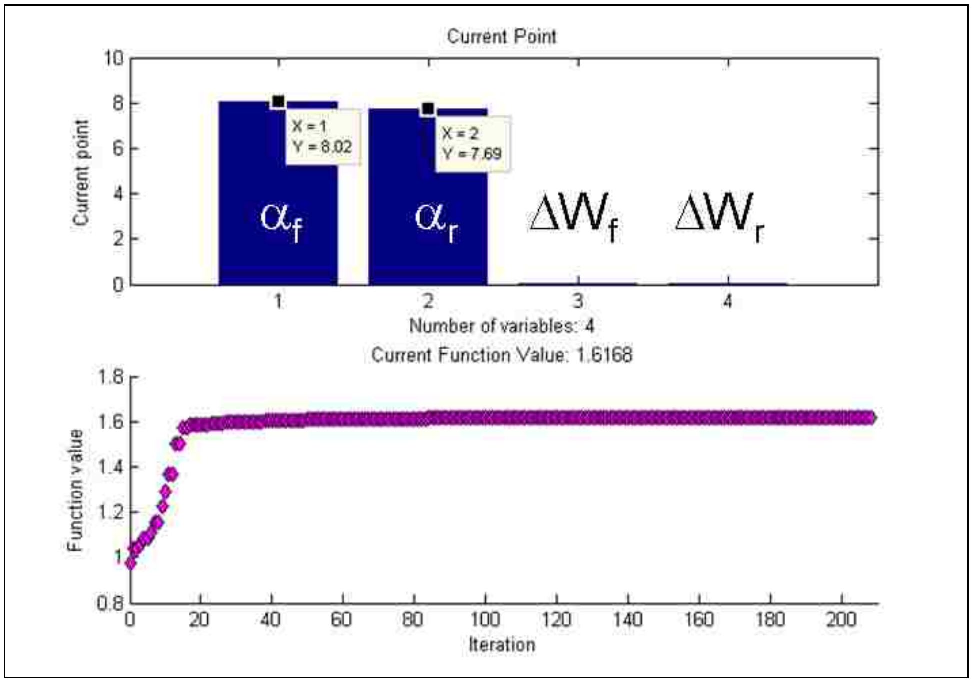

Figure 4.11 – Optimized Goodyear Tire Lateral Force

By inspection of Figure 4.11, the optimization (see Appendix A) has shown that a maximum of 1.6168 g's can be produced by the car if the front tire slip angles are 8.02°, rear tire slip angles are 7.69°, and no weight is transferred from the inside to the outside tires at the front and rear of the vehicle. It should be noted that the 2007 FSAE vehicle was capable of producing ~1.5 lateral g's, so the optimization has shown that the vehicle could, in theory, produce more lateral acceleration if lateral weight transfer were reduced. The implication is that, in fact, the postulated equally loaded tires produces maximum lateral acceleration, and minimizing weight transfer should be the objective function for optimizing the nonlinear damper and inerter.

4.5.2 Optimization Algorithm

The Matlab function “lsqnonlin” solves nonlinear, least-squares problems as required for the damper optimization for a road race vehicle. In general, starting at an initial vector the routine seeks the vector values for which the sum of squares of the function is minimized. The vector in this case contains the damping curve or inertance. Optimization procedure item 5 of section 3.2 discusses the fact the damper was optimized separately from the inerter.

The function “lsqnonlin” uses the trust-region reflective algorithm which, at each iteration, involves the approximate solution of a large linear system using the method of preconditioned conjugate gradient; a very efficient method which seeks to minimize the function by minimizing its gradient by generating a succession of search directions [12]. This approximate solution is known as the trust-region. A trial step is computed by minimizing inside the trust-region, which is known as the trust-region subproblem. The current point is then updated or the region of trust is then shrunk and the trial step is repeated.

In the standard trust-region method, the quadratic approximations are defined by the first two terms of Taylor approximation. The Matlab solver restricts the trust-region subproblem to a two-dimensional subspace where the solution to the subproblem is simple. The next step is finding the two-dimensional subspace using the preconditioned conjugate gradient process. The solver defines the linear space spanned by a vector in the direction of the gradient and another vector in the approximate Gauss-Newton direction.

It should be noted that the routine only finds a local optimum as it only seeks a place where the gradient is zero. This implies that the optimization only finds local minima and requires refinement to obtain a global minimum. The low-speed damping of the FSAE damper was first chosen as the initial damping. A much larger number was then chosen as the initial damping. The direction these values were changed from their initial values by the optimization was indicative of the location of the global maximum (as can be seen in the results tables of section 5). By choosing the next initial damping value as the average of the two previous optimized damping values, another indicator of the location of the global optimum was predicted. This process continued until convergence was achieved and a global minimum was found.

To assess the weight transfer with a scalar value, a vector was formed by subtracting the sum of forces at the right tires ($F_{yr}(t_i)$) by the force at the left tires ($F_{yl}(t_i)$) for each time step. The norm of this vector was then taken which was referred to as the weight transfer norm ($||\Delta W||$) and was evaluated at the end of each optimization iteration to assess the reduction in weight being transferred laterally. This is similar to RMS although the value was not normalized by a time period. The equation for the weight transfer norm is shown in Eq. 4.11.

$$\|\Delta W\| = \sqrt{\sum_{i=1}^{n} [F_{yr}(t_i) - F_{yl}(t_i)]^2} \qquad \text{Eq. 4.11}$$

4.5.3 Suspension Model Considerations

The quarter suspension model cannot be optimized since weight transfer is not a parameter in the model. The model is constructed mainly as a means of validation for root locus creation in section 4.6.

The full-car model benefits from not having the simplifications necessary in the previous models. The necessary assumptions made, however, are listed in section 4.1. The various model complexities (DOFs) were also beneficial to decide whether simpler models are reasonable approximations or whether higher-order models need to be constructed to simulate a vehicle properly.

4.6 Identification of System Mode Shapes & Root Locus

Identifying the mode shapes means the primary components of motions of the system can be viewed in relation to each other. Occurring at the natural frequencies of the system, the mode shapes provide insight as to how the optimization routine has changed the system behavior to minimize the objective function.

The natural frequencies of a system are the square of the eigenvalues, and the mode shapes are the associated eigenvectors. This implies the natural frequencies in the root locus plot have associated mode shapes. The mode shapes are found by neglecting damping and computing the eigenvalues and eigenvectors of the mass and stiffness matrices from the EOMs. The natural frequencies can then be found by taking the square root of the eigenvalues. It should be noted that mode shapes and root locus figures are only applicable to the linear systems; therefore no root locus figures are created for the full-car nonlinear damper model. An example of identifying mode shapes is shown in section 4.6.1.

Palm [10] presents a method for constructing a root locus for the 2DOF quarter suspension-model. The method arranges the transfer function into the standard form for creating the root locus and solves for the damping rate as the gain. This method would be very beneficial for all the models; however, no closed form solution for the other models can be created due to higher-order system complexities. In order to create a root locus, the average of the damping rates from each damper was parametrically varied in the model followed by eigenvalues extraction at each iteration. The eigenvalues were then plotted on the s-plane to create the root locus figure. It was decided that the best way to plot the eigenvalues was by varying the damping (gain) from zero to the optimized value. This means that the final pole of the root locus corresponds to the optimal damping rate and was marked with a red dot to signify it is the last value. The first pole was marked with a green dot for further assistance. With exception to the quarter-suspension model, the root locus figures for the models are presented in section 5.

4.6.1 2DOF Quarter Suspension Model

4.6.1.1 Mode Shape Identification Example

As previously mentioned, the mode shapes are the eigenvectors of the system with associated eigenvalues. Finding the eigenvectors by neglecting damping more readily identifies the branch of the root locus figure with the associated natural frequency. Assuming no forcing is present, the system of differential equations from Eq. 4.1 is then simplified to Eq. 4.12.

$$[\mathbf{M}]\{\ddot{\mathbf{x}}\}+[\mathbf{K}]\{\mathbf{x}\}=\mathbf{0} \qquad \text{Eq. 4.12}$$

The homogeneous solution to the undamped system is simple harmonic motion which is represented by Eq. 4.13.

$$\{\mathbf{x}\} = \{\mathbf{u}\}\sin(\omega_n t) \qquad \text{Eq. 4.13}$$

Substituting Eq. 4.13 into Eq. 4.12 and simplifying yields Eq. 4.14, where **u** is the eigenvector with the associated eigenvalue ω_n^2.

$$[\mathbf{K}]\{\mathbf{u}\} = [\mathbf{M}]\{\mathbf{u}\}\omega_n^{\,2} \qquad \text{Eq. 4.14}$$

Substituting the values m_1 = 250kg, m_2 = 40kg, k_1 = 15,000 N/m, k_2=150,000 N/m into Eq. 4.2 yields:

$$\mathbf{M} = \begin{pmatrix} 250 & 0 \\ 0 & 40 \end{pmatrix}$$

$$\mathbf{K} = \begin{pmatrix} 15{,}000 & -15{,}000 \\ -15{,}000 & 165{,}000 \end{pmatrix} \qquad \text{Eq. 4.15}$$

The eigenvalues (ω_i^2) and eigenvectors (u_i) can then be found by use of the Matlab function “eig” or by some other means of calculating them so that they satisfy Eq. 4.14. The state vector is $\mathbf{x} = \{x_1 \quad x_2\}^{\mathrm{T}}$ where x_1 is the motion of the sprung mass and x_2 is the motion of the unsprung mass. The mode shapes and natural frequencies are given in Table 4.2. The first mode (i=1) shows that at the natural frequency of 7.38 rad/s, the primary motion is that of the sprung mass since 1.0>>0.0921. Conversely, the second mode shows primarily unsprung motion.

Table 4.2 – 2DOF Quarter-Suspension Natural Frequencies and Mode Shapes

i	ω_i^2	ω_i	u_i	Primary Motion
1	54.5	7.3805	$\{1.0, 0.0921\}^T$	Sprung Mass
2	4130.5	64.2692	$\{0.01471, -1.0\}^T$	Unsprung Mass

4.6.1.2 Root Locus Benchmark

To verify that the proposed method of extracting eigenvalues to create a root locus was valid, the Palm example was used as a benchmark [10]. The transfer function for sprung mass to road input is given in Eq. 4.16.

$$\frac{X_1(s)}{Y(s)} = \frac{k_2(Cs + k_1)}{(m_1 s^2 + Cs + k_1)(m_2 s^2 + Cs + k_1 + k_2) - (Cs + k_1)^2} \qquad \text{Eq. 4.16}$$

Factoring out c, rearranging into root locus form and substituting the values m_1 = 250kg, m_2 = 40kg, k_1 = 15,000 N/m, k_2=150,000 N/m the denominator becomes Eq. 4.17.

$$s^4 + 4185 s^2 + 2.25\times10^5 + K(s^3 + 150s + 0.29s^3) = 0 \qquad \text{Eq. 4.17}$$

Finally, the root locus for the quarter-suspension model shown in Figure 4.12 was created using Matlab functions with the damping coefficient as the gain. In addition, data cursors were placed at the poles on the imaginary axis, representing the system without damping, for the purpose of determining the natural frequency. The natural frequencies of these poles correspond to those in Table 4.2 and therefore correspond to the associated eigenvectors (mode shapes). The primary motions of the root locus branches have been labeled to indicate the mode shapes.

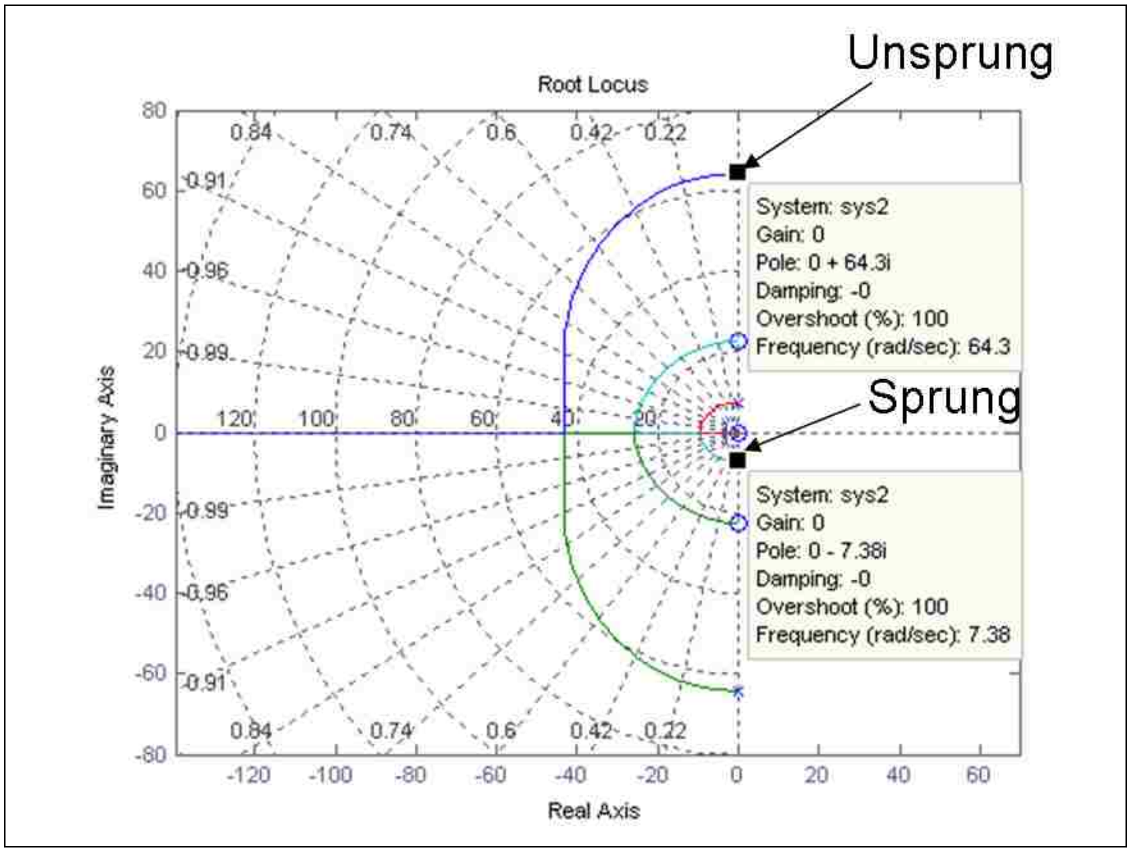

Figure 4.12 – 2DOF Quarter Suspension Benchmark Root Locus

The branches of the root locus begin at the poles of the undamped system and end at the zeros where the gain (damping coefficient) goes to infinity. Figure 4.12 shows that the motion of the system becomes oscillatory if the damping rate is high. This is to be expected as Milliken [1] describes, “if damping is too high, dampers tend to control the suspension motion and overpower the spring rates,” and the system essentially behaves as a 1DOF since the suspension is nearly rigid with the only deflection coming from the tire.

4.6.1.3 Root Locus from Eigenvalue Extraction Example

The proposed method to create a root locus plot by iteratively varying the damping and finding the eigenvalues produces Figure 4.13, superimposed on the Matlab root locus plot. Small increments were used to vary the damping rate of the 2DOF quarter-suspension model and subsequently find the eigenvalues of the simulated system using Matlab functions at each step. The eigenvalue at each iteration was plotted as a dot on the s-plane which shows up as a black line on Figure 4.13 since the plotted points overlap. The iterative method and benchmark root locus show good agreement. It should be noted that an infinite damping rate (gain) is required to populate the branch that breaks away from the real axis with the iterative method as the gain heads toward infinity quickly. The black dots of the manual plotting method show that only a finite gain can be used to create the manual root locus figure due to computing limitations; however, infinite damping is not realistic for design purposes.

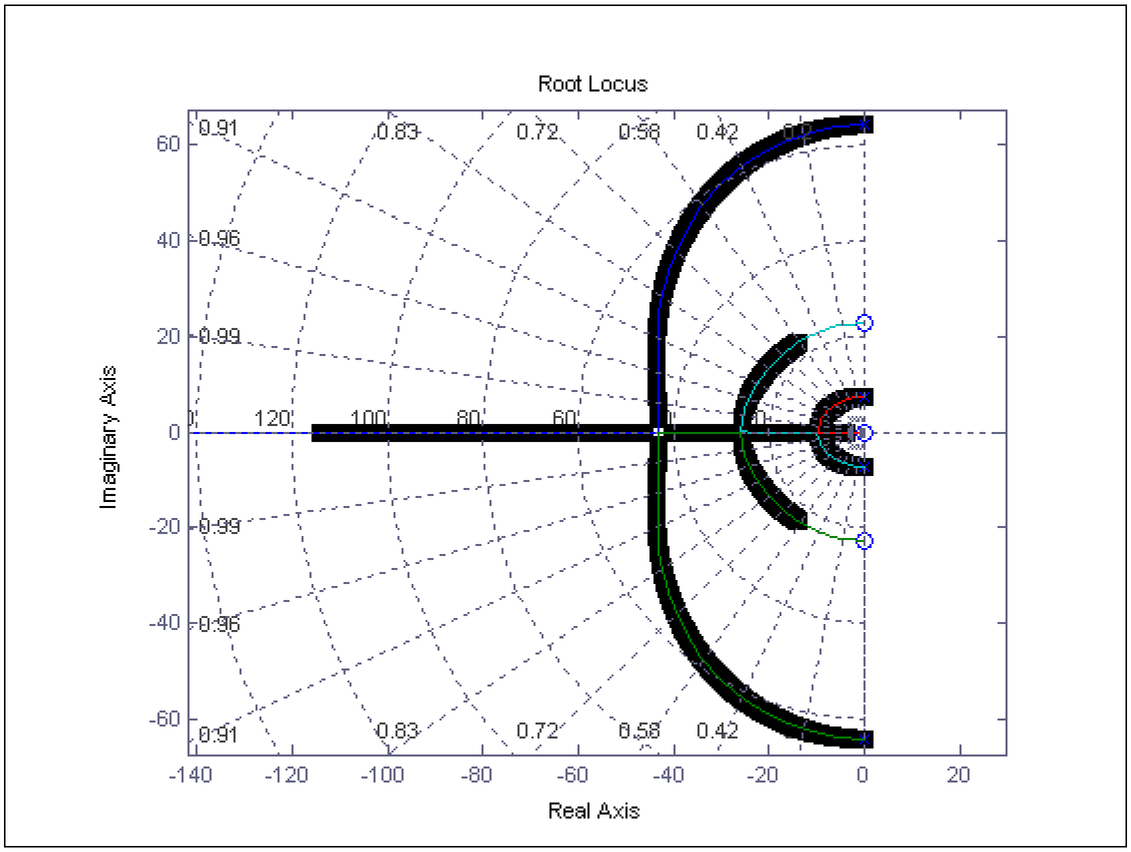

Figure 4.13 – Iterative Plotting Technique and Benchmark 2DOF Quarter Suspension Root Locus

4.6.2 2DOF Half-Car Roll Model

The mass and stiffness matrices of the 2DOF Roll model are in section 4.2.2. The state vector is $\mathbf{x} = \{Z \quad \psi\}^{\mathrm{T}}$ where Z is chassis heave and ψ is chassis roll depicted in Figure 4.3. The mode shapes and natural frequencies are given in Table 4.3. The first mode shows chassis roll with no heave, conversely, the second mode shows heave without roll which is defined as uncoupled motion. This implies that the roll motion has no effect on the heave motion as expected from section 4.2.2.

Table 4.3 – 2DOF Roll Suspension Natural Frequencies and Mode Shapes

i	ω_i	u_i	Primary Motion
1	0.8088	$\{0\ -1\}^T$	Roll
2	3.2554	$\{-1\ 0\}^T$	Heave

4.6.3 4DOF Half-Car Model

The mass and stiffness matrices of the 4DOF suspension model are given in Eq. 4.6. The state vector is $\mathbf{x} = \{Z \quad z_1 \quad z_2 \quad \psi\}^T$ where Z is chassis heave, ψ is chassis roll, z_1 is the left wheel motion and z_2 is the right wheel motion depicted in Figure 4.4. The mode shapes and natural frequencies are given in Table 4.4. The first mode and third modes are mainly wheels moving in opposite directions (out-of-phase) with some body roll. The second mode is primarily heave with mild wheel motion. The fourth mode consists of wheels moving together synchronously (in-phase) with almost no chassis heave.

Table 4.4 – 4DOF Suspension Natural Frqequencies and Mode Shapes

i	ω_i	u_i	Primary Motion
1	1.0051	$\{0.0000, 1.0000, -1.0000, -0.1220\}^T$	Counter-Wheel
2	4.4472	$\{-1.0000, -0.3521, -0.3521, 0.0000\}^T$	Heave
3	19.4990	$\{0.0000, 1.0000, -1.0000, 0.0002\}^T$	Counter-Wheel
4	19.7599	$\{0.0848, -1.0000, -1.0000, 0.0000\}^T$	Synchronous-Wheel

4.6.4 7DOF Full-Car Model

The mass and stiffness matrices of the 7DOF suspension model are given in Eq. 4.8. The state vector is $\mathbf{x} = \{z_{lf} \quad z_{lr} \quad z_{rr} \quad z_{rf} \quad Z \quad \psi \quad \theta\}^T$ where Z is chassis heave, ψ is chassis roll, θ is chassis pitch, z_{lf} is the left-front wheel motion, z_{lr} is the left-rear wheel motion, z_{rr} is the right-rear wheel motion and z_{rf} is the right-front wheel motion depicted in Figure 4.5. The mode shapes and natural frequencies are given in Table 4.5. The first mode primarily shows the front wheels out-of-phase with the rear wheels, chassis heave and pitch. The second mode shows the left-side wheels out-of-phase with the right-side wheels as well as chassis roll. The third mode shows primarily chassis heave and modes four through seven show primarily wheel motion occurring at much higher frequencies due to the tire stiffness being much higher than the suspension stiffness.

Table 4.5 – 7DOF Suspension Natural Frqequencies and Mode Shapes

i	ω_i	u_i	Primary Motion
1	0.3892	$\{0.9981, -1.0000, -1.0000, 0.9981, -0.8139, -0.0000, -0.1769\}^T$	Front Wheels opposite Rear Wheels, Heave, Pitch
2	1.1091	$\{-1.0000, -0.7662, 0.7662, 1.0000, -0.0000, -0.1883, 0.0000\}^T$	Left Wheels opposite Right Wheels
3	4.9396	$\{0.2196, 0.1761, 0.1761, 0.2196, 1.0000, 0.0000, 0.0000\}^T$	Heave
4	24.6694	$\{-0.0014, 1.0000, -1.0000, 0.0014, 0.0000, 0.0000, 0.0000\}^T$	Left-Rear opposite Right-Rear
5	24.7129	$\{-0.0293, 1.0000, 1.0000, -0.0293, -0.0218, 0.0000, 0.0000\}^T$	Left Rear synchronous w/ Right Rear
6	26.9172	$\{-1.0000, -0.0011, 0.0011, 1.0000, 0.0000, 0.0000, 0.0000\}^T$	Left-Front opposite Right-Front
7	26.9862	$\{1.0000, 0.0216, 0.0216, 1.0000, -0.0253, 0.0000, 0.0000\}^T$	Left Front synchronous w/ Right Front

4.7 Analysis Chapter Recapitulation

Section 4 has implemented the methodology procedure to which will maximize the lateral acceleration of a vehicle by optimizing a nonlinear damper to minimize lateral load transfer. By using some assumptions in creating the EOMs for each of the various vehicle suspension models, a simulation was, in turn, created to optimize the nonlinear damper and inerter. It was also determined that minimizing lateral load transfer would maximize lateral acceleration, thus lateral load transfer is the objective function of the optimization. A method to understand how the optimization has changed the suspension system to accomplish its objective was also presented by identifying the mode shapes of each model. The mode shapes have associated natural frequencies which correspond to branches on the root locus figures which are presented in section 5 along with the results of the optimization.

The root locus figures use the linear damping rate as the gain and were created by varying the linear damping coefficient and extracting the eigenvalues of the model at each iteration. The eigenvalues were then plotted on the s-plane to create the root locus figure. It was decided that the best way to plot the eigenvalues was by varying the damping (gain) from zero to the optimized value. This means that the final pole of the root locus corresponds to the optimal damping rate and was marked with a red dot to signify it is the last value. The first pole was marked with a green dot for further assistance. The EOMs and root locus figures are only applicable to the linear systems, however, and are not created for the nonlinear damper.

The results of the analysis of each suspension model are presented in section 5.

5 RESULTS

The methodology procedure to maximize lateral acceleration by minimizing lateral load transfer and identify the optimized suspension system modes was constructed in section 4. The results of the procedure for each suspension model are presented here.

The optimization of the linear damper was first used on simpler models culminating with the full-car model. The root locus figures were created by varying the linear damping coefficient and extracting the eigenvalues of the model at each iteration. These eigenvalues were then plotted on the s-plane using a green dot to represent the starting point with no damping and a red dot to indicate the optimized damping coefficient.

The nonlinear damper was then optimized on the full-car model. No root locus figure was created for the nonlinear model since root locus is only applicable to linear models. Finally the inerter was optimized on the full-car separately, using the optimized nonlinear damper in the model.

5.1 Linear Damper Optimization

This section presents the results of the optimization of the linear dampers as well as the creation of the root locus figures for each of the models.

5.1.1 2DOF Half-Car Roll Model

Initially, a sinusoid with low input frequency was used to drive the half-car roll model. The optimization routine made little change to the damping coefficient to minimize the lateral load transfer or weight transfer norm ($||\Delta W||$). After examining the natural frequencies of the model it was decided to use an input frequency of 1 rad/s as it is closer to the roll mode natural frequency of the system. The optimization was then able to change the damping coefficient further. It was found that the optimization required the system to have sufficient excitation by having a forcing frequency near a natural frequency to make a substantial change in the initial damping. The results in Figure 5.1 used the 4.33 rad/s spatial frequency found from the FSAE rules for a slalom and an amplitude equivalent to 1.6 g's from the optimized tire data of Figure 4.11. Figure 5.1 shows the tire load on the left and right tires in the top subplot. The optimization had the objective of minimizing the difference between these two curves. As expected, the chassis heave subplot shows no motion since the model was driven with a roll moment and the modes are uncoupled as seen in Table 4.2.

The optimal linear damping rate for the half-car model was found to be 21.6 lb-s/in. This value is much lower than the initial 164 lb-s/in from the FSAE damper so it is likely that the model is over-simplified using the total vehicle mass as the sprung mass. The combination of high mass and low stiffness creates a chassis roll natural frequency lower than that seen in the more complex models so less damping is required to achieve a high or moderate damping ratio.

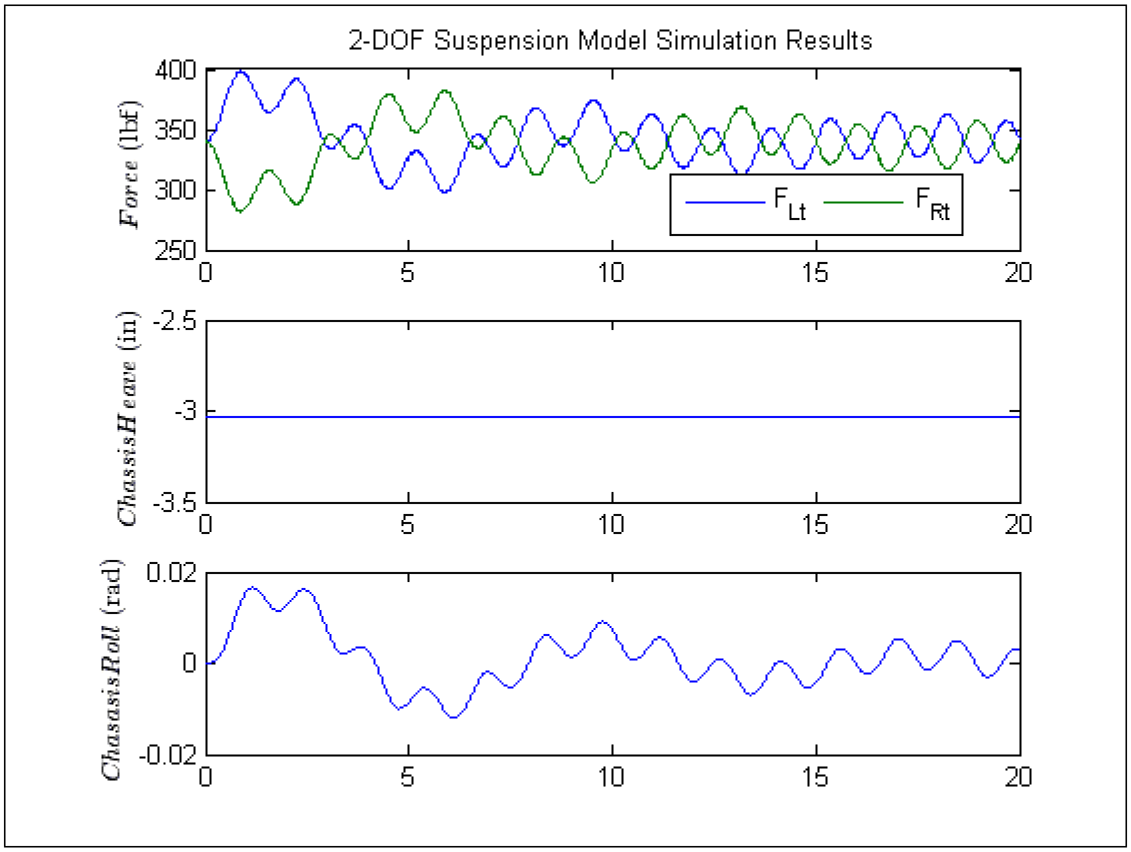

Figure 5.1 – 2DOF Roll Optimized Results

By inspection of Figure 5.2, the heave mode (branch at 3.2554 rad/s) is more sensitive to changes in the damping coefficient than the roll mode (branch at 0.80877 rad/s). This is to be expected as the vehicle model does not have a damper specifically for roll, but for the heave motion. The roll mode shows a very low damping ratio while the heave mode has a significantly higher damping ratio. There is no significance to the high damping ratio on the heave mode since there is no heave input or output.

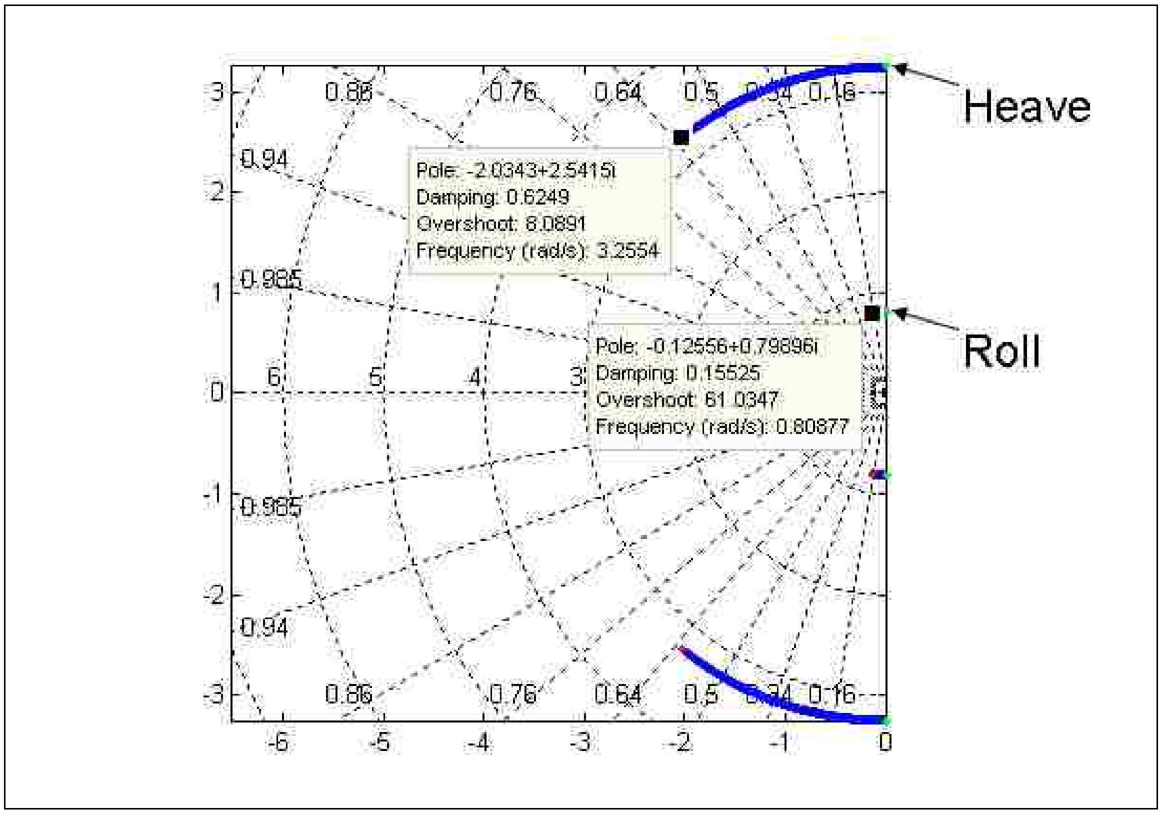

Figure 5.2 – 2DOF Roll Root Locus

5.1.2 4DOF Half-Car Roll Model

The half-car model simulation used roll input taken from the data acquisition system scaled to represent the FSAE vehicle parameters and uniform random noise to represent road input. The results of the optimization at each iteration (n) are shown in Table 5.1. The table includes the initial damping value (C_0), the optimized damping value (C_{opt}) and the weight transfer norm ($||\Delta W||$) after optimization. After several iterations of varying the initial value, the damping coefficient converged to 330 lb-s/ft. The optimal linear damping coefficient of the 4DOF half-car is more reasonable than that of the 2DOF roll model since it is closer to the low-speed damping of the 2007 FSAE vehicle.

Table 5.1 – 4DOF Initial Damping and Optimization Results

n	C_0 (lbf-s/in)	C_{opt} (lbf-s/in)	$\|\|\Delta W\|\|$ (lbf)
1	164	174	103980
2	300	310	99329
3	600	584	103890
4	500	468	100310
5	400	330	99253
6	320	330	99253

Figure 5.3 shows the vertical load on the left and right tires in the top subplot; the difference of which is the objective function for the optimization. The middle subplot shows the chassis heave motion has a decaying sinusoid indicating the heave mode is likely underdamped. The bottom subplot shows the wheel motion in inches. This subplot was expected to have ±1 inch of wheel travel since the suspension was designed by UNM FSAE to do so. The wheel motion in Figure 5.3 shows that a range of roughly ±1 inch of travel occurred in the simulation, further validating the results of the simulation.

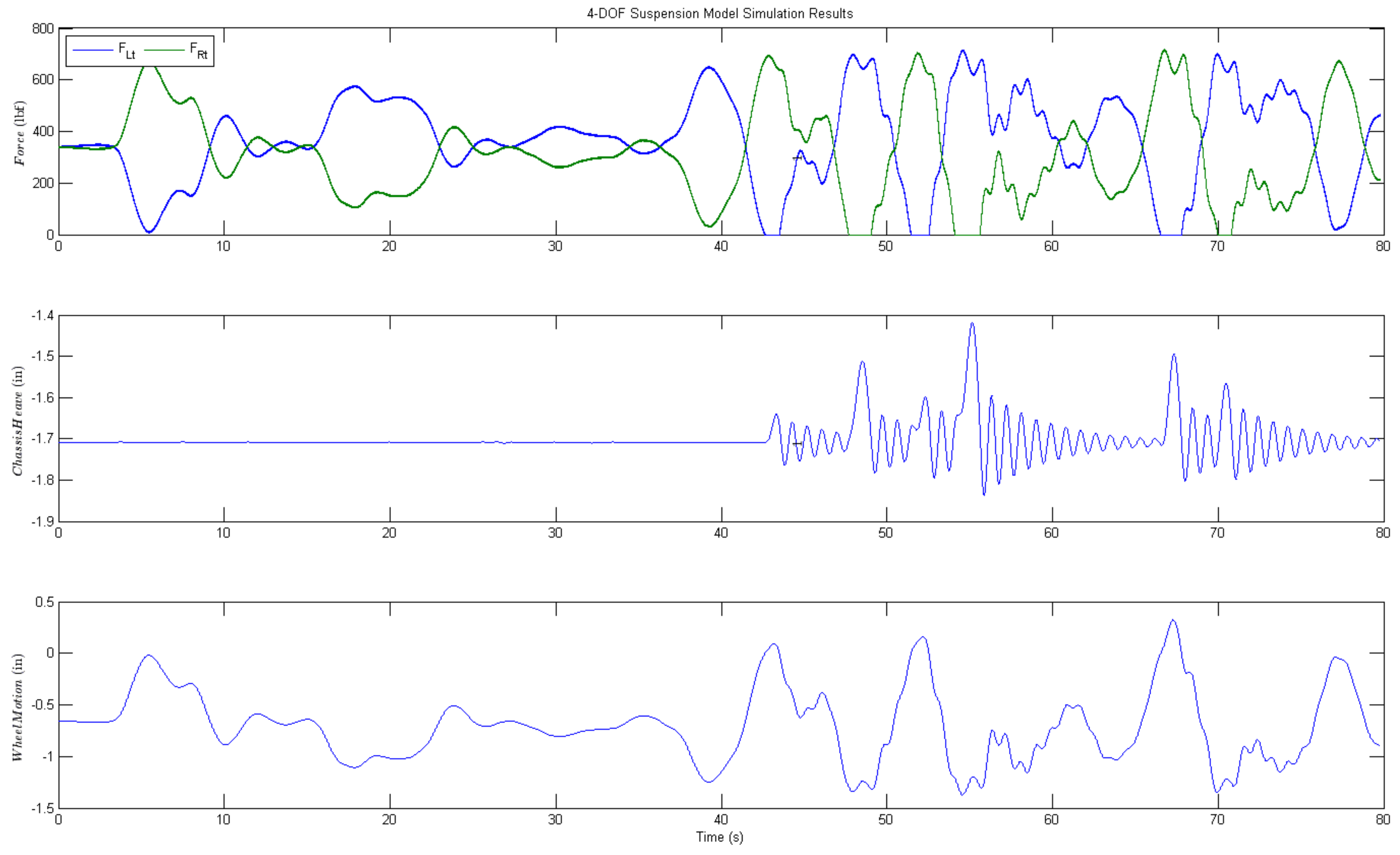

Figure 5.3 – 4DOF Optimized Behavior

Figure 5.4 shows the root locus after optimization. No information is readily available from this figure until the data cursors were placed on the branches of the root locus. Figure 5.5 shows the data cursors placed on the branches at the imaginary axis of the root locus simply for the purpose of identifying the associated mode shapes from Table 4.4.

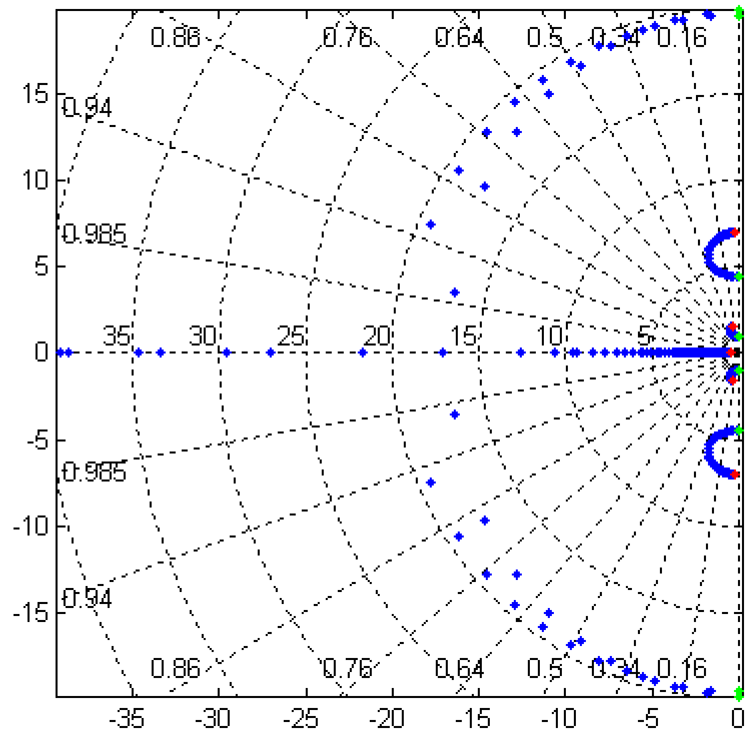

Figure 5.4 – 4DOF Root Locus

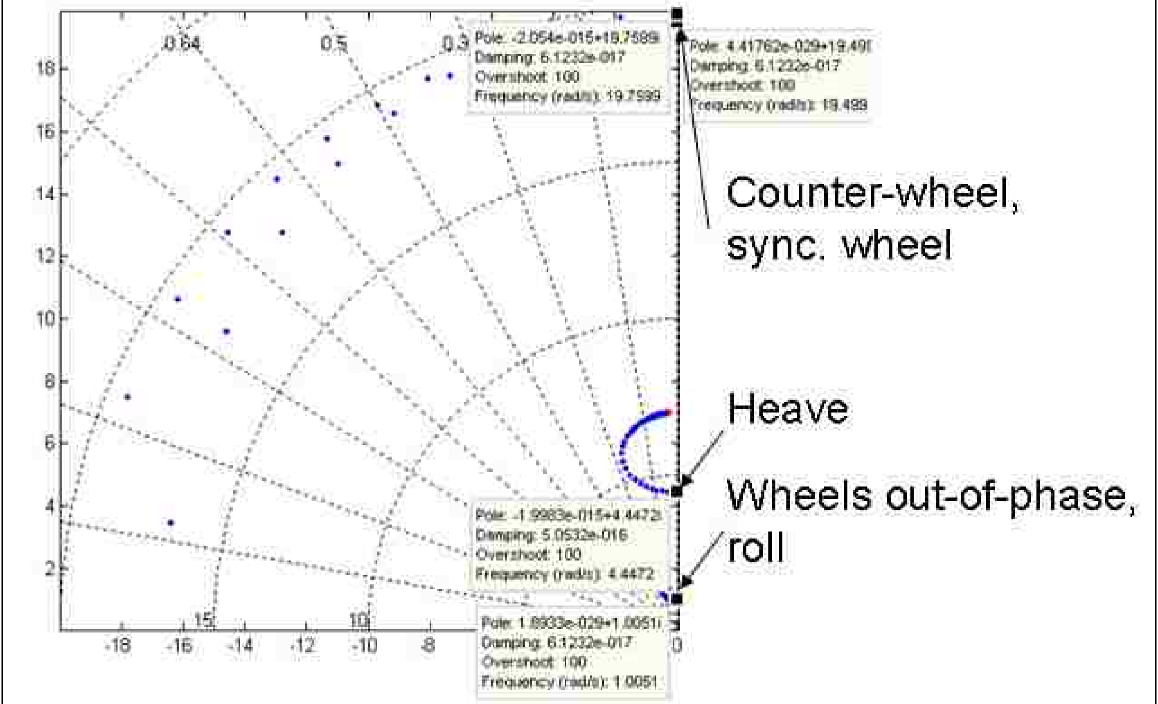

Figure 5.5 – 4DOF Root Locus / Mode Correspondance

The data cursors in Figure 5.6 show the pole, damping ratio, percent overshoot and natural frequency based on the selected eigenvalues. Figure 5.6 shows the upper-half plane of the root locus from Figure 5.4 with data cursors on the optimal gain. Inspection of Figure 5.6 verifies that the heave mode is, in fact, underdamped corresponding to the chassis heave oscillations seen in Figure 5.3. By further inspection of Figure 5.6, the optimization has given a relatively high natural frequency to the first (counter-wheel) and second (heave) modes and low natural frequency to the third (counter-wheel) and fourth modes (synchronous wheel). The third (counter-wheel) and fourth (synchronous-wheel) modes are critically damped, while the first (counter-wheel) and second (heave) modes have very little damping. The root locus has shown that the wheel motions are preferentially treated by the optimization since load transfer occurs at the wheels.

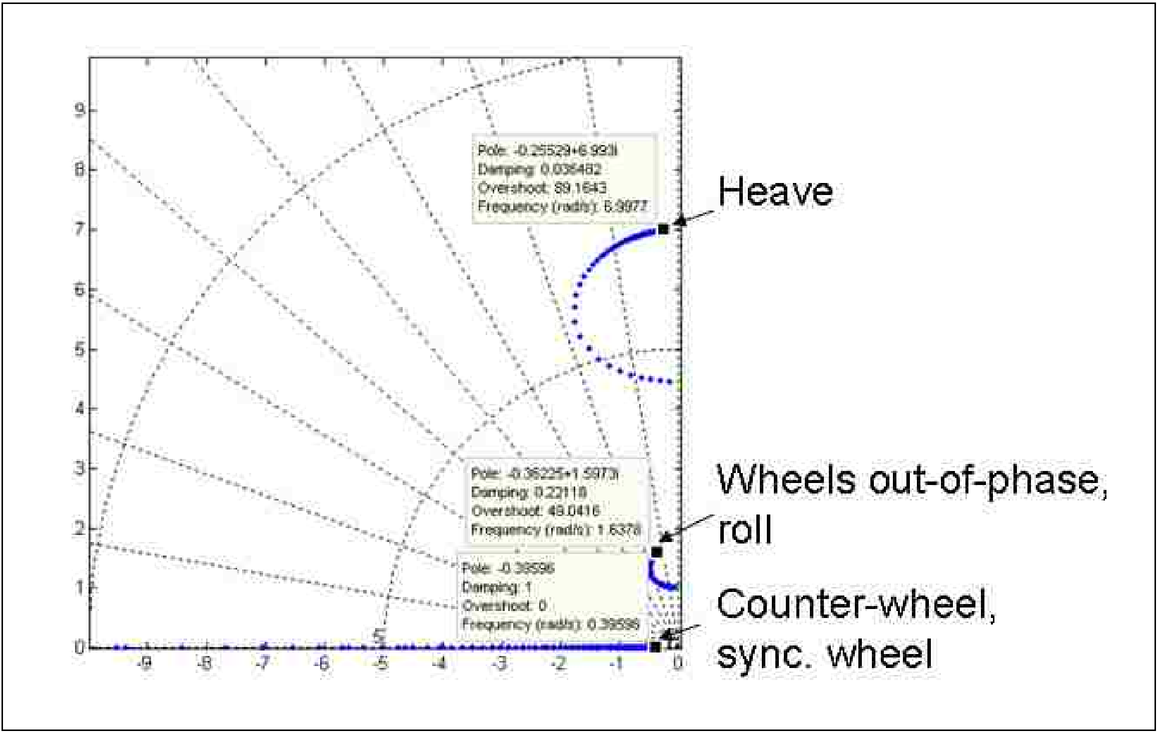

Figure 5.6 – 4DOF Root Locus with Optimal Damping Data Cursors

5.1.3 7DOF Full-Car Model

The full-car model used the scaled lateral, longitudinal and vertical accelerations from the data acquisition system to drive the simulation. Road noise was again modeled as uniform random noise with a 0.005 in amplitude. The optimization for the full-car linear damper to minimize weight transfer was conducted first to provide a closer estimate of the nonlinear force-velocity curve needed. This reduces the number of iterations required in the nonlinear case as the initial guess is closer to the optimal value.

In a similar fashion to the method used in the 4DOF optimization, an iterative approach to the initial linear damping value was used to converge on the global minimum. However, because front and rear damper values are allowed to be independent, the front and rear installation ratios from section 4.3.2 were used to create a front/rear distribution for the initial damping. The installation ratio (or velocity ratio) is the mechanical advantage from the suspension geometry that relates the velocity of the wheel to the velocity of the damper. The installation ratio creates an effective, or "installed," damper rate that is observed vertically at the wheel center. The installation ratio for the front of the car is larger than that in the rear so a front/rear distribution was created. Using this distribution places the relative values nearer their optimal positions and requires fewer iterations to find the optimum. The installed damping rate was found by use of Eq. 5.1.

$$C_w = C \cdot IR^2 \qquad \text{Eq. 5.1}$$

A better approach to create a front/rear distribution for the dampers would have been to use the roll rate distribution of the vehicle. The roll rate distribution is determined by the suspension designer and used to place a front/rear distribution of the installed spring rates on a car. The same method applied to the dampers would have been more logical but was over-looked at the time.

The optimization routine was then used to minimize the total lateral load transfer. The results of the optimization at each iteration (n) are shown in Table 5.2. The table includes the initial front damping (C_{f0}), the initial rear damping (C_{r0}), the initial installed front damping rate (C_{wf0}), initial installed rear damping rate (C_{wr0}) and optimized weight transfer norm ($||\Delta W||$). The optimized damping values were C_f = 426 lb-s/in and C_r = 239 lb-s/in shown in Table 5.2.

Table 5.2 – Full-Car Linear Damper Optimization Results

	Initial Values				Optimized Values		
n	C_{f0} (lbf-s/in)	C_{r0} (lbf-s/in)	C_{wf0} (lbf-s/in)	C_{wr0} (lbf-s/in)	$C_{wf\ Opt}$ (lbf-s/in)	$C_{wr\ Opt}$ (lbf-s/in)	$\|\|\Delta W\|\|$ (lbf)
1	164	164	97	48	452	261	90328
2	1000	1000	593	292	430	284	90377
3	500	500	296	146	302	154	91520
4	750	750	445	219	426	229	90299
5	825	825	489	241	412	243	90297
6	720	820	427	239	426	239	90294

It should be noted that the front installed spring rate is higher than the rear installed spring rate and the vehicle weight is distributed toward the rear (i.e. the weight is higher in the rear than the front). Considering the combination of these two, a higher damping rate is expected in the front than in the rear. This was the result at which the optimization arrived and explains the large disparity between the front and rear optimized damping values.

Figure 5.7 shows the results of the chassis response to the scaled lateral, longitudinal and vertical acceleration using the optimized linear damper. The vertical motion of the chassis is not as easily identified as underdamped, critically damped or overdamped by the oscillations as in the 4DOF since the full-car used vertical accelerations to force the motion. However, the maximum amplitude of the vertical motion appears to be fairly small so the heave mode of the full-car is probably closer to critically damped than the 4DOF half-car. The roll and pitch motions do no convey any information on what the damping ratio might be since they are also forced by acceleration input.

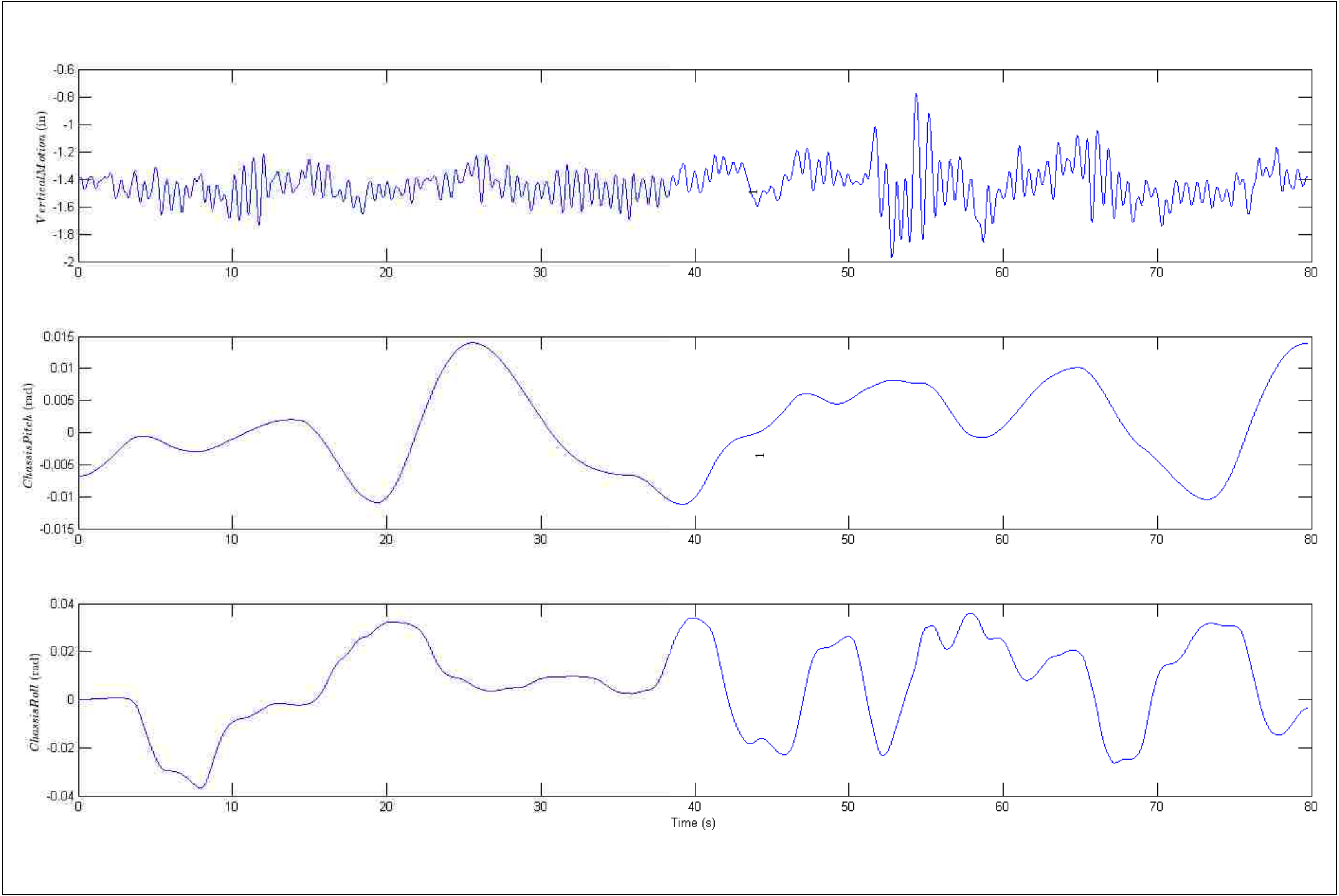

Figure 5.7 – 7DOF Chassis Response with Optimal Linear Damping

Figure 5.8 shows the root locus after optimization. No information is readily available from this figure until data cursors were placed on the branches of the root locus. Figure 5.9 shows the data cursors placed on the branches at the imaginary axis simply for the purpose of identifying the associated mode shapes from Table 4.5. The mode shapes are labeled by number in Figure 5.9 and are listed again here:

1: Front wheels out of phase with rear wheels, heave, pitch

2: Left wheels out of phase with right wheels, roll

3: Heave

4: Left rear wheels out of phase with right rear wheels

5: Left rear wheels in phase with right rear wheels

6: Left front wheels out of phase with right front wheels

7: Left front wheels in phase with right front wheels

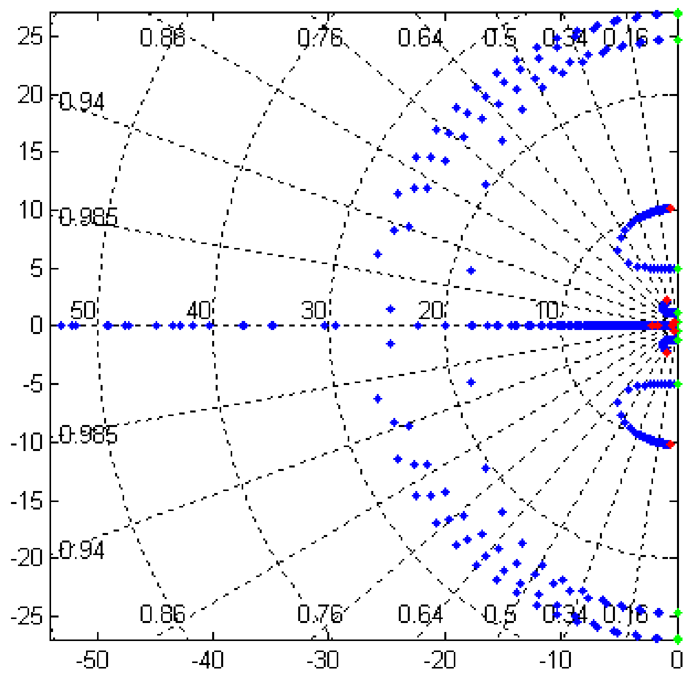

Figure 5.8 – 7DOF Linear Damper Root Locus

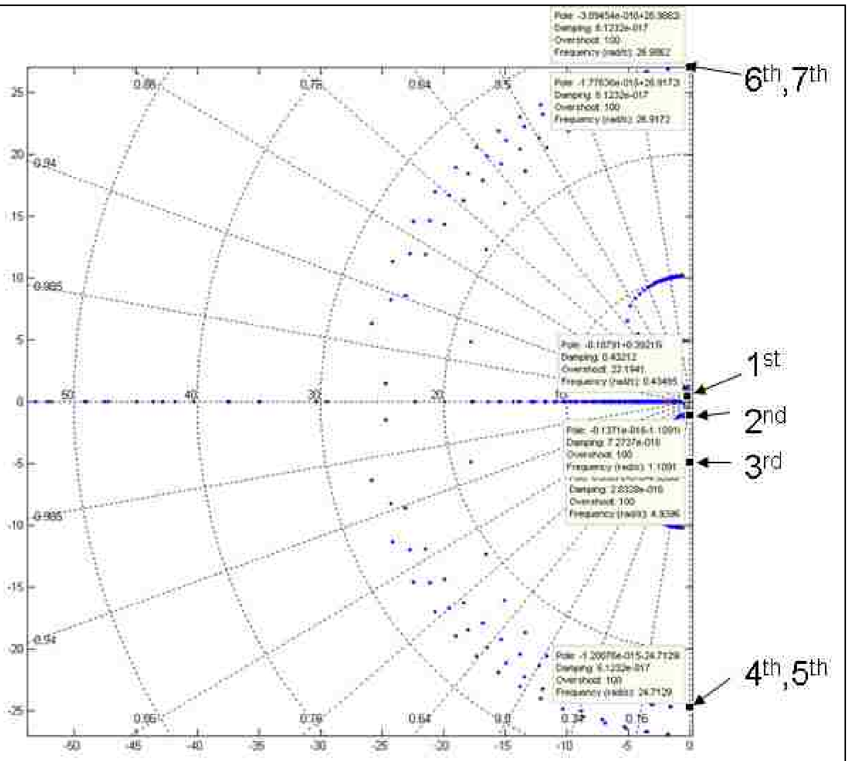

Figure 5.9 – 7DOF Root Locus / Mode Correspondance

The data cursors in Figure 5.10 and Figure 5.11 show the pole, damping ratio, percent overshoot and natural frequency based on the selected eigenvalues. Figure 5.10 and Figure 5.11 show the upper-half plane of the root locus from Figure 5.8 with data cursors on the optimal gain.

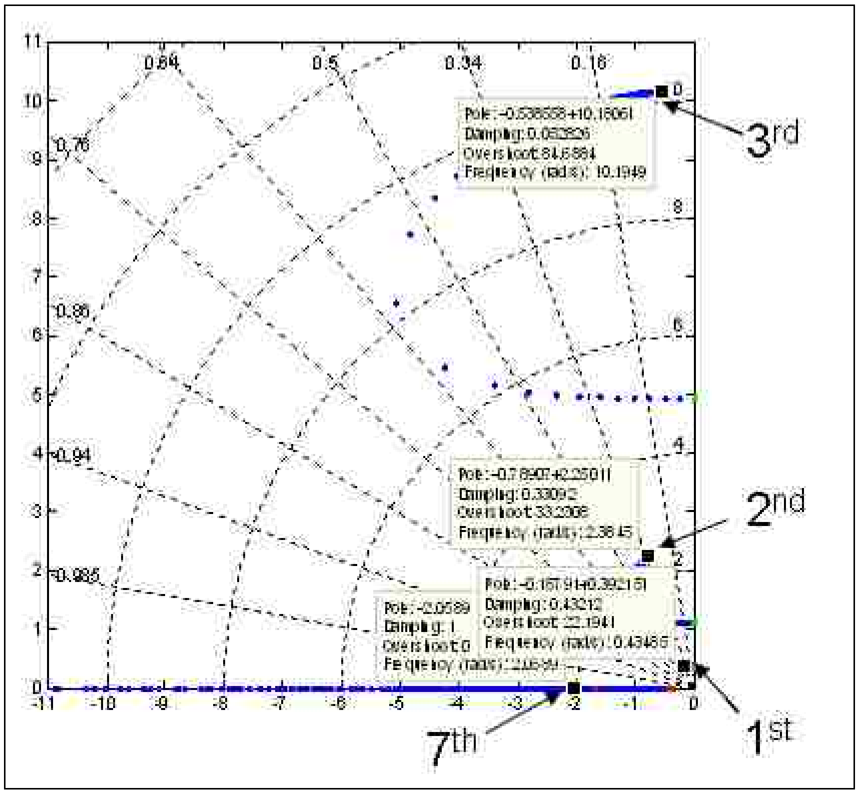

Figure 5.10 – 7DOF Root Locus with Optimal Damping Data Cursors

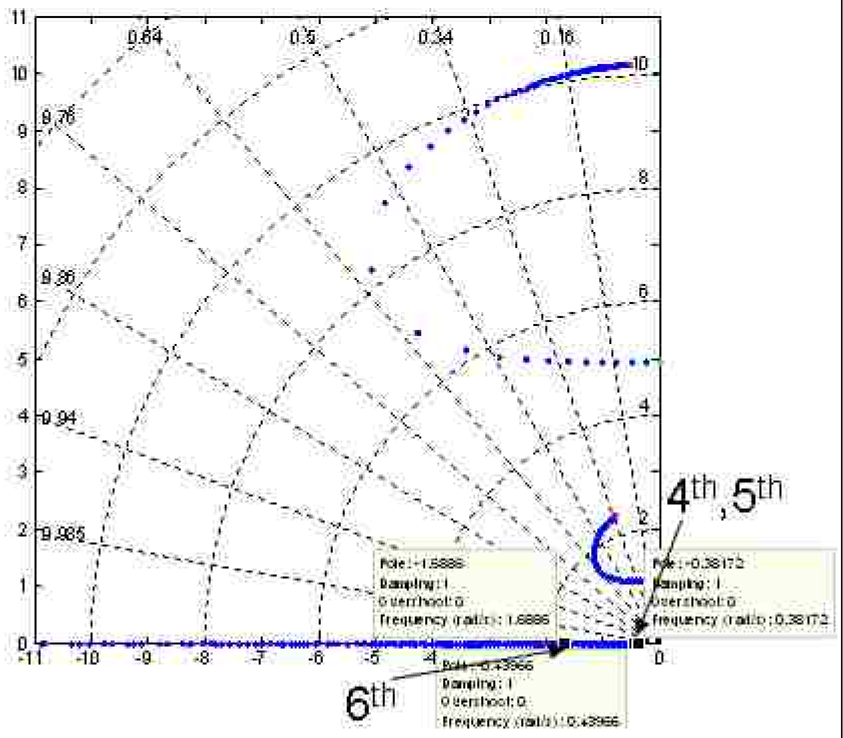

Figure 5.11 – 7DOF Root Locus with Optimal Damping Data Cursors

By inspection of Figure 5.10 and Figure 5.11, the optimization has given a relatively high natural frequency to the second mode (left wheels opposite right wheels) and third mode (heave) and a relatively low natural frequency to the first mode (front wheels opposite rear wheels, heave). Further investigation shows the fourth mode (left-rear wheel opposite right-rear wheel) and fifth mode (left-rear wheel in-phase with right-rear wheel) are the eigenvalues close to the origin and thus have low natural frequencies. The sixth mode (left-front wheel opposite right-front wheel) and seventh mode (left-front wheel synchronous with right-front wheel) have relatively high natural frequencies as well. The last four modes are critically damped, while the first three have very little damping. The root locus has shown that the wheel motions are preferentially treated by the optimization since load transfer occurs at the wheels where the objective function was placed. The road race car suspension should be designed around controlling wheel motions since the tires are crucial to maximizing lateral acceleration.

The nonlinear damper should further reduce lateral load transfer; however, the root locus is not a usable tool for analyzing the optimization results since the root locus technique is only applicable to linear models. Section 5.2 presents the results of the nonlinear damper for the full-car model.

5.2 Nonlinear Damper Optimization

This section presents the results of the optimization of the nonlinear dampers. The nonlinear damper has three parameters associated with the force-velocity curve to be optimized as opposed to the single coefficient for the linear case. These nonlinear parameters are the low-speed damping (C_1), high-speed damping (C_2) and transition velocity (V_t) which can be seen in Figure 5.12. As mentioned previously, the root locus technique cannot be applied to nonlinear models.

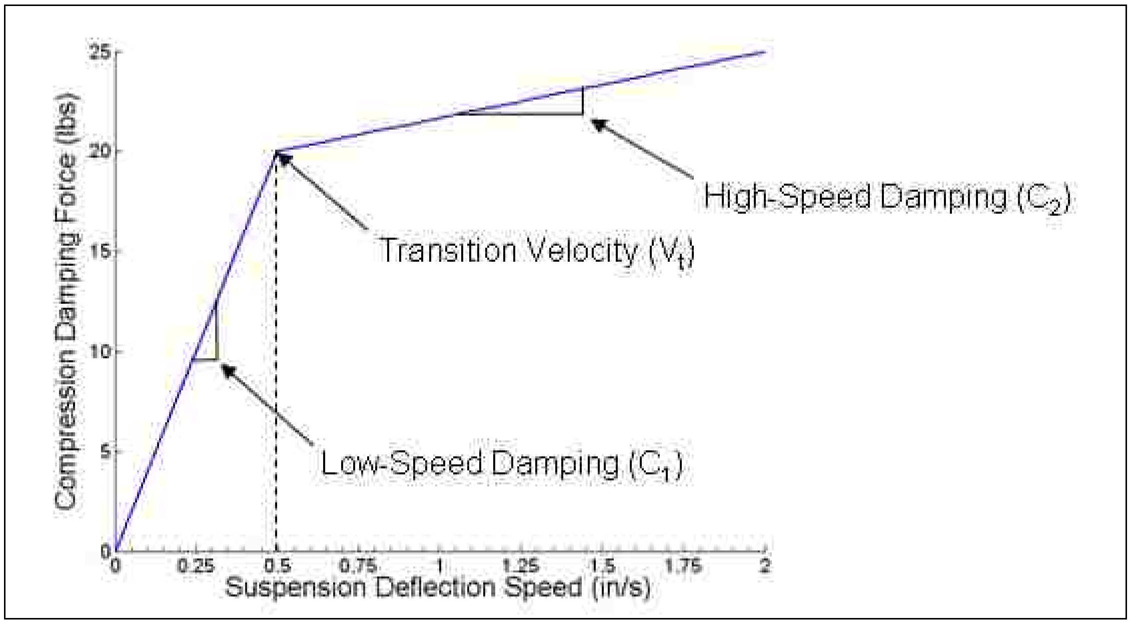

Figure 5.12 – Nonlinear Damper Force-Velocity Curve

Due to the number of variables significantly increasing for the nonlinear damper, two different approaches were explored. The first approach was to optimize the three variables associated with the nonlinear damper which resulted in a total of six variables being optimized; three for the front dampers and three for the rear dampers. The second approach was to choose a constant transition velocity reducing the number of variables to four.

5.2.1 Six Variable Nonlinear Damper Optimization

The initial values of the low-speed damping for the front (C_{1f0}) and rear (C_{1r0}) suspension were chosen as the optimized linear values; $C_{f\,Opt}$ = 412 lbf-s/in and $C_{r\,Opt}$ = 243 lbf-s/in. At each iteration (n), the optimization changed the damping values in a direction from their initial value. The initial value in the next iteration was then slightly modified in the direction of the previous optimized value as they should be near the optima already. The initial values for the high-speed damping for the front (C_{2f0}) and rear (C_{2r0}) were chosen as the same respective values for low-speed damping. This implies the initial force-velocity curve was linear. The initial transition velocity (V_{t0}) was chosen from the UNM FSAE vehicle parameters. The optimized parameters are the front low-speed damping (C_{1fOpt}), front high-speed damping (C_{2fOpt}), front transition velocity (V_{tfOpt}), rear low-speed damping (C_{1rOpt}), rear high-speed damping (C_{2rOpt}) and rear transition velocity (V_{trOpt}). Table 5.3 shows the results of each iteration of optimizing the six-variable nonlinear damper.

Table 5.3 – Full-Car Six Variable Nonlinear Damper Optimization Results

	Initial Values					Optimized Values						
n	C_{1f0} (lbf-s/in)	C_{2f0} (lbf-s/in)	C_{1r0} (lbf-s/in)	C_{2r0} (lbf-s/in)	V_{t0} (in/s)	$\|\|\Delta W\|\|$ (lbf)	C_{1fOpt} (lbf-s/in)	C_{2fOpt} (lbf-s/in)	V_{tfOpt} (in/s)	C_{1rOpt} (lbf-s/in)	C_{2rOpt} (lbf-s/in)	V_{trOpt} (in/s)
1	301	301	154	154	0.25	91372	307	302	0.25	161	158	0.25
2	412	412	243	243	0.25	90256	417	408	0.25	248	237	0.25
3	426	426	229	229	0.25	90257	430	422	0.25	236	224	0.25
4	430	430	284	284	0.25	90340	433	426	0.25	284	275	0.25
5	416	100	248	100	0.75	89856	416	100	0.62	248	100	0.67
6	416	150	248	150	0.75	90022	416	150	0.61	248	150	0.68
7	416	75	248	75	0.75	89745	416	75	0.62	248	75	0.67

The weight transfer norm of the optimal nonlinear damper shows improvement over the optimal linear damper. This is to be expected as nonlinearity generally improves system behavior. The high-speed damping was insensitive to the optimization for all iterations. C_{2f0} and C_{2r0} should have been further decreased at iteration eight as convergence had yet to be achieved. The transition velocity was also insensitive to optimization until it was later changed to a larger value.

The optimization results seemed to be sensitive to the initial low-speed and high-speed damping of the force-velocity curve. Along with the high number of variables, the first few iterations did not accurately represent the nonlinear damper of the UNM FSAE vehicle since the initial force-velocity curve was linear resulting in poor optimization performance. The noise in the input data also creates many local minima which the optimization finds. Experimenting with the initial conditions seemed to be the only way to decrease the weight transfer norm, defeating the purpose of using optimization.

5.2.2 Four Variable Nonlinear Damper Optimization

The four variable nonlinear damper used a constant transition velocity to reduce the number of variables for the optimization. Several values for the front (V_{tf}) and rear (V_{tr}) transition velocity were examined in order to determine that which would minimize load transfer, including the 0.25 in/s transition velocity from the UNM FSAE damper data shown in Figure 4.10. In the last iteration, the value was changed to the optimized value from the six variable optimization case. The optimal nonlinear damper for the front suspension had a low-speed damping rate of 417 lb-s/in and a high-speed damping rate of 4 lb-s/in. The rear suspension had an optimal low-speed damping rate of 256 lb-s/in and high-speed damping rate of 8 lb-s/in.

Table 5.4 – Full-Car Four Variable Nonlinear Damper Optimization Results

	Initial Values						Optimized Values				
n	C_{1f0} (lbf-s/in)	C_{2f0} (lbf-s/in)	C_{1r0} (lbf-s/in)	C_{2r0} (lbf-s/in)	V_{tf} (in/s)	V_{tr} (in/s)	$\|\|\Delta W\|\|$ (lbf)	C_{1fOpt} (lbf-s/in)	C_{2fOpt} (lbf-s/in)	C_{1rOpt} (lbf-s/in)	C_{2rOpt} (lbf-s/in)
1	412	412	243	243	0.75	0.75	90289	417	409	247	236
2	412	412	243	243	0.5	0.5	90276	416	409	249	236
3	412	412	243	243	0.25	0.25	90256	417	408	248	237
4	412	412	243	243	0.125	0.125	90267	418	408	248	238
5	412	412	243	243	0.0625	0.0625	90281	418	409	248	238
6	416	400	248	200	0.25	0.25	90123	421	396	255	196
7	416	375	248	175	0.25	0.25	90050	421	372	256	174
8	416	20	248	20	0.25	0.25	97835	421	25	253	25
9	416	100	248	100	0.25	0.25	90618	421	105	253	105
10	416	100	248	100	0.75	0.75	89987	402	92	248	94
11	416	75	248	75	0.62	0.67	89712	416	69	250	68
12	416	50	248	50	0.62	0.67	89568	416	44	252	43
13	416	25	248	25	0.62	0.67	89408	416	19	254	20
14	416	10	248	10	0.62	0.67	89294	417	4	256	8

The result of the four-variable damper optimization was an improvement over the six-variable optimization. The optimization routine seemed to find several local minima due to the input noise as it took several iterations to find a global minimum. The four-variable nonlinear damper case further reduced the weight transfer from the six-variable case.

Figure 5.13 shows the chassis heave, pitch and roll of the vehicle with the optimal nonlinear damper. No conclusions can be drawn from this figure since all the motions are forced.

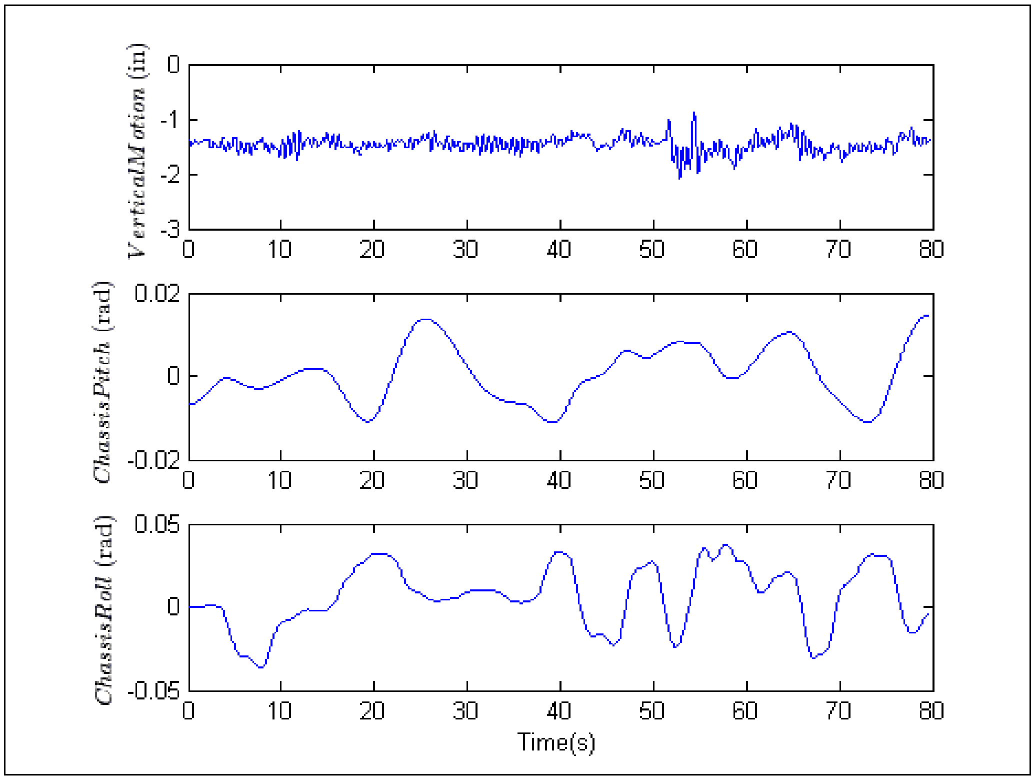

Figure 5.13 – 7DOF Chassis Response with Optimal Nonlinear Damping

Figure 5.14 shows the wheel motions with the optimal nonlinear damper. The wheels have relatively low amplitude implying that they are controlled well by the suspension. A spike in all wheel motions is evident at approximately 55 seconds, corresponding to a spike in the chassis heave motion in Figure 5.13.

Figure 5.15 shows the load on the left tires and right tires in the top subplot. Frequently the force of the right and left tires drops to zero indicating the wheels are lifting off the ground. The front inner wheel of the 2007 UNM FSAE vehicle would also lift in hard turns, indicating that the simulation is a reasonable representation of the actual vehicle. The bottom subplot displays the difference in the force of the left and right tires which is the total lateral load transfer or objective function. The maximum lateral load transfer also occurs at approximately 55 seconds corresponding to the spikes in the wheel and chassis heave motions.

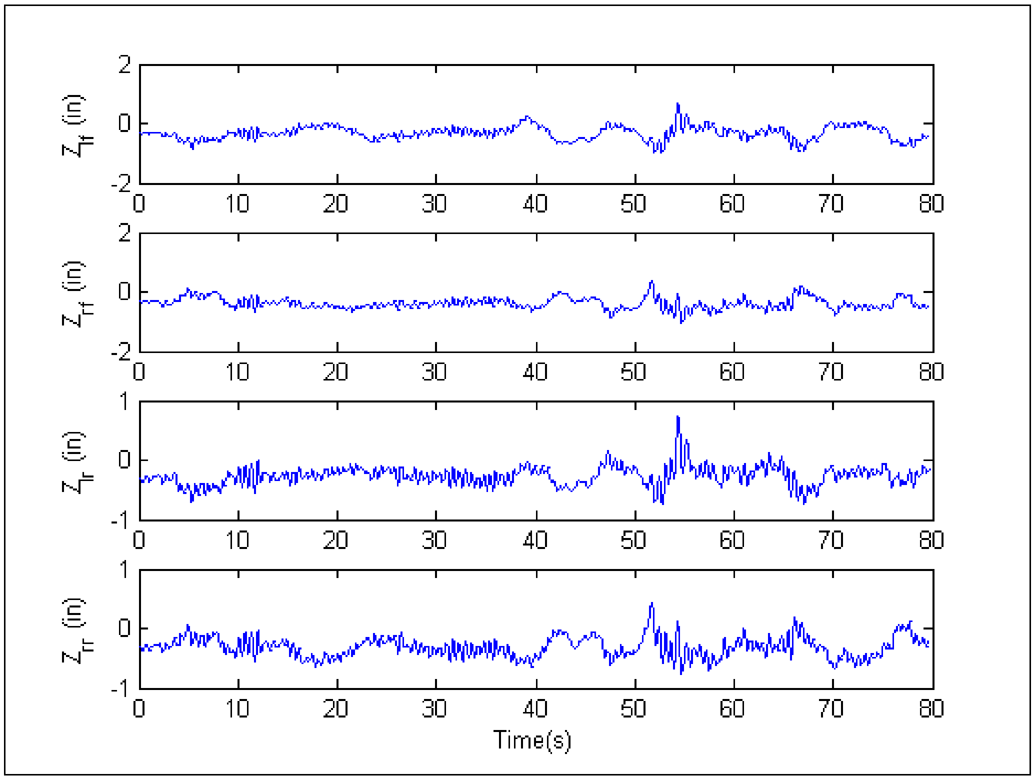

Figure 5.14 – 7DOF Wheel Motions with Optimal Nonlinear Damping

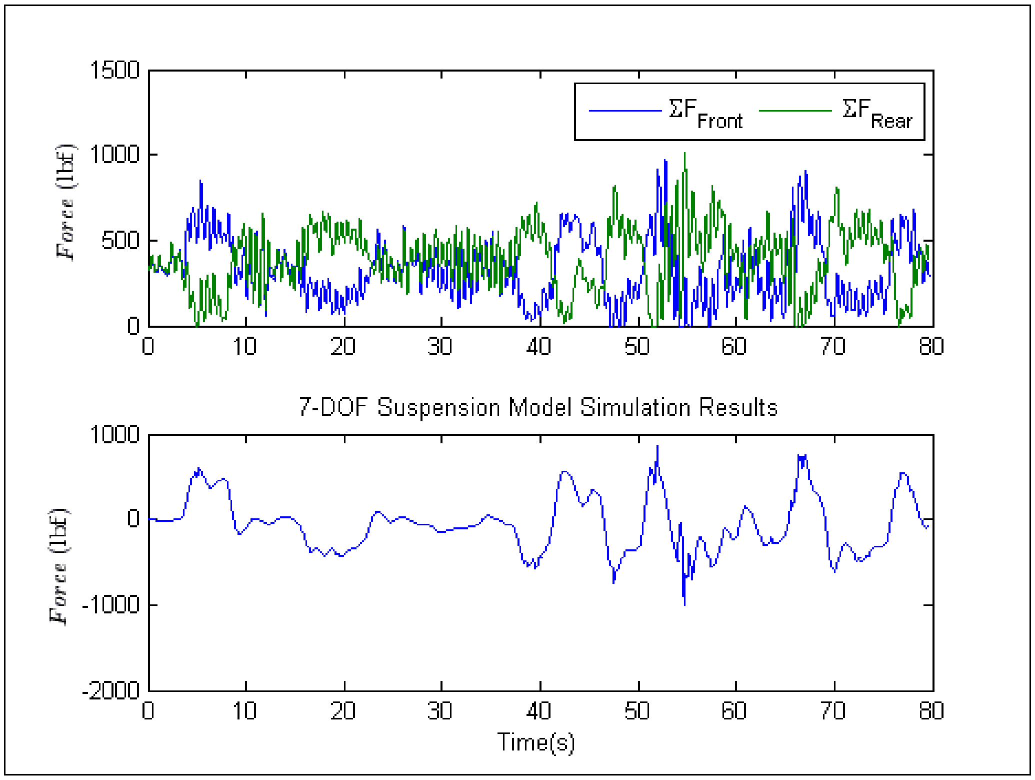

Figure 5.15 – 7DOF Objective Function with Nonlinear Damping

5.3 Inerter Optimization

As mentioned in assumption 5, the inerter was optimized separately from the damper. This is mainly due to the large number of variables that would over-complicate the process, although it is also true that an inerter is not available to UNM FSAE and is not currently used in their vehicle design. The root locus was not able to be created in this case since the nonlinear damper was employed. Table 5.5 contains the results of the inerter optimization. The initial front inertance (b_{f0}) and rear inertance (b_{r0}) were chosen at random for the first iteration as no information is available for a FSAE vehicle inerter. The optimal front inertance (b_{fOpt}) was 1.4 slugs and optimal rear inertance (b_{rOpt}) was 6.5 slugs.

Table 5.5 – Full-Car Inerter Optimization Results

b_{f0}	b_{r0}	$\|\|\Delta W\|\|$	b_{fOpt}	b_{rOpt}
2.5	2.5	89049	6.2	2.8
10	10	89039	10	7.5
50	50	92297	45	56
25	25	89167	26	18
18	13	88956	3.6	33
4.5	20	88996	1.5	21
4.5	26	88991	1.4	6.5

The inerter further decreased the weight transfer norm from the nonlinear damper case but not to the degree that the nonlinear damper decreased the weight transfer norm from the linear damper case. The inerter optimization was unable to converge to a global optimum as the damper optimizations had. However, the weight transfer norm changed very little in the last iteration. Figure 5.16 shows the chassis roll, pitch and heave motion of the vehicle with the optimal inerters. Figure 5.17 shows the wheel motions with the optimal inerters. The same spike in wheel motions noticed in the nonlinear damper case is noticed in the wheel motion at approximately 55 seconds corresponding to the spike in chassis heave at the same time. Figure 5.18 shows the load on the left tires and right tires in the top subplot and the total lateral load transfer or objective function. The maximum lateral load transfer is again seen at approximately 55 seconds.

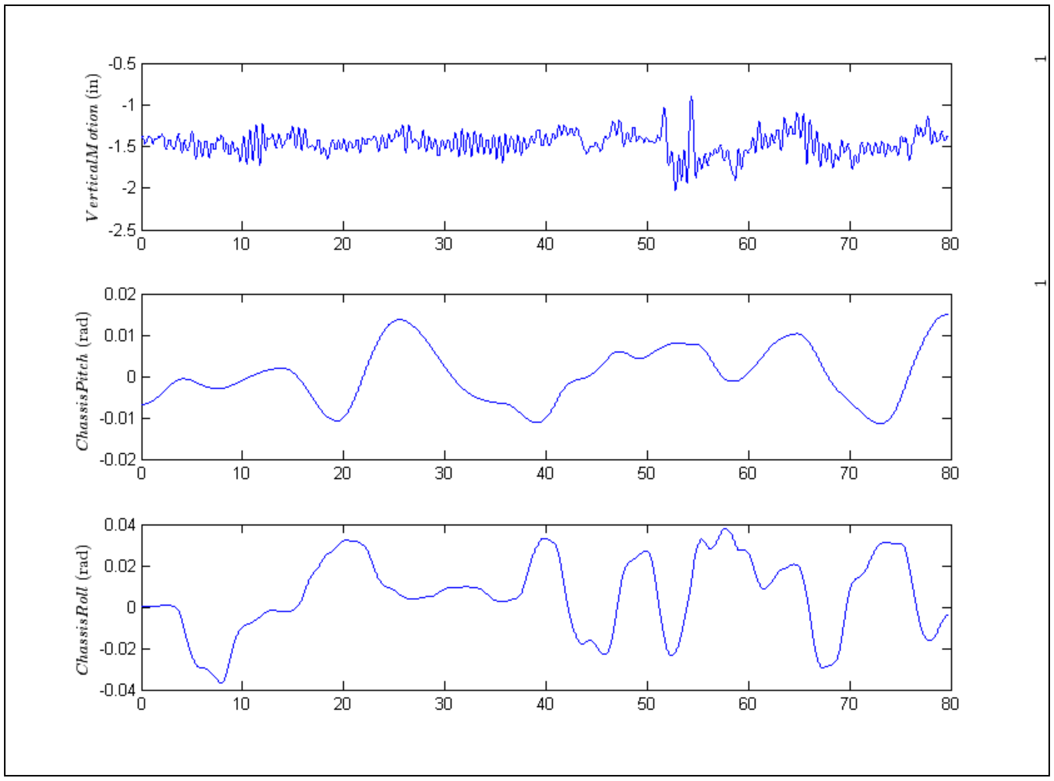

Figure 5.16 – 7DOF Chassis Response with Optimal Inerter and Nonlinear Damping

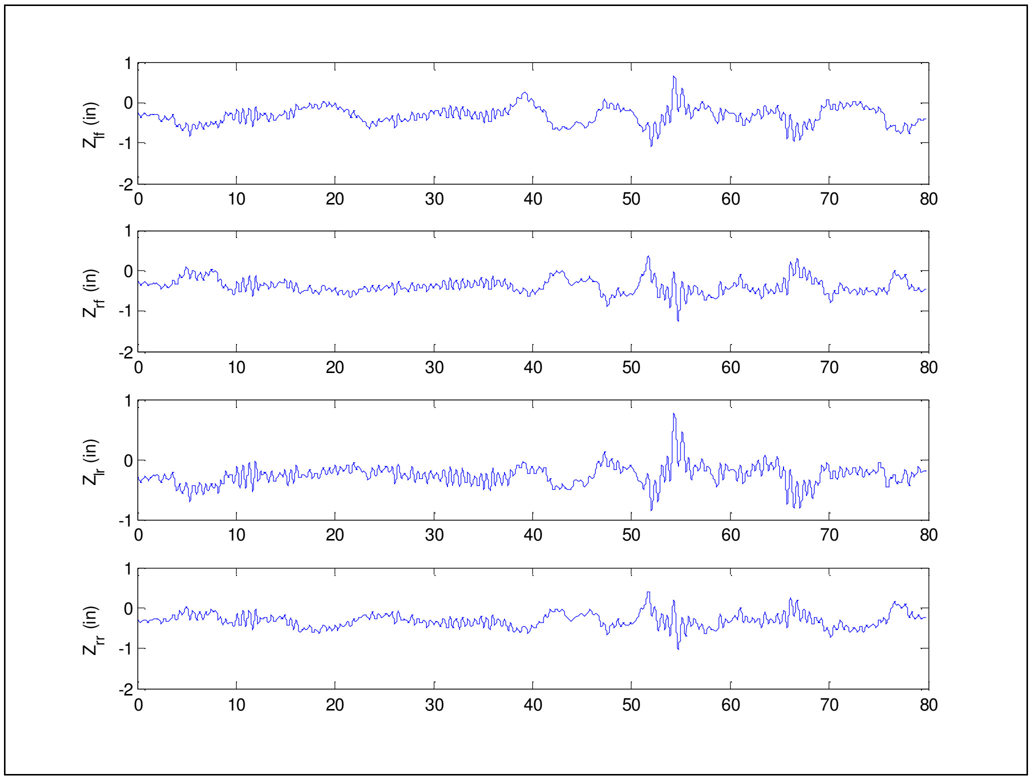

Figure 5.17 – 7DOF Wheel Motions with Optimal Inerter and Nonlinear Damping

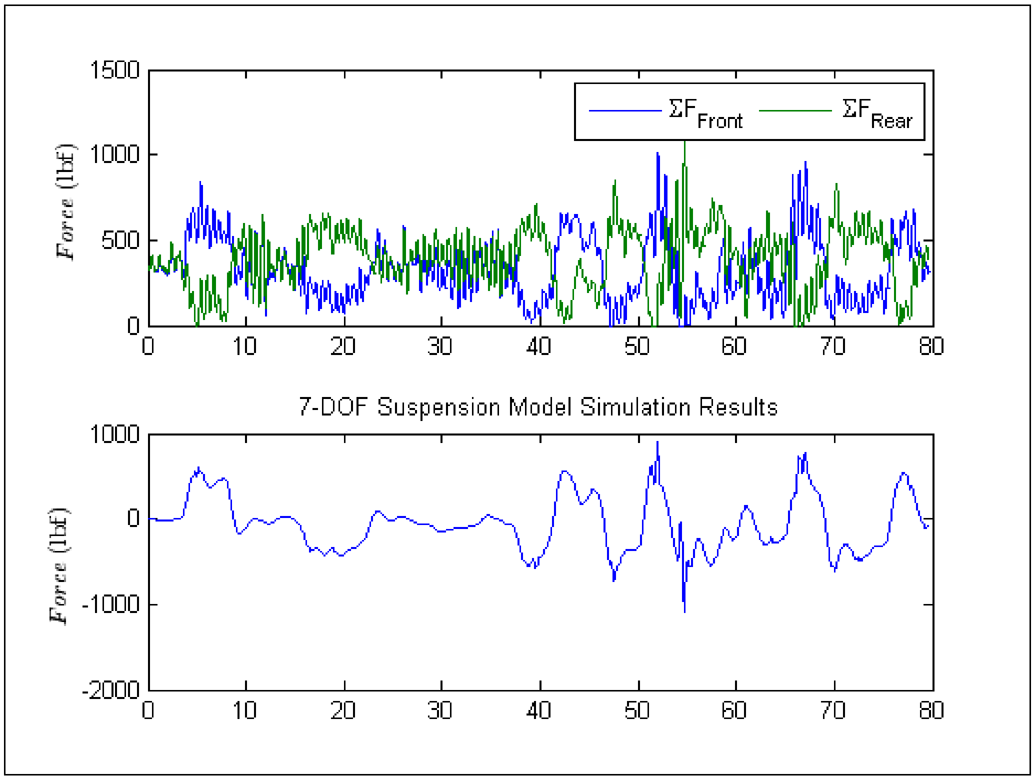

Figure 5.18 – 7DOF Objective Function with Optimal Inerter and Nonlinear Damping

The reason for the convergence problem is apparent in observance of the left-front suspension relative velocity and acceleration in Figure 5.19. Part of the problem with the optimization of the inerter is a spike in the input data which is exacerbated through the suspension. This spike is not as prominent in the suspension velocity as shown in Figure 5.19, but was still noticeable in the simulation results of the nonlinear damper. This is probably the cause of the convergence problems for the inerter optimization.

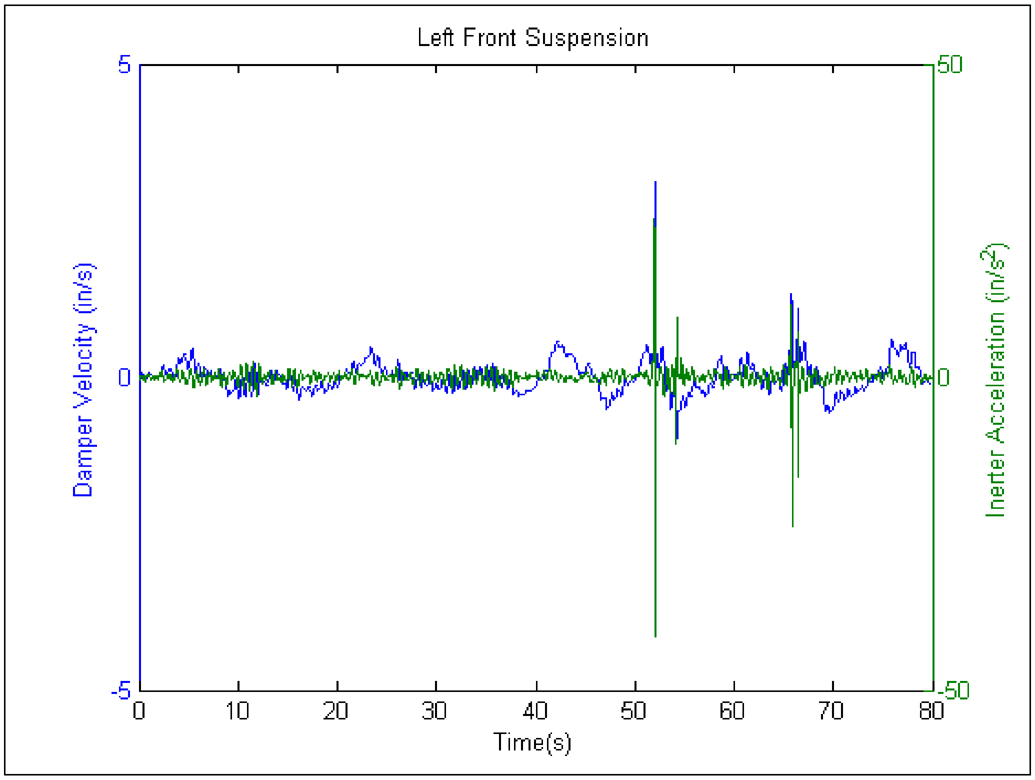

Figure 5.19 – 7DOF Left Front Suspension Relative Velocity/Acceleration

5.4 Results Chapter Recapitulation

The results of the optimizations of the linear and nonlinear dampers converged for each model with exception to the six-variable design. The optimization of the inerter did not converge, likely due to a spike in the input data. Convergence for the full-car linear and nonlinear damper took several more iterations than the simpler models.

The root locus has shown that the optimization preferentially treats the wheel modes since the 4DOF and 7DOF linear models had critically damped wheel modes and the objective function for the models was placed on lateral load transfer at the tires.

6 CONCLUSIONS

The goal of the damper design methodology was to find an optimal nonlinear damper for a conceptual car. The method required the obtainment of an objective function which would maximize the lateral acceleration of a road race vehicle, the input from a road race vehicle to excite the modes of the vehicle and the optimization of the nonlinear damper and inerter to minimize the objective function. Understanding the results of the optimization was accomplished through inspection of the root locus plots.

The damper design methodology arrived at an optimal solution for each case, minimizing lateral load transfer. The 4DOF and 7DOF system root loci have shown in general, the chassis heave mode frequency has increased along with a lateral, opposite wheel motion. The chassis roll mode frequency was relatively low compared to the other modes as was the damping after the optimization; however, it seems that more emphasis was placed on a higher frequency than a higher damping ratio. The wheel motion modes decreased in natural frequency from their initial position on the imaginary axis and became critically damped. This indicates that the wheels are more sensitive to lateral load transfer than body roll.

The simpler 2DOF Roll and 4DOF models do not seem to have good agreement with the full-car model as far as the result of the optimal damper because the mode shapes for these models differ. The 2DOF used a sinusoidal input which eventually led the vehicle to steady-state by inspection of Figure 5.1. This is due to the fact that the 4.33 rad/s frequency was not close enough to the roll frequency to excite it. Comparing Table 4.3 to Table 4.4 it is possible the assumption to make the entire vehicle mass the sprung mass may have also decreased the natural frequency of the system to an unrealistic value, thus reducing the amount of damping required.

The heave motion that seems to have been excited in the 4DOF model would be unacceptable in race application. However, heave does not affect lateral load transfer so the optimization did not have reason to control it. The settling time is also very long due to the underdamped heave mode. The heave resonance seems to be drowned out by noise in the 7DOF model due to the input of vertical acceleration.

The results tables from each of the cases show that weight transfer is fairly insensitive to small changes in the damping rate or inertance. This means the actual shock absorber or inerter constructed does not have to exactly match the optimized force-velocity curve to maintain overall performance; this also demonstrates the robustness of the results of the methodology.

Although the optimal inertance seemed relatively small, its effect was quite profound in terms of weight transfer. By comparing Table 5.4 and Table 5.5 the simple linear inerter performed about 25% as well as the nonlinear damper in decreasing load transfer. The increased mass of the inerter may become a factor in determining its true performance benefits as it had to be neglected for this analysis due to a lack of available information. Packaging considerations may also eliminate the device from conception from a designer's perspective.

It was necessary to use several iterations to arrive at the global minimum weight transfer. Future optimization methodology should include a method to choose or vary the initial damping. Due to the fairly random nature of the input signals, local minima are abundant which terminates the optimization routine often. Future models may also benefit from using data smoothing techniques on the inertial input which would have reduced the acceleration spike seen in Figure 5.19. Simultaneously optimizing spring stiffness would also be beneficial to designers. However, a constraint would have to be placed on FSAE designs as the rules require ± 1 inch of wheel travel.

APPENDICES

Appendix A: Optimization Code

The optimization code and root locus plotting code is contained in this appendix

```
% function [x,fval,exitflag,output] = optimizer(x0)
% This is an auto generated M-file to do optimization with the
Optimization Toolbox.
% Optimization Toolbox is required to run this M-file.

% optimizer([2;2;109;156])
x0=[2;2;109;156];

% Start with the default options
options = optimset;
% Modify options setting
options = optimset(options,'Display' ,'off');
options = optimset(options,'PlotFcns' ,{  @optimplotx @createfigure });
options = optimset(options,'LargeScale' ,'off');
[x,fval,exitflag,output] = ...
fminsearch(@Suspension_Solver_mod,x0,options);

function stop = createfigure(x,optimValues,state)
% OPTIMPLOTFVAL Plot value of the objective function at each iteration.
%
%   STOP = OPTIMPLOTFVAL(X,OPTIMVALUES,STATE) plots OPTIMVALUES.fval.
If
%   the function value is not scalar, a bar plot of the elements at the
%   current iteration is displayed.  If the OPTIMVALUES.fval field does
not
%   exist, the OPTIMVALUES.residual field is used.
%
%   Example:
%   Create an options structure that will use OPTIMPLOTFVAL as the plot
%   function
%     options = optimset('PlotFcns',@optimplotfval);
%
%   Pass the options into an optimization problem to view the plot
%     fminbnd(@sin,3,10,options)

%   Copyright 2006 The MathWorks, Inc.
%   $Revision: 1.1.6.2 $  $Date: 2006/06/20 20:10:00 $

stop = false;
switch state
    case 'iter'
        if isfield(optimValues,'fval')
            if isscalar(optimValues.fval)
                plotscalar(optimValues.iteration,optimValues.fval);
            else
                plotvector(optimValues.iteration,optimValues.fval);
            end
        else
            plotvector(optimValues.iteration,optimValues.residual);
        end
end
```

```
function plotscalar(iteration,fval)
% PLOTSCALAR initializes or updates a line plot of the function value
% at each iteration.

if iteration == 0
    plotfval = plot(iteration,fval,'kd','MarkerFaceColor',[1 0 1]);
    title(['Current Function Value:
',num2str(1/fval)],'interp','none');
    xlabel('Iteration','interp','none');
    set(plotfval,'Tag','optimplotfval');
    ylabel('Function value','interp','none')
else
    plotfval = findobj(get(gca,'Children'),'Tag','optimplotfval');
    newX = [get(plotfval,'Xdata') iteration];
    newY = [get(plotfval,'Ydata') 1/fval];
    set(plotfval,'Xdata',newX, 'Ydata',newY);
    set(get(gca,'Title'),'String',['Current Function Value:
',num2str(1/fval)]);
end

function plotvector(iteration,fval)
% PLOTVECTOR creates or updates a bar plot of the function values or
% residuals at the current iteration.
if iteration == 0
    xlabelText = ['Number of function values: ',num2str(length(fval))];
    % display up to the first 100 values
    if numel(fval) > 100
```

```
        xlabelText = {xlabelText,'Showing only the first 100 values'};
        fval = fval(1:100);
    end
    plotfval = bar(fval);
    title('Current Function Values','interp','none');
    set(plotfval,'edgecolor','none')
    set(gca,'xlim',[0,1 + length(fval)])
    xlabel(xlabelText,'interp','none')
    set(plotfval,'Tag','optimplotfval');
    ylabel('Function value','interp','none')
else
    plotfval = findobj(get(gca,'Children'),'Tag','optimplotfval');
    % display up to the first 100 values
    if numel(fval) > 100
        fval = fval(1:100);
    end
    set(plotfval,'Ydata',fval);
end


function LatG=Suspension_Solver_mod(VAR)
% Solves for front and rear slip angles (alphaf and alphar)
% clear all
% clc
% Global variable declarations


global ny R W h L tf tr Xf Xr Kphif Kphir zrr zrf H N
global DWf DWr Wf Wr FyfoN FyfiN FyroN FyriN LatG
```

```
alphaf=VAR(1);

alphar=VAR(2);

DWf=VAR(3);

DWr=VAR(4);


% ny = 1.59        % estimated g'

Kphif =.61;      % ratio of front roll rate to total roll rate

Kphi = 1;        % roll gradient (deg/g)


g  = 32.17;

R  = 25;         % Turn radius, ft

W  = 500+180;    % Total vehicle weight, lb

h  = 12/12;      % CG height, ft

L  = 60/12;      % Wheelbase, ft

tf = 50/12;      % Front track, ft

tr = 48/12;      % Rear track, ft

Xf = .48;        % Front static weight distribution, per cent

Xr = 1-Xf;       % Rear static weight distribution, per cent

zrr =1.6/12;    % Rear roll center, ft

zrf= 1.7/12;   % Front roll center, ft


Df = 1;          % Allowable front travel from ride height, in

Dr = 1;          % Allowable rear travel from ride height, in


Kphir =1-Kphif;

H = h-(Xf*(zrr-zrf)+zrr);


tiref(alphaf);
```

```
tirer(alphar);

LatG=((FyfoN+FyfiN+FyroN+FyriN)/(W*N))^-1;
% DWf=156.93/2;
% DWr=109.08/2;

% UnderSteer or OverSteer

% deltaack = (L/R)*180/pi
% delta = deltaack + alphaf - alphar
% US =  deltaack - delta

function efront = tiref(xf)

global ny W tf Xf Xr Kphif zrf H %R Kphir zrr tr h L
global DWf N FyfoN FyfiN %DWr Wf Wr

gammafo = 0;        % Camber - front outside, deg
gammafi = 0;        % Camber - front inside, deg

Wf= Xf*W/2;     % Static weight on each front tire, lb
Wr= Xr*W/2;     % Static weight on each rear tire, lb

% DWf= (ny*W/tf)*(H*Kphif+Xr*zrf);  % Weight transfer

Zfo = Wf+DWf;                         % Load on front outside, lb
Zfi = Wf-DWf;                         % Load on front inside, lb
```

```
KN= .00444822162;                       % LB-KN conversion factor


ZfoKN= -KN*Zfo;

ZfiKN= -KN*Zfi;




% %Goodyear 20.0x8.0-13 15psi Pacjeka Formula '94 coefficients

% ktfNM =239086.7;

%  a0    =   1.5000000e+000;

%  a1    =   4.4255739e+001;

%  a2    =   1.5149295e+003;

%  a3    =  -3.8868435e+003;

%  a4    =  -1.5215217e+001;

%  a5    =   6.6089451e-002;

%  a6    =  -1.0564438e-002;

%  a7    =   2.9197666e-001;

%  a8    =  -3.7541013e-002;

%  a9    =   8.5183467e-002;

%  a10   =  -3.1927707e-002;

%  a11   =  -4.7678765e+001;

%  a12   =  -1.4027698e+002;

%  a13   =   8.8456962e-001;

%  a14   =  -2.0369299e+001;

%  a15   =  -3.8214868e-003;

%  a16   =  -0.2874285e-001;

%  a17   =   2.0000000e-001;
% Goodyear 20.0x6.5-13 15psi  Pacjeka Formual '94 coefficients

ktfNM =202968.5;  % Front tire stiffness, N/m

a0    =   1.5000000e+000;

a1    =   4.4255739e+001;

a2    =   1.5098362e+003;

a3    =  -3.7230395e+003;

a4    =  -1.5215217e+001;

a5    =   6.6089451e-002;

a6    =  -1.0564438e-002;

a7    =   2.9197666e-001;

a8    =  -3.7541013e-002;

a9    =   8.5183467e-002;

a10   =  -3.1927707e-002;

a11   =  -4.7678765e+001;

a12   =  -1.4027698e+002;

a13   =   8.8456962e-001;

a14   =  -2.0369299e+001;

a15   =  -3.8214868e-003;

a16   =  -0.2874285e-001;

a17   =   2.0000000e-001;


C=a0;


Dfo = (a1*ZfoKN+a2)*(1-a15*gammafo^2)*ZfoKN;

Dfi = (a1*ZfiKN+a2)*(1-a15*gammafi^2)*ZfiKN;


BCDfo = (2*a3*sin(atan(ZfoKN/a4)))*(1-a5*abs(gammafo));
```

```
BCDfi = (2*a3*sin(atan(ZfiKN/a4)))*(1-a5*abs(gammafi));


Bfo = BCDfo/(C*Dfo);

Bfi = BCDfi/(C*Dfi);


Shfo = a8*ZfoKN+a9*a10*gammafo;

Shfi = a8*ZfiKN+a9*a10*gammafi;


Svfo = a11*ZfoKN+a12+(a13.*ZfoKN^2+a14*ZfoKN)*gammafo;

Svfi = a11*ZfiKN+a12+(a13.*ZfiKN^2+a14*ZfiKN)*gammafi;


N = 4.44822162;


if Zfi<=0

FyfiN=0;

else

FyfiN = -(Dfi*sin(C*atan(Bfi*(xf+Shfi)-

(a6*ZfiKN+a7)*(a16*gammafi+a17*sign(xf+Shfi))*(Bfi*(xf+Shfi)-

atan(Bfi*(xf+Shfi)))))+ Svfi);

end


FyfoN = -(Dfo*sin(C*atan(Bfo*(xf+Shfo)-

(a6*ZfoKN+a7)*(a16*gammafo+a17*sign(xf+Shfo))*(Bfo*(xf+Shfo)-

atan(Bfo*(xf+Shfo)))))+ Svfo);
```

```
% efront = (FyfoN +FyfiN)/N -2*ny*Wf;  % error in front loads, alphaf
which
% makes error zero is found in solver

function erear = tirer(xr)

global ny  W tr Xf Xr Kphir zrr H
global DWr FyroN FyriN

gammaro = 0;         % Camber - rear outside tire, deg
gammari = 0;         % Camber - rear inside tire, deg

Wr= Xr*W/2;         % Static weight on each rear tire, lb

% DWr= (ny*W/tr)*(H*Kphir+Xf*zrr);    % weight transfer, lb

Zro = Wr+DWr;       % Load on rear outside tire, lb
Zri = Wr-DWr;       % Load on rear insode tire, lb

KN= .00444822162;   %converts loads from pounds to KiloNewtons

ZroKN= -KN*Zro;
ZriKN= -KN*Zri;
```

```
% Goodyear 20.0x8.0-13 15psi  Pacjeka Formual '94 coefficients

ktrNM =239086.7;  % Rear tire stiffness, N/m

 a0    =   1.5000000e+000;

 a1    =   4.4255739e+001;

 a2    =   1.5149295e+003;

 a3    =  -3.8868435e+003;

 a4    =  -1.5215217e+001;

 a5    =   6.6089451e-002;

 a6    =  -1.0564438e-002;

 a7    =   2.9197666e-001;

 a8    =  -3.7541013e-002;

 a9    =   8.5183467e-002;

 a10   =  -3.1927707e-002;

 a11   =  -4.7678765e+001;

 a12   =  -1.4027698e+002;

 a13   =   8.8456962e-001;

 a14   =  -2.0369299e+001;

 a15   =  -3.8214868e-003;

 a16   =  -0.2874285e-001;

 a17   =   2.0000000e-001;


C=a0;


Dro = (a1*ZroKN+a2)*(1-a15*gammaro^2)*ZroKN;

Dri = (a1*ZriKN+a2)*(1-a15*gammari^2)*ZriKN;


BCDro = (2*a3*sin(atan(ZroKN/a4)))*(1-a5*abs(gammaro));

BCDri = (2*a3*sin(atan(ZriKN/a4)))*(1-a5*abs(gammari));
```

```
Bro = BCDro/(C*Dro);

Bri = BCDri/(C*Dri);


Shro = a8*ZroKN+a9*a10*gammaro;

Shri = a8*ZriKN+a9*a10*gammari;


Svro = a11*ZroKN+a12+(a13*ZroKN^2+a14*ZroKN)*gammaro;

Svri = a11*ZriKN+a12+(a13*ZriKN^2+a14*ZriKN)*gammari;


N = 4.44822162;       % Used to convert FY in N to Fy in lb


FyroN = -(Dro*sin(C*atan(Bro*(xr+Shro)-
(a6*ZroKN+a7)*(a16*gammaro+a17*sign(xr+Shro))*(Bro*(xr+Shro)-
atan(Bro*(xr+Shro)))))+ Svro);
FyriN = -(Dri*sin(C*atan(Bri*(xr+Shri)-
(a6*ZriKN+a7)*(a16*gammari+a17*sign(xr+Shri))*(Bri*(xr+Shri)-
atan(Bri*(xr+Shri)))))+ Svri);


% erear  = (FyroN +FyriN)/N -2*ny*Wr; % Error in rear loads, alphar
which
% makes error zero is found in solver


%2DOF Root Locus Analysis
close all
clear
clc
```

```
m1=250;       % body mass in kg

m2=40;         % unsprung mass in kg

k1=15000;      % front suspension stiffness in N/m

k2=150000;  %tire stiffness

b=0;

% c=0;




%%%%%%%%%%Root Locus Analysis%%%%%%%%%%%

%The form of the root locus is the denominator of the T.F.=0.

%Furthermore, D(s)+K*N(s)=0

%Poles of Denominator

% p0=4*k*L^2/(4*b*L^2+Iyy);%0;

% p1=0;

% p2=1;

% p3=0;

% p4=0;

% %Zeros of Numerator

% z0=0;

% z1=1;

% z2=0;

% z3=0;

% z4=0;

%

% sys2=tf([z4,z3,z2,z1,z0],[p4,p3,p2,p1,p0])

% rlocus(sys2);
%   axis equal
%   sgrid
% % K=
% % c=K*(4*b*L^2+Iyy)/(4*L^2)
% %%%%%%%%%%%%%%%Full TF System%%%%%%%%%%%%%%%%
% s=tf('s');
% num=1;
% den=s^2*(4*b*L^2+Iyy)+s*(4*c*L^2)+(4*k*L^2);
% sys=num/den
% damp(sys)

%%%%%%%%%%%%%%%Simulink Model Verification%%%%%%%%%%%%%%%%%%
n=5;
fin=500;
for c=0:n:fin

    [A,B,C,D]=linmod('Palm2DOF_RLocus');
%     [num, den]=linmod('Palm2DOF_RLocus');
%     sys1=ss(A,B,C,D);
    [num,den]=ss2tf(A,B,C,D,1);   %choose "1" for the 1st input
    sys1=tf([num],[den]);
    [wn,zeta,p]=damp(sys1);

    i=c/n+1
    re(i,:)=real(p); im(i,:)=imag(p);

end
```

```
plot(re,im,'ks','MarkerSize',8,'MarkerFaceColor','k')

axis equal

sgrid



%%%%%%%%%%%%%%%%%Palm2DOF CORRRECT root locus%%%%%%%%%%%%%%%%%%%

sys2=tf([1,0,517.24,0],[1,0,4185,0,2.25e5]);

hold on

rlocus(sys2)





function [b br c1 c2]=TwoDOFoptimization



%initialize plant variables in model

TwoDOFinerter

%ms=250,mu=35,kt=150

% ms=20;

% mu=10;

% ks=15;

% kt=15;

% b0=0.5;

% br0=.15;

% c1_0=1;

% c2_0=1;

ms=20;

mu=4;

ks=15;

kt=15;

b0=3;
```

```
br0=.1;

c1_0=15;

c2_0=15;


bc0=[b0 br0 c1_0 c2_0];




options = optimset('LargeScale','off','Display','iter',...

      'TolX',0.001,'TolFun',0.001);

coef = lsqnonlin(@tracklsq, bc0, [], [], options);

b=coef(1); br=coef(2); c1=coef(3); c2=coef(4);


    function F = tracklsq(coef)

        % Track the output of optsim to a signal of 1

       b=coef(1);

       br=coef(2);

       c1=coef(3);

       c2=coef(4);


        % Compute function value

        simopt=simset('SrcWorkspace','current');  % Initialize sim

options

        [t,x,y] = sim('TwoDOFInerter',10,simopt);

        F = y-1;


    end

end
```

```
 % plot(t,y)


function [cl,cr]=suspn_2dofroll_optim


global Ll Lr Mb Ixx k kl kr bl br cl0 cr0 g


%load model
suspn_2dof_roll_inerter


%initialize plant variables in model
% L=24.5;
% Ll = L;             % left hub displacement from body CG (in)
% Lr = L;             % right hub displacement from body CG (in)
% Mb = 680/32.17;     % body mass +180lb driver in slugs (lbf/grav)
% Ixx = 205559.63;    % body moment of inertia about roll-axis in
lb*in^2
% k=(1/(148+112)+550)/2;          % equivalent spring [k_front=148,
k_rear=112]
% kl = k;             % left suspension stiffness in lb/in
% kr = k;             % rear suspension stiffness in lb/in
% b=0;                % inertance
% br=b;
% bl=b;
% cl0 = 15;           % initial left suspension damping in lb/(in/s)
% cr0 = 15;           % initial right suspension damping in lb/(in/s)
c0=[cl0 cr0];       % create initial condition vector


% set tolerance on damping to 10 lb/(in/s), tolerance on weight...
% transfer to 1/4lb

options = optimset('Display','iter','TolX',0.25,'TolFun',.25);

coef = lsqnonlin(@track,c0,0,[],options);

cl=coef(1); cr=coef(1);


    function F = track(coef)

                cl=coef(1);

                cr=coef(1);


        % Compute function value

        simopt=simset('SrcWorkspace','current','MaxStep',.01);  %
Initialize sim options

        [t,x,y] = sim('suspn_2dof_roll_inerter',20,simopt);

        F = y(:,1)-y(:,2);


    end

end


%% 2DOF(Roll) Root Locus Analysis

close all

clear

clc


global Ll Lr Mb Ixx k kl kr bl br cl0 cr0 g


g=-32.17;              %gravity (ft/s^2)

L=24.5;
```

```
Ll = L;             % left hub displacement from body CG (in)

Lr = L;             % right hub displacement from body CG (in)

Mb = 680/32.17;     % body mass +180lb driver in slugs (lbf/grav)

Ixx = 205559.63;    % body moment of inertia about roll-axis in lb*in^2

k=(1/(148+550)+112)/2;          % equivalent spring [k_front=148,
k_rear=112]

kl = k;             % left suspension stiffness in lb/in

kr = k;             % rear suspension stiffness in lb/in

b=0;                % inertance

br=b;

bl=b;

cl0=15;

cr0=15;


%% Optimization

[cl,cr]=suspn_2dofroll_optim


%% %%%%%%%%Root Locus Analysis%%%%%%%%%%%

%The form of the root locus is the denominator of the T.F.=0.

%Furthermore, D(s)+K*N(s)=0

%Poles of Denominator

% p0=4*k*L^2/(4*b*L^2+Iyy);%0;

% p1=0;

% p2=1;

% p3=0;

% p4=0;

% %Zeros of Numerator

% z0=0;
% z1=1;

% z2=0;

% z3=0;

% z4=0;

%

% sys2=tf([z4,z3,z2,z1,z0],[p4,p3,p2,p1,p0])

% rlocus(sys2);

%   axis equal

%   sgrid

% % K=

% % c=K*(4*b*L^2+Iyy)/(4*L^2)

% %%%%%%%%%%%%%%%%Full TF System%%%%%%%%%%%%%%%%%

% s=tf('s');

% num=1;

% den=s^2*(4*b*L^2+Iyy)+s*(4*c*L^2)+(4*k*L^2);

% sys=num/den

% damp(sys)


%% Signals

    sim('suspn_2dof_roll_postopt_analysis',20);    %Create logsout
structure file

    norm(logsout.Flt.Data-logsout.Frt.Data)

    time = logsout.Frt.Time;

    figure

    subplot(3,1,1),

    plot(time,logsout.Flt.Data,time,logsout.Frt.Data);

    ylabel('$$Force$$ (lbf)','Interpreter','LaTex');

    %text(2.2,0.002,'d\theta/dt')
    title('2-DOF Suspension Model Simulation Results')
    legend('F_L_t','F_R_t','Orientation','horizontal',...
        'Location','Best')

    subplot(3,1,2),
    plot(time,logsout.Z.Data);
    ylabel('$$Chassis Heave$$ (in)','Interpreter','LaTex');
    ylim([-3.5 -2.5]);
    %text(2.2, 0.03, 'dz/dt')

    subplot(3,1,3),
    plot(time,logsout.Theta.Data);
    ylabel('$$Chasasis Roll$$ (rad)','Interpreter','LaTex');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%
%% %%%%%%%%%%%%%%%Eigenvectors and Eigenvalues
(modes)%%%%%%%%%%%%%%%%%%%%
% c=0;
% cr=c;
% cl=c;

M=[-Mb-4*b, 0; 0, -4*b*L^2-Ixx];  %Mass matrix (see 2DOF(Roll)_1.nb)
K=[-4*k, 0; 0, -4*k*L^2];      %Stiffness matrix (see 2DOF(Roll)_1.nb)

[vector,value]=eig(K,M)   %"vector" columns are eigenvectors
display('Nat. Freq.')
sqrt(value)
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%
%% %%%%%%%%%%%%Build Root Locus w/ damping coef. as
gain%%%%%%%%%%%%%%%%%
n=.1;       % step size
fin=(cr+cl)/2;     % final damping value
i=0;        % initialize index

% create root locus
for c=0:n:fin
    cl=c;       % give suspension symmetry
    cr=c;

    [A,B,C,D]=linmod('TwoDofRoll_RLocus');      % find LTI model
    sys1=ss(A,B,C,D);           % change LTI model to state space
    [wn,zeta,p]=damp(sys1);     % find eigenvalues

    i=i+1       % increment index
    dampcoef(i,:)=c;        % collect "c's" used
    re(i,:)=real(p); im(i,:)=imag(p);       % find Re(pole), Im(pole)
%     natfreq(i,:)=wn;          % collect natural frequency value
%     damprat(i,:)=zeta;        % collect damping ratio value

%%variable step size
%      if i>1
%         if (sqrt((re(i)-re(i-1))^2+(im(i)-im(i-1))^2)>.2)
%         n=2;
%         cnt=cnt+1

%         end

%     end

end


%%Create root locus figure

fig=figure;

dcm_obj=datacursormode(fig);

set(dcm_obj,'UpdateFcn',@myupdatefcn)   %customized datatip function

plot(re,im,'.b',re(1,:),im(1,:),'.g',re(i,:),im(i,:),'.r')

axis([2*min(min(im)) 0 min(min(im)) max(max(im))])

axis square

%axis([-14 0 .47 .52])

% axis equal

% axis tight

sgrid


%%Use custom datatip

% xlimits=xlim;

% ylimits=ylim;

% display('Press any button to continue')

% while waitforbuttonpress==0

%     info_struct=getCursorInfo(dcm_obj);

%     ind=info_struct.DataIndex;

%     refreshdata

%     text(0.9*xlimits(1),0.8*ylimits(2),['Damping Coefficient: ',
num2str(dampcoef(ind))],...

%         'BackgroundColor',[1 1 .93],'EdgeColor',[.8 .8 .8])
% end

function output_txt = myfunction(obj,event_obj)
% Display the position of the data cursor
% obj          Currently not used (empty)
% event_obj    Handle to event object
% output_txt   Data cursor text string (string or cell array of
strings).

pos = get(event_obj,'Position');
% get(event_obj,'Target');
% dampcoef(1)
% dcm_obj=datacursormode(fig);
% info_struct=getCursorInfo(dcm_obj);
% ind = info_struct.DataIndex;
ang=atan(pos(2)/pos(1));
zeta=cos(ang);
freq=norm([pos(1) pos(2)]);
pole=pos(1)+j*pos(2);
os=100*exp(-pi*zeta/(sqrt(1-(zeta)^2)));
output_txt = {['Pole: ',num2str(pole)],...
    ['Damping: ',num2str(zeta)],...
    ['Overshoot: ',num2str(os)],...
    ['Frequency (rad/s): ',num2str(freq)]};

% If there is a Z-coordinate in the position, display it as well
if length(pos) > 2
    output_txt{end+1} = ['Z: ',num2str(pos(3),4)];
```

```
end


clear

close all

clc


%%Initial Conditions & Parameters

L=24.5;

Ll = L;              % left hub displacement from body CG (in)

Lr = L;              % right hub displacement from body CG (in)

m_u=(31+35)/(2*32.17);    % front/rear corner average unsprung mass
(lbf/grav)

Mb = 680/32.17-4*m_u;     % body mass +180lb driver in slugs (lbf/grav)

m1=m_u;               % car is symmetric

m2=m_u;

Ixx = 205559.63;     % body moment of inertia about roll-axis in lb*in^2

k=(148+112)/2;       % equivalent wheelrate [k_front=148, k_rear=112]

kl = k;              % left suspension wheelrate in lb/in

kr = k;              % rear suspension wheelrate in lb/in

kt= 519/2;             % tire stiffness (lb/in)

b=0;                 % inertance

br=b;

bl=b;

IR_f=0.77;

IR_r=0.54;

g=-32.17;

y0=g*(Mb/2+2*m_u)/(2*kt);    %initial unsprung mass height;

Z0=g*Mb/(2*kr+2*kl)+y0;      %initial sprung mass height;
```

```
c0=320;               % FSAE stock damping

cr0=c0;        % effective damping

cl0=c0;


%% Optimization

cl=halfcar_symdamper_optim(g,Ll,Lr,Mb,Ixx,m_u,kl,kr,kt,bl,br,cr0)

%

[cl,cr]=halfcar_damper_optim(g,Ll,Lr,Mb,Ixx,m_u,kl,kr,kt,bl,br,cl0,cr0)

% optimized suspension damping in N(m/s)

cr=cl;

% br=bl;


%% Bode etc.

% halfcar_bode_analysis

% [A,B,C,D]=linmodv5('halfcar_bode_analysis');

% sys=ss(A,B,C,D);

% bode(sys)

% damp(sys)   %output Eigenvalues/Damping/Natural Frequency

%sldebug halfcar_bode_analysis %>>states


%% Signals

    sim('halfcar_post_opt_analysis',79.785);    %Create logsout

structure file

    norm(logsout.Flt.Data-logsout.Frt.Data)

    time = logsout.Frt.Time;

    figure

    subplot(3,1,1),

    plot(time,logsout.Flt.Data,time,logsout.Frt.Data);
    ylabel('$$Force$$ (lbf)','Interpreter','LaTex');
    %text(2.2,0.002,'d\theta/dt')
    title('4-DOF Suspension Model Simulation Results')
    legend('F_L_t','F_R_t','Orientation','horizontal',...
        'Location','NorthWest')

    subplot(3,1,2),
    plot(time,logsout.Z.Data);
    ylabel('$$Chassis Heave$$ (in)','Interpreter','LaTex');
    %text(2.2, 0.03, 'dz/dt')

    subplot(3,1,3),
    plot(time,logsout.('Left Suspension').y.Data);
    ylabel('$$Wheel Motion$$ (in)','Interpreter','LaTex');

%% Mode Shapes
%%%coordinate order: (Z,z1,z2,Theta)
M=[-2*bl-2*br-Mb, 2*bl, 2*br, 2*bl*Ll-2*br*Lr  %Mass matrix (see
4DOF.nb)
    2*bl, -2*bl-2*m1, 0, -2*bl*Ll
    2*br, 0, -2*br-2*m2, 2*br*Lr
    2*bl*Ll-2*br*Lr, -2*bl*Ll, 2*br*Lr, -2*bl*Ll^2-2*br*Lr^2-Ixx];

K=[-2*kl-2*kr, 2*kl, 2*kr, 2*kl*Ll-2*kr*Lr  %Stiffness matrix (see
4DOF.nb)
    2*kl, -2*kl-2*kt, 0, -2*kl*Ll
    2*kr, 0, -2*kr-2*kt, 2*kr*Lr
    2*kl*Ll-2*kr*Lr, -2*kl*Ll, 2*kr*Lr, -2*kl*Ll^2-2*kr*Lr^2];
```

```
[eigenvectors,eigenvalues]=eig(K,M);   %"vector" columns are
eigenvectors
eigenvectors
display('Nat. Freq.')
sqrt(eigenvalues)

%% Build Root Locus w/ damping coef. as gain%%%%%%%%%%%%%%%%
fin=(cl+cr)/2;    % final damping value
n=fin/100;       % step size
i=0;        % initialize index

% create root locus
for c=0:n:fin
    cl=c;       % give suspension symmetry
    cr=c;

    [A,B,C,D]=linmod('halfcar_bode_analysis');      % find LTI model
    sys1=ss(A,B,C,D);           % change LTI model to state space
    [wn,zeta,p]=damp(sys1);     % find eigenvalues

    i=i+1       % increment index
    dampcoef(i,:)=c;        % collect "c's" used
    re(i,:)=real(p); im(i,:)=imag(p);       % find Re(pole), Im(pole)
%     natfreq(i,:)=wn;          % collect natural frequency value
%     damprat(i,:)=zeta;        % collect damping ratio value
```

```
%%variable step size
%      if i>1
%         if (sqrt((re(i)-re(i-1))^2+(im(i)-im(i-1))^2)>.2)
%         n=2;
%         cnt=cnt+1
%         end
%     end
end


%% Create root locus figure
fig=figure;
dcm_obj=datacursormode(fig);
set(dcm_obj,'UpdateFcn',@myupdatefcn)   %customized datatip function
plot(re,im,'.b',re(1,:),im(1,:),'.g',re(i,:),im(i,:),'.r')
axis([2*min(min(im)) 0 min(min(im)) max(max(im))])
axis square
% axis equal
% axis tight
sgrid


%%Use custom datatip
% xlimits=xlim;
% ylimits=ylim;
% display('Press any button to continue')
% while waitforbuttonpress==0
%     info_struct=getCursorInfo(dcm_obj);
%     ind=info_struct.DataIndex;
%     refreshdata
%     text(-45,20,['Damping Coefficient: ', num2str(dampcoef(ind))],...

%         'BackgroundColor',[1 1 .93],'EdgeColor',[.8 .8 .8])

% end


function

cl=halfcar_symdamper_optim(g,Ll,Lr,Mb,Ixx,m_u,kl,kr,kt,bl,br,c0)


%load model

halfcar


%initialize plant variables in model

% cl0 = 2500;     % front suspension damping in N/(m/s)

% cr = 18759;   % rear suspension damping in N/(m/s)

y0=g*(Mb/4+m_u)/(kt);      %initial unsprung mass height;

Z0=g*Mb/(2*kr+2*kl)+y0;    %initial sprung mass height;

% c0=[cl0 cr0];




options = optimset('Display','iter','TolX',.25,'TolFun',.05);%

coef = lsqnonlin(@track,c0,0,[],options);

cl=coef(1); cr=coef(1);


    function F = track(coef)

                cl=coef(1);

                cr=coef(1);


        % Compute function value
        simopt=simset('SrcWorkspace','current');   % Initialize sim
options
        [t,x,y] = sim('halfcar',79.785,simopt);   % runtime = 79.7852s
        F = y(:,1)-y(:,2);

    end
end

function
[cl,cr]=halfcar_damper_optim(g,Ll,Lr,Mb,Ixx,m_u,kl,kr,kt,bl,br,cl0,cr0)

%load model
halfcar

%initialize plant variables in model
% cl0 = 2500;     % front suspension damping in N/(m/s)
% cr = 18759;    % rear suspension damping in N/(m/s)
y0=g*(Mb/4+m_u)/(kt);      %initial unsprung mass height;
Z0=g*Mb/(2*kr+2*kl)+y0;    %initial sprung mass height;
c0=[cl0 cr0];



options = optimset('Display','iter');%,'TolX',.25,'TolFun',.05
coef = lsqnonlin(@track,c0,0,[],options);
cl=coef(1); cr=coef(2);

    function F = track(coef)
```

```
                cl=coef(1);
                cr=coef(2);

        % Compute function value
        simopt=simset('SrcWorkspace','current');   % Initialize sim
options
        [t,x,y] = sim('halfcar',79.785,simopt);   % runtime = 79.7852s
        F = y(:,1)-y(:,2);

    end
end

function output_txt = myfunction(obj,event_obj)
% Display the position of the data cursor
% obj          Currently not used (empty)
% event_obj    Handle to event object
% output_txt   Data cursor text string (string or cell array of
strings).

pos = get(event_obj,'Position');
% get(event_obj,'Target');
% dampcoef(1)
% dcm_obj=datacursormode(fig);
% info_struct=getCursorInfo(dcm_obj);
% ind = info_struct.DataIndex;
ang=atan(pos(2)/pos(1));
zeta=cos(ang);
freq=norm([pos(1) pos(2)]);
```

```
pole=pos(1)+j*pos(2);

os=100*exp(-pi*zeta/(sqrt(1-(zeta)^2)));

output_txt = {['Pole: ',num2str(pole)],...

    ['Damping: ',num2str(zeta)],...

    ['Overshoot: ',num2str(os)],...

    ['Frequency (rad/s): ',num2str(freq)]};


% If there is a Z-coordinate in the position, display it as well

if length(pos) > 2

    output_txt{end+1} = ['Z: ',num2str(pos(3),4)];

end


clear

close all

clc


global g tf tr Lf Lr mf mr Mb Ixx Iyy kf kr kt

%%%%%%%%%%%%%%%%%%Initial Conditions &

Parameters%%%%%%%%%%%%%%%%%%%%%%

g=-32.17;

tf=50;              % front trackwidth

tr=48;              % rear trackwidth

fdist=0.48;         % front weight distribution

L=60;               % wheelbse

Lf=(1-fdist)*L;         % left hub displacement from body CG (in)

Lr=fdist*L;             % right hub displacement from body CG (in)

mf=31/(-g);    % front/rear corner average unsprung mass (lbf/grav)

mr=35/(-g);
```

```
Mb=680/(-g)-2*mf-2*mr;     % body mass +180lb driver in slugs
(lbf/grav)
Ixx=205559.63;      % body moment of inertia about roll-axis in lb*in^2
Iyy=2459865.33;     % body moment of inertia about pitch-axis in
lb*in^2
kf=148;             % front suspension wheelrate in lb/in
kr=112;             % rear suspension wheelrate in lb/in
kt=550;             % tire stiffness (lb/in)
IR_f=0.77;          % front installation ratio
IR_r=0.54;          % rear installation ratio
bf=0;               % front inertance
br=0;               % rear inertance
%c0=500;             % FSAE stock damping
cf0=412;       % effective front damping
cr0=243;      % effective rear damping
[yf0,yr0,Z0,Theta0,Psi0]=EOMs;%(g,Mb,mf,mr,tf,tr,Lf,Lr,kf,kr,kt);  %
I.C.'s


%% optimization
% [cf,cr]=fullcar_symdamper_optim(bf,br,cf0,cr0)
cf=426;
cr=239;


%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%Bode
etc.%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% halfcar_bode_analysis
% [A,B,C,D]=linmodv5('fullcar_bode_analysis');
% sys=ss(A,B,C,D);
% bode(sys)
% damp(sys)   %output Eigenvalues/Damping/Natural Frequency
% sldebug halfcar_bode_analysis %>>states

%%
%%%%%%%%%%%%%%%%%%%%%%%%%%Signals%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
    sim('fullcar_postopt_analysis',79.785);    %Create logsout
structure file
    display('Norm of weight transfer vectors')
    norm(logsout.Fleft.Data-logsout.Fright.Data)
    time = logsout.Fleft.Time;

    figure('Name','Chassis','NumberTitle','off')
    subplot(3,1,1),
    plot(time,logsout.Heave.Z.Data);
    ylabel('$$Vertical Motion$$ (in)','Interpreter','LaTex');
    %text(2.2, 0.03, 'dz/dt')
    subplot(3,1,2),
    plot(time,logsout.Pitch.Psi.Data);
    ylabel('$$Chassis Pitch$$ (rad)','Interpreter','LaTex');
    subplot(3,1,3),
    plot(time,logsout.Roll.Theta.Data);
    ylabel('$$Chassis Roll$$ (rad)','Interpreter','LaTex');

    figure('Name','Wheels','NumberTitle','off')
    subplot(4,1,1),
    plot(time,logsout.('Left Front Suspension').zlf.Data);
    ylabel('Z_l_f (in)');

    subplot(4,1,2),

    plot(time,logsout.('Right Front Suspension').zrf.Data);

    ylabel('Z_r_f (in)');

    subplot(4,1,3),

    plot(time,logsout.('Left Rear Suspension').zlr.Data);

    ylabel('Z_l_r (in)');

    subplot(4,1,4),

    plot(time,logsout.('Right Rear Suspension').zrr.Data);

    ylabel('Z_r_r (in)');


    figure('Name','Objective Function','NumberTitle','off')

    subplot(2,1,1),

    plot(time,logsout.Fleft.Data,time,logsout.Fright.Data);

    ylabel('$$Lateral Loads$$ (lbf)','Interpreter','LaTex');

    %text(2.2,0.002,'d\theta/dt')


legend('\SigmaF_{Front}','\SigmaF_{Rear}','Orientation','Horizontal',..
.
        'Location','NorthEast')

%     title('7-DOF Suspension Model Simulation Results')

    subplot(2,1,2)

    plot(time,logsout.Fleft.Data-logsout.Fright.Data);

    ylabel('$$Weight Transfer$$ (lbf)','Interpreter','LaTex');




%% Mode Shapes
```

```
%%%coordinate order: (zlf,zlr,zrr,zrf,Z,Theta,Psi)
% M=[-bf-mf, 0, 0, 0, bf, bf*tf/2, -bf*Lf;  %Mass matrix (see 7DOF.nb)
%    0, -br-mr, 0, 0, br, br*tr/2, br*Lr;
%    0, 0, -br-mr, 0, br, -br*tr/2, br*Lr;
%    0, 0, 0, -bf-mf, bf, -bf*tf/2, -bf*Lf;
%    bf, br, br, bf, -bf-br-bf-br-Mb, -bf*tf/2-
br*tr/2+bf*tf/2+br*tr/2,...
%       bf*Lf+bf*Lf-br*Lr-br*Lr;
%   bf*tf/2, br*tr/2, -br*L*tr/2, -bf*tf/2, -bf*tf/2-
br*tr+bf*tf/2+br*tr/2,...
%       -Ixx-bf*(tf/2)^2-br*(tr/2)^2-bf*(tf/2)^2-br*(tr/2)^2,...
%       bf*Lf*tf/2-br*tr/2*Lr-bf*Lf*tf/2+br*Lr*tr/2;
%   -bf*Lf, br*Lr, br*Lr, -bf*Lf, bf*Lf+bf*Lf-br*Lr-br*Lr,...
%       bf*Lf*tf/2-br*tr/2*Lr-bf*Lf*tf/2+br*Lr*tr/2,...
%       -Iyy-bf*Lf^2-bf*Lf^2-br*Lr^2-br*Lr^2];
%
%  K=[-kf-kt, 0, 0, 0, kf, kf*tf/2, -kf*Lf;   %Stiffness matrix (see
7DOF.nb)
%    0, -kr-kt, 0, 0, kr, kr*tr/2, kr*Lr;
%    0, 0, -kr-kt, 0, kr, -kr*tr/2, kr*Lr;
%    0, 0, 0, -kf-kt, kf, -kf*tf/2, -kf*Lf;
%    kf, kr, kr, kf, -kf-kr-kf-kr, -kf*tf/2-kr*tr/2+kf*tf/2+kr*tr/2,...
%       kf*Lf+kf*Lf-kr*Lr-kr*Lr;
%    kf*tf/2, kr*tr/2, -kr*tr/2, -kf*tf/2, -kf*tf/2-kr*tr/2+kf*tf/2+...
%       kr*tr/2, -kf*(tf/2)^2-kr*(tr/2)^2-kf*(tf/2)^2-kr*(tr/2)^2,...
%       kf*Lf*tf/2-kr*tr/2*Lr-kf*Lf*tf/2+kr*Lr*tr/2;
%    -kf*Lf, kr*Lr, kr*Lr, -kf*Lf, kf*Lf+kf*Lf-kr*Lr-kr*Lr,...
%       kf*Lf*tf/2-kr*tr/2*Lr-kf*Lf*tf/2+kr*Lr*tr/2,...
%       -kf*Lf^2-kf*Lf^2-kr*Lr^2-kr*Lr^2];
```

```
%
% [eigenvectors,eigenvalues]=eig(K,M);   %"vector" columns are
eigenvectors
% %%%coordinate order: (zlf,zlr,zrr,zrf,Z,Theta,Psi)
% eigenvectors
% display('Nat. Freq.')
% sqrt(eigenvalues)

%% %%%%%%%%%%%%Build Root Locus w/ damping coef. as
gain%%%%%%%%%%%%%%%%%%
fin=(cf+cr)/2;    % final damping value
n=fin/100;       % step size
i=0;        % initialize index

% create root locus
for c=0:n:fin
     cf=c;       % give suspension symmetry
     cr=c;

    [A,B,C,D]=linmod('fullcar_bode_analysis');      % find LTI model
    sys1=ss(A,B,C,D);           % change LTI model to state space
    [wn,zeta,p]=damp(sys1);     % find eigenvalues

    i=i+1;       % increment index
%    dampcoef(i,:)=c;        % collect "c's" used
    re(i,:)=real(p); im(i,:)=imag(p);       % find Re(pole), Im(pole)
%     natfreq(i,:)=wn;          % collect natural frequency value
%     damprat(i,:)=zeta;        % collect damping ratio value
```

```
%%variable step size
%      if i>1
%         if (sqrt((re(i)-re(i-1))^2+(im(i)-im(i-1))^2)>.2)
%         n=2;
%         cnt=cnt+1
%         end
%     end
end


%% Create root locus figure
fig=figure;
dcm_obj=datacursormode(fig);
set(dcm_obj,'UpdateFcn',@myupdatefcn)   %customized datatip function
plot(re,im,'.b',re(1,:),im(1,:),'.g',re(i,:),im(i,:),'.r')
axis([2*min(min(im)) 0 min(min(im)) max(max(im))])
axis square
% axis equal
% axis tight
sgrid


%%Use custom datatip
% xlimits=xlim;
% ylimits=ylim;
% display('Press any button to continue')
% while waitforbuttonpress==0
%     info_struct=getCursorInfo(dcm_obj);
%     ind=info_struct.DataIndex;
%     refreshdata
%     text(-45,20,['Damping Coefficient: ', num2str(dampcoef(ind))],...
%         'BackgroundColor',[1 1 .93],'EdgeColor',[.8 .8 .8])
% end

function [cf,cr]=fullcar_symdamper_optim(bf0,br0,cf0,cr0)
% function
% [c1f,c2f,vtf,c1r,c2r,vtr]=fullcar_symdamper_optim(bf0,br0,cf0,cr0)

global g tf tr Lf Lr mf mr Mb Ixx Iyy kf kr kt

%load model
fullcar

%initialize plant variables in model
bf=bf0;
br=br0;
[yf0,yr0,Z0,Theta0,Psi0]=EOMs;%(g,Mb,mf,mr,tf,tr,Lf,Lr,kf,kr,kt);  %
I.C.'s
c0=[cf0 cr0];



options = optimset('Display','iter','TolX',.25,'TolFun',.05);%
coef = lsqnonlin(@track,c0,0,[],options);
cf=coef(1); cr=coef(2);

    function F = track(coef)
```

```
                  cf=coef(1);

                  cr=coef(2);


          % Compute function value

          simopt=simset('SrcWorkspace','current');    % Initialize sim

options

          [t,x,y] = sim('fullcar',79.785,simopt);    % runtime = 79.7852s

          F = y(:,1)-y(:,2);

    end

end


function

[ylf0,ylr0,Z0,Theta0,Psi0]=EOMs%(g,Mb,mf,mr,tf,tr,Lf,Lr,kf,kr,kt)


global g Mb mf mr tf tr Lf Lr kf kr kt


% g=-32.17;

% tf=50;              % front trackwidth

% tr=48;              % rear trackwidth

% fdist=0.48;         % front weight distribution

% L=60;               % wheelbse

% Lf=(1-fdist)*L;         % left hub displacement from body CG (in)

% Lr=fdist*L;             % right hub displacement from body CG (in)

% mf=31/(-g);     % front/rear corner average unsprung mass (lbf/grav)

% mr=35/(-g);

% Mb=680/(-g)-2*mf-2*mr;     % body mass +180lb driver in slugs

(lbf/grav)

% kf=148;             % front suspension wheelrate in lb/in
% kr=112;              % rear suspension wheelrate in lb/in
% kt=550;              % tire stiffness (lb/in)

 K=[-kf-kt, 0, 0, 0, kf, kf*tf/2, -kf*Lf;
   0, -kr-kt, 0, 0, kr, kr*tr/2, kr*Lr;
   0, 0, -kr-kt, 0, kr, -kr*tr/2, kr*Lr;
   0, 0, 0, -kf-kt, kf, -kf*tf/2, -kf*Lf;
   kf, kr, kr, kf, -kf-kr-kf-kr, -kf*tf/2-kr*tr/2+kf*tf/2+kr*tr/2,...
      kf*Lf+kf*Lf-kr*Lr-kr*Lr;
   kf*tf/2, kr*tr/2, -kr*tr/2, -kf*tf/2, -kf*tf/2-kr*tr/2+kf*tf/2+...
      kr*tr/2, -kf*(tf/2)^2-kr*(tr/2)^2-kf*(tf/2)^2-kr*(tr/2)^2,...
      kf*Lf*tf/2-kr*tr/2*Lr-kf*Lf*tf/2+kr*Lr*tr/2;
   -kf*Lf, kr*Lr, kr*Lr, -kf*Lf, kf*Lf+kf*Lf-kr*Lr-kr*Lr,...
      kf*Lf*tf/2-kr*tr/2*Lr-kf*Lf*tf/2+kr*Lr*tr/2,...
      -kf*Lf^2-kf*Lf^2-kr*Lr^2-kr*Lr^2];

b=[mf*g; mr*g; mr*g; mf*g; Mb*g;0;0];
%%%coordinate order: (zlf,zlr,zrr,zrf,Z,Theta,Psi)
ans=(inv(K)*b);
ylf0=ans(1,1);
ylr0=ans(2,1);
yrr=ans(3,1);
yrf=ans(4,1);
Z0=ans(5,1);
Theta0=ans(6,1);
Psi0=ans(7,1);

function output_txt = myfunction(obj,event_obj)
```

```
% Display the position of the data cursor
% obj          Currently not used (empty)
% event_obj    Handle to event object
% output_txt   Data cursor text string (string or cell array of
strings).

pos = get(event_obj,'Position');
% get(event_obj,'Target');
% dampcoef(1)
% dcm_obj=datacursormode(fig);
% info_struct=getCursorInfo(dcm_obj);
% ind = info_struct.DataIndex;
ang=atan(pos(2)/pos(1));
zeta=cos(ang);
freq=norm([pos(1) pos(2)]);
pole=pos(1)+j*pos(2);
os=100*exp(-pi*zeta/(sqrt(1-(zeta)^2)));
output_txt = {['Pole: ',num2str(pole)],...
    ['Damping: ',num2str(zeta)],...
    ['Overshoot: ',num2str(os)],...
    ['Frequency (rad/s): ',num2str(freq)]};

% If there is a Z-coordinate in the position, display it as well
if length(pos) > 2
    output_txt{end+1} = ['Z: ',num2str(pos(3),4)];
end
```

```
clear

close all

clc


global g tf tr Lf Lr mf mr Mb Ixx Iyy kf kr kt


%%%%%%%%%%%%%%%%%%%%Initial Conditions &
Parameters%%%%%%%%%%%%%%%%%%%%%%%
g=-32.17;
tf=50;              % front trackwidth
tr=48;              % rear trackwidth
fdist=0.48;         % front weight distribution
L=60;               % wheelbse
Lf=(1-fdist)*L;     % left hub displacement from body CG (in)
Lr=fdist*L;         % right hub displacement from body CG (in)
mf=31/(-g);         % front/rear corner average unsprung mass
(lbf/grav)
mr=35/(-g);
Mb=680/(-g)-2*mf-2*mr;     % body mass +180lb driver in slugs
(lbf/grav)
Ixx=205559.63;      % body moment of inertia about roll-axis in lb*in^2
Iyy=2459865.33;     % body moment of inertia about pitch-axis in
lb*in^2
kf=148;             % front suspension wheelrate in lb/in
kr=112;             % rear suspension wheelrate in lb/in
kt=550;             % tire stiffness (lb/in)
IR_f=0.77;          % front installation ratio
IR_r=0.54;          % rear installation ratio
```

```
% optimization I.C.'s
bf0=0;                % front inertance
br0=0;                % rear inertance
% c0=164;         % FSAE stock damping
c1f0=416;%c0*IR_f^2;     % effective front damping (low speed)
c2f0=75;          % effective front damping (high speed)
vtf0=0.75;
c1r0=248;%c0*IR_r^2;     % effective rear damping (low speed)
c2r0=75;          % effective rear damping (low speed)
vtr0=0.75;
[yf0,yr0,Z0,Theta0,Psi0]=EOMs;%(g,Mb,mf,mr,tf,tr,Lf,Lr,kf,kr,kt);  %
I.C.'s

%% optimization
[c1f,c2f,vtf,c1r,c2r,vtr,bf,br]=fullcar_nonlindamper_optim(c1f0,c2f0,vt
f0,c1r0,c2r0,vtr0,bf0,br0)
% cr=cl;
% br=bl;

%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Bode
etc.%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% halfcar_bode_analysis
% [A,B,C,D]=linmodv5('fullcar_bode_analysis');
% sys=ss(A,B,C,D);
% bode(sys)
% damp(sys)   %output Eigenvalues/Damping/Natural Frequency
% sldebug halfcar_bode_analysis %>>states
```

```
%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%Signals%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
    sim('nonlinfullcar_postopt_analysis',79.785);    %Create logsout
structure file
    display('Norm of weight transfer vectors')
    norm(logsout.Fleft.Data-logsout.Fright.Data)
    time = logsout.Fleft.Time;

    figure('Name','Chassis','NumberTitle','off')
    subplot(3,1,1),
    plot(time,logsout.Heave.Z.Data);
    ylabel('$$Vertical Motion$$ (in)','Interpreter','LaTex');
    %text(2.2, 0.03, 'dz/dt')
    subplot(3,1,2),
    plot(time,logsout.Pitch.Psi.Data);
    ylabel('$$Chassis Pitch$$ (rad)','Interpreter','LaTex');
    subplot(3,1,3),
    plot(time,logsout.Roll.Theta.Data);
    ylabel('$$Chassis Roll$$ (rad)','Interpreter','LaTex');

    figure('Name','Wheels','NumberTitle','off')
    subplot(4,1,1),
    plot(time,logsout.('Left Front Suspension').zlf.Data);
    ylabel('Z_l_f (in)');
    subplot(4,1,2),
    plot(time,logsout.('Right Front Suspension').zrf.Data);
    ylabel('Z_r_f (in)');
    subplot(4,1,3),
    plot(time,logsout.('Left Rear Suspension').zlr.Data);
    ylabel('Z_l_r (in)');
    subplot(4,1,4),
    plot(time,logsout.('Right Rear Suspension').zrr.Data);
    ylabel('Z_r_r (in)');


    figure('Name','Objective Function','NumberTitle','off')
    subplot(2,1,1),
    plot(time,logsout.Fleft.Data,time,logsout.Fright.Data);
    ylabel('$$Force$$ (lbf)','Interpreter','LaTex');
    %text(2.2,0.002,'d\theta/dt')

legend('\SigmaF_{Front}','\SigmaF_{Rear}','Orientation','Horizontal',..
.
        'Location','NorthEast')
    subplot(2,1,2)
    plot(time,logsout.Fleft.Data-logsout.Fright.Data);
    ylabel('$$Force$$ (lbf)','Interpreter','LaTex');
    title('7-DOF Suspension Model Simulation Results')



%% Mode Shapes
%%%coordinate order: (zlf,zlr,zrr,zrf,Z,Theta,Psi)
% M=[-bf-mf, 0, 0, 0, bf, bf*tf/2, -bf*Lf;  %Mass matrix (see 7DOF.nb)
%    0, -br-mr, 0, 0, br, br*tr/2, br*Lr;
%    0, 0, -br-mr, 0, br, -br*tr/2, br*Lr;
%    0, 0, 0, -bf-mf, bf, -bf*tf/2, -bf*Lf;

%    bf, br, br, bf, -bf-br-bf-br-Mb, -bf*tf/2-
br*tr/2+bf*tf/2+br*tr/2,...

%       bf*Lf+bf*Lf-br*Lr-br*Lr;

%   bf*tf/2, br*tr/2, -br*L*tr/2, -bf*tf/2, -bf*tf/2-
br*tr+bf*tf/2+br*tr/2,...

%       -Ixx-bf*(tf/2)^2-br*(tr/2)^2-bf*(tf/2)^2-br*(tr/2)^2,...

%       bf*Lf*tf/2-br*tr/2*Lr-bf*Lf*tf/2+br*Lr*tr/2;

%   -bf*Lf, br*Lr, br*Lr, -bf*Lf, bf*Lf+bf*Lf-br*Lr-br*Lr,...

%       bf*Lf*tf/2-br*tr/2*Lr-bf*Lf*tf/2+br*Lr*tr/2,...

%       -Iyy-bf*Lf^2-bf*Lf^2-br*Lr^2-br*Lr^2];

%

%  K=[-kf-kt, 0, 0, 0, kf, kf*tf/2, -kf*Lf;   %Stiffness matrix (see
7DOF.nb)

%    0, -kr-kt, 0, 0, kr, kr*tr/2, kr*Lr;

%    0, 0, -kr-kt, 0, kr, -kr*tr/2, kr*Lr;

%    0, 0, 0, -kf-kt, kf, -kf*tf/2, -kf*Lf;

%    kf, kr, kr, kf, -kf-kr-kf-kr, -kf*tf/2-kr*tr/2+kf*tf/2+kr*tr/2,...

%       kf*Lf+kf*Lf-kr*Lr-kr*Lr;

%    kf*tf/2, kr*tr/2, -kr*tr/2, -kf*tf/2, -kf*tf/2-kr*tr/2+kf*tf/2+...

%       kr*tr/2, -kf*(tf/2)^2-kr*(tr/2)^2-kf*(tf/2)^2-kr*(tr/2)^2,...

%       kf*Lf*tf/2-kr*tr/2*Lr-kf*Lf*tf/2+kr*Lr*tr/2;

%    -kf*Lf, kr*Lr, kr*Lr, -kf*Lf, kf*Lf+kf*Lf-kr*Lr-kr*Lr,...

%       kf*Lf*tf/2-kr*tr/2*Lr-kf*Lf*tf/2+kr*Lr*tr/2,...

%       -kf*Lf^2-kf*Lf^2-kr*Lr^2-kr*Lr^2];

%

% [eigenvectors,eigenvalues]=eig(K,M);   %"vector" columns are
eigenvectors

% %%%coordinate order: (zlf,zlr,zrr,zrf,Z,Theta,Psi)
% eigenvectors
% display('Nat. Freq.')
% sqrt(eigenvalues)


%% %%%%%%%%%%%%Build Root Locus w/ damping coef. as
gain%%%%%%%%%%%%%%%%%
% fin=(cf+cr)/2;    % final damping value
% n=fin/100;       % step size
% i=0;        % initialize index
%
% % create root locus
% for c=0:n:fin
%      cf=c;       % give suspension symmetry
%      cr=c;
%
%     [A,B,C,D]=linmod('fullcar_bode_analysis');      % find LTI model
%     sys1=ss(A,B,C,D);           % change LTI model to state space
%     [wn,zeta,p]=damp(sys1);     % find eigenvalues
%
%     i=i+1;       % increment index
% %    dampcoef(i,:)=c;        % collect "c's" used
%     re(i,:)=real(p); im(i,:)=imag(p);       % find Re(pole), Im(pole)
% %     natfreq(i,:)=wn;          % collect natural frequency value
% %     damprat(i,:)=zeta;        % collect damping ratio value
%
% %%variable step size
% %      if i>1
% %         if (sqrt((re(i)-re(i-1))^2+(im(i)-im(i-1))^2)>.2)
% %         n=2;
% %         cnt=cnt+1
% %         end
% %     end
% end
%
%% Create root locus figure
% fig=figure;
% dcm_obj=datacursormode(fig);
% set(dcm_obj,'UpdateFcn',@myupdatefcn)   %customized datatip function
% plot(re,im,'.b',re(1,:),im(1,:),'.g',re(i,:),im(i,:),'.r')
% axis([2*min(min(im)) 0 min(min(im)) max(max(im))])
% axis square
% % axis equal
% % axis tight
% sgrid


%%Use custom datatip
% xlimits=xlim;
% ylimits=ylim;
% display('Press any button to continue')
% while waitforbuttonpress==0
%     info_struct=getCursorInfo(dcm_obj);
%     ind=info_struct.DataIndex;
%     refreshdata
%     text(-45,20,['Damping Coefficient: ', num2str(dampcoef(ind))],...
%         'BackgroundColor',[1 1 .93],'EdgeColor',[.8 .8 .8])
% end
```

```
function
[c1f,c2f,vtf,c1r,c2r,vtr,bf,br]=fullcar_nonlindamper_optim(c1f0,c2f0,vt
f0,c1r0,c2r0,vtr0,bf0,br0)

global g tf tr Lf Lr mf mr Mb Ixx Iyy kf kr kt

%load model
nonlinfullcar

%initialize plant variables in model
bf=bf0;
br=br0;
[yf0,yr0,Z0,Theta0,Psi0]=EOMs;  % I.C.'s
c0=[c1f0,c2f0,vtf0,c1r0,c2r0,vtr0];%,c1r0,c2r0,vtr0];



options = optimset('Display','iter','TolX',.25,'TolFun',.05);%
coef = lsqnonlin(@track,c0,0,[],options);
c1f=coef(1); c2f=coef(2); vtf=coef(3);
c1r=coef(4); c2r=coef(5); vtr=coef(6);
% bf=coef(); br=coef()

    function F = track(coef)
                c1f=coef(1); c2f=coef(2); vtf=coef(3);
                c1r=coef(4); c2r=coef(5); vtr=coef(6);
%                 bf=coef(); br=coef()
        % Compute function value
        simopt=simset('SrcWorkspace','current');   % Initialize sim
options
        [t,x,y] = sim('nonlinfullcar',79.785,simopt);   % runtime =
79.7852s
        F = y(:,1)-y(:,2);
    end
end


clear
close all
clc


global g tf tr Lf Lr mf mr Mb Ixx Iyy kf kr kt vtf vtr


%%%%%%%%%%%%%%%%%%%%Initial Conditions &
Parameters%%%%%%%%%%%%%%%%%%%%%%
g=-32.17;
tf=50;                % front trackwidth
tr=48;                % rear trackwidth
fdist=0.48;           % front weight distribution
L=60;                 % wheelbse
Lf=(1-fdist)*L;       % left hub displacement from body CG (in)
Lr=fdist*L;           % right hub displacement from body CG (in)
mf=31/(-g);           % front/rear corner average unsprung mass
(lbf/grav)
mr=35/(-g);
Mb=680/(-g)-2*mf-2*mr;     % body mass +180lb driver in slugs
(lbf/grav)
```

```
Ixx=205559.63;      % body moment of inertia about roll-axis in lb*in^2

Iyy=2459865.33;     % body moment of inertia about pitch-axis in

lb*in^2

kf=148;             % front suspension wheelrate in lb/in

kr=112;             % rear suspension wheelrate in lb/in

kt=550;             % tire stiffness (lb/in)

IR_f=0.77;          % front installation ratio

IR_r=0.54;          % rear installation ratio

vtf=0.62;

vtr=0.67;


% optimization I.C.'s

bf=0;               % front inertance

br=0;               % rear inertance

c1f=417;%c0*IR_f^2;     % effective front damping (low speed)

c2f=4;          % effective front damping (high speed)

c1r=248;%c0*IR_r^2;     % effective rear damping (low speed)

c2r=256;          % effective rear damping (low speed)

[yf0,yr0,Z0,Theta0,Psi0]=EOMs;%(g,Mb,mf,mr,tf,tr,Lf,Lr,kf,kr,kt);  %

I.C.'s


%% optimization

%

[c1f,c2f,c1r,c2r,bf,br]=fullcar_nonlindamper_optim(c1f0,c2f0,c1r0,c2r0,

bf0,br0)

% cr=cl;

% br=bl;
```

```
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%Bode
etc.%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% halfcar_bode_analysis
% [A,B,C,D]=linmodv5('fullcar_bode_analysis');
% sys=ss(A,B,C,D);
% bode(sys)
% damp(sys)   %output Eigenvalues/Damping/Natural Frequency
% sldebug halfcar_bode_analysis %>>states


%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Signals%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
    sim('nonlinfullcar_postopt_analysis',79.785);    %Create logsout
structure file
    display('Norm of weight transfer vectors')
    norm(logsout.Fleft.Data-logsout.Fright.Data)
    time = logsout.Fleft.Time;


    figure('Name','Chassis','NumberTitle','off')
      subplot(3,1,1),
          plot(time,logsout.Heave.Z.Data);
          ylabel('$$Vertical Motion$$ (in)','Interpreter','LaTex');
          %text(2.2, 0.03, 'dz/dt')
      subplot(3,1,2),
          plot(time,logsout.Pitch.Psi.Data);
          ylabel('$$Chassis Pitch$$ (rad)','Interpreter','LaTex');
      subplot(3,1,3),
          plot(time,logsout.Roll.Theta.Data);
```

```
            ylabel('$$Chassis Roll$$ (rad)','Interpreter','LaTex');


      figure('Name','Wheels','NumberTitle','off')
        subplot(4,1,1),
            plot(time,logsout.('Left Front Suspension').zlf.Data);
            ylabel('Z_l_f (in)');
        subplot(4,1,2),
            plot(time,logsout.('Right Front Suspension').zrf.Data);
            ylabel('Z_r_f (in)');
        subplot(4,1,3),
            plot(time,logsout.('Left Rear Suspension').zlr.Data);
            ylabel('Z_l_r (in)');
        subplot(4,1,4),
            plot(time,logsout.('Right Rear Suspension').zrr.Data);
            ylabel('Z_r_r (in)');


%     figure('Name','Velocities','NumberTitle','off')
%         subplot(4,1,1),
%             plot(time,logsout.('Left Front
Suspension').LFShaftVelocity.Data,...
%                 time,logsout.('Left Front Suspension').zlfdot.Data);
%             legend('Shaft Velocity','Wheel Velocity')
%             ylabel('Left Front');
%         subplot(4,1,2),
%             plot(time,logsout.('Right Front
Suspension').RFShaftVelocity.Data,...
%                 time,logsout.('Right Front Suspension').zrfdot.Data);
%             ylabel('Right Front');
%         subplot(4,1,3),
%             plot(time,logsout.('Left Rear
Suspension').LRShaftVelocity.Data,...
%                 time,logsout.('Left Rear Suspension').zlrdot.Data);
%             ylabel('Left Rear');
%         subplot(4,1,4),
%             plot(time,logsout.('Right Rear
Suspension').RRShaftVelocity.Data,...
%                 time,logsout.('Right Rear Suspension').zrrdot.Data);
%             ylabel('Right Rear');

    figure('Name','Objective Function','NumberTitle','off')
      subplot(2,1,1),
          plot(time,logsout.Fleft.Data,time,logsout.Fright.Data);
          ylabel('$$Force$$ (lbf)','Interpreter','LaTex');
          %text(2.2,0.002,'d\theta/dt')

legend('\SigmaF_{Front}','\SigmaF_{Rear}','Orientation','Horizontal',..
.
        'Location','NorthEast')
      subplot(2,1,2)
          plot(time,logsout.Fleft.Data-logsout.Fright.Data);
          ylabel('$$Force$$ (lbf)','Interpreter','LaTex');
          title('7-DOF Suspension Model Simulation Results')



%% Mode Shapes
%%coordinate order: (zlf,zlr,zrr,zrf,Z,Theta,Psi)
```

```
M=[-bf-mf, 0, 0, 0, bf, bf*tf/2, -bf*Lf;  %Mass matrix (see 7DOF.nb)
   0, -br-mr, 0, 0, br, br*tr/2, br*Lr;
   0, 0, -br-mr, 0, br, -br*tr/2, br*Lr;
   0, 0, 0, -bf-mf, bf, -bf*tf/2, -bf*Lf;
   bf, br, br, bf, -bf-br-bf-br-Mb, -bf*tf/2-
br*tr/2+bf*tf/2+br*tr/2,...
      bf*Lf+bf*Lf-br*Lr-br*Lr;
  bf*tf/2, br*tr/2, -br*L*tr/2, -bf*tf/2, -bf*tf/2-
br*tr+bf*tf/2+br*tr/2,...
      -Ixx-bf*(tf/2)^2-br*(tr/2)^2-bf*(tf/2)^2-br*(tr/2)^2,...
      bf*Lf*tf/2-br*tr/2*Lr-bf*Lf*tf/2+br*Lr*tr/2;
  -bf*Lf, br*Lr, br*Lr, -bf*Lf, bf*Lf+bf*Lf-br*Lr-br*Lr,...
      bf*Lf*tf/2-br*tr/2*Lr-bf*Lf*tf/2+br*Lr*tr/2,...
      -Iyy-bf*Lf^2-bf*Lf^2-br*Lr^2-br*Lr^2];


 K=[-kf-kt, 0, 0, 0, kf, kf*tf/2, -kf*Lf;   %Stiffness matrix (see
7DOF.nb)
   0, -kr-kt, 0, 0, kr, kr*tr/2, kr*Lr;
   0, 0, -kr-kt, 0, kr, -kr*tr/2, kr*Lr;
   0, 0, 0, -kf-kt, kf, -kf*tf/2, -kf*Lf;
   kf, kr, kr, kf, -kf-kr-kf-kr, -kf*tf/2-kr*tr/2+kf*tf/2+kr*tr/2,...
      kf*Lf+kf*Lf-kr*Lr-kr*Lr;
   kf*tf/2, kr*tr/2, -kr*tr/2, -kf*tf/2, -kf*tf/2-kr*tr/2+kf*tf/2+...
      kr*tr/2, -kf*(tf/2)^2-kr*(tr/2)^2-kf*(tf/2)^2-kr*(tr/2)^2,...
      kf*Lf*tf/2-kr*tr/2*Lr-kf*Lf*tf/2+kr*Lr*tr/2;
   -kf*Lf, kr*Lr, kr*Lr, -kf*Lf, kf*Lf+kf*Lf-kr*Lr-kr*Lr,...
      kf*Lf*tf/2-kr*tr/2*Lr-kf*Lf*tf/2+kr*Lr*tr/2,...
      -kf*Lf^2-kf*Lf^2-kr*Lr^2-kr*Lr^2];
```

```
[eigenvectors,eigenvalues]=eig(K,M);   %"vector" columns are
eigenvectors
%%%coordinate order: (zlf,zlr,zrr,zrf,Z,Theta,Psi)
eigenvectors
display('Nat. Freq.')
sqrt(eigenvalues)


%% %%%%%%%%%%%%Build Root Locus w/ damping coef. as
gain%%%%%%%%%%%%%%%%%
fin=(c1f+c1r)/2;    % final damping value
n=fin/100;       % step size
i=0;        % initialize index


% create root locus
for c=0:n:fin
     c1f=c;       % give suspension symmetry
     c1r=c;


    [A,B,C,D]=linmod('nonlinfullcar_bode_analysis');      % find LTI
model
    sys1=ss(A,B,C,D);           % change LTI model to state space
    [wn,zeta,p]=damp(sys1);     % find eigenvalues


    i=i+1;       % increment index
    dampcoef(i,:)=c;        % collect "c's" used
    re(i,:)=real(p); im(i,:)=imag(p);       % find Re(pole), Im(pole)
    natfreq(i,:)=wn;          % collect natural frequency value
    damprat(i,:)=zeta;        % collect damping ratio value

%variable step size
%      if i>1
%         if (sqrt((re(i)-re(i-1))^2+(im(i)-im(i-1))^2)>.2)
%         n=2;
%         cnt=cnt+1
%         end
%     end
end
%
%% Create root locus figure
fig=figure;
% dcm_obj=datacursormode(fig);
% set(dcm_obj,'UpdateFcn',@myupdatefcn)   %customized datatip function
plot(re,im,'.b',re(1,:),im(1,:),'.g',re(i,:),im(i,:),'.r')
axis([2*min(min(im)) 0 min(min(im)) max(max(im))])
axis square
% axis equal
% axis tight
sgrid

% %Use custom datatip
% xlimits=xlim;
% ylimits=ylim;
% display('Press any button to continue')
% while waitforbuttonpress==0
%     info_struct=getCursorInfo(dcm_obj);
%     ind=info_struct.DataIndex;

%     refreshdata

%     text(-45,20,['Damping Coefficient: ', num2str(dampcoef(ind))],...

%         'BackgroundColor',[1 1 .93],'EdgeColor',[.8 .8 .8])

% end


function

[c1f,c2f,c1r,c2r,bf,br]=fullcar_nonlindamper_optim(c1f0,c2f0,c1r0,c2r0,

bf0,br0)


global g tf tr Lf Lr mf mr Mb Ixx Iyy kf kr kt vtf vtr


%load model

nonlinfullcar


%initialize plant variables in model

bf=bf0;

br=br0;

[yf0,yr0,Z0,Theta0,Psi0]=EOMs;  % I.C.'s

c0=[c1f0,c2f0,c1r0,c2r0];


options =

optimset('Display','iter','TolX',.25,'TolFun',.05,'GradObj','on');%

coef = lsqnonlin(@track,c0,0,[],options);

c1f=coef(1); c2f=coef(2);

c1r=coef(3); c2r=coef(4);

% bf=coef(); br=coef()
```

```
    function F = track(coef)
                c1f=coef(1); c2f=coef(2);
                c1r=coef(3); c2r=coef(4);
%                 bf=coef(); br=coef()
        % Compute function value
        simopt=simset('SrcWorkspace','current');   % Initialize sim
options
        [t,x,y] = sim('nonlinfullcar',79.785,simopt);   % runtime =
79.7852s
        F = y(:,1)-y(:,2);
    end
end


clear
close all
clc


global g tf tr Lf Lr mf mr Mb Ixx Iyy kf kr kt vtf vtr c1f c2f c1r c2r


%%%%%%%%%%%%%%%%%%%%Initial Conditions &
Parameters%%%%%%%%%%%%%%%%%%%%%%
g=-32.17;
tf=50;              % front trackwidth
tr=48;              % rear trackwidth
fdist=0.48;         % front weight distribution
L=60;               % wheelbse
Lf=(1-fdist)*L;     % left hub displacement from body CG (in)
```

```
Lr=fdist*L;          % right hub displacement from body CG (in)

mf=31/(-g);          % front/rear corner average unsprung mass

(lbf/grav)

mr=35/(-g);

Mb=680/(-g)-2*mf-2*mr;     % body mass +180lb driver in slugs

(lbf/grav)

Ixx=205559.63;       % body moment of inertia about roll-axis in lb*in^2

Iyy=2459865.33;      % body moment of inertia about pitch-axis in

lb*in^2

kf=148;              % front suspension wheelrate in lb/in

kr=112;              % rear suspension wheelrate in lb/in

kt=550;              % tire stiffness (lb/in)

IR_f=0.77;           % front installation ratio

IR_r=0.54;           % rear installation ratio


% optimization I.C.'s

bf0=2.5;                % front inertance

br0=2.5;                % rear inertance

c1f=417.2874;         % effective front damping (low speed)

c2f=3.9375;           % effective front damping (high speed)

vtf=0.62;

c1r=255.5894;         % effective rear damping (low speed)

c2r=8.0028;           % effective rear damping (low speed)

vtr=0.67;

[yf0,yr0,Z0,Theta0,Psi0]=EOMs;%(g,Mb,mf,mr,tf,tr,Lf,Lr,kf,kr,kt);  %

I.C.'s


%% optimization
```

```
[bf,br]=fullcar_inerter_optim(bf0,br0)
% cr=cl;
% br=bl;


%% %%%%%%%%%%%%%%%%%%%%%%%%%%%Bode
etc.%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% halfcar_bode_analysis
% [A,B,C,D]=linmodv5('fullcar_bode_analysis');
% sys=ss(A,B,C,D);
% bode(sys)
% damp(sys)   %output Eigenvalues/Damping/Natural Frequency
% sldebug halfcar_bode_analysis %>>states


%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Signals%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
    sim('nonlinfullcar_postopt_analysis',79.785);    %Create logsout
structure file
    display('Norm of weight transfer vectors')
    norm(logsout.Fleft.Data-logsout.Fright.Data)
    time = logsout.Fleft.Time;


%     figure('Name','Chassis','NumberTitle','off')
%       subplot(3,1,1),
%           plot(time,logsout.Heave.Z.Data);
%           ylabel('$$Vertical Motion$$ (in)','Interpreter','LaTex');
%           %text(2.2, 0.03, 'dz/dt')
%       subplot(3,1,2),
%           plot(time,logsout.Pitch.Psi.Data);
%           ylabel('$$Chassis Pitch$$ (rad)','Interpreter','LaTex');
%       subplot(3,1,3),
%           plot(time,logsout.Roll.Theta.Data);
%           ylabel('$$Chassis Roll$$ (rad)','Interpreter','LaTex');

%     figure('Name','Wheels','NumberTitle','off')
%       subplot(4,1,1),
%           plot(time,logsout.('Left Front Suspension').zlf.Data);
%           ylabel('Z_l_f (in)');
%       subplot(4,1,2),
%           plot(time,logsout.('Right Front Suspension').zrf.Data);
%           ylabel('Z_r_f (in)');
%       subplot(4,1,3),
%           plot(time,logsout.('Left Rear Suspension').zlr.Data);
%           ylabel('Z_l_r (in)');
%       subplot(4,1,4),
%           plot(time,logsout.('Right Rear Suspension').zrr.Data);
%           ylabel('Z_r_r (in)');

    figure('Name','Velocities','NumberTitle','off')
        subplot(4,1,1),
            plot(time,logsout.('Left Front
Suspension').LFShaftVelocity.Data,...
                time,logsout.('Left Front Suspension').zlfdot.Data);
            legend('Shaft Velocity','Wheel Velocity')
            ylabel('Left Front');
        subplot(4,1,2),
            plot(time,logsout.('Right Front
Suspension').RFShaftVelocity.Data,...
                time,logsout.('Right Front Suspension').zrfdot.Data);
            ylabel('Right Front');
        subplot(4,1,3),
            plot(time,logsout.('Left Rear
Suspension').LRShaftVelocity.Data,...
                time,logsout.('Left Rear Suspension').zlrdot.Data);
            ylabel('Left Rear');
        subplot(4,1,4),
            plot(time,logsout.('Right Rear
Suspension').RRShaftVelocity.Data,...
                time,logsout.('Right Rear Suspension').zrrdot.Data);
            ylabel('Right Rear');


%     figure('Name','Objective Function','NumberTitle','off')
%       subplot(2,1,1),
%           plot(time,logsout.Fleft.Data,time,logsout.Fright.Data);
%           ylabel('$$Force$$ (lbf)','Interpreter','LaTex');
%           %text(2.2,0.002,'d\theta/dt')
%
legend('\SigmaF_{Front}','\SigmaF_{Rear}','Orientation','Horizontal',..
.
%         'Location','NorthEast')
%       subplot(2,1,2)
%           plot(time,logsout.Fleft.Data-logsout.Fright.Data);
%           ylabel('$$Force$$ (lbf)','Interpreter','LaTex');
%           title('7-DOF Suspension Model Simulation Results')
```

```
%% Mode Shapes
%%coordinate order: (zlf,zlr,zrr,zrf,Z,Theta,Psi)
% M=[-bf-mf, 0, 0, 0, bf, bf*tf/2, -bf*Lf;  %Mass matrix (see 7DOF.nb)
%    0, -br-mr, 0, 0, br, br*tr/2, br*Lr;
%    0, 0, -br-mr, 0, br, -br*tr/2, br*Lr;
%    0, 0, 0, -bf-mf, bf, -bf*tf/2, -bf*Lf;
%    bf, br, br, bf, -bf-br-bf-br-Mb, -bf*tf/2-
br*tr/2+bf*tf/2+br*tr/2,...
%       bf*Lf+bf*Lf-br*Lr-br*Lr;
%   bf*tf/2, br*tr/2, -br*L*tr/2, -bf*tf/2, -bf*tf/2-
br*tr+bf*tf/2+br*tr/2,...
%       -Ixx-bf*(tf/2)^2-br*(tr/2)^2-bf*(tf/2)^2-br*(tr/2)^2,...
%       bf*Lf*tf/2-br*tr/2*Lr-bf*Lf*tf/2+br*Lr*tr/2;
%   -bf*Lf, br*Lr, br*Lr, -bf*Lf, bf*Lf+bf*Lf-br*Lr-br*Lr,...
%       bf*Lf*tf/2-br*tr/2*Lr-bf*Lf*tf/2+br*Lr*tr/2,...
%       -Iyy-bf*Lf^2-bf*Lf^2-br*Lr^2-br*Lr^2];
%
%  K=[-kf-kt, 0, 0, 0, kf, kf*tf/2, -kf*Lf;   %Stiffness matrix (see
7DOF.nb)
%    0, -kr-kt, 0, 0, kr, kr*tr/2, kr*Lr;
%    0, 0, -kr-kt, 0, kr, -kr*tr/2, kr*Lr;
%    0, 0, 0, -kf-kt, kf, -kf*tf/2, -kf*Lf;
%    kf, kr, kr, kf, -kf-kr-kf-kr, -kf*tf/2-kr*tr/2+kf*tf/2+kr*tr/2,...
%       kf*Lf+kf*Lf-kr*Lr-kr*Lr;
%    kf*tf/2, kr*tr/2, -kr*tr/2, -kf*tf/2, -kf*tf/2-kr*tr/2+kf*tf/2+...
%       kr*tr/2, -kf*(tf/2)^2-kr*(tr/2)^2-kf*(tf/2)^2-kr*(tr/2)^2,...
%       kf*Lf*tf/2-kr*tr/2*Lr-kf*Lf*tf/2+kr*Lr*tr/2;
%    -kf*Lf, kr*Lr, kr*Lr, -kf*Lf, kf*Lf+kf*Lf-kr*Lr-kr*Lr,...
%       kf*Lf*tf/2-kr*tr/2*Lr-kf*Lf*tf/2+kr*Lr*tr/2,...
%       -kf*Lf^2-kf*Lf^2-kr*Lr^2-kr*Lr^2];
%
% [eigenvectors,eigenvalues]=eig(K,M);   %"vector" columns are
eigenvectors
% %%%coordinate order: (zlf,zlr,zrr,zrf,Z,Theta,Psi)
% eigenvectors
% display('Nat. Freq.')
% sqrt(eigenvalues)


%% %%%%%%%%%%%%Build Root Locus w/ damping coef. as
gain%%%%%%%%%%%%%%%%%
% fin=(c1f+c1r)/2;    % final damping value
% n=fin/100;       % step size
% i=0;        % initialize index
%
% % create root locus
% for c=0:n:fin
%      c1f=c;       % give suspension symmetry
%      c1r=c;
%
%     [A,B,C,D]=linmod('nonlinfullcar_bode_analysis');      % find LTI
model
%     sys1=ss(A,B,C,D);           % change LTI model to state space
%     [wn,zeta,p]=damp(sys1);     % find eigenvalues
%
```

```
%     i=i+1;       % increment index
%     dampcoef(i,:)=c;        % collect "c's" used
%     re(i,:)=real(p); im(i,:)=imag(p);       % find Re(pole), Im(pole)
%     natfreq(i,:)=wn;          % collect natural frequency value
%     damprat(i,:)=zeta;        % collect damping ratio value
%
% %variable step size
% %      if i>1
% %         if (sqrt((re(i)-re(i-1))^2+(im(i)-im(i-1))^2)>.2)
% %         n=2;
% %         cnt=cnt+1
% %         end
% %     end
% end


%% Create root locus figure
% fig=figure;
% % dcm_obj=datacursormode(fig);
% % set(dcm_obj,'UpdateFcn',@myupdatefcn)   %customized datatip
function
% plot(re,im,'.b',re(1,:),im(1,:),'.g',re(i,:),im(i,:),'.r')
% axis([2*min(min(im)) 0 min(min(im)) max(max(im))])
% axis square
% % axis equal
% % axis tight
% sgrid
%
% % %Use custom datatip
% % xlimits=xlim;
% % ylimits=ylim;
% % display('Press any button to continue')
% % while waitforbuttonpress==0
% %     info_struct=getCursorInfo(dcm_obj);
% %     ind=info_struct.DataIndex;
% %     refreshdata
% %     text(-45,20,['Damping Coefficient: ',
num2str(dampcoef(ind))],...
% %         'BackgroundColor',[1 1 .93],'EdgeColor',[.8 .8 .8])
% % end


function [bf,br]=fullcar_inerter_optim(bf0,br0)


global g tf tr Lf Lr mf mr Mb Ixx Iyy kf kr kt vtf vtr c1f c2f c1r c2r


%load model
nonlinfullcar


%initialize plant variables in model
[yf0,yr0,Z0,Theta0,Psi0]=EOMs;  % I.C.'s
b0=[bf0,br0];


options =
optimset('Display','iter','TolX',.25,'TolFun',.05,'GradObj','on');%
coef = lsqnonlin(@track,b0,0,[],options);
bf=coef(1); br=coef(2);
```

```
    function F = track(coef)

                bf=coef(1); br=coef(2);

        % Compute function value
        simopt=simset('SrcWorkspace','current');   % Initialize sim
options
        [t,x,y] = sim('nonlinfullcar',79.785,simopt);   % runtime =
79.7852s
        F = y(:,1)-y(:,2);
    end
end
```

Appendix B: Simulink Block Diagrams

This appendix contains the block diagrams used in the Simulink models for each suspension system.

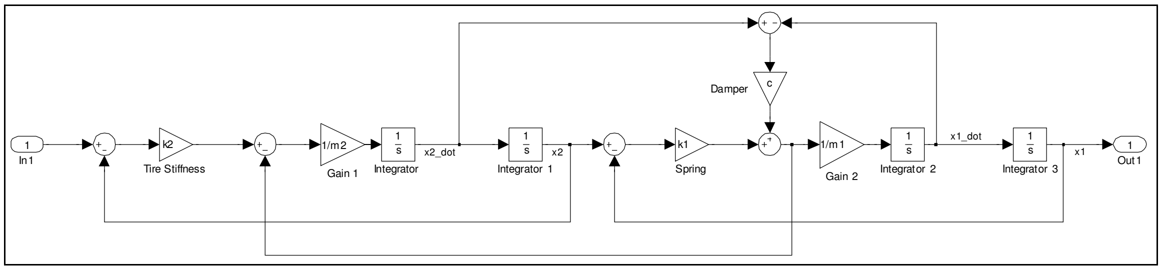

Figure B.1 – 2DOF Quarter-Suspension Model

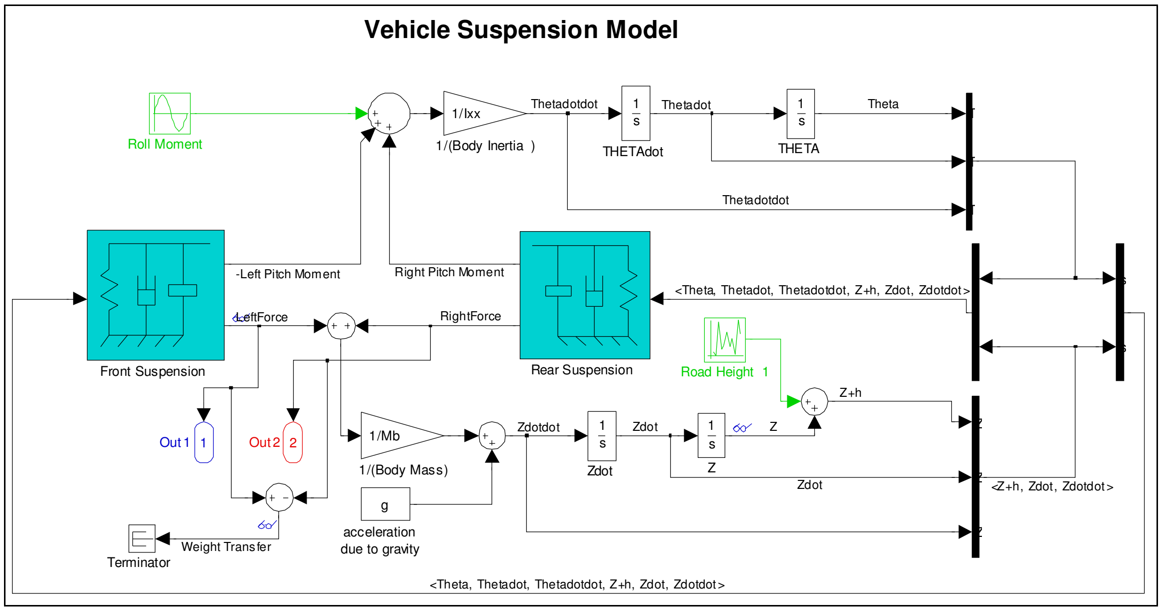

Figure B.2 – 2DOF Roll Model

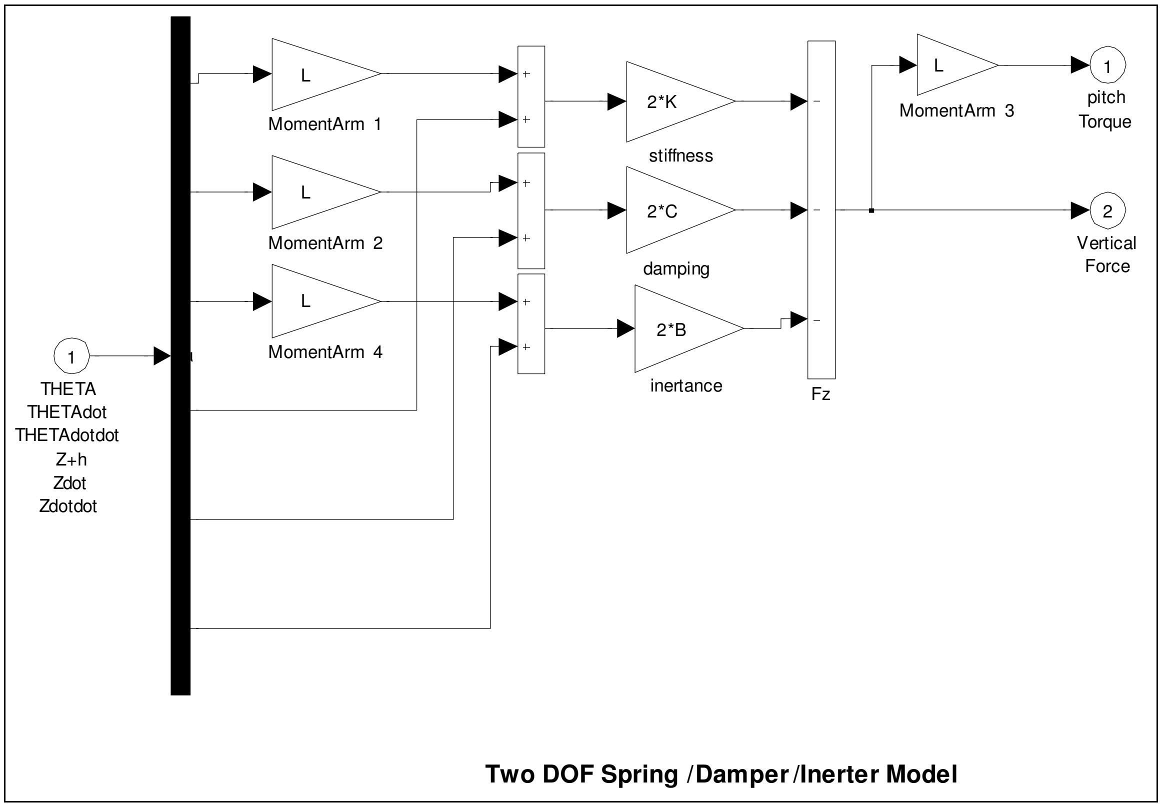

Figure B.3 – 2DOF Roll Suspension Subsystem

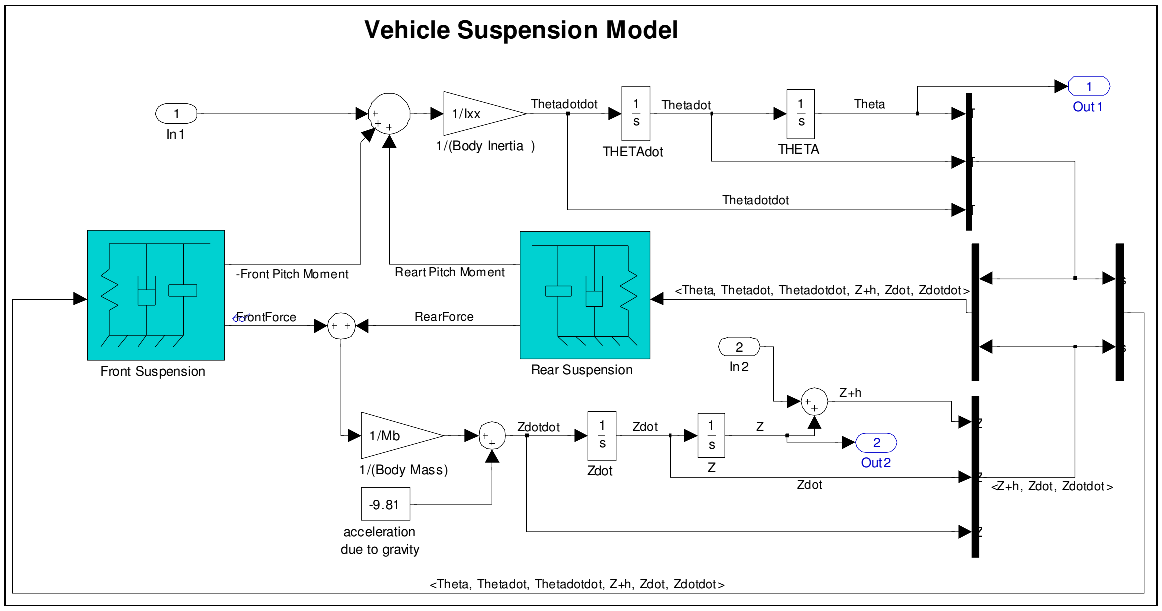

Figure B.4 – 2DOF Roll Model for Root Locus Analysis

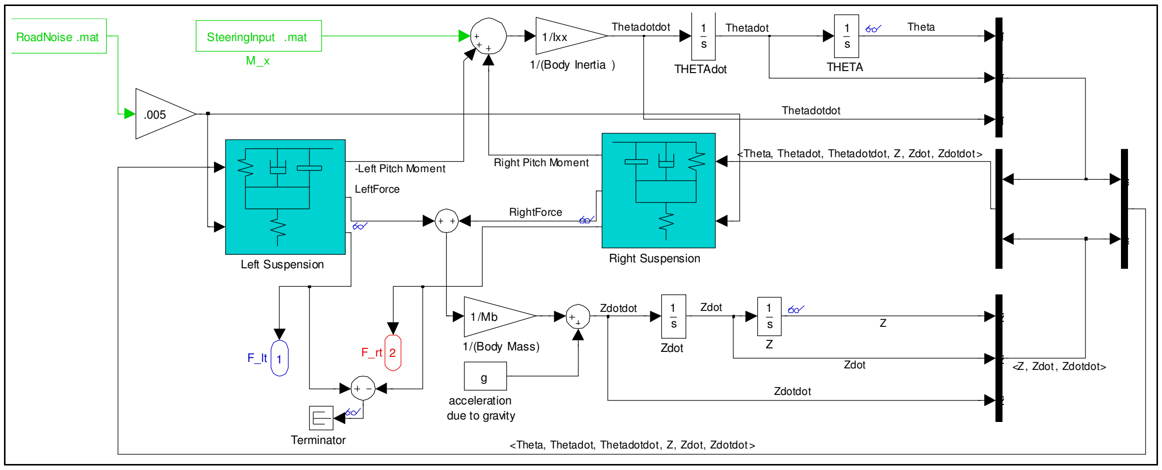

Figure B.5 – 4DOF Model

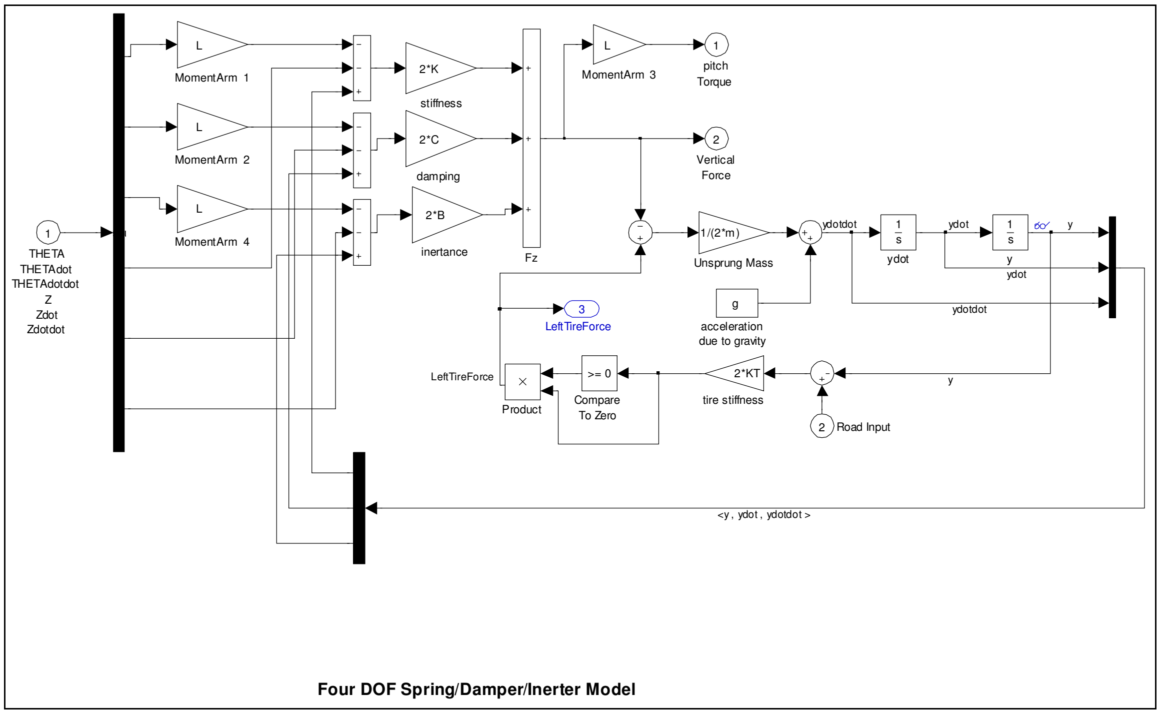

Figure B.6 – 4DOF Suspension Subsystem

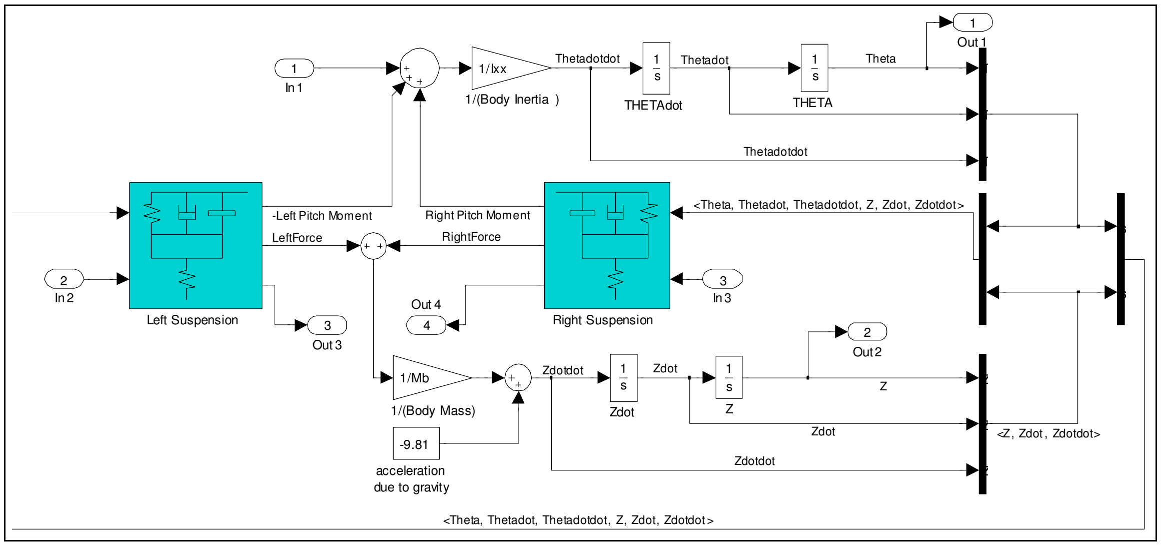

Figure B.7 – 4DOF Model for Root Locus Analysis

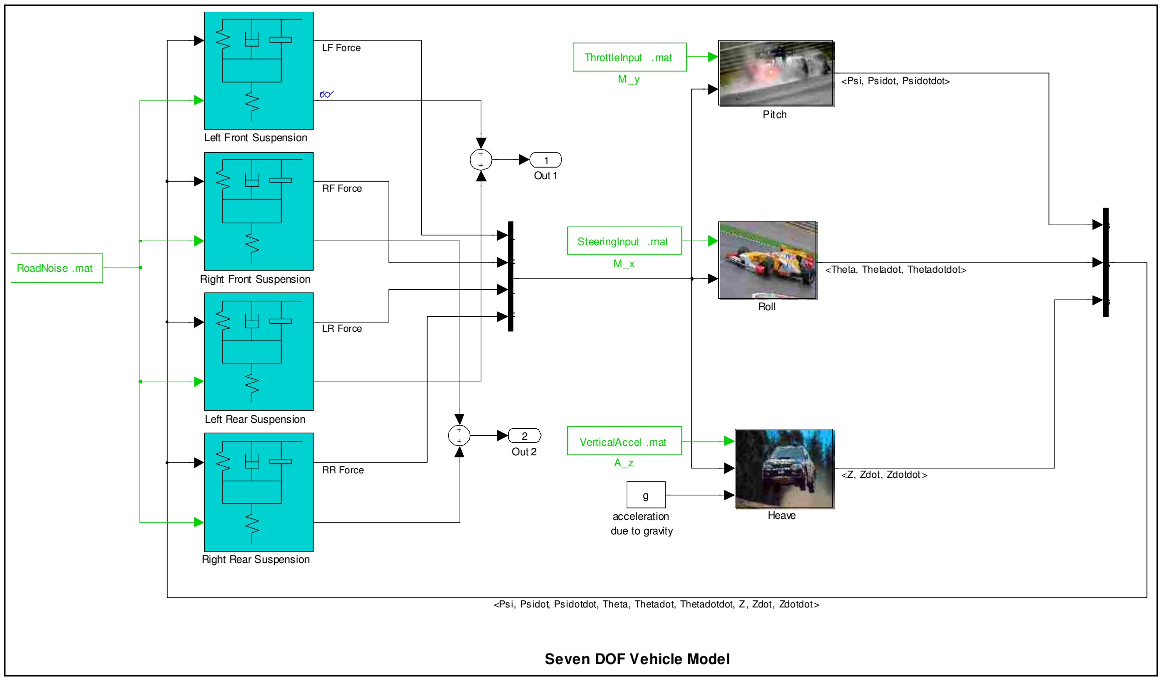

Figure B.8 – 7DOF Model

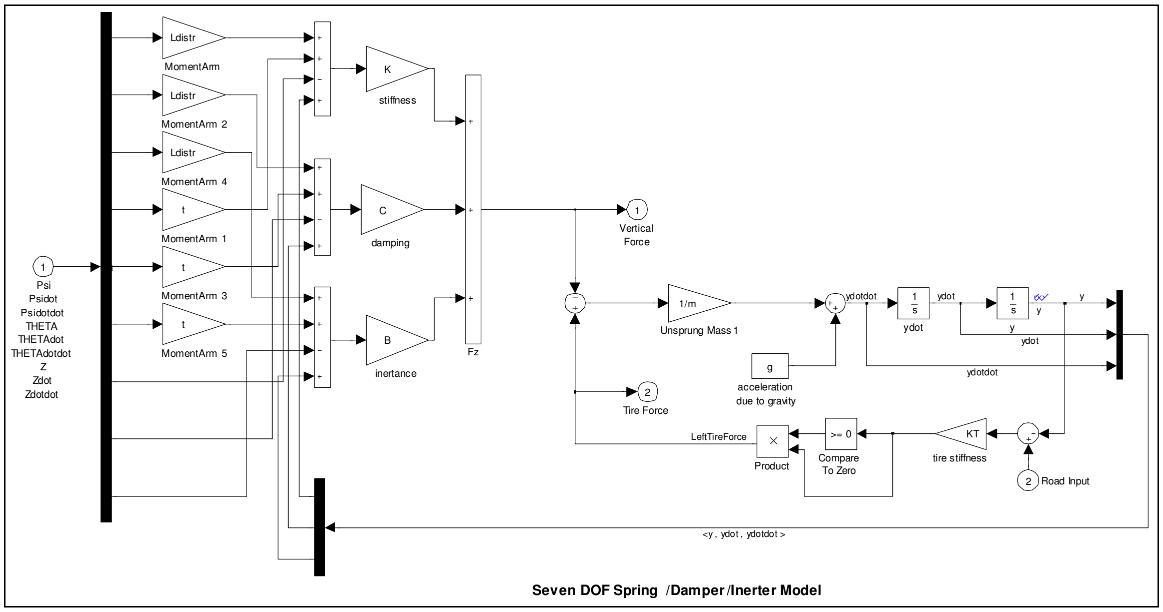

Figure B.9 – 7DOF Suspension Subsystem with Linear Damper

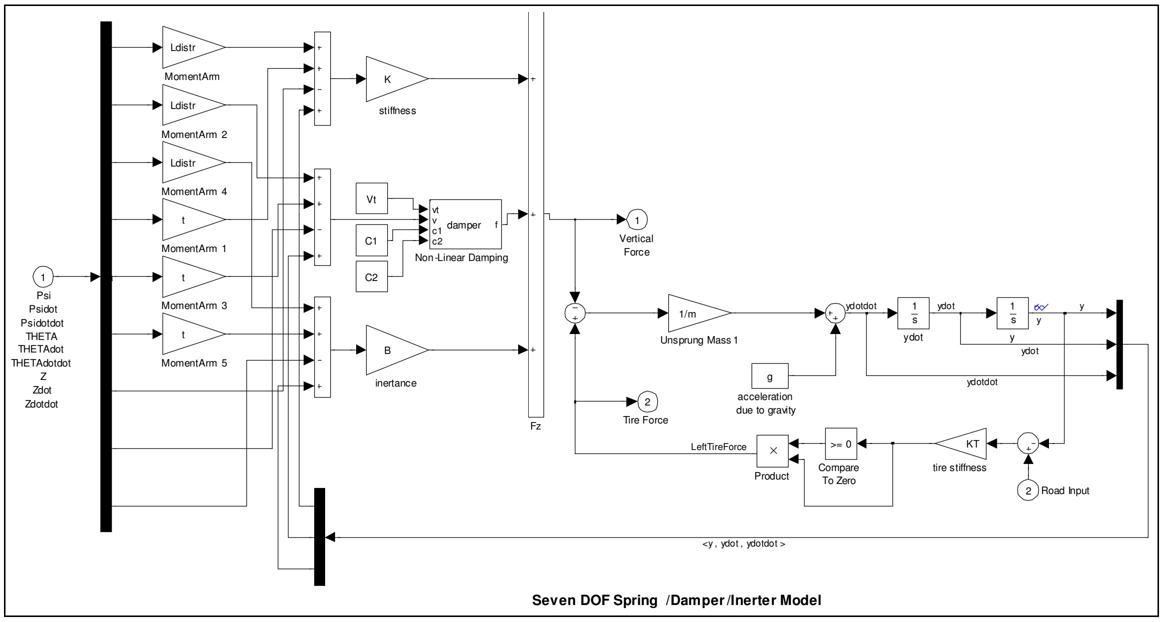

Figure B.10 – 7DOF Suspension Subsystem with Nonlinear Damper

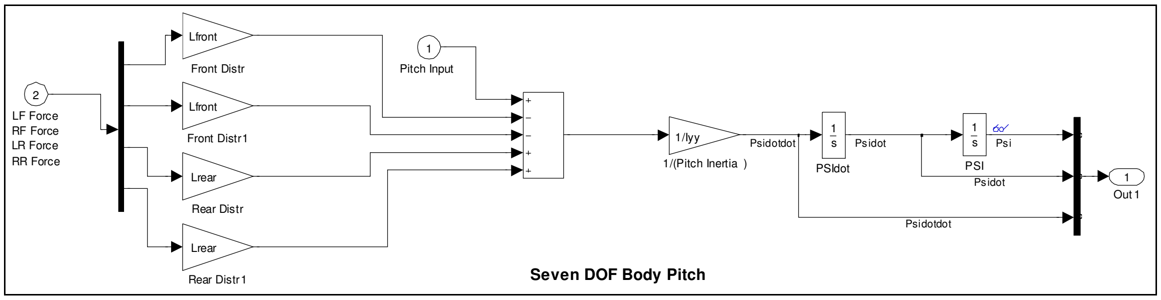

Figure B.11 – 7DOF Chassis Pitch Subsystem

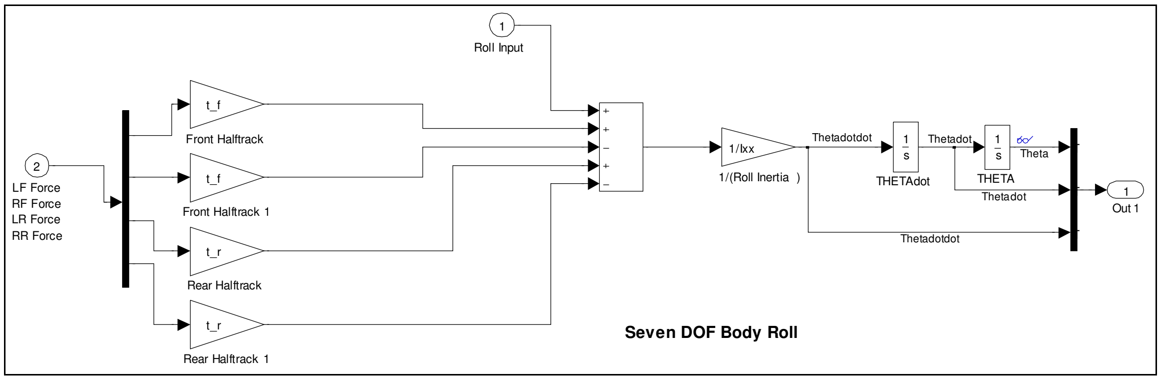

Figure B.12 – 7DOF Chassis Roll Subsystem

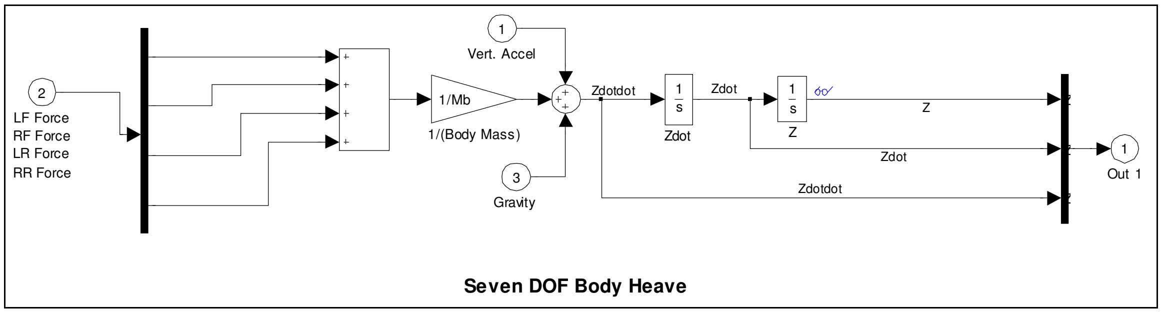

Figure B.13 – 7DOF Chassis Heave Subsystem

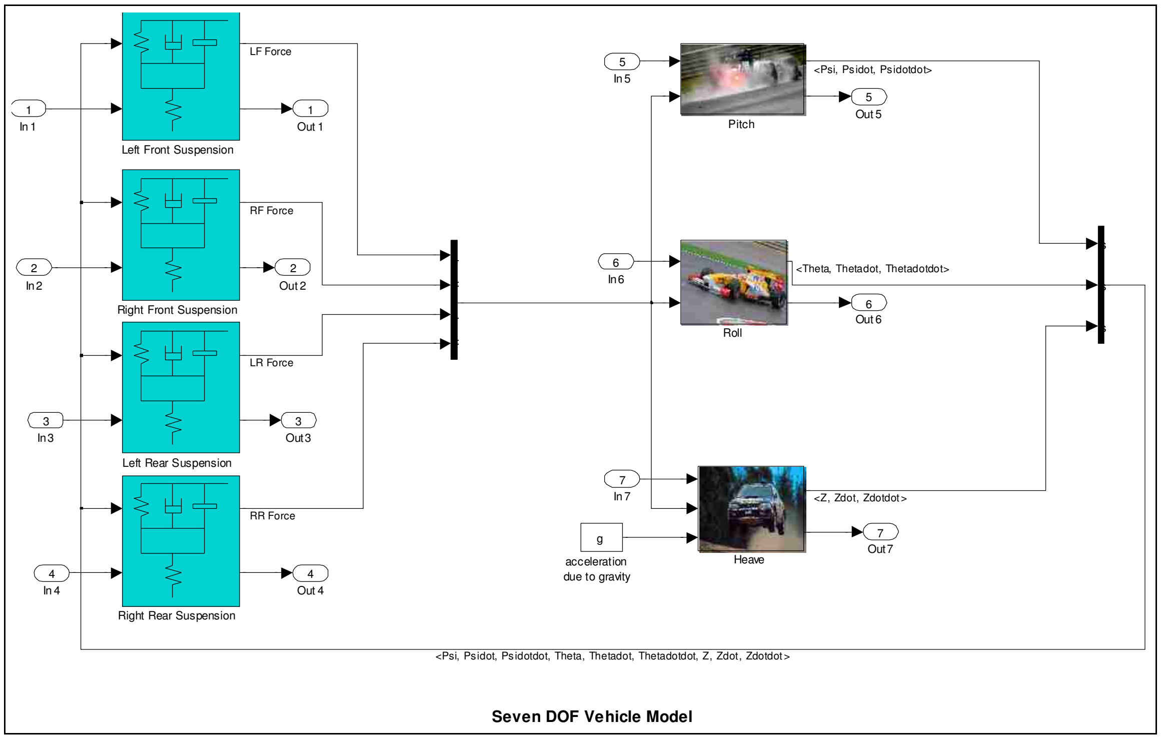

Figure B.14 – 7DOF Model for Root Locus Analysis

Appendix C: 2007 UNM FSAE Design Specification Sheet

This appendix contains the parameters used in the suspension models and optimization.

Table C.1 – 2007 FSAE Spec Sheet

FSAE Design Spec Sheet 2007

Competitors please replace the sample specification values in the table below with those appropriate for your vehicle and submit this to with your design report. This information will be reviewed by the design judges and may be referred to during the design event.

--Please do not modify format of this sheet. Common formatting will help keep the judges happy!

--The sample values are fictional and should be used as a baseline for your designs.

Car No.	71
School	University of New Mexico

Dimensions	**Front**	**Rear**
Overall Length, Width, Height	107.5 inches, 57.0 inches, 45.0 inches	
Wheelbase	60 inches	
Track	50 inches	48 inches
Weight with 150lb driver	310 lbs	340 lbs

Suspension Parameters	**Front**	**Rear**
Suspension Type	Short-Long A-Arm. Push rod actuated semi-horizontally oriented spring and damper	Sort-Long A-Arm. Push rod actuated semi-vertically oriented spring and damper
Tire Size and Compound Type	20.5x7.0-13 R25A Hoosier	20x7.5-13 R25A Hoosier
Wheels	6 inch wide, 2 pc Al Rim, 1 inch neg. offset	6 inch wide, 2 pc Al Rim, 1 inch neg. offset
Design ride height (chassis to ground)	3.5 inches	3.3 inches
Center of Gravity Design Height	12 inches above ground	
Suspension design travel	1.1 inches jounce/ 1.1 inch rebound	1.4 inches jounce/ 1.4 inch rebound
Wheel rate (chassis to wheel center)	148 lb/in	126 lb/in
Roll rate (chassis to wheel center)	0.8 degrees per g	
Sprung mass natural frequency	2.94 Hz	2.6 Hz
Jounce Damping	Designed to exploit tire traction	Designed to exploit tire traction

Rebound Damping	Designed to exploit tire traction	Designed to exploit tire traction
Motion ratio / type	0.77 / progressive	0.53 / progressive rate
Camber coefficient in bump (deg / in)	1.5 deg / in	1.7 deg / in
Camber coefficient in roll (deg / deg)	.34 deg / deg	.30 deg /deg
Static Toe and adjustment method	0.25 to 0.5 inch toe out via adj steering links	0 inch toe in via adj toe links
Static camber and adjustment method	'-0.5 deg via rod end adjustment	0 deg
Front Caster and adjustment method	8.2 degrees adjustable	
Front Kingpin Axis	7.6 degrees adjustable	
Kingpin offset and trail	.9 inches offset .5 inches trail	
Static Akermann and adjustment method	60% adjustable	
Anti dive / Anti Squat	0% adjustable	0% adjustable
Roll center position static	1.8 inches above ground	1.6 inches above ground
Roll center position at 1g lateral acc	.4 inches above ground, 5.7 inches toward unladen side	0.2 inches above ground, 11.7 inches toward laden side
Steer location, Gear ratio, Steer Arm Length	Front steer inplane with lower A-arm, 3.33" c-factor, 3.72" steer arm	

7 REFERENCES

1) Milliken and Milliken, “Race Car Vehicle Dynamics,” Society of Automotive Engineers, Inc. Warrendale, PA, 1995.

2) Armstrong-Wilson, Charles, “How J Damper really work,” Racecar Engineering. 8 Mar 2008.

3) Lacroix, Benoit et al, “A Passive Nonlinear Damping Design for a Road Race Car Application.” Society of Automotive Engineers, Warrendale, PA, 2006

4) Fukushima, N., K. Hidaka, and K. Iwata, “Optimum Characteristics of Automotive Shock Absorbers under Various Driving Conditions and Road Surfaces,” *JSAE Review*, March 1983.

5) Sugasawa, F., et al., “Electronically Controlled Shock Absorber System Used as a Road Sensor which Utilizes Supersonic Waves,” SAE Paper No. 851652, Society of Automotive Engineers, Warrendale, PA, 1985.

6) Rill, G., “Steady-State Cornering on Uneven Roadways,” SAE Paper No. 860575, Society of Automotive Engineers, Warrendale, PA, 1986.

7) P.S. Els, et al. “Gradient-based approximation methods applied to the optimal design of vehicle suspension systems using computational models with severe inherent noise.”

8) The MathWorks, “Penske Technology Group Enables Motorsport Teams to Improve Vehicle Performance with MathWorks Tools,” *User Story*, Paper No. 91351v01, June 2009.

9) Smith, Malcolm C. and Wang, Fu-Cheng, “Performance Benefits in Passive Vehicle Suspensions Employing Inerters.” University of Cambridge, 2004.

10) Palm, William J., “Mechanical Vibration.” 1st editionWiley, John & Sons Incorporated. Hoboken, NJ, 2006.

11) The MathWorks, “Using Simulink® and Stateflow™ in Automotive Applications.” *Simulink-Stateflow Technical Examples*, 1998: pp. 32-34.

12) Press, William H., et al. “Numerical Recipes in Fortran 77: The Art of Scientific Computing,” Second Edition (1992). Cambridge University Press, Copyright 1986-1992.