UNCERTAINTY QUANTIFICATION IN GROUND VEHICLE DYNAMICS THROUGH HIGH FIDELITY CO-SIMULATION

By
Makarand Vilas Datar

A dissertation submitted in partial fulfillment of the requirements for the degree of

Master of Science
(Mechanical Engineering)

AT THE
UNIVERSITY OF WISCONSIN – MADISON
2008
Dedicated to my parents Vilas Datar, Manisha Datar and my sister Ketaki Datar for their love and sacrifices,
# Table of Contents

1 Abstract  \(i\)

2 Acknowledgments  \(iv\)

3 List of Figures  \(v\)

4 Introduction  \(1\)

5 Vehicle Model  \(5\)
   2.1 Steering  \(5\)
   2.2 Suspension  \(6\)
   2.3 Chassis  \(8\)
   2.4 Powertrain  \(10\)
   2.5 Test Rig  \(11\)
   2.6 Event Builder  \(12\)
   2.7 ADAMS/Car native tire models  \(13\)
      2.7.1 Pacejka Tire Model  \(14\)
      2.7.2 Fiala Tire model  \(14\)
   2.8 Road Modeling  \(15\)
   2.9 Numerical experiments for model validation  \(17\)
      2.9.1 Simple Straight-Line Maneuver with Sinusoidal Road Profile  \(17\)
      2.9.2 Ride Over an Obstacle  \(19\)
      2.9.3 Design of Experiments  \(21\)
      2.9.4 Simulation of a vehicle on flat road with variable friction values  \(22\)
   2.10 References  \(25\)

6 Co-simulation ADAMS-PSAT  \(26\)
   3.1 Motivation  \(26\)
   3.2 Simulation Results  \(28\)
   3.3 References  \(33\)

7 Co-simulation ADAMS-FTire  \(34\)
A framework for uncertainty quantification in multi-body dynamics simulation

5.1. Uncertainty handling methodology
   5.1.1. Parameter Estimation
   5.1.2. Covariance Function Selection

5.2. Uncertainty Quantification results: Parametric model sensitivity
   5.2.1. Tire and Road Models
   5.2.2. Numerical Results, Square Exponential
   5.2.3. Numerical Results, Ornstein-Uhlenbeck
   5.2.4. Numerical Results, Matérn
   5.2.5. Discussion of Numerical Results

5.3. References

9 Conclusion

10 Appendix: ADAMS – PSAT co-simulation process
   A.1. Launching ADAMS/Car
   A.2. Opening an assembly
   A.3. Modifying ADAMS/Car assembly for ADAMS/Controls
      A.3.1. Modify powertrain to get driving torque values from PSAT
      A.3.2. Modify brakes to get mechanical brake demand from PSAT
      A.3.3. Use the existing state variable to measure vehicle longitudinal velocity in ADAMS/Car
      A.3.4. Create a state variable to measure ADAMS driveline speed
      A.3.5. Create/Modify/Delete Plant Inputs/Outputs
      A.3.6. Load ADAMS/Controls plugin
   A.4. Perform a simulation to export files for ADAMS/Controls
   A.5. Create a Simulink model of PSAT vehicle model
      A.5.1. Setup and Build the Simulink Vehicle
      A.5.2. Launching PSAT
      A.5.3. Setting up the vehicle
      A.5.4. Using an existing vehicle
      A.5.5. Using an empty configuration
A.5.6. Building the Simulink

A.5.7. Saving the vehicle

A.6. Initialize PSAT Simulink model to be simulated outside of PSAT GUI

A.7. Modify PSAT model

A.8. Add ADAMS block to Simulink model of PSAT

A.9. Setup ADAMS/Controls block in PSAT Simulink model

A.10. Run Simulation

A.11. Post-process Results

A.12. References
Abstract

Makarand Vilas Datar

Under the Supervision of Dr. Dan Negrut
At the University of Wisconsin - Madison

Today, the state of the art in vehicle virtual prototyping in the automobile industry has reached a level where full vehicles are analyzed through simulation. However, difficulties continue to arise particularly in interfacing the vehicle model with accurate powertrain models and in developing adequate formulations for the contact between tire and terrain (specifically, scenarios such as tire sliding on ice and rolling on sand or other very deformable surfaces). This thesis addresses these limitations and outlines a virtual environment for high-fidelity vehicle simulation by combining several third party
simulation packages for the vehicle, powertrain, and tires into one unified framework.

Even with sophisticated simulation environments, the analysis results cannot be fully trusted due to uncertainty in the model inputs and in defining the physical properties of the model components. Hence, the thesis also outlines a methodology for determining the statistics associated with the time evolution of a nonlinear multi-body dynamic system operated under input uncertainty. The focus is on the dynamics of ground vehicle systems in environments characterized by multiple sources of uncertainty: road topography, friction coefficient at the road/tire interface and aerodynamic force loading. Drawing on parametric maximum likelihood estimation, the methodology outlined is general and can be applied to systematically study the impact of sources of uncertainty characterized herein by random processes. The proposed framework is demonstrated through a study that characterizes the uncertainty induced in the loading of the lower control arm of an SUV type vehicle by uncertainty associated with road topography.

For the work presented in this thesis, ADAMS/Car is used for modeling the
vehicle, the Powertrain Systems Analysis Toolkit (PSAT) or native ADAMS powertrain system are used for the powertrain modeling and simulation, and FTire is used for tire-terrain contact simulation.

Approved:

Dr Dan Negrut
Department of Mechanical Engineering
University of Wisconsin – Madison
Acknowledgments

First and foremost, I would like to thank Dr Dan Negrut for being a wonderful advisor and always eager to help out. I would also like to thank Andy Dyer of MSC software and Sylvain Pagerit of Argonne National Laboratory for helping out with simulation software ADAMS/Car and PSAT respectively. I also thank my committee members, Dr. Roxann Engelstad and Dr. Suresh Krishnan, for their participation and feedback. I thank all my SBEL colleagues for being around and helping out with proofreading of the thesis.

Last but not the least I thank my parents Vilas Datar and Manisha Datar for being extremely supportive and making my graduate studies possible.
List of Figures

Figure 2.1: Closeup of the steering ..................................................................................... 6
Figure 2.2: Closeup of the suspension ................................................................................ 7
Figure 2.3: Vehicle model, no chassis, no powertrain ......................................................... 7
Figure 2.4: Schematic of how different subsystems are assembled to make a full vehicle model ................................................................................................... 9
Figure 2.5: Vehicle path .................................................................................................... 12
Figure 2.6: Top: Road tessellation, Bottom: Triangulation applied to a road profile ............................................................................................................. 16
Figure 2.7: Comparison of three tire models ..................................................................... 18
Figure 2.8: Vehicle encountering the obstacle .................................................................. 19
Figure 2.9: Force in the front left tire using the FTire model ............................................. 20
Figure 2.10: Variation of max vertical acceleration .......................................................... 21
Figure 2.11: Triangulated road with variable coefficient of friction .................................... 23
Figure 2.12: Vehicle turning on a road with constant and variable coefficient of friction ........................................................................................................... 24
Figure 3.1: Fuel efficiency (mpg) versus vehicle mass (kg) for the 60 sec long analysis. .......................................................................................................... 29
Figure 3.2: Velocity versus Time, for each vehicle mass (60 sec run) ................................. 30
Figure 3.3: CO2 emission plot ......................................................................................... 32
Figure 4.1: FTire modeling approach ............................................................................... 35
Figure 4.2: Forces acting on tire while riding over a bump (FTire) ....................................... 38
Figure 5.1: Left: four samples drawn from the prior distribution, Right: situation after two data points have been observed. ................................................................. 47
Figure 5.2: Proposed Uncertainty Quantification Framework .......................................... 48
Figure 5.3: Coarse grid for learning, and fine grid employed in sampling for Monte Carlo analysis. ........................................................................................................ 49
Figure 5.4: Top: Road tessellation (Triangulation applied to a road profile) ..................... 61
Figure 5.5: Vehicle on two different road surfaces. The two roads are generated using the same training data. ................................................................. 62

Figure 5.6: Vehicle response, obtained using ADAMS. Displayed is only a subset of 10 out of the 200 simulations used to generate the statistics of the normal force response. ................................................................. 63

Figure 5.7: Statistics of vertical load, bushing attached to LCA. ........................................ 67

Figure 5.8: Statistics of vertical load, bushing attached to LCA. Detailed view focused on the last part of run. ................................................................. 67

Figure 5.9: Statistics of vertical acceleration, as measured at the CG position of the chassis: mean and 95% confidence interval. ........................................ 68

Figure 5.10: Vertical load statistics, bushing attached to LCA. ....................................... 70

Figure 5.11: Statistics of vertical load, bushing attached to LCA. Detailed view focused on the last part of run. ................................................................. 71

Figure 5.12: Statistics of vertical acceleration, as measured at the CG position of the chassis: mean and 95% confidence interval. ........................................ 71

Figure 5.13: Vertical load statistics, bushing attached to LCA. ....................................... 72

Figure 5.14: Statistics of vertical load, bushing attached to LCA. Detailed view focused on the last part of run. ................................................................. 73

Figure 5.15: Statistics of vertical acceleration, as measured at the CG position of the chassis: mean and 95% confidence interval. ........................................ 73

Figure A.1: ADAMS/Car interface ............................................................................... 81

Figure A.2: Opening an assembly .................................................................................. 83

Figure A.3: Full vehicle assembly .................................................................................. 84

Figure A.4: Open template ............................................................................................ 86

Figure A.5: Database navigator ...................................................................................... 87

Figure A.6: Modifying a State variable ........................................................................... 88

Figure A.7: Driveline speed ........................................................................................... 91

Figure A.8: Create plant output .................................................................................... 92

Figure A.9: Create plant output dialog box .................................................................... 93
Figure A.10: Create plant input ....................................................................................... 95
Figure A.11: Plug-in Manager .......................................................................................... 96
Figure A.12: Straight line acceleration ............................................................................. 99
Figure A.13: ‘Files only’ simulation ............................................................................... 100
Figure A.14: PSAT GUI ................................................................................................ 104
Figure A.15: Open Vehicle file in PSAT ........................................................................ 106
Figure A.16: Selecting a Configuration ......................................................................... 107
Figure A.17: Select Components .................................................................................. 108
Figure A.18: Set Parameters ......................................................................................... 109
Figure A.19: Select Controller ...................................................................................... 110
Figure A.20: Adding a Cycle .......................................................................................... 111
Figure A.21: Cycle Characteristics ................................................................................ 112
Figure A.22: Build Simulink ........................................................................................ 113
Figure A.23: Modify PSAT Simulink Model .................................................................. 116
Figure A.24: ADAMS Simulink Block ......................................................................... 117
Figure A.25: Insert ADAMS Block in PSAT Model ..................................................... 118
Figure A.26: ADAMS Block in PSAT Model .............................................................. 119
Figure A.27: ‘adams_sub’ Block Details ....................................................................... 120
Figure A.28: Setup ADAMS Block ............................................................................. 121
Figure A.29: Flowchart for Setting Up Co-Simulation Analysis ................................... 123
Chapter 1

Introduction

The concept of simulation-based engineering has been embraced by virtually every research and industry sector as the engineering and science communities have become increasingly aware that computer simulation is an indispensable tool for resolving a multitude of scientific and technological problems. Design and analysis engineers are simulating increasingly complex mechanical systems. The Virtual Prototyping approach to the Product Life Cycle is adopted in industry due to its economic advantages, i.e., reduced cost and time to market. It is clearly desirable to gain a reliable perspective on the behavior of a system early in the design stage, long before actually building costly prototypes. The potential of Virtual Prototyping further increases when the final system is the assembly of components/subsystems contributed by several manufacturers at different geographic locations. These subsystems might be in various stages of
development, and building the physical prototype is either impractical (due to cost and
time constraints) or outright impossible.

Today, the state of the art in vehicle virtual prototyping in the automobile industry has
reached a level where full vehicles are analyzed through simulation. However, difficulties
continue to arise particularly in interfacing the vehicle model with accurate powertrain
models and in developing adequate formulations for the contact between tire and terrain
(specifically, scenarios such as tire sliding on ice and rolling on sand or other very
deformable surfaces). This thesis addresses these limitations and outlines a virtual
environment for high-fidelity vehicle simulation by combining several third party
packages for the vehicle, powertrain, and tires. Specifically, this effort aims to investigate
techniques and to develop infrastructure to enable co-simulation of vehicle models that
contain high-fidelity representations of tire, terrain, powertrain, driver, and controls
subsystems.

A single simulation can very rarely predict accurately the behavior of a system. Even with
sophisticated simulation environments, the analysis results cannot be fully trusted due to
uncertainty in the model inputs and in defining the physical properties of the model components. These uncertainties in the simulation environment are modeled as random variables with some expected value (usually the ‘ideal’ value) and a variation (variance or spread). Then a number of simulations are run with a different value for the input variable (using normal distribution to capture probability of occurrence of any particular value) at each occasion. When the results from these simulations are post-processed, we get an upper and lower bound on the possible outcomes. Hence, even though we cannot accurately predict an output (force in a component, deflection etc.) we can certainly get an idea about the range in which the value can be expected to fall.

One such uncertainty in the input data results from the road models. This thesis explores a way to quantify this uncertainty by looking at the forces in vehicle’s suspension and their range.

For the work presented in this thesis, ADAMS/Car is used for modeling the vehicle, the Powertrain Systems Analysis Toolkit (PSAT) is used for the powertrain modeling and simulation, and FTire is used for tire-terrain contact simulation. High-fidelity vehicle
analysis is carried out drawing on these three packages and relying on MATLAB/Simulink as the co-simulation framework. A detailed road model is considered at the interface between ADAMS and FTire that allows for tessellated roads with variable friction coefficients and elevation. The driver model available in ADAMS/Car is leveraged for maneuvering the vehicle. Finally, the statistical analysis for uncertainty quantification is done using a MATLAB based code developed in house and explained in details later. Henceforth, this simulation framework will be referred to as Digital Car.
Chapter 2

Vehicle Model

The purpose of this section is to present in detail the ADAMS/Car component that enables the Digital Car concept. ADAMS/Car is a full vehicle simulation package distributed by MSC.Software. The vehicle of interest in this work is a SUV-type vehicle similar, although not identical, to the Army’s High Mobility Multi-Purpose Wheeled Vehicle (HMMWV). All the simulations described here, (co-simulation, uncertainty quantification) make use of the ADAMS/Car HMMWV model for the vehicle component unless mentioned otherwise.

2.1 Steering

The steering subsystem is a rack-and-pinion type subsystem. It is modeled with parts and joints to form a single-degree-of-freedom system starting from the steering wheel to the steering rack. A detailed image of steering subsystem is shown in Figure 2.1.
2.2 Suspension

An Ackerman Arm suspension (See Figure 2.2) is used which differs in dimensions depending upon whether it is a front or rear suspension. The topology however remains the same.

Figure 2.1: Closeup of the steering
Figure 2.2: Closeup of the suspension

Figure 2.3: Vehicle model, no chassis, no powertrain
The location of the suspension subsystem is parameterized with respect to the chassis of the vehicle to allow for easy editing of the assembly topology. Further, the stiffness of the suspension springs and damping coefficients of the dampers can be easily altered if required. Figure 2.3 shows the topology of a vehicle with a front and rear suspension, wheels, and steering subsystems. The chassis of the vehicle is not shown.

2.3 Chassis

The vehicle body is modeled as a single component with mass-inertia properties. The chassis can be made rigid or flexible. In the latter case, a finite element analysis would be required to produce a modal representation of the element.
Figure 2.4: Schematic of how different subsystems are assembled to make a full vehicle model
Figure 2.4 shows how various subsystems are assembled together to make the full vehicle assembly. The bottom picture shows the model of HMMWV rendered in 3ds max.

2.4 Powertrain

The powertrain system models the interaction between various blocks in a vehicle driveline. The dynamics is mostly captured using algebraic and simple differential equations rather than using parts. A module called “Driving Machine” demands steering, throttle, brake, gear, and clutch inputs from the user to simulate custom vehicle maneuvers. These inputs are given to the powertrain subsystem by a test rig subsystem (discussed later). The powertrain subsystem references external files to get data necessary during the simulations, e.g., engine torque map with respect to engine speed and throttle position. The powertrain system modeled for the HMMWV is a four-wheel drive topology. The torque produced by the engine is split between the front and the rear tires. Changing the values in the relevant mathematical equations captures the fractions going into the front and the rear. The powertrain part of the simulation proceeds as follows. The engine torque value is read from a file which gives the torque values for various
engine speeds and throttle positions. The clutch is modeled as a torsional spring-and-damper element. The torque lost in the clutch slippage is subtracted from the engine torque to calculate the torque at the clutch output. This torque, which is also the input torque to the transmission, is scaled by the gear ratio (depending upon which gear the vehicle is in) to calculate the output torque from the transmission. The torque is finally scaled by the final drive ratio and then split between the two wheels to be the torque acting on the wheel hub. The more sophisticated third party powertrain simulation package called PSAT is discussed in Chapter 3.

2.5 Test Rig

The test rig conveys user inputs to the model. The control data provided by the user for standard and custom full vehicle simulations is communicated to the vehicle subsystems through ADAMS/View variables called “communicators”. Communicators are the primary connectors between various subsystems and facilitate data exchange between them.
2.6 Event Builder

Custom events can be built as a sequence of mini-maneuvers. Maneuvers are defined by steering, throttle, brake, gear, and clutch inputs. When a set of end conditions is reached for a particular mini-maneuver, it triggers the start of the next mini-maneuver.

An event with two mini maneuvers is illustrated in Figure 2.5: (1) The first mini maneuver settles the vehicle and starts moving it with a constant speed. (2) Once this constant speed has been maintained for a specific period of time called the “Filter Time”, the maneuver ends and triggers the start of a next mini maneuver. (3) In the second mini
maneuver, the vehicle makes a left turn on a flat road. (4) Finally, when the turn has been
achieved, the vehicle continues going straight. Simulation output data such as vehicle
lateral acceleration, chassis roll, pitch and yaw angles can be plotted by using the
ADAMS/PostProcessor. A wide range of maneuvers can be simulated on a wide range of
road profiles.

2.7 ADAMS/Car native tire models

Two types of external forces act under normal conditions on the vehicle and influence its
dynamics: forces at the tire-road interface and aerodynamic forces. Given the range of
speeds at which the vehicle is driven, the former forces are prevalent and high-fidelity
simulation critically depends on their accurate characterization. In this context, the SUV
configuration of interest is designed to ride over a variety of terrains, from flat and
smooth pavement to off-road conditions. A number of different approaches for tire
modeling are supported in the co-simulation framework developed, each suitable for
specific driving conditions. From very simple models to complex non-linear finite
element formulations, they all have merits and downsides as they reach a compromise
between fidelity and simulation time. A discussion of the models, along with a description of the effects they are able to account for and the assumptions they rely on is presented next.

2.7.1 Pacejka Tire Model

The Pacejka '89 and Pacejka '91 handling tire models are special versions of the Magic-Formula (MF) Tire model ((Bakker et al., 1987), (Pacejka and Bakker, 1991)). In general, an MF tire model describes the tire behavior on mostly smooth roads (road obstacle wavelengths longer than the tire radius) up to a frequency of 8 Hz.

2.7.2 Fiala Tire model

The Fiala Tire (FT) used by the proposed simulation capability relies on the Fiala Handling Force element available in ADAMS and is essentially an extended version of the Fiala model (Fiala, 1964). This model provides reasonable results for simple maneuvers where road inclination angle is not a major factor and where longitudinal and lateral slip effects may be considered unrelated.
2.8 Road Modeling

In terms of the road models supported, they should accommodate the particular tire model they work in conjunction with. For the MF tires this is not an issue, since the road description is very basic. In this and the case of Fiala tire model, the road environment of choice is the one provided by ADAMS, where the road is defined by a text based data file (rdf). This file contains the information about road size, type (flat, periodic obstacles, stochastic 3D) and coefficients of friction over the road surface. Defining a flat road is trivial and an obstacle in the road (curb, roof-shaped) can be defined by specifying the size and shape of the obstacle in the definition file. For roads with varying elevations in both lateral and longitudinal directions, a tessellated road definition is supported. A tessellated road is described by a set of vertices/nodes, which are grouped in sets of three to create a triangulated mesh that describes the entire road surface. A coefficient of friction can be specified for each triangle. Figure 2.6 shows an example of a tessellated road with 6 nodes and triangles A, B, C and D that form a mesh while a full road is
defined using the tessellation approach. This road definition works well with both synthesized and measured road data.

![Diagram showing road tessellation and triangulation applied to a road profile](image)

**Figure 2.6:** Top: Road tessellation, Bottom: Triangulation applied to a road profile

The disadvantage of this road definition is that for each time step, the simulation has to check every triangle for contact with the tire patch, which becomes a major
computational bottleneck (extremely long simulation times for large road profiles with high resolution). In order to simulate large road profiles, the Regular Grid Road (RGR) file format is used in conjunction with FTire. A conversion from tessellated to the new RGR format leads to smaller file sizes and significantly reduces CPU time per simulation step in that the CPU time required for the tire/terrain interaction does not depend on the dimensions (length/width) of the road profile.

2.9 Numerical experiments for model validation

A set of four numerical experiments have been carried out in order to demonstrate some of the capabilities of the simulation environment: (1) a straight-line maneuver over a sinusoidal road profile; (2) a ride over an obstacle maneuver; and (3) a design of experiments (DOE) study.

2.9.1 Simple Straight-Line Maneuver with Sinusoidal Road Profile

In this simulation the HMMWV described in the previous section was run using three different tire models: FTire, Fiala, and Pacejka. The tire models are rather similar although no concerted effort was made at this time to ensure that the results obtained
with these models were identical. The straight-line acceleration maneuver was run for 5 seconds on a road with a sinusoidal profile (amplitude is 50 mm and with a wavelength of 2.5 meters). The initial velocity of the vehicle is 30 km/h and the initial throttle is zero. The throttle starts rising after one second and is 40% open at the end of the fourth second. This throttle value is maintained for the rest of the simulation time. The gears automatically shift based upon the vehicle and engine speed. There is no input from the steering wheel and the vehicle follows a straight line at all times.

Figure 2.7: Comparison of three tire models

Figure 2.7 presents the results of this simulation for normal force in the front left tire. As indicated, using different tire models leads to different results. Of the three models,
FTire is the most suitable to deal with the sinusoidal road profile. The other two models, while expeditious in nature, are particularly suitable for road profiles that are smooth or have a wavelength significantly larger than the tire radius.

2.9.2 Ride Over an Obstacle

The height of the step shaped obstacle was 75 mm and the width was approximately equal to the tire width. FTire is used as a tire model in running these simulations. There is a slight bevel edge on either side of the step to avoid heavy impact of the vehicle onto the step.

Figure 2.8: Vehicle encountering the obstacle
Figure 2.8 shows a wire frame model of the vehicle climbing the step. Various values can be plotted and a Design of Experiments can be run to optimize some particular measure based on different vehicle parameters. These parameters include size and shape of various vehicle components, as well as damping and stiffness coefficients of the suspension.

![Graph showing forces on the front left tire.](image)

**Figure 2.9: Force in the front left tire using the FTire model.**

Figure 2.9 shows the force in the front left tire of the vehicle as it climbs over the step and crosses it.
2.9.3 Design of Experiments

A design of experiments (DOE) test involves running a set of simulations by changing select values of certain design parameters. A design objective is defined and its value is tracked for each experiment. This DOE is run as a co-simulation of FTire as the tire model with the ADAMS/Car HMMWV model.

Figure 2.10: Variation of max vertical acceleration
A straight-line acceleration maneuver is run for 3 seconds on a road with a bump. The height of this bump is 100 mm. The initial velocity of the vehicle is 40 km/h and the initial throttle is zero. The throttle starts rising after 0.5 seconds, is 40% open after 1.5 seconds and stays like this afterwards.

The absolute maximum value of the vertical acceleration was decided to be the design objective. The two variables that were changed were (a) damping coefficient of the front left side suspension damper and (b) damping coefficient of the front right side suspension damper.

The design of experiments was run by varying the values of above mentioned two variables by +30% and -50% in steps of 5%, which resulted in 17 X 17 (289) trials. The variation of maximum vertical acceleration over these 289 trials is shown in Figure 2.10.

2.9.4 Simulation of a vehicle on flat road with variable friction values

In this analysis, FTire is used with an ADAMS/Car generic sedan vehicle model to monitor the behavior of a vehicle as it navigates a turn on road profiles with the friction coefficient assuming a user-defined distribution. Given a set of road friction data points,
triangular elements are created with sets of three data points to define the entire road profile. Friction values from these points can simply be averaged over each triangle.

![Triangulated road with variable coefficient of friction](image)

Figure 2.11: Triangulated road with variable coefficient of friction

The road shown in Figure 2.11 has varied coefficient of friction values; a minimum value of $\mu_{\text{min}} = 0.1240$ and a maximum value of $\mu_{\text{max}} = 0.8786$ are present in this particular simulation.
In order to demonstrate the impact of a stochastic distribution of the friction coefficient over the road profile, a reference simulation is run with a constant coefficient of friction. The trajectory of the vehicle’s center of mass is tracked, and results of the simulations with constant and a stochastic road friction distribution are both illustrated in Figure 2.12. In this context, the proposed co-simulation environment has recently been used by Schmitt, Madsen et al. (2008) in a stochastic analysis that relies on Gaussian Random Processes to quantify uncertainty in vehicle dynamics.
2.10 References


Chapter 3

Co-simulation ADAMS-PSAT

3.1 Motivation

Although ADAMS/Car provides full vehicle simulation capability, the powertrain model in ADAMS is not very versatile. The powertrain system used in ADAMS/Car has some limitations, as listed below:

- Only conventional powertrain systems
- The fuel economy cannot be predicted
- Not very sophisticated
- Is not validated extensively

The Powertrain Systems Analysis Toolkit (Rousseau and Pagerit, 2001) offers more sophisticated powertrain system with various features, as listed below:

- Conventional, Electric, fuel cell
Several varieties of hybrid (parallel, series, power split, series-parallel) powertrains for light and heavy duty vehicles

Able to predict fuel economy

The Powertrain Systems Analysis Toolkit (PSAT) powertrain system also offers prediction of fuel efficiencies within 2% to 5% accuracy.

PSAT powertrain models are built by integrating sub-component models. This automated powertrain configuration setup is the cornerstone of PSAT's flexibility and reusability. Three hundred predefined configurations can be simulated including conventional, electric, fuel cell and several varieties of hybrid (parallel, series, power split, series-parallel) powertrains. Light, medium, and heavy-duty vehicle platforms can be simulated. PSAT has been used to evaluate fuel consumption and performance for a variety of driving cycles and profiles, to develop realistic control strategies, and to select drivetrain component size and technology. These models are validated within 2% fuel economy for conventional and mild hybrid vehicles (Honda Insight, Ford P2000) and up to 5% for full hybrid vehicles (Toyota Prius) [(Rousseau, 2001), (Cao et al., 2007)].
powertrain systems are modeled using MATLAB, Simulink, and Stateflow. For co-simulation, the powertrain system in ADAMS/Car is not used. Instead, PSAT is used to simulate the powertrain. The ADAMS vehicle model representing the chassis, tires, etc., is combined with a PSAT model inside MATLAB/Simulink.

In order to carry out an ADAMS/Car – PSAT co-simulation, some basic knowledge of ADAMS/Car and PSAT is assumed. However, an effort has been made to provide explanations in greater detail for users less familiar with the software. The procedure of ADAMS-PSAT co-simulation is explained in details in the appendix.

3.2 Simulation Results

The co-simulation results can be post-processed using both ADAMS and PSAT. Information about load histories in vehicle components, vehicle accelerations and driver comfort can be obtained using ADAMS. Information about vehicle fuel efficiency, engine efficiency and CO2 emission is obtained through PSAT. Information regarding the tire/road interaction is reported both in ADAMS and FTire.
In order to demonstrate the capability of the simulation environment, a simple straight-line acceleration test is performed on a generic sedan. The initial velocity of vehicle is 10 m/s. For this initial velocity, three different scenarios are considered: (1) the vehicle
Figure 3.2: Velocity versus Time, for each vehicle mass (60 sec run)
attempts acceleration to 18 m/s in 10 seconds, (2) the vehicle attempts acceleration to 18 m/s in 40 seconds, and (3) the vehicle attempts acceleration to 18 m/s in 60 seconds.

For each of these situations, the vehicle mass is varied between 1000 kg and 3500 kg in steps of 500 kg resulting in a total of 18 simulations.

For each acceleration run, the fuel efficiency decreases as the vehicle mass increases.

Figure 3.1 shows a plot of engine fuel efficiency on y-axis versus vehicle mass on x-axis, for the 60 seconds acceleration case. The plot clearly shows a reduction in fuel efficiency as expected.

Figure 3.2 shows the velocity profile for the 10 to 18 m/s vehicle acceleration maneuver over 60 seconds for a variety of vehicle masses (corresponding to a gradual increase in payload). The velocity profile of the vehicle provides useful information. As the vehicle gets heavier it becomes harder for the engine to reach the target velocity. The instant the vehicle shifts into a higher gear can be clearly noticed by the drop in the slope of velocity profile at the gearshift event, which confirms that higher gears produce less acceleration.
In this co-simulation environment, exhaust gas emission can be predicted using PSAT post processing. Figure 3.3 presents a plot of CO2 emission rate versus vehicle mass for a 60 second long acceleration from 10m/s to 18m/s. It can be seen that as the vehicle mass increases, the engine burns more fuel to generate sufficient power, leading to an increase in CO2 emission.
3.3 References


Chapter 4

Co-simulation ADAMS-FTire

4.1 Introduction and Motivation

As indicated in the online documentation, the FTire (Flexible Ring Tire), model serves as a more sophisticated tire force element (Gipser, 2005). It can be used in multibody system models for vehicle ride comfort investigations as well as other vehicle dynamics simulations on even or uneven roadways. Specifically, FTire is designed for vehicle comfort simulations on road irregularities even with extremely short wavelengths. At the same time, it serves as a physically based, highly nonlinear, dynamic tire model for investigating handling characteristics under the above-mentioned excitation conditions.
FTire is fast (cycle time is only 5 to 20 times real-time) and numerically robust. The tire belt is described as an extensible and flexible ring carrying bending loads, elastically founded on the rim by distributed, partially dynamic stiffness values in radial, tangential, and lateral directions. The degrees of freedom of the ring are such that both in-plane and out-of-plane rim movements are possible. The ring is numerically approximated by a finite number of discrete masses called the belt elements. These belt elements are coupled with their direct neighbors by stiff springs with in- and out-of-plane bending stiffness. In-plane bending stiffness is realized by means of torsional springs about the lateral axis. The torsional deflection of these springs is determined by the angle between three consecutive belt elements projected onto the rim mid-plane. Similarly, the out-of-plane
bending stiffness is described by means of torsional springs about the radial axis. Here, the torsional deflection is determined by the angle between three consecutive belt elements projected onto the belt tangential plane. Note that in Figure 4.1, the plates do not represent the belt elements themselves but rather the connecting lines between the elements. FTire calculates the stiffness and the damping factors during preprocessing by fitting the prescribed modal properties. A number of massless tread blocks (5 to 50, for example) are associated with every belt element. These blocks carry nonlinear stiffness and damping properties in the radial, tangential, and lateral directions. The radial deflections of the blocks depend on the road profile and orientations of the associated belt elements. FTire determines tangential and lateral deflections using the sliding velocity on the ground and the local values of the sliding coefficient. The latter depends on ground pressure and sliding velocity. FTire calculates all six components of tire forces and moments acting on the rim by compounding the forces in the elastic foundation of the belt. Because of this modeling approach, the resulting overall tire model is accurate up to relatively high frequencies both in longitudinal and in lateral directions. There are few
restrictions in its applicability with respect to longitudinal, lateral, and vertical vehicle
dynamics situations. FTire deals with long- and/or short-wavelength obstacles. It works
out of, and up to, a complete standstill, with no additional computing effort or any model
switching. Finally, it is applicable with high accuracy in demanding applications such as
ABS braking on uneven roadways. In a full 3D variant, FTire additionally takes into
account belt element rotation and bending about the circumferential axis. These new
degrees of freedom enable FTire to use contact elements that are distributed not only
along a single line but also over the whole contact patch. The contact elements can be
either randomly distributed or distributed along several parallel lines. In the full 3D
variant, belt torsion about the circumferential axis is described by (1) torsional stiffness
between belt elements and rim about the circumferential axis (represented by torsion
springs between two belt elements) and (2) torsional stiffness between adjacent belt
elements about the circumferential axis. The right side of the Figure 4.1 outlines the belt
bending stiffness about the circumferential axis. This is done in a somewhat simplified
manner: lateral belt bending is taken into account by introducing a parabolic shape
function for each belt element. The curvature of this shape function is treated as a belt element’s additional degree of freedom.

Figure 4.2: Forces acting on tire while riding over a bump (FTire)

Figure 4.2 shows a graphical representation (from a FTire simulation) of forces acting on the tire when one of the wheels is riding over a bump.
4.2 References

Chapter 5

A framework for uncertainty quantification in multi-body dynamics simulation

The work described in this section focuses on establishing an analytically sound and computationally efficient framework for quantifying uncertainty in the dynamics of complex multi-body systems. The motivating question for this effort is as follows: how can one predict an average behavior and produce a confidence interval for the time evolution of a complex multi-body system that is subject to uncertain inputs? Herein, of interest is answering this question for ground vehicle systems whose dynamics are obtained as the solution of a set of differential-algebraic equations (Brenan et al., 1996). The differential equations follow from Newton's second law. The algebraic equations represent nonlinear kinematic equations that constrain the evolution of the bodies that make up the system (Haug, 1989).
The motivating question above is relevant for vehicle Condition-Based Maintenance (CBM), where the goal is to predict durability and fatigue of system components. For instance, the statistics of lower control arm loading in a High-Mobility Multi-Purpose Wheeled Vehicle (HMMWV) obtained through a multi-body dynamics simulation become the input to a durability analysis that can predict, in a stochastic framework, the condition of the part and recommend or postpone system maintenance. A stochastic characterization of system dynamics is also of interest in understanding limit behavior of the system. For instance, providing a confidence interval in real time for certain maneuvers is useful in assessing the control of a vehicle operating on icy road conditions.

Vehicle dynamics analysis under uncertain environment conditions, e.g. road profile (elevation, roughness, friction coefficient) and aerodynamic loading, requires approaches that draw on random functions. The methodology is substantially more involved than the one required for handling uncertainty that enters the problem through discrete design parameters associated with the model. For instance, uncertainty in suspension spring stiffness or damping rates can be handled through random variables. In this case,
methods such as the polynomial chaos (PC), see, for instance, (Xiu and Karniadakis, 2002) are suitable provided the number of random variables is small. This is not the case here, since a discretization of the road leads to a very large number of random variables (the road attributes at each road grid point). Moreover, the PC methodology requires direct access and modification of the computer program used to run the deterministic simulations to produce first and second order moment information. This represents a serious limitation if relying on commercial off-the-shelf (COTS) software, which is most often the case in industry when running complex high-fidelity vehicle dynamics simulations.

In conjunction with Monte Carlo analysis, the alternative considered herein relies on random functions to capture uncertainty in system parameters and/or input. Limiting the discussion to three-dimensional road profiles, the methodology samples a posterior distribution that is conditioned on available road profile measurements. Two paths can be followed to implement this methodology; the first draws on a parametric representation of the uncertainty, the second being nonparametric in nature. The latter approach is
general yet expensive to implement. It can rely on smoothing techniques (nonparametric regression) that use kernel estimators such as Nadaraya-Watson or variants; see, for instance, (Wasserman, 2006). The parametric approach is used in this thesis by considering Gaussian Random Functions as priors for the road profiles. Furthermore, the discussion will be limited to stationary processes although current research is also investigating the nonstationary case.

The use of a parametric model raises two legitimate questions: why use a particular parametric model, and why is it fit to capture the statistics of the problem? Gaussian Random Functions (GRF) are completely defined by their correlation function, also known as a variogram (Adler, 1990 and Cramér and Leadbetter, 1967). Consequently, scrutinizing the choice of a parametric GRF model translates into scrutinizing the choice of correlation function. There are several families of correlation functions, the more common ones being exponential, Matérn, linear, spherical, and cubic (see, for instance (Santner et al., 2003)). In this context, and in order to demonstrate the proposed framework for uncertainty quantification in multi-body dynamics, a representative
problem will be investigated in conjunction with the selection of a GRF-based prior. Specifically, an analysis will be carried out to assess the sensitivity of the response of a vehicle to uncertainty in system input, the uncertainty in this case being in the road profile. The outcome of interest will be the load history for the lower-control arm of an HMMWV, a key quantity in the CBM of the vehicle. The parametric priors considered are (i) a GRF with a squared exponential correlation function, (ii) the Ornstein-Uhlenbeck process and (iii) the Matérn correlation function. Pronounced sensitivity of the statistics of the loads acting on the lower control arm with respect to the choice of parametric model would suggest that serious consideration needs to be given to the nonparametric choice, where the empirical step of variogram selection is avoided at the price of a more complex method and increase in simulation time.
5.1. Uncertainty handling methodology

The discussion herein concerns handling uncertainty in spatial data. More specifically, it is assumed that limited information is used to generate road profiles that are subsequently used in the dynamic analysis of a ground vehicle. The handling of uncertainty in aerodynamic loads can be addressed similarly but is not of primary interest in this study and will be omitted.

The problem at hand is concerned with supervised learning, which is the problem of learning input-output mappings from empirical data (the training dataset) (Rasmussen and Williams, 2006). Depending on the characteristics of the output, this problem is known as either regression, for continuous outputs, or classification, when outputs are discrete.

The input (or the independent variable) for the observed data will be denoted as $x$, which in general is a vector variable. The output is denoted as $y$ and there can be multiple input and output variables. Hence we have a dataset of $n$ observations,
\{(x_i, y_i) \mid i = 1, 2, \ldots, n-1, n\}. In the context of this thesis, the input data for road terrain is a collection of points along the lateral and longitudinal direction of the road and the output data is the elevation of the road at these locations.

With the aid of this input data, the goal is to predict the elevations of the road at a finer grid than the one available with observed data. This new set of inputs where predictions need to be made is denoted by \(x_\ast\). Thus, from a finite set of observed data, predictions are made over any number of new input locations. The methodology used to achieve this relies on the idea of Bayesian inference and Gaussian processes. Gaussian process is a generalization of the Gaussian probability distribution. Whereas a probability distribution describes random variables which are scalars or vectors (for multivariate distributions), a Gaussian process describes distribution of functions instead of variables. Bayesian inference is used to give a prior probability to every possible function, where higher probabilities are given to functions that are more likely. This choice about likeliness is made after studying the observed data and locations where predictions need to be made. The Bayesian inference is illustrated graphically using a simple regression
and classification example. Consider a simple 1-d regression problem, mapping from an input \( x \) to an output \( f(x) \).

![Figure 5.1: Left: four samples drawn from the prior distribution. Right: situation after two data points have been observed. The mean prediction is shown as the solid line and four samples from the posterior are shown as dashed lines. In both plots the shaded region denotes twice the standard deviation at each input value \( x \) (Rasmussen and Williams, 2006).](image)

Figure 5.1 shows number of sample functions drawn at random from a prior distribution based on some Gaussian distribution. This prior is taken to represent prior beliefs over the kinds of functions that are expected before seeing any observed data. The average value of all the infinitely many sample functions at each \( x \) is assumed to be zero. At any value of \( x \) the variability of the sample functions can be gauged by computing the variance at that point. The shaded region denotes twice the pointwise standard deviation.
With this background, given a dataset \( \{(x_1, y_1), (x_2, y_2)\} \) consisting of two observations, only those functions that exactly pass through these points are considered. This situation is illustrated in Figure 5.1 (right). The dashed lines show sample functions which are consistent with the dataset, and the solid line depicts the mean value of such functions. The uncertainty is reduced closer to the data points and is zero at the data points. The combination of the prior and the data leads to the posterior distribution over functions. If more data points were added, the overall uncertainty at input locations is reduced.

![Diagram](image)

Figure 5.2: Proposed Uncertainty Quantification Framework
The uncertainty quantification framework proposed is described in Figure 5.2. An assumption is made that learning data has been made available as the result of field measurements.

![Figure 5.3: Coarse grid for learning, and fine grid employed in sampling for Monte Carlo analysis. Here \( d = 2 \).](image)

Referring to Figure 5.3, the measured data is available on a “coarse” measurement grid. For dynamic analysis, road information is ideally available everywhere on the road as continuous data. Since, this is not possible however, the data is only provided on a fine grid (right image in Figure 5.3). If working with a parametric model, a correlation function is selected and a learning stage follows. Its outcome, a set of hyper-parameters
associated with the correlation function, is instrumental in generating the mean and covariance matrix to be used in generating sample road surfaces on the user specified fine grid.

The learning data available is the road elevation as a discrete function of x-y co-ordinates. An example of such a road can be seen in Figure 5.3. Note that this represents a two-dimensional problem ($d = 2$).

The use of Gaussian Random Functions (GRF) or processes is a very versatile approach for the simulation of infinite dimensional uncertainty. In general, a spatially distributed random variable $y(x), x \in \mathbb{R}^d$, is a GRF with mean function $m(x; \theta_1)$ and correlation function $k(x, x'; \theta_2)$ if, for any set of space points $X = \{x_1, x_2, \ldots, x_M\}$,

$$y(X) = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_M \end{pmatrix} \sim N \left( m(X; \theta_1), K(X, X; \theta_2) \right). \quad (1)$$

Here $m \in \mathbb{R}^M$, $K \in \mathbb{R}^{M \times M}$, and $N(m, K)$ is the M-variate normal distribution with mean $m$ and covariance $K$ given by
\[ m(X; \theta_1) = \begin{pmatrix} m(x_1, \theta_1) \\ m(x_2, \theta_1) \\ \vdots \\ m(x_M, \theta_1) \end{pmatrix} \] (2)

\[ K(X, X'; \theta_2) = \begin{pmatrix} k(x_1, x_1'; \theta_2) & \cdots & k(x_1, x_N'; \theta_2) \\ k(x_2, x_1'; \theta_2) & \cdots & k(x_2, x_N'; \theta_2) \\ \vdots & \vdots & \vdots \\ k(x_M, x_M'; \theta_2) & \cdots & k(x_M, x_N'; \theta_2) \end{pmatrix} \] (3)

where \( X' = \{ x_1', x_2', \ldots, x_N' \} \). The hyper-parameters \( \theta_1 \) and \( \theta_2 \) associated with the mean and covariance functions are obtained from a data set \( y(D) \) at nodes \( D = \{ d_1, \ldots, d_M \} \).

The posterior distribution of the variable \( y(S) \) at node points \( S = \{ s_1, \ldots, s_N \} \), consistent with \( y(D) \), is \( N(f^*, K^*) \) (Rasmussen and Williams, 2006), where

\[ f^* = K(S, D; \theta_2)K^{-1}(D, D; \theta_2)(y(D) - m(D; \theta_1)) + m(S; \theta_1) \]

\[ K^* = K(S, S; \theta_2) - K(S, D; \theta_2)K^{-1}(D, D; \theta_2)K(D, S; \theta_2) \]

The key issues in sampling from this posterior are a) how to obtain the hyper-parameters from data, and b) how to sample from \( N(f^*, K^*) \) especially in the case where \( M \) is very large. The classical way of sampling relies on a Cholesky factorization of \( K^* \), a costly
A brief description of the hyper-parameter calculation follows.

5.1.1. Parameter Estimation

The method used herein for the estimation of the hyper-parameters from data is maximum likelihood estimation (MLE) (Rasmussen and Williams, 2006). The method relies on the maximization of the log-likelihood function. In the multivariate Gaussian with mean \( \mathbf{m}(\theta) \in \mathcal{R}^M \) and covariance matrix \( \mathbf{K}(\theta) \in \mathcal{R}^{M \times M} \) case, the log-likelihood function assumes the form

\[
\log p(\mathbf{y} | \theta) = -\frac{1}{2} \mathbf{W}^T \mathbf{K}(\theta)^{-1} \mathbf{W} - \frac{1}{2} \log | \mathbf{K}(\theta) | - \frac{M}{2} \log 2\pi
\]

Here, \( \mathbf{W} = \mathbf{y} - \mathbf{m}(\theta) \) and \( \mathbf{y} \) is the observed data. Note that the dependence on the hyper-parameters \( \theta \) appears in terms of coordinates \( \mathbf{x} \), and \( \theta = \{ \theta_1, \theta_2 \} \). The gradients of the likelihood function can be computed analytically (Rasmussen and Williams, 2006):

\[
\frac{\partial}{\partial \theta_{1j}} \log p(\mathbf{y} | \theta) = - \left( \frac{\partial}{\partial \theta_{1j}} \mathbf{m}(\theta) \right)^T \mathbf{K}(\theta)^{-1} \mathbf{W}
\]
\[
\frac{\partial \log p(y \mid \theta)}{\partial \theta_{2j}} = \frac{1}{2} \text{tr} \left( (\mathbf{K}(\theta)^{-1}\mathbf{W}(\mathbf{K}(\theta)^{-1}\mathbf{W})^T - \mathbf{K}(\theta)^{-1}) \frac{\partial \mathbf{K}(\theta)}{\partial \theta_{2j}} \right)
\]

MATLAB’s \textit{fsolve} function, which implements a quasi-Newton approach for nonlinear equations, was used to solve the first order optimality conditions \(\frac{\partial \log p(y \mid \theta)}{\partial \theta_{1j}} = 0\) and \(\frac{\partial \log p(y \mid \theta)}{\partial \theta_{2j}} = 0\) to determine the hyper-parameters \(\theta_1\) and \(\theta_2\). The entire approach hinges, at this point, upon the selection of the parametric mean and covariance. It is common to select a zero mean prior \(\mathbf{m} \equiv 0\), in which case only the \(\theta_{2j}\) hyper-parameters associated with the covariance matrix remain to be inferred through MLE.

5.1.2. Covariance Function Selection

The parametric covariance function adopted determines the expression of the matrix \(\mathbf{K}(\theta)\) from the previous subsection, and it requires an understanding of the underlying statistics associated with the data. In what follows, the discussion focuses on three common choices of correlation function: squared exponential (SE), Ornstein-Uhlenbeck (OU) (Uhlenbeck and Ornstein, 1930) and Matérn (MTR) (Mátrén, 1960).
The SE correlation function assumes the form

\[ k(x, x'; \theta_2) = \exp \left\{ -\frac{(x_1 - x'_1)^2}{\theta_{21}} - \frac{(x_2 - x'_2)^2}{\theta_{22}} \right\}, \tag{4} \]

where \( \gamma = 1 \). The hyper-parameters \( \theta_{21} \) and \( \theta_{22} \) are called the characteristic lengths associated with the stochastic process. They control the degree of spatial correlation; large values of these coefficients lead to large correlation lengths, while small values reduce the spatial correlation leading in the limit to white noise, that is, completely uncorrelated data. The SE is the only continuously differentiable member of the family of exponential GRF. As such, it is not commonly used for capturing road profiles, which are typically not characterized by this level of smoothness. To this end, Stein (Stein, 1999) recommends the Matérn family with the correlation function

\[ k(r; \theta_2) = \frac{2^{1-\nu}}{\Gamma(\nu)} \left( \frac{\sqrt{2\nu}r}{l} \right)^\nu K_{\nu} \left( \frac{\sqrt{2\nu}r}{l} \right), \tag{5} \]

with positive parameters \( \nu \) and \( l \), where \( K_{\nu} \) is the modified Bessel function. The degree of smoothness of the ensuing GRF can be controlled through the parameter \( \nu \): the corresponding GRF is \( p \)-times differentiable iff \( \nu > p \). Note that selecting \( \gamma = 2 \) in
Eq. (4) leads to the OU random process, which is a nonsmooth process although not as
general as the Matérn family.

The three covariance models discussed so far: SE, OU, and Matérn are stationary.

Referring to Eq. (1), this means that for any set of points \( X = \{ x_1, x_2, \ldots, x_M \} \), where
\( M \) is arbitrary, and for any vector \( h \in \mathbb{R}^d \), \( y(X) \) and \( y(X + h) \) always have the same
mean and covariance matrix. In particular, when \( M = 1 \), this means that the GRF
should have the same mean and variance everywhere. Clearly, the stationary assumption
does not hold in many cases. For vehicle simulation, consider the case of a road with a
pothole in it, which cannot be captured by stationary processes. A nonstationary neural
network covariance function has been proposed by Neal (1996):

\[
k(x, x'; \theta_2) = \frac{2}{\pi} \sin^{-1}\left( \frac{2 \tilde{x}^T \Sigma \tilde{x}'}{\sqrt{1 + 2 \tilde{x}^T \Sigma \tilde{x}' \left( 1 + 2 \tilde{x}'^T \Sigma \tilde{x}' \right)}} \right), \tag{6}
\]

where \( \tilde{x} = (1, x_1, \ldots, x_d)^T \) is an augmented input vector; the symmetric positive definite
matrix \( \Sigma \) contains the parameters associated with this GRF and are determined based on
the MLE approach described in 5.1.1. In this context, Rasmussen and Williams (2006) suggest \( \Sigma = \text{diag}(\sigma_0, \sigma_1, \ldots, \sigma_d) \). Recall that for the road profile problem \( d = 2 \).

Using this parameter estimation approach, a mean and co-variance function for the Gaussian process is determined. This is then used to generate new roads which are statistically equivalent to the road used in the learning process.
5.2. **Uncertainty Quantification results: Parametric model sensitivity**

Uncertainty quantification (UQ) is the quantitative characterization of uncertainty in applications. Three types of uncertainties can be identified. The first type is uncertainty due to variability of input and/or model parameters, where, the characterization of the variability is given (for example, with probably density functions). The second type is similar to the first type except that the corresponding variability characterization is not available, in which case work needs modifications to gain better knowledge. The third type, which is the most challenging, is uncertainty due to an unknown process or mechanism (Sometimes type 1 is referred to as aleatory; type 2 and 3 are referred to as epistemic uncertainties). Type 1 UQ is relatively straightforward to perform. Techniques such as Monte Carlo are frequently used. To evaluate type 2 uncertainties, many methods such as fuzzy logic or evidence theory have been proposed. An objective of UQ is to work towards reducing type 2 and 3 uncertainties to type 1. It often happens in real life
applications that all three types of uncertainties are present in the systems under study.

[(Tong, 2006), (Wojtkiewicz et al., 2001)]

It becomes apparent that UQ is essential to the whole model development process from conception to maturity, and that it should probably not be viewed as just a discipline to study the propagation of uncertainty from the inputs to the output of interest. Thus UQ in the generic sense encompasses (not an exhaustive list)

- uncertainty analysis
- sensitivity analysis
- design exploration
- design optimization
- design validation and calibration

The reasons responsible to introduce uncertainty in a model may be:

- The model structure, i.e., how accurately does a mathematical model describe the true system for a real-life situation
The numerical approximation, i.e., how appropriately a numerical method is used in approximating the operation of the system.

The initial / boundary conditions, i.e., how precise are the data / information for initial and / or boundary conditions.

The data for input and/or model parameters.

The previous sections outlined how the time evolution of a system under consideration (HMMWV) is studied using a simulation tool, ADAMS/Car or a combination of simulation tools, PSAT, FTire and ADAMS/Car. Although the simulation tool can be trusted for its high fidelity multibody simulation capabilities, there is an inherent uncertainty in the very model that is being simulated. The vehicle model being simulated is an assembly of a vehicle chassis, a powertrain system, a tire model, a road model and a driver model. There are numerous sources of uncertainty in the model input parameters. For example, the vehicle is simulated assuming some stiffness value for the springs in the suspension system (which is usually the stiffness of the spring when the vehicle is first assembled). After a few years of operation, the changes in material properties, length and
loads, will most definitely change the stiffness value of the spring. Not only that, but
different vehicles will be in different conditions. Thus, a single simulation cannot be
trusted for its results. What we need is a study set of simulations that will put upper and
lower bounds on the results with some confidence. A 95% confidence interval is used in
this work. What this means is that one can say with a 95% confidence that the simulation
result is going to be within the specified upper and lower bound. The numerical
experiments carried out illustrate how the proposed uncertainty quantification framework
is used to predict an average behavior and produce a confidence interval in relation to the
time evolution of a complex multi-body system. A high-fidelity nonlinear ground vehicle
model is considered, and its time evolution is subject to data uncertainty stemming from
measurements of the road profile. This setup was chosen due to its relevance in CBM,
where the interest is the statistics of the loads acting on the vehicle for durability analysis
purposes. Note that a similar analysis is carried out for a simplified scenario that does not
involve the MLE learning stage by (Schmitt et al., 2008) in conjunction with quantifying
the uncertainty of vehicle dynamics when running on icy roads with a stochastic distribution of the tire/road friction coefficient.

5.2.1. Tire and Road Models

Under normal conditions, two types of external forces act on the vehicle and influence its
Figure 5.5: Vehicle on two different road surfaces. The two roads are generated using the same training data.

dynamics: forces at the tire-road interface and aerodynamic forces. Given the range of speeds at which the considered ground vehicle is driven, the former forces are prevalent
and high-fidelity simulation depends critically on their accurate characterization. In this context, the SUV configuration of interest is designed to drive over a variety of terrains, from flat smooth pavement to off-road conditions.

Figure 5.6: Vehicle response (force in the front left lower control arm), obtained using ADAMS. Displayed is only a subset of 10 out of the 200 simulations used to generate the statistics of the normal force response.

The road models supported should run efficiently and accommodate the fidelity level of the tire model. ADAMS uses a road which is defined by a text based road data file (*.rdf). This file contains the road characteristics such as size, type, profile, friction coefficients etc. The road profile can be defined in a number of ways. A triangle mesh is generally used to define a road with varying elevation in both the longitudinal and lateral
directions. Sets of vertices are grouped to define triangles that make up the mesh. Figure 5.4 (Top) shows a section of road made up of this type of triangle mesh. Although the level of fidelity can be as high as the total number of known vertices, the simulation time required per time step increases exponentially with the number of triangles in the mesh. It was determined that for each time step, the simulation had to check every triangle for contact with the tire patch. The bottleneck in the simulation was due to the large number of triangles that describe road profiles. In order to simulate large road profiles with high fidelity the FTire specific regular grid road (*.rgr) data file format was selected. Under the assumption that the input road data points are equally spaced in the lateral and longitudinal directions, a conversion from an ADAMS triangle mesh (*.rdf) to a regular grid road (*.rgr) file type is possible. The advantages of using this type of road data file type are that the file size is smaller and the CPU time per simulation step is independent of the size or fidelity of the road. The obvious disadvantage is that a conversion is necessary and if incorrect parameters are chosen, the file size can balloon or the accuracy of the road profile can diminish. The similarity between a section of road in both (*.rdf)
and (*.rgt) file types when conversion parameters are carefully selected can be seen in Figure 5.4 (Bottom).
5.2.2. Numerical Results, Square Exponential Correlation Function

The SUV model discussed in Chapter 2 was equipped with a set of four tires generated in the FTire modeling and simulation package discussed in subsection 5.2.1 and standard powertrain system in ADAMS. This complete vehicle model was exercised through a straight-line maneuver over a road profile for which information is available on a grid as follows (see also Figure 5.4): in the x-direction, information is provided every 0.25 feet in 180 slices. In the y-direction, the data is provided at a distance of four feet apart in 3 slices. The length of the course in the x-direction was approximately 45 feet. The width of the road was 8 feet. Although not reported here, simulations up to one mile long have been run using this vehicle configuration. The stochastic analysis proceeded according to the work flow in Figure 5.2.
Results: Squared Exponential (Figure 5.7 to Figure 5.9)

Figure 5.7: Statistics of vertical load, bushing attached to LCA.

Figure 5.8: Statistics of vertical load, bushing attached to LCA. Detailed view focused on the last part of run.
A road profile was provided and considered the outcome of a set of field measurements on a 180X3 grid as indicated above. MLE was carried out, the resulting characteristic lengths were $\theta_x = 4.5355$ and $\theta_y = 0.8740$, which are identified in Eq. (4) with $\theta_{21}$ and $\theta_{22}$, respectively. A set of 200 samples were generated and Figure 5.5 illustrates two road profiles out of the 200 generated by sampling of the posterior. The road profile was relatively smooth, in the sense that there was no road geometric feature of length
comparable to the length of the tire/road contact patch; additionally, the terrain was considered rigid and with a constant friction coefficient.

A batch of 200 ADAMS simulations was subsequently carried out to determine the statistics of the vehicle response. One such response is illustrated in Figure 5.6, which reports the force response in the bushing connecting the lower control arm (LCA) to the chassis. There are two such bushings connecting the LCA to the chassis, and there are two more connecting the upper control arm (UCA) to the chassis. There is a grand total of 16 such bushing elements (eight UCA and eight LCA) in the vehicle model. Average behavior and a 95% confidence interval are provided for the normal force in the LCA in Figure 5.7. Figure 5.8 is a zoom-in to better gauge the 95% confidence interval for the LCA load. The vertical acceleration associated with the SE process is reported in Figure 5.9. Note that all results reported in the plots are in SI units.

5.2.3. Numerical Results, Ornstein-Uhlenbeck Correlation Function

The results reported in this subsection are obtained setting $\gamma = 2$ in Eq. (4). This change leads to GRF posteriors that lack differentiability. In fact, the SE ($\gamma = 1$) is the
only exponential GRF that is continuously differentiable. Figure 5.10 shows the load history in the same bushing element as was reported in Figure 5.7. Figure 5.11 is a zoom-in to better gauge the 95% confidence interval for the LCA load. Finally, the vertical acceleration associated with the OU process is reported in Figure 5.12. Note that for OU the two characteristic lengths are \( \theta_x = 1.0849 \times 10^3 \) and \( \theta_y = 0.1830 \).

Results: Ornstein-Uhlenbeck (Figure 5.10 to Figure 5.12)

![Figure 5.10: Vertical load statistics, bushing attached to LCA.](image-url)
Figure 5.11: Statistics of vertical load, bushing attached to LCA. Detailed view focused on the last part of run.

Figure 5.12: Statistics of vertical acceleration, as measured at the CG position of the chassis: mean and 95% confidence interval.
5.2.4. Numerical Results, Matérn Correlation Function

The results reported in this section are obtained by using a Matérn correlation function. The characteristic lengths for this correlation function were identified to be were $\theta_x = 14.1291$ and $\theta_y = 0.7338$. Average behavior and a 95% confidence interval are provided for the LCA bushing load in Figure 5.13. Figure 5.14 is a zoom-in to better gauge the 95% confidence interval for the LCA load. Note that all results reported in the plots are in SI units. The vertical acceleration associated with the SE process is reported in Figure 5.15.

Results: Matérn (Figure 5.13 to Figure 5.15)

![Diagram](image)

Figure 5.13: Vertical load statistics, bushing attached to LCA.
Figure 5.14: Statistics of vertical load, bushing attached to LCA. Detailed view focused on the last part of run.

Figure 5.15: Statistics of vertical acceleration, as measured at the CG position of the chassis: mean and 95% confidence interval.
5.2.5. Discussion of Numerical Results

The results reported in Figure 5.8 and Figure 5.11 suggest that the experimental data is gathered on a dense enough grid because there is a relatively small variance in the response of the vehicle. This can also be seen in Figure 5.8, which illustrates the statistics associated with the last part of the simulation. This is a good indication that the grid used is fine enough to eliminate a majority of the uncertainty that comes from the road description. In other words, the measured road data is provided at a level of granularity sufficient to pinpoint, with good precision, the load history for the force acting at a hot point of the LCA.

The results reported in Figure 5.9 show the statistics associated with the vertical acceleration of the center of gravity (CG) of the chassis. In this context, information regarding the vertical acceleration and the jerk measured at a location where the vehicle driver is positioned is valuable as it is used to gauge ride comfort and potential of vibration induced fatigue for long term exposure of a vehicle driver. Although the plot does not report this information at the driver location but rather at the CG location, it is
expected that Figure 5.9 captures the quantitative trends in the evolution of the vertical acceleration experienced by the driver.

Finally, the comparison of results reported in Figure 5.7, Figure 5.10 and Figure 5.13, or Figure 5.8, Figure 5.11 and Figure 5.14 or Figure 5.9, Figure 5.12 and Figure 5.15 demonstrate the qualitative difference between the SE, OU exponential and MTR GRFs. The OU family leads to processes that are nonsmooth, while the SE family leads to road profiles that are continuously differentiable. This difference is reflected in the smoothness of the output: for SE the response is smooth while OU leads to more roughness in the outcome. Nonetheless, there is good qualitative agreement between the results obtained with the OU and SE. It remains to be investigated whether the roughness associated with OU leads to any significant change in CBM related outcomes. This undertaking falls outside the scope of this study.
5.3. References


Chapter 6

Conclusion

The co-simulation framework developed and described in this thesis can be used to simulate a range of vehicles that can be modeled drawing on a collection of several powertrain systems, tire models and road models. The commercial simulation packages ‘ADAMS/Car’, ‘PSAT’ and ‘FTire’ are leveraged to enable full vehicle analysis through co-simulation. The simulation results can be post-processed using any of the three software packages. This allows one to get useful information about vehicle dynamics (using ADAMS/Car); engine performance, fuel economy, CO₂ emission (using PSAT), and tire-terrain dynamics (using FTire).

The uncertainty quantification framework proposed describes a formal methodology to capture the impact of the input uncertainty on vehicle dynamics response. The focus herein is on uncertainty stemming from the definition of the terrain profile. The approach uses three types of correlation functions namely ‘Squared Exponential’,
‘Ornstein-Uhlenbeck’ and ‘Matérn’. The force in the rear lower control arm was chosen as a representative quantity that was studied to capture uncertainty in the dynamics of the vehicle induced by uncertain terrain definition. The results obtained from all three correlation functions specified above were compared to gauge the quality of input/observed data. The uncertainty quantification framework allows prediction of average (based of 95%, 99% confidence intervals etc.) behavior of a vehicle subject to uncertain model and/or environmental inputs. This is of particular interest for designing a vehicle component or conduct ‘Condition Based Maintenance’ (CBM) studies for an existing vehicle system.
Chapter 7

Appendix: ADAMS – PSAT co-simulation process

This appendix describes the co-simulation software setup between ADAMS and PSAT in greater detail.
A.1. Launching ADAMS/Car

To start ADAMS/Car on Windows, go to start menu select Programs, point to MSC.Software, point to MSC.ADAMS 2005 r2, point to ACar, and then select ADAMS – Car. This will launch the ADAMS/Car user interface as shown in Figure A.1.

Figure A.1: ADAMS/Car interface
Select the “Standard Interface” option. If this choice is not displayed, then it is necessary
to change the configuration file.

Expert and standard users each have a private configuration file with a default name of
.acar.cfg. The template-based product accesses this file at the beginning of every session.

The private configuration file is found at $HOME/.acar.cfg, where $HOME is the
location of the home directory.

Note that the private configuration file is not located in the installation directory. Never
change the acar.cfg file located in the installation directory.

Open the private configuration file .acar.cfg with any text editor. Change the user mode
from “standard” to “Expert”. Save the file and re-launch ADAMS/Car.
A.2. Opening an assembly

Select the “standard interface” option after launching ADAMS/Car as explained in the previous step.

Figure A.2: Opening an assembly
Select File→Open→Assembly as shown in Figure A.2. Right click next to “Assembly Name” in the open assembly dialog box and point to “Search” as shown in Figure A.2. This will show all the databases in the current ADAMS/Car session. Select <acar_shared>assemblies. This will show all the vehicle assemblies in this database. Select “MDI_Demo_Vehicle_lt.asy” and open it. This will open the full vehicle assembly as shown in Figure A.3.
Go to File ➔ Select Directory and set the current working directory to any desired location. This will be the directory where files will be saved after a simulation is run.
A.3. Modifying ADAMS/Car assembly for ADAMS/Controls

A.3.1. Modify powertrain to get driving torque values from PSAT

The assembly uses the “_powertrain_lt.tpl” template for the powertrain system. To make
the co-simulation work, changes need to be made in the powertrain system of
ADAMS/Car. Open the “_powertrain_lt.tpl” template within the template builder.

Notice that templates cannot be opened in the standard interface. Use the F9 key to
switch between “Standard interface” and “Expert interface/template builder”. This can
also be done by selecting Tools → ADAMS/Car template builder. Once the correct
template builder is selected, go to File → open. This will launch the “Open Template”
dialog box as shown in Figure A.4. Right click in the box and select search →
<acar_shared> database. This will show all the templates in the database. Select
“_powertrain_lt.tpl” and click open. Before any changes are made to the database, it is recommended to make a backup of the entire database. This will still allow access the original shared database after the modifications have been made.

Figure A.5: Database navigator

Once the powertrain template is opened, go to Build → System Element → State Variable → Modify. This will launch the database navigator as shown in Figure A.5
Double click _powertrain_lt. This will show a list of all the state variables in powertrain template. See Figure A.5 (right).

Here, any state variable can be selected and modified. PSAT is used as the powertrain model and ADAMS as the vehicle model. This means that the value of wheel axle torque will be computed in PSAT and should be an input to ADAMS as torque acting on the wheels. The state variable holding the value of axle torque in the powertrain template is “total_axle_torque”. This variable needs to be modified so that its value can be populated by PSAT during the co-simulation.

![Figure A.6: Modifying a State variable](image)
Double click “total_axle_torque”. This will launch a “modify state variable” dialog box for “total_axle_torque” as shown in Figure A.6. Set its value for Run-Time Expression to zero. This state variable will later be defined as a plant input and its value will be populated at run time by PSAT (Dyer et al. 2007).

**A.3.2. Modify brakes to get mechanical brake demand from PSAT**

Since the vehicle driving torque is supplied by PSAT to ADAMS, the braking should also be controlled by PSAT. This avoids ambiguous situations where both the driving and braking torques are high.

Open the “_brake_system_4Wdisk” template in the template builder from the shared database. Follow the same steps that were done to open the powertrain template.

Go to Build → System Elements → State Variables → New. This will launch a “create new state variable” dialog box. Create a state variable and give it some relevant name eg. “mechanical_brake_demand_psat”

After this state variable has been created, the following four entities need to be modified in the “_brake_system_4Wdisk” template.
1. left_front_brake_line_pressure

2. right_front_brake_line_pressure

3. rear_brake_line_pressure

4. brake_line_pressures

These are all state variables in the template and use “cis_brake_demand_adams_id”, which is a communicator from the test rig, in their expressions. Replace this with the state variable created earlier, for example “mechanical_brake_demand_psat”.

A.3.3. Use the existing state variable to measure vehicle longitudinal velocity in ADAMS/Car

The longitudinal velocity of the vehicle needs to be reported as feedback to PSAT from ADAMS. The velocity relative to the ground projected onto the vehicle’s X axis (longitudinal axis) is desired. The “longitudinal velocity” variable measures this velocity and can be found in the _rigid_chassis.tpl in the shared database. This records the vehicle velocity in the units of the model (default is mm/s). Note that the value of velocity
reported by this variable is a negative value for the forward velocity. Hence this should be changed later in Simulink by adding a “-1” gain.

A.3.4. Create a state variable to measure ADAMS driveline speed

As described previously, open the “_powertrain_Lt” template in the template builder. Go to Build → System Elements → State Variables → New. This will launch the “create new state variable” dialog box. Give some name for the state variable, eg. “input_axle_speed_PSAT”

![Create State Variable dialog box](image)

Figure A.7: Driveline speed

The run time expression for this state variable is a function which uses values from other state variables in the template. This function is created by selecting appropriate state variables as shown in Figure A.7.
A.3.5. Create/Modify/Delete Plant Inputs/Outputs

In the process of creating the files for Simulink, ADAMS/Car will search for all Plant Inputs and Plant Outputs in the model to specify the inputs and outputs from the ADAMS S-Function block within Simulink.

Figure A.8: Create plant output
Many Plant Outputs exist in the _MDI_SDI_TESTRIG; these can optionally be removed to help simplify the block.

All values to be transferred from ADAMS to PSAT should be specified as plant outputs and all values needed as input to ADAMS from PSAT should be specified as plant inputs.

As described above, “vehicle longitudinal velocity” and “driveline speed” should be plant outputs. To specify these plant outputs, save all modified templates thus far, and open the full vehicle assembly (which requires using the standard interface). Go to Tools → Command navigator to launch the command navigator window.

Then go to data_element → create → plant → output as shown in Figure A.8.

Figure A.9: Create plant output dialog box
This will launch the create plant dialog box as shown in Figure A.9. Give an appropriate name for the plant output and select the variable by right clicking in the window next to “Variable Name” and selecting the correct variables from the subsystems. In this case, select the variables for “vehicle longitudinal velocity” and “driveline speed” and click OK. This will create the plant output for those two variables.

Similarly, “axle torque” and “braking demand” need to be added as plant inputs. To specify these plant inputs, save all modified templates, and open the full vehicle assembly (by switching to the standard interface). Go to Tools → Command navigator. This will again launch the command navigator window.
Then go to data element → create → plant → input as shown in Figure A.10. This will launch the create plant dialog box. Like before, give an appropriate name for the plant input and select the variable by right clicking in the window next to variable name and selecting the correct variables from the subsystems. In this case, select the variables for...
“axle torque” and “brake demand” and click OK. This will create the plant input for those two variables.

A.3.6. Load ADAMS/Controls plugin

ADAMS/Controls is needed to create a Simulink model of the vehicle. To work with ADAMS/Controls in ADAMS/Car, add the controls plug-in to ADAMS/Car.

To do this, go to tools → Plug-in Manager; this will launch the plug-in dialog box as shown in Figure A.11. Check the box under “Load” in front of ADAMS/Controls and click OK. This will load the ADAMS/Controls plug-in for use in ADAMS/Car. If the
plug-in needs to be loaded every time ADAMS/Car is started, check the “Load at
Startup” box.
A.4. **Perform a simulation to export files for ADAMS/Controls**

The next step is to decide which simulation to run. Most of the built-in Full-Vehicle Events can be run, but they must be transient events. Events can always be scripted via the Event Builder if none of the pre-built events can be used. Only models intended for dynamic analysis are valid for ADAMS/Controls co-simulations. Quasi-static vehicle events in ADAMS/Car that are simulations are not valid because ADAMS/Controls does not issue quasi-static analysis commands.

Run a “Straight line acceleration” maneuver ‘files only’ simulation to generate all the necessary files for creating an ADAMS/Car Simulink model.
To do this, open the full vehicle assembly in the standard interface. Go to Simulate → Full vehicle analysis → Straight line events → Acceleration, as shown in Figure A.12.

This will launch the Straight line acceleration simulation dialog box (Figure A.13). Fill in the boxes with the necessary simulation parameters.
Since PSAT is replacing the powertrain in this model, the “Quasi-static Setup” option for any full-vehicle event must be deactivated. A quasi-static setup is a prephase analysis prior to running the transient analysis on full-vehicle assemblies, which uses a static (equilibrium) simulation to achieve a particular throttle and steering demand to drive the
vehicle at a user-set initial speed in a straight line or cornering event (the internal 
ADAMS/Car keywords for this are STRAIGHT and SKID_PAD).

If this option is not selected, ADAMS/Car performs a SETTLE analysis, which simply 
sets the vehicle at the initial speed and does not attempt to change the initial throttle or 
steering demands.

The initial velocity setup in ADAMS/Car is important to consider, since it is output to 
the PSAT longitudinal controller, will ideally be set to the same initial velocity of the 
desired speed (duty cycle) to mitigate initial transients in the model.

Note that the output step size will determine the communication interval between 
Simulink and ADAMS and must be the same (subsequently, the communication must be 
set within the ADAMS block in Simulink, which is explained later). Otherwise, the co-
simulation will not be synchronized.

After the simulation has completed, all simulation files are saved in the working 
directory. To setup a working directory, go to File → Select Directory and select a
location for this directory. After doing this, run a file only simulation as shown in Figure A.13. Then click OK.

This will create the following files in the working directory (File ✔ Select Directory)

<output_prefix>.m – ADAMS/Controls setup file for Simulink

<output_prefix>.inf – ADAMS/Controls setup file for Easy5 (not used here)

<output_prefix>.adm – ADAMS/Car model to be simulated

<output_prefix>_controls.acf – ADAMS/Car Solver Command File for co-simulation

<output_prefix>.xml – Event File to describe ADAMS/Car maneuver
A.5. Create a Simulink model of PSAT vehicle model

As of now, the PSAT vehicle has to be initialized in two steps for co-simulation.

1. Start PSAT with the GUI to define the parameters of the vehicle drivetrain and build the Simulink.

2. Start PSAT manually in MATLAB to integrate the S-function and run the simulation.

A.5.1. Setup and Build the Simulink Vehicle

IMPORTANT: before launching PSAT, make sure that MATLAB R14 SP3 or later is installed and can be used as ActiveX. To test it, open MATLAB, and type:

```matlab
h=actxserver('Matlab.Application.Single')
```

If it returns `h=COM.Matlab_Application_Single`, then it is working. Else, type

```bash
!matlab -regserver
```

and try again to verify it worked.
A.5.2. Launching PSAT

To launch PSAT, go in the Start menu, and then select ‘All Programs’ → PSAT → PSAT.

The Welcome screen should appear as shown in Figure A.14. On this screen, possible selections include,

![PSAT GUI](image)
- Specify a user name, PSAT will create a corresponding folder where all data will be saved. This folder is located in the PSATROOT\users folder, where PSATROOT is the PSAT Installation folder.

- Open the documentation for a full description of PSAT and a step by step example of how to use it.

- Specify which version of MATLAB will be used with PSAT. This is useful if several versions of MATLAB are installed on a computer.

- Specify the location of the PSAT folder. This is useful if several versions/copies of PSAT are installed on a computer.

- Launch PSAT Light Duty or Heavy Duty. The GUI is almost the same for both versions, but the list of configurations, components and drive cycles is different.

Once PSAT has been launched, a ‘MATLAB Command Window’ should appear in the taskbar. This is the MATLAB used by PSAT to read/write the data.
A.5.3. Setting up the vehicle

From the PSAT GUI, there are 2 ways to setup a vehicle in PSAT.

1. Using an existing vehicle file

2. Building the vehicle from a predefined configuration

A.5.4. Using an existing vehicle

To open a PSAT Vehicle file, click on the Open Button at the top left of the screen, and select the file to open. PSAT will load the information in the GUI and MATLAB.

Navigate through the tabs to look at or change the parameters of this vehicle.

![Open Vehicle file in PSAT](image)

Figure A.15: Open Vehicle file in PSAT

A.5.5. Using an empty configuration

Creating a vehicle from an empty configuration requires the following steps:

1. Select the configuration
2. Select the component models and parameters

3. Select the controller models and parameters

**A.5.5.1. Select a Configuration**

To select a configuration, navigate through the Configuration tree until the desired configuration appears. Then, simply double click or drag and drop that configuration in the panel below to initialize the GUI (Figure A.16).

![Figure A.16: Selecting a Configuration](image-url)
A.5.5.2. Select the Component Models and Parameters

In the ‘Drivetrain Components’ tab, a model and an initialization file needs to be specified for each component. To select them, navigate through the Component/Model/Technology tree, which will populate the Initialization File list with the corresponding files.

Then double click or drag and drop the Initialization file to select the model and file at the same time (Figure A.17).

If the exact match cannot be found in the initialization files, use the file that is the closest, and modify its parameters in the ‘Initialization Parameters’ tab at the bottom of the GUI.
For parameters such as “engine power” or “efficiency”, scaling files can be used. They are selected in the upper right list, similar to the initialization files.

Component values can be modified in the ‘Scaling Parameters’ tab as shown in the Figure A.18.

A.5.5.3. Select the Controller Models and Parameters

For most vehicles, the controller is split into 3 parts.

- the propelling: control the components during accelerations
- the shifting: control the gear shifting
- the braking: control the components during braking

Select the initialization file of each of these sub-controller the same way that for the component ones. Some of the controller parameters in the list below can also be modified (Figure A.19).
A.5.5.4. The drive cycle

The drive cycle selected in PSAT has to reproduce the same longitudinal behavior as the one in ADAMS. If none of the existing cycles correspond, a new one will need to be created.

A.5.5.4.1. Creating a new cycle

The PSAT cycles are MAT files containing at least 3 variables:

- **sch_cycle**: matrix which contains the time (sec) in the first column, and the corresponding speed (m/s) in the second column

- **sch_grade**: matrix which contains the time (sec) in the first column, and the corresponding grade (%) in the second column
- `sch_key_on`: matrix which contains the time (sec) in the first column, and the corresponding key ON/OFF (1/0) in the second column.

Once a MAT file with these 3 variables has been created, save it in the `PSATROOT\component\initialization\drive_cycle` folder so PSAT can access it. `PSATROOT` refers to the installation folder of PSAT.

### A.5.5.4.2. Adding a new cycle

If there is already an existing MAT-file, it will be added to the PSAT database to use it.

![Figure A.20: Adding a Cycle](image)
To do so, go to the ‘Simulation Setup’ tab and select a Cycle type. Then right click on one of the cycles in the list, and select ‘Add Cycle to…’ The new cycle will be added at the end of the list (Figure A.20).

A.5.5.4.3. Selecting a cycle

To select a cycle, first select the cycle type which will be used in the ‘Simulation Setup’ tree. Then, double click or drag and drop the cycle which needs to be used. To display the characteristic of a selected cycle, click on it. They will be shown in the ‘Simulation Statistics’ tab (Figure A.21).
A.5.6. Building the Simulink

Once the vehicle has been setup and the cycle selected, the full vehicle Simulink model can be generated.

In the ‘Run Simulations’ tab, make sure that ‘Current Simulation’ is selected in the ‘Simulation List’ tree, and then double click or Drag & Drop the simulation file ‘rerun01’.

To build the vehicle without running the simulation, check the ‘Manual Simu Stop’ at the bottom right of the GUI.

Figure A.22: Build Simulink
Then launch the building and vehicle initialization by clicking on the ‘Run the Simulation…’ link (Figure A.22).

Now save the vehicle and its data before modifying and using it for the co-simulation.

**A.5.7. Saving the vehicle**

Save both the Simulink diagram and the workspace data, for the use in co-simulation. As explained ahead.

**A.5.7.1. Saving the Simulink Model**

In the File Menu of the Simulink Vehicle Model, select ‘Save as…’.

**A.5.7.2. Saving the initialization data**

Use the ‘MATLAB Command Window’ started by PSAT to save the data.

In the ‘MATLAB Command Window’, type the following commands:

```matlab
cd('Name_Of_Folder_Where_Simulink_Was_Saved')
global psat simulation
save data.mat
```

The file data.mat will contain all the data needed to initialize the vehicle Simulink.
A.6. Initialize PSAT Simulink model to be simulated outside of PSAT GUI

Issue the following command (the text in upper case needs to be modified depending upon user specific directory paths) in MATLAB before opening the PSAT Simulink model.

```matlab
%% set user
user_name = 'USER_NAME_TO_BE_EDITED'

%% add paths via psat function "set_path"
cd C:\psatv61\root
set_path
%% working dir should now be set to \psat\users\<user_name>

%% change to simulation directory
cd save_simu
cd DIRECTORY_WITH_MODEL_TO_BE_EDITTED\n
%% load data
load data.mat

%% initialize psat processing parameters
psat.simulation = simulation
psat.current.field = 'simulation'
psat.current.simulation = 1
psat.current.model_name = 'ADAMS_PSAT'
%% Name of the .mdl model to run, without the .mdl extension, e.g.: ADAMS_PSAT in this case

%% length of the simulation, e.g.: 500 seconds
psat.global.gbl_stop_time = 10
initialize_parameters('all')
```
A.7. Modify PSAT model

Remove the differential, wheel, and vehicle blocks from the rest of the powertrain model.

Open the Simulink model from the directory where it was saved in PSAT.

![Simulink Model](image)

Figure A.23: Modify PSAT Simulink Model

Disconnect the differential, wheel and vehicle blocks as shown in Figure A.23.
A.8. Add ADAMS block to Simulink model of PSAT

Run the `<output_prefix>.m` file from the Plant Export performed earlier. This will setup variables for the ADAMS block and add the ADAMS/Controls installation to the MATLAB path.

Enter “adams_sys” at the MATLAB command line.

```
>> adams_sys
```

![Figure A.24: ADAMS Simulink Block](image)
This will find the *adams_sys.m* file in the ADAMS/Controls installation and launch a library of ADAMS blocks, as shown in Figure A.24.

The “State-Space” block is for a linear model, which will not be used here. The other two blocks are essentially the same – the ‘adams_sub’ block contains the S-Function block for the non-linear model – however, the ‘adams_sub’ block has extra variables and scopes.

![Figure A.25: Insert ADAMS Block in PSAT Model](image)

The ‘adams_sub’ block is added to the PSAT model as shown in Figure A.25 and Figure A.26.
The key connection points are as follows

1. The output torque of the gearbox is an input into the ADAMS model (e.g., "total_axle_torque").

2. The mechanical brake command from PSAT ("cmd_wh") is input to the ADAMS model (e.g., "mechanical_brake_command").

3. The longitudinal velocity from ADAMS is sent to the PSAT variable "veh_lin_spd_out_simu".

Figure A.26: ADAMS Block in PSAT Model
4. The transmission speed from ADAMS (e.g., “input_axle_speed”) is sent as feedback to import 3 of the gearbox.

If the existing variable “longitudinal_velocity” from the chassis in ADAMS is used, the velocity must be negated to get a positive forward velocity. This should be done by adding a gain of “-1” before it is reported to PSAT as shown in Figure A.27. The Simulink model seen in Figure A.27 is obtained by double clicking the orange colored “adams_sub” block from Figure A.26.
A.9. Setup ADAMS/Controls block in PSAT Simulink model

Figure A. 28: Setup ADAMS Block
Double click the “MSC Software/ADAMS Plant” block as seen in Figure A.27, and open the window to set up the ADAMS block (See Figure A. 28).

Key settings

- Simulation mode – This must be set to discrete so that ADAMS integrates the ADAMS model and Simulink integrates the PSAT model.

- Animation mode – This must be set to batch to simulate via external ADAMS/Solver.

- ADAMS Solver Type – This must be set to Fortran to use the ADAMS Fortran Solver.

- Communication Interval – This must be set to the output step size for the ADAMS simulation to synchronize the co-simulation.

- Output Files Prefix – Set this to any valid string to generate the ADAMS output files (.res, .req, .msg).
- Use single quotes around all the texts in this window.

**A.10. Run Simulation**

After setting up the PSAT model, select Simulation → Start to initiate the co-simulation. This should launch ADAMS solver.

![Figure A.29: Flowchart for Setting Up Co-simulation Analysis](image-url)
A.11. Post-process Results

The ADAMS results generated by the co-simulation may be imported back into ADAMS/PostProcessor for review. The simulation files will be saved in the working directory. A simple flow chart for the entire co-simulation process is shown in Figure A.29.
A.12. References