Supervisory Control Optimization with Sequential Quadratic Programming for Parallel Hybrid Vehicle with Synchronous Power Sources

by

Rashad Kamal Maady

A Thesis Presented in Partial Fulfillment
of the Requirements for the Degree
Master of Science

Approved February 2017 by the
Graduating Supervisory Committee:

Abdel Raouf Mayyas Co-Chair
Sangram Redkar Co-Chair
Wenlong Zhang

ARIZONA STATE UNIVERSITY

May 2017

ABSTRACT

The thesis covers the development and modeling of the supervisory hybrid controller using two different methods to achieve real-world optimization and power split of a parallel hybrid vehicle with a fixed shaft connecting the Internal Combustion Engine (ICE) and Electric Motor (EM). The first strategy uses a rule based controller to determine modes the vehicle should operate in. This approach is well suited for real-world applications. The second approach uses Sequential Quadratic Programming (SQP) approach in conjunction with an Equivalent Consumption Minimization Strategy (ECMS) strategy to keep the vehicle in the most efficient operating regions. This latter method is able to operate the vehicle in various drive cycles while maintaining the SOC with-in allowed charge sustaining (CS) limits. Further, the overall efficiency of the vehicle for all drive cycles is increased. The limitation here is the that process is computationally expensive; however, with advent of the low cost high performance hardware this method can be used for the hybrid vehicle control.

ACKNOWLEDGMENTS

I want to thank my main advisor Dr. Sangram Redkar for helping me throughout my graduate work, and for caring about my overall progress beyond just my thesis work. I also deeply appreciate him finding time for me every time when I needed it and without his guidance I would not have grown into the person I am now.

I want to thank Mohammed Alzorgan for allowing me to bounce ideas off of him and for double checking my ideas and logic. He also kept me sane when I was trying to balance regular classes, the EcoCAR program, and my thesis. It is good to have someone to joke around with and can relate to exactly what you are going through.

I want to thank my former co-workers Dr. Jeffrey Wishart and Tyler Gray. If it wasn't for their guidance in the early part of my career, I wouldn't have pursued the graduate degree. Also, thanks for listening to me rant a few times a week. Jeff thank you for your help in the early and late parts of my master's degree. Discussing ideas that I wanted to pursue and helping me narrow down what would actually be worth researching. Also thanks for reading and my paper and making edits so that it sounded like an intelligent person wrote it.

I want to thank all of my family, because without you taking care of my dog and pretty much my daily life, I would have had less time to focus on my thesis and my projects. Without them supporting me this would have taken me even longer to accomplish and I would also look more of a disaster than I do now. I want to thank my brother for always being there when I needed you even though you were physically half a world away.

TABLE OF CONTENTS

Page

LIST OF TABLES vi

LIST OF FIGURES vii

LIST OF ABBREVIATIONS ix

CHAPTER

1. INTRODUCTION 1

1.1 Project Purpose 1

1.2 Ecocar 2

1.3 Problem Statement 3

1.4 Limitation Of Study 3

1.5 Summary 4

2. LITERATURE REVIEW 6

3. VEHICLE MODEL 22

3.1 Engine Model 23

3.2 Electric Motor Model 24

3.3 Ess Model 25

3.4 Transmission Model 26

CHAPTER Page

3.5 Vehicle Dynamics Model 27

4. RULE-BASED CONTROLLER STRATEGY 30

4.1 VSC Torque Parameter Calculations 30

4.2 VSC CD Mode Select 33

4.3 VSC CS Mode Selection 34

5. SQP IMPLEMENTATION 38

5.1 SQP In Rule Based Hierarchy 38

5.2 Efficiency Maps 39

5.3 Objective And Constraints Equations 42

5.4 Fmincon Validation 51

5.5 Sequential Dynamic Programming Algorithm 52

6. RESULTS FROM SIMULATION AND HARDWARE TESTS 56

6.1 Simple Drive Cycle Results 56

6.2 Us06 Drive Cycle Results 59

6.3 Hwyfet Drive Cycle Results 64

CHAPTER Page

6.4 505 Drive Cycle Results 67

6.5 Hardware Results 71

7. CONCLUSION AND FUTURE WORK 76

7.1 Conclusion 76

7.2 Future Work 77

REFERENCES 78

APPENDIX

A DATA COLLECTED FROM FMINCON SIMULTAIONS 81

B FMINCON CODE FOR SIMULATIONS 88

C EQUATIONS FOR ENGINE EFFICIENCY MAP 91

D EQUATIONS FOR MOTOR EFFICIENCY MAP 94

E DATA COLLECTED FROM SQP SIMULATIONS 96

F SQP CODE FRO SIMULATIONS 103

LIST OF TABLES

Table Page

1: VSC State Machine Operating Modes (Phillips, 2000, p.299) 12

2: Vehicle Model Specs 29

3: Specification Protected by NDA 30

4: CS SOC Limits 36

5: Engine Constant Values 43

6: Motor Efficiency Constants 47

7: Simple Drive Cycle Results 58

8: US06 Drive Cycle Results 64

9: HWYFET Cycle Results 67

10: 505 Drive Cycle Results 71

11: Hardware Computation Time Results 73

12: Energy Percent Difference 76

13: Region 1 Constant Values 92

14: Region 2 Constant Values 92

15: Region 3 Constant Values 93

16: Region 4 Constant Values 93

17: Motor Positive Torque Efficiency Constant Values 95

LIST OF FIGURES

Figure Page

1: Torque Distribution Strategy for Mild Parallel Powertrain (Song, 2011, p.3) 8

2: DP Node Example Approach (Perez, 2006, p.249) 14

3: Vehicle Configuration.....22

4: Vehicle Model Layout 23

5: Electric Motor efficiency calculation 25

6: 8-Speed Transmission.....27

7: ESS Power Torque Check.....32

8: CD Flow Chart.....34

9: CS Mode Flow Chart 37

10: SQP Flow Logic.....39

11: Engine Efficiency Map 40

12: Motor Efficiency Map 41

13: Surface Curve Fit for Engine Efficiency 42

14: Engine Efficiency Region 1.....44

15: Engine Efficiency Region 2.....45

16: Engine Efficiency Region 3.....45

17: Engine Efficiency Region 4.....46

18: Surface Curve Fit for Motor Efficiency.....46

19: Motor Efficency Map Postive Torque 48

20: Simple Drive Cycle.....57

Figure Page

21: SQP Torque from Simple Drive Cycle 58

22: RB Torque from Simple Drive Cycle 58

23: SQP and RB ESS SOC 59

24: US06 Drive Cycle 60

25: Section of US06 Drive Cycle 60

26: SQP Torque Request US06 Cycle 61

27: SQP Torque Request Closer Look US06 62

28: RB Torque Request US06 63

29: US06 ESS SOC 64

30: HWYFET Drive Cycle 65

31: SQP Torque for HWYFET Cycle 66

32: RB Torque for HWYFET Cycle 66

33: HWYFET ESS SOC 67

34: 505 Drive Cycle 68

35: SQP Torque for 505 Cycle 69

36: RB Torque for 505 Cycle 70

37: 505 ESS SOC 71

38: HIL Vehicle Trace For SQP and RB 73

39: HIL ESS SOC for SQP and RB 74

40: SQP HIL Torque Results 75

41: RB HIL Torque Results 75

LIST OF ABBREVIATIONS

The following are terms used in the following report-

505- Suburban based drive cycle provided by EPA

Blended- Electric motor and engine provide torque to the wheels

CD- Charge depleting

CS- Charge Sustaining

DP- Dynamic programing

ECMS- Equivalent consumption minimization strategy

ECU - Electronic computer unit

EPA- Environmental Protection Agency

ESS- Energy storage system

EM- Electric Motor

HEV- Hybrid electric vehicle

HIL – Hardware in the loop

HWFET- Highway based drive cycle provided by EPA

ICE- Internal combustion engine

MIL- Model in the loop

PHEV- Plug-in hybrid electric vehicle

RB – Rule-Based

SIL – Software in the Loop

SQP- Sequential quadratic programing

US06- Aggressive drive cycle provided by EPA

VSC- Vehicle supervisory controller

1. INTRODUCTION

1.1 PROJECT PURPOSE

The objective of this thesis is to provide a solution to an ever growing demand for fuel efficiency in the automotive sector partially due to the Cooperate Average Fuel Economy (CAFE) Standards adopted in 2008 by the federal government. The CAFÉ regulations currently dictate that automotive industry companies must have a company-wide fuel economy of 54.54 mpg by 2025 (Administration, National Highway Traffic Safety, 2010) This standard has drastically affected the way the automotive industry designs their vehicles. One major vehicle segment being affected is the sports car segment. This segment of vehicles is not usually known for being fuel efficient. The requirements for this segment include high performance and maneuverability. In order to keep this segment vibrant and keep the auto makers profitable, hybridization is gradually being introduced. Hybridization is defined as two or more energy storage systems connected to two or more energy converters, with mechanical energy to the wheels in at least one direction for each energy converter (Wishart, 2010, p.19). The vast majority of hybrid vehicles are hybrid electric vehicles (HEVs), where one or more electric EM (EMs) is/are paired with an internal combustion engine (ICE) along with a high-voltage battery. This move to hybridization comes with a need for a hybrid controller that can source the proper power from the various power producing components. The hybrid controller must be able to increase the vehicles overall fuel economy and reduce its emissions. Secondly the controller must be able to provide the performance required of a sports car.

1.2 ECOCAR

The EcoCAR program, run by the U.S Department of Energy and General Motors, is for university students as a way of getting hands-on experience in converting a conventional vehicle to an advanced vehicle. With the EcoCAR project, the ability to test and validate various key engineering designs and controller algorithm allows students as well as companies to test various architectures and see how well the chosen type of vehicle will appeal to the market. The EcoCAR 3 program, the third iteration of the EcoCAR program, has the specific goal of converting a conventional sports car to an advanced sport car namely a Chevrolet Camaro. ASU successfully bid for a position in the EcoCAR 3 competition, and chose to design and build the conversion to a parallel P2 hybrid sports car. In hybridization there are multiple architectures available. For parallel architectures, a P# designation indicates the location of the EM. P1 indicates that the EM is connected upstream of the ICE usually at the crankshaft. P2 indicates that the EM is connected downstream of the ICE, but upstream of the transmission. P3 indicates that the EM is connected downstream of the transmission. Finally, P4 indicates that the EM is on a completely separate axle to the ICE. The P2 parallel architecture provided an interesting challenge for designing the hybrid controller. Due to the fact that the ICE and EM are connected together to produce power to the wheels, through a clutch and not a Planetary Gear Set (PSG), required that both the ICE and EM angular velocities be synchronous. The reason the ASU team selected fixed shaft design instead of a PSG is due to difficulty in the mechanical design of a PSG.

1.3 PROBLEM STATEMENT

This research deals with the design, and validation of the supervisory controller for the EcoCAR 3 vehicle. The objective of the project is to study the rule- based control strategy with empirical data that minimizes energy consumption that is implemented in the competition vehicle. The secondary objective is to design an algorithm that offers better energy efficiency than the rule-based method and can still be implemented. There have been multiple approaches to solving the problem of a hybrid supervisory controller that reduces energy consumption. Some of the approaches increase the vehicle efficiency of the overall vehicle but are global optimization algorithms, such as dynamic programing (DP). Another approached is using Equivalence Consumption Minimization Strategy (ECMS). These two approaches require either prior data or the finial optimization point and are computationally complex and not real-time implementable. A more in depth analysis of the various approaches is discussed in the literature review. The approached used in this thesis looks is Sequential Quadratic Programing (SQP) because of its ability to quickly calculate a solution and provide an optimized solution using the efficiency maps of the power producing components.

1.4 LIMITATION OF STUDY

There are various strategies that can be implemented to design an efficient power distribution algorithm. This study only looks at the most common forms of controls for modern vehicles and provides one possible alternative to the standard. Many factors such as drivability, emissions, comfort, and thermal effects can dramatic affect the efficiency of a vehicle. The thermal aspect could greatly affect the emissions of the ICE when the ICE is cold. It also affects the battery of the vehicle limiting the amount of power the Energy

Storage System (ESS) could provide during a given drive cycle. The model uses ideal torque and speed values and conditions for the powertrain components. These values are what is used when determining the efficiency of the overall powertrain to meet a given EPA regulated trace.

1.5 SUMMARY

The thesis presents the design and development of the control strategy for the ASU EcoCar 3 vehicle. It will analyze the logic used for the rule based design as well as the propose alternate algorithm that shows improvement in efficiency from the original. It will prove that both approaches are implementable in real time and can be processed with the available hardware.

The thesis is organized in the following manner;

- The introduction is presented in Chapter 1, outlining the need for the thesis work. The background history surrounding the project and the market significance this project plans might have on current OEM's is also discussed.
- Chapter 2, contains the literature review of control strategies. Possible strategies that can be used to improve upon current control strategies to make the overall vehicle more efficient are outlined.
- Chapter 3, discusses the vehicle model and the governing equations of the vehicle model.
- Chapter 4, outlines the details on the rule based control strategy and the strategy used for this thesis work.

- Chapter 5, provides details on the SQP algorithm used and the constraints and the calculations SQP uses to calculate the optimal operating points.
- Chapter 6, compares and contrasts the SIL results of the rule-based and SQP algorithms. It demonstrates the results from testing performed on the hardware.
- Chapter 7, discusses the conclusions gleaned from the thesis results, future applications the controls strategies have, limitations, and possibility of further improvements on the current control strategy design for different varieties of hybridization.

2. LITERATURE REVIEW

With all hybrid vehicles there are supervisory controllers controlling the vehicle system's safety, torque split, and battery health (where applicable), depending on the hybridization design and complexity of the vehicle. The controller will have to be able to control the various operating modes of the hybrid vehicle. In EM only mode, the EM provides all the torque requested by the driver. In the hybrid mode, the ICE and EM work together to provide the power. Within the Hybrid mode, there are multiple options for controlling ICE and EM. Each algorithm has a different focus. Some algorithms look at increased efficiency, using various techniques like DP or ECMS. Others look at the drivability and emissions produced. The controlling and operating points of different modes have been thoroughly researched and explored by researchers. (Bailey, 2002, p.3708-3712)

There are multiple levels of hybridization for Hybrid Electric Vehicles (HEV) micro, mild, and full. Micro hybrids are the lowest form, as at this level, the EM will only be used capture regenerative braking energy and provide start/stop features. The next level is mild hybrids; in this level the EM assists the ICE in propelling the vehicle as well as provides start/stop feature. At full hybridization, the EM can provide large proportion of the wheel torque load and can in some cases provide more torque than the ICE. (Wishart, 2010, p. 22) Examples of vehicles that are parallel hybrids in the current market place are: Honda Civic Hybrids, Honda Insight, Hyundai Sonata Hybrid, Audi A3 PHEV, Volkswagen Jetta Hybrid, and BMW X3 Hybrid.

Recently, researchers (Song, 2011, p.1-5) attempted to increase the efficiency of an inefficient six-cylinder ICE by using an EM in a parallel configuration changing the vehicle into a mild hybrid. The ICE and EM specification for the research are 36 kW for the ICE

and 10 kW for the EM. The overall attempt of the author was to make the system stable in all operating conditions, correctly distribute torque between the EM and ICE, and finally maintain the battery in a suitable SOC range. The control strategy was broken up into two components. The first component controlled the driving action while the second was dealing with energy management. The driving controller examined the driver's actions by monitoring the overall vehicle inputs, key position, accelerator pedal, brake pedal, clutch position, and transmission gear. It determined if the vehicle was in start-up, idle, braking, and driving modes. The second controller then determined the energy distribution and torque distribution between the ICE and EM. The control strategy took into account multiple criteria to turn on and off the ICE. Some of the criterion for shutting down the ICE were the vehicle coming to a complete stop, the vehicle was in neutral, or the SOC of the battery was high enough to supply the vehicle based on its energy demand. The criterion that would have caused the ICE to turn on would have been the brake pedal being released or shifting the vehicle back into drive. The driving controller's objective was to maintain efficiency as high as possible. Using the accelerator pedal input the controller determined the best torque distribution of the powertrain. In Figure Figure *1*, M1 and M2 determine the running mode of the EM, and K1 and K2 determine the EM ability to be a generator or power assistance.

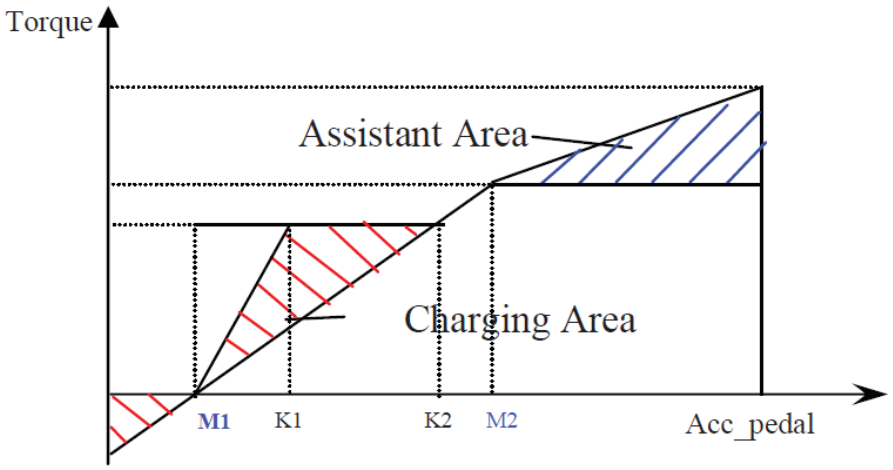

Figure 1. Torque Distribution Strategy for Mild Parallel Powertrain (Song, 2011, p.3)

If the accelerator is less than M1. There is no drive torque demand and the vehicle is braking and the ICE and EM are acting as inertial loads. If the accelerator pedal is between M1 and K2 the ICE ideal efficiency torque points are greater than torque demand. In this area the EM can act as a generator and charge the SOC. If the accelerator is between K2 and M2 the ICE is providing torque on its own because the torque demand is at its most efficient points. Finally, if the accelerator is greater than M2 the ICE and EM both provide torque to the wheels. The overall approach looks at the ICE efficiency map selects the points on the torque map of the IE that the EM can keep the ICE in its most efficient region. Many strategies use the similar approach to keep the battery SOC with in a healthy range from an SOC standpoint.

The commercial vehicle that uses a similar strategy is the Honda Insight. In the Honda Insight, the ICE can provide more power than the EM. This is usually the case for many hybrid vehicles due to the cost of hybridization components like the inverter, EM, ESS, and DC/DC converters.

The control strategy to reduce energy consumption and increase fuel economy for the Honda Insight is using EM at lower speeds and power demand (Bedir, 2009 p.803-807). Keeping the ICE off at lower speeds helps reduce fuel consumptions and emissions. The approach consists of looking at the battery SOC and determining if the battery can provide the power needed to move the vehicle. Then the vehicle can be run on electric power only. At higher speeds the power demand is greater so the logic determines how much power the EM can provide and the rest of the request is sent to the ICE. However, with this approach the electric EM is the main power source and ICE supplies the extra power when needed. The approach does not make the ICE operate in its most efficient range. This will cause a loss of efficiency every time the ICE is utilized.

When determining the proper control strategy, many approaches look to improve the performance of a particular component. One example is to utilize the efficiency map to decrease energy and fuel consumption by looking at the efficiency map of the ICE and manually selecting the working area of the ICE. The designer would create a new torque map that sets the output torque of the ICE to a fixed value regardless of the range the efficiency values available. This approach is the simplest approach however; the selected torque values could still not be the most optimal efficiency point for the overall efficiency of the vehicle. This approach ignores the efficiency of the EM completely. Source (Liu, 2012, p.350-353) looks at the ICE efficiency map and determines at what points the ICE will require assistance to maintain the optimal torque production as well as regions where the EM should act as a generator to force the ICE to provide more torque and recharge the batteries. There are a few equations that are used to determining the torque that must be provided to propel the vehicle based on driver demand. The demand torque is given by

$$T_d(n) = a(T_{e\_\max}(n) + T_{m\_\max}(n)) \tag{0.1}$$

Where a is the accelerator pedal position, n is the ICE and EM speed, T_{e_max} and T_{m_max} are the maximum output torque of the ICE and EM at the current speed, respectively. The equation states that at each given rpm of the EM and ICE there is a max torque available. The accelerator pedal is a percent value from 0 to 1. It is then multiplied to the max torque value to determine the total torque demanded based on the percentage of accelerator pedal The general form of equation 1.1.2 must also hold true.

$$T_{ref} = T_{Mot} + T_{Eng} \tag{0.2}$$

$T_{ref} = T_d(n)$ the accelerator torque request is what will usually be used to determine the torque needed from the powertrain. T_{Mot} and T_{Eng} are the torque requested from both the EM and ICE. The torque request between the two components can differ based on optimization strategies; which will be discussed later.

Another example is (Huang, 2010, p.1-6) In this approach the controller looks at the SOC of the battery and creates a SOC region of operation that will change the torque split between the ICE and EM when the vehicle enters the specific region of the battery SOC, the controller adjusts the ICE and EM to either recharge the ESS or discharge to increase efficiency. The original SOC controls approach is

$$\begin{aligned} T_e &= T_{e\_\min} \quad when \quad SOC > SOC_{high} \\ T_e &= \min(T_{req}, T_{e\_\max}) \end{aligned} \tag{0.3}$$

$$T_e = \min(T_{req} + T_{ch}, T_{e\_\max}) \;\; when \;\; SOC_{low} < SOC < SOC_{high} \tag{0.4}$$

$$T_m = \min(T_{req} - T_e, T_{m\_\max}) \;\; when \;\; SOC < SOC_{low} \tag{0.5}$$

The T_e is the ICE torque T_m is the EM torque, T_{e_min} is the minimum ICE torque, T_{e_man} is the maximum ICE torque T_{m_max} is the maximum EM torque, T_{req} is the torque request from the driver. T_{ch} is the torque to charge the battery, and SOC_{low} and SOC_{high} are the lower and upper bounds of the ESS respectively. With this approach, the torque strategy changes based on the SOC value. This make the overall torque split calculation vary within the torque calculations and if the battery is in-between two states, the logic becomes unstable in the sense that it flickers between the torque calculation states. The paper provides a solution by creating a buffer in the SOC. The SOC envelope is broken down into five different sections. SOC_{High}, SOC_{buff_high}, $SOC_{Nominal,}$ SOC_{buff_low}, and SOC_{low} in each SOC region torque split will be calculated differently. The two additional zones SOC_{buff_high} and SOC_{buff_low} will smooth the torque change and attempt to return the SOC to its optimal state before reach SOC_{high} or SOC_{low}. It also determines if the EM should charge or discharge or keep the EM at a zero torque. There is an extra variable added to these section as well; the optimal torque. The Optimal Torque is a term used for keeping the ICE in the most optimal torque range.

In using the optimal torque value, there are many parameters that need to be determined in keeping the ICE in the optimal area. Those parameters could be the ICE efficiency map, emissions map and fuel consumption map.

Each controller uses state machines for the rule-based approach. Using state machines allows for the ability to transition to as many modes that the vehicle architecture can accommodate. The state machine can also control the criteria when the ICE turns on and off, as well as what torque split algorithm that can be used during specific driving conditions and vehicle state. In (Philips, 2000, p.297-302), the controller has ten different

states that the controller can execute. Each state will determine how the vehicle should operate and provide transitions between modes. The list of modes is shown in Table 1

Table 1: Vehicle State Machine Operating Modes (Phillips, 2000, p.299)

VSC State	ICE	Clutch	EM	Description
Off	ICE Off	Disengaged	Off	Vehicle off state
EM Drive	ICE Off	Disengaged	Tractive Force	EM propelling the vehicle
Regen-Low Velocity	ICE Off	Disengaged	Generating	Regenerative Braking with engine disconnected
Regen-High Velocity	ICE Off	Engaged	Generating	Regenerative Braking with engine connected
Engine Drive	ICE On	Engaged	Off	ICE propelling the vehicle
Boost	ICE On	Engaged	Tractive Force	ICE and EM both propelling the vehicle
Charging	ICE On	Engaged	Generating	ICE propelling the vehicle and charging the battery
Engine Stop	ICE Off	Disengaged	Tractive Force	EM propelling the vehicle and starting the engine
Engine Start	ICE On	Engaged	Tractive Force	EM propelling the vehicle and stopping the engine
Bleed	ICE On	Engaged	Tractive Force	ICE propelling the vehicle and motor discharging the battery

In each operating mode there is a potential to optimize the power distribution between the ICE and EM. Potential ways of optimizing the transitions and decision matrix for entering modes could be done by using fuzzy logic.

In another approach (Salman, 2000, p.524-528), the controller used fuzzy logic to determine conditions the vehicle should enter based on inputs and rules set by the fuzzy logic. This approach allowed for multiple possible transitions and a way to make the vehicle enter modes that were optimal based on the current states of the vehicle. The approach took into account three inputs, the driver demand, battery SOC, and EM speed. Nine rules were used in the fuzzy logic. The logic then produced two outputs to the torque split. First would be the available generator power the EM could provide to recharge the battery when needed. Second was a scaling factor that determined if the EM could act like

a generator or a torque producing component. If the battery had to be recharged the scaling factor would be zero. If the battery had enough charge and could provide torque to keep the ICE in its optimal torque producing region the scaling factor would be 1. The rules in-between then scaled the scaling factor based on the region the fuzzy logic. This approach was helpful in making the transition between modes more "fluid" in the sense that small changes could allow the mode to change compared to a pure rule-based approach, which required defined values to be met before allowing a transition to a different state. However, this approach did not really improve upon the overall efficiency of the powertrain. This approach was still similar to a rule-based approach when it comes to vehicle performance.

This approach did add more computation time with similar results to rule based controllers. As discussed earlier there are multiple approaches to achieve the overall goal of reducing the fuel consumption and increase efficiencies. One approach is to use DP, as this approach identifies all the available possible options in which the vehicle can distribute power based on the objective function as well as the constraints to the objective function. In (Perez, 2006, p.244-254) the objective function was

$$V[P_{FT}(t)] = \int_0^T \frac{P_{FT}(t)}{\eta_{FT}(P_{FT}(t))} \tag{0.6}$$

Where $V[P_{FT}(t)]$ is the velocity at the given power of the fuel tank at a giving time step. η_{FT} is the efficiency of the ICE. The objective function had constraints imposed on it:

$$\dot{E}_{Ess}(t) = f(P_{req} - P_{FT}(t)) \forall t \in [0,T] \tag{0.7}$$

$$E_{ESS}(0) = 0 \tag{0.8}$$

$$0 \le P_{FT}(t) \le P_{FT_{Max}} \forall t \in [0,T] \tag{0.9}$$

$$P_{ESS_{Min}} \le P_{req} - P_{FT}(t) \le P_{ESS_{Max}} \; \forall t \in [0,T] \quad (0.10)$$

$$E_{ESS_{\min}} \le E_{ESS}(T) \le E_{ESS_{Max}} \; \forall t \in [0,T] \quad (0.11)$$

In these equations, E_{ESS} is the energy of the ESS, and P_{ESS} is the power available from the ESS. With these constraints the DP algorithm can determine the paths that are truly feasible based on the given vehicle parameters.

The algorithm creates an iterative process that will take the initial starting point and then determine the possible next points based on the given power and energy available. It will than weight each transition. Once it has calculated all possible transition to the optimal point, which will need to be known, the algorithm will back track through the nodes and determine the least expensive route to achieve the optimal point. With this approach the full drive cycle needs to be known. If the whole drive cycle is not known, then an arbitrary horizon point needs to be determined but this will reduce the efficiency of the algorithm. Figure Figure *2* shows and example of how DP works in determining the power split between the EM and ICE.

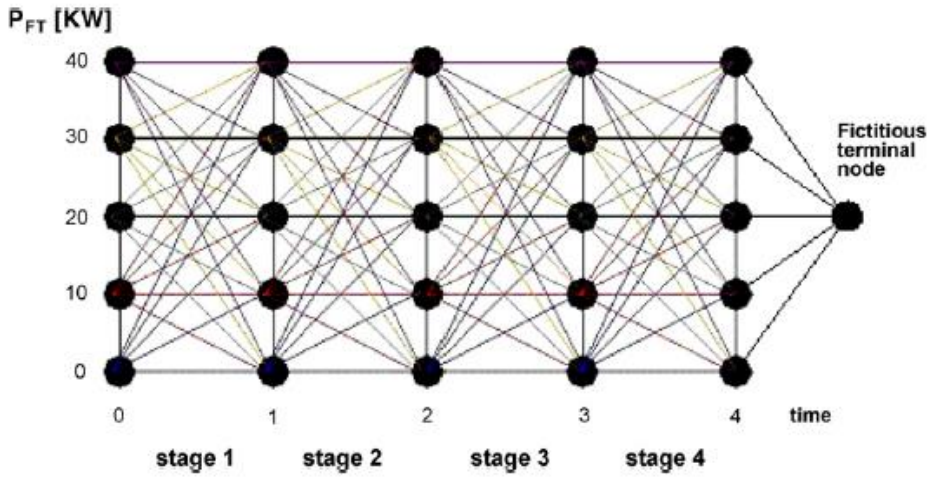

Figure 2. DP Node Example Approach (Perez, 2006, p.249)

Due to the number of calculations the algorithm requires DP isn't possible to be implemented in to real world applications. The computing time to achieve a solution is not

possible on present-day microcontrollers. This approach is also focusing on a single component to optimize.

The next approach is ECMS. The ECMS algorithm looks at the current and future energy use to determine the overall efficiency of the vehicle. The optimization looks at the instantaneous optimal power-split between the ICE and EM while working with in vehicle constraints. The algorithm looks at the amount of energy consumed by electric energy. It also determines how much electrical energy must be returned back into the ESS to maintain SOC and provided discharge later during a drive. The algorithm uses two different coefficients for charging (s_{chg}) and discharging (s_{dis}) statues. These coefficients are optimized for the complete drive cycle, which affects the overall energy balance (Kim, 2010. P.1279-1287). The governing equation for ECMS is as follows:

$$J_t = \dot{m}_{ice}(P_{ICE}(t)) + \zeta(P_{EM}(t)) \tag{0.12}$$

Where $\zeta(P_{EM}(t))$ is the fuel equivalent of the electrical energy. To determine the electrical energy equivalent, the following equations are used:

$$\gamma = \frac{1+\sin(P_{EM}(t))}{2} \tag{0.13}$$

$$\zeta(P_{EM}(t)) = \gamma * s_{dis} \frac{1}{\eta_{batt}(P_{EM})\eta_{EM}(P_{EM})} \frac{P_{EM}(t)}{H_{LHV}} + (1-\gamma) * s_{chg} * \eta_{batt}(P_{EM})\eta_{EM}(P_{EM}) \frac{P_{EM}(T)}{H_{LHV}} \tag{0.14}$$

Where H_{LHV} is the lower heating value of fuel. ECMS depends entirely on the equivalence factors. If the values are not accurately tuned for the given drive cycle, the resulting performance would be poor or would not maintain the charge sustaining conditions.

To improve on this approach, researchers proposed (Shankar, 2012, p.4892-4923) the instantaneous power-split, that can be represented with β, and the base equations are

$$P_{ICE} = \beta * P_{dmd} \tag{0.15}$$

$$P_b = (1-\beta) * P_{dmd} \tag{0.16}$$

$$P_{ICE\_MIN} \le P_{ICE} \le P_{ICE\_MAX} \tag{0.17}$$

$$P_{B\_MIN} \le P_B \le P_{\_MAX} \tag{0.18}$$

The β is calculated at each time step by minimizing the cost function (J)

$$J = MIN(g + g_{equiv} * \zeta) \tag{0.19}$$

Here, g represents the instantaneous mass flow rate for the ICE and g_{equiv} represents the amount of electric energy used by the ESS and the units converted to equivalent fuel energy to determine the amount of fuel respectively used by the ESS to return the SOC back to original value. The variable ζ defines the charge-sustaining penalty function. The penalty function is calculated from and PI controller that outputs a value 0 to 10 in order to avoid potential instabilities that can occur from the PI controller. The input of the PI controller is the SOC_{ref} set by the controller to maintain the SOC as well as the current SOC value.

Another approach to improve on the ECMS algorithm is using the Adaptive ECMS algorithm. The improvement to the ECMS algorithm requiring knowledge of the drive cycle to optimize the constants s_{chg} and s_{dis}. (Rizzoni, 2005, p.509-524), which is infeasible in real-world applications. The idea of using GPS and vehicle system information to predict the coming drive cycle and setting the equivalence factors to the proper value will allow the strategy to be used in real time. This will replace the knowledge requirement for the drive cycle priori and also be able to adapt the vehicle to the changing road conditions or driver input. It also increases the stability of the algorithm and reduces the sensitivity to the equivalence factors so that tuning is unnecessary. For s_{chg} and s_{dis}, a bi-dimensional

minimization problem reduces to a single dimensional nonlinear optimization assuming that

$$s_{dis}(t) = s_{chg}(t) = s(t) \tag{0.20}$$

where the value of *s(t)* is the averaged between s_{dis} and s_{chg}. This approach gives similar results to ECMS; however, it requires additional information from outside controllers and GPS positioning in order to replace the priori drive cycle knowledge requirement.

Another approach is to use an iterative process that can determine the local optimal point of operation for various objectives. Sequential Quadratic Programming (SQP) is an optimization approach that is able to find the minimization of an objective function using various sub algorithms that decrease its computation time and number of iterations to solve the optimal problem. Due to the number of methods SQP uses and can incorporate into finding a solution, a lot of work has gone into make the SQP method as efficient as possible some examples will be discussed in this thesis. An algorithm like SQP uses a matrix approach to solving the problem. Some of the problems are complex to handle, one approach uses semi-infinite nonlinear equations (Wilde, 2000. p.317-350), while another uses the interior point method to handle large scale nonlinear programming (Albuquerque, 1999, p.543-544).

Due to the intensive nature of SQP, for application use, a lot of work has been done to help SQP calculate a limited range. One example is using Evolutionary Programming (EP) (Attaviriyanupap, 2002, p.411-417). In this process the region in which the SQP is solving is narrowed. This is possible because, the EP algorithm determines the best local solution that can be passed to the SQP algorithm from a solution based on the EP local solution. The EP is a global stochastic optimization method which can start from multiple points,

but it requires a long computation time and suffers from convergence problems. So, to combat the convergence issue, SQP is used as it can start at a singular point and use gradient methods to solve the solution with a low computation time. The process initially starts with EP finding a point that the SQP can start with, then transfers the results to the SQP algorithm, where SQP will find the local minimum. The SQP algorithm will have the same constraints that EP uses.

SQP can also be used as standalone solution for finding the minimum. The approach looks at the desired EM power $P_{m\_desired}$ the weighting W, and the EM peak power P_{peak}

$$P_{m\_desired} = W * P_{peak} \tag{0.21}$$

The weighting W is based on different states of the accelerator pedal, vehicle velocity, and the ESS SOC. The weights take values X_1, X_2, X_3, X_4, X_5, and X_6 that will be provided from the optimization process as well as the vehicle parameters. The vehicle parameters are state space equations of the state variables describing vehicle dynamics:

$$x(t) = \begin{pmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \\ X_5 \\ X_6 \end{pmatrix} = \begin{pmatrix} T_{ce} \\ T_{em} \\ i \\ FC \\ SOC \\ V \end{pmatrix} \tag{0.22}$$

Here, $X_{(t)}$ is the state vectors, T_{ce} is ICE torque, T_{em} is the EM torque, I is the speed ratio, FC is the ICE fuel consumption, and V is the vehicle velocity the equations for the state

variables. The vehicle dynamic governing equations that are used as part of the X values are

$$\dot{T}_{CE} = \frac{T_{CE\_desired} - T_{CE}}{\tau_{CE}} \tag{0.23}$$

$$\dot{T}_{EM} = \frac{T_{EM\_desired} - T_{EM}}{\tau_{EM}} \tag{0.24}$$

$$i = \beta(i)\omega_p(F_p - F_{\dot{P}}) \tag{0.25}$$

$$F\dot{C} = T_{CE} * \omega_{CE} * BSFC \tag{0.26}$$

$$SOC = SOC(t_i) - \int_i^f i_a(t)dt \tag{0.27}$$

$$\dot{V} = \frac{i\frac{N_d}{R_t}(T_{CE} + T_{EM}) - F_L - F_b - \frac{N_d^2 i\frac{di}{dt}J}{R_t^2}V}{M + (\frac{iN_d}{R_t})^2 J} \tag{0.28}$$

Where $\beta(i)$ is the constraint which is a function of the CVT ratio I, ω_p is the primary pulley speed, F_p is the primary thrust at a steady state, $BSFC$ is the rake specific fuel consumption, ω_{ce} is the ICE speed, i_a is the current, M is the vehicle mass, R_t is the tire radius, N_d is the final reduction gear ratio, J is the vehicle equivalent inertia, F_L is the road load and F_b is the braking force.

The objective function and constraint are:

$$minimize\ f(u) = T_{ce} * \omega_{ce} * BSFC * tf(u) \tag{0.29}$$

$$subject\ to\ g(u)\ \ g(u) = 1 - \frac{SOC_{(tf)}}{SOC_{(tf)}} = 0 \tag{0.30}$$

Here $f(u)$ is the fuel consumption, and $g(u)$ is a function of ESS SOC. After the optimization finishes, the output variable will be a weighting to which vehicle dynamic effects the overall efficiency and is then placed in the large weight to determine the final drive torque split. In this approach (Oh Kyoungcheol, 2005)the objective function is only looking at the ICE performance and constraints it with the battery SOC. The authors also only compare their results between two different architecture types; one in which the ICE cannot be clutched from the system, the second in which it can be clutched from the system. The overall vehicle architecture is needed to provide the solution, and requires every aspect of the vehicle to be known and modeled. The calculation is a backwards torque calculation where the power at the wheels is determined, and then the torque required from the powertrain is then calculated to achieve the desired power.

The SQP approach can also take in different equations to adjust the overall optimal solution using the constraints of the problem. The algorithm can also handle different forms of sensitive and weighting to the objective functions. The various weights can affect the final optimal solution provided by the SQP algorithm. In the work by Kim (Tae Soo Kim, 2009) the implementation of a weighting for the SOC constraint using the bases of ECMS to determining the weighting for the amount of electrical energy is used. The SQP based on the weighting adjusted the final value of the optimal torque split. Adding the constraint and changing the SOC constraint cause an increase in computation time. It as well, had trouble with the weightings of the SOC to maintain the SOC during the drive cycle. The value for a fixed weighting of the SOC gave unphysical values that the powertrain could not physical perform. However, the approach of using weightings to adjust the results did give the SQP algorithm more dimensions of freedom allowing it to be affect by the constraints allowing

the SQP algorithm to handle more dynamic situations. The research from (Yuan Zhu Y. C., 2006) indicated that the SQP is not robust against disturbances with the weighting the SQP algorithm could better handle the disturbances.

The SQP algorithm can handle future architectures like fuel cells. In the work of (Young-Bae Kim, 2011) the SQP algorithm is used to determine the minimal hydrogen consumption of the fuel cell stack. The author uses a D-optimality method to select the experimental points for the controller to operate in. Then the SQP algorithm is used to find the system's optimal operating parameters and the power distribution from the fuel cell stack and the battery.

With all approaches, overall constrains are usually similar. The torque requested by the driver must be achieved, the SOC of the ESS must be within a specified region to maintain the health of the ESS, and the ICE and EM must remain within the allowed torque limits.

3. VEHICLE MODEL

The vehicle model used for this thesis is the same model used in the design process for ASU's ECOCAR. The architecture is a parallel P2 full plug-in hybrid vehicle. The model was designed using Matlab Simulink and Simscape environments. The vehicle architecture, main components and power flow are shown in Figure Figure *3* while the overall model setup is shown in Figure Figure *4*. The components used in this architecture are a GM LEA 2.4L ICE, GKN AF-130 EM, GM 8L90 Transmission, and an A123 18.9 kWh ESS. The specifications and model equations will be shown later on in this chapter.

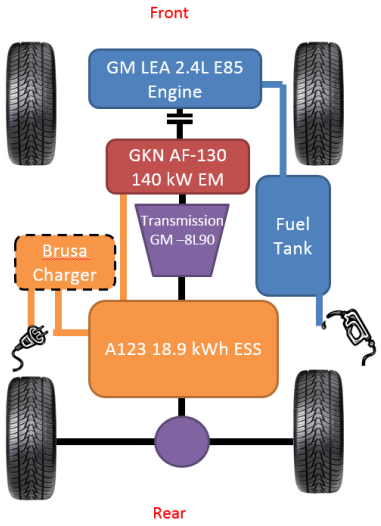

Figure 3. Vehicle Configuration

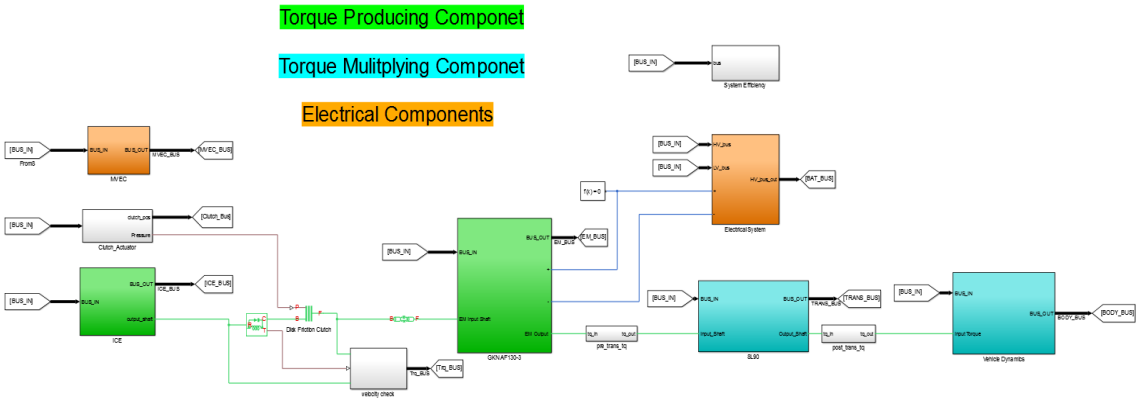

Figure 4. Vehicle Model Layout

Majority of the vehicle models, torque producing, high voltage, and vehicle body components are developed in Simscape for easier implementation. Simscape was the choice of modeling environment due to its high fidelity pre-built equations, and the modeling environment does not require the developer to design all the equations and account for all the possible equations need to handle vehicle dynamics.

3.1 ENGINE MODEL

A simple ICE model block is used to handle the dynamics created from the ICE. This block uses tabular data to determine the amount of torque requested from the driver based on an input of throttle position request. The throttle position has a range from 0 to 1. The tabular data query is used in order to make the ICE respond with the correct torque request at any given speed of the ICE. The throttle position was normalized to the maximum torque curve. The block also uses fuel consumption maps to determine the amount of fuel used. The block uses the following functions to determine the torque output (Matlab Engine Documentation):

$$T = (P_{\max} / \omega) * [p(w) / w] \qquad (2.1)$$

Where *T* is the torque output will be in the units based on the data provided to the block, *Pmax* is the maximum power at the given speed, *p(ω)* is the power of the polynomial based on a third order equation of the given data. *ω* is the current speed. The ICE block uses a third order polynomial equation to solve for *p(ω)* that satisfies equation 2.3:

$$p(\omega) = p_1 * w + p_2 * w^2 + p_3 * w^3 \tag{2.2}$$

$$p_1 + p_2 - p_3 = 1\,,\ p_1 + 2p_2 - 3p_3 = 0 \tag{2.3}$$

> "$P_{(i)}$ are positive constant values. This polynomial has three zeros, one w = 0, and a conjugate pair. One of the pair is positive and physical; the other is negative and unphysical:" (Matlab Engine Documentation)

$$W_{\pm} = \frac{1}{2}\left(-p_2 \pm \sqrt{p_2^2 + 4p_1p_3}\right) \tag{2.4}$$

3.2 ELECTRIC MOTOR MODEL

The next torque-producing component is the EM. The EM takes the torque request directly and produces the torque request based on the maximum torque curve and speed data. It then calculates the amount of current needed from the ESS to produce the torque requested. The EM block has two different electric loss calculations. The first uses tabular data if available; the other calculation uses a fixed value to determine efficiency throughout the EM speed and torque range. To make calculations easier and with more control, the EM block was set to use the latter calculation of a fixed value; the value was set to 100% efficiency. The purpose of 100% fixed value used for the EM block was avoid the summation of two different efficiency values affect the power needed from the ESS. The

EM block in Simscape didn't allow the use of the efficiency table available. This allows a separate calculation outside the block to determine the correct amount of power needed from the ESS using the efficiency maps. Figure Figure 5 shows the calculation used to determine the power needed from the ESS.

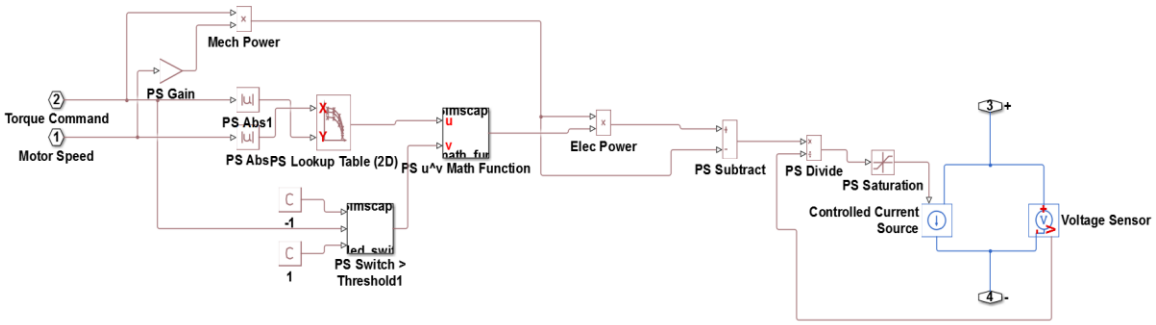

Figure 5. Electric Motor efficiency calculation

The EM torque and speed is input, and a 2D look-up table to determine the efficiency of the EM. That efficiency value then gets multiplied by power value of 1 or -1, depending on whether the EM is providing torque to the wheels or generating torque via regenerative braking for recharging the ESS. That value is then multiplied by the mechanical power to determine total power by the EM. Then the mechanical power is subtracted from the total power needed to determine the exact electric power needed from the ESS. The power value is then divided by the current ESS voltage to determine the amount of current needed from the ESS.

3.3 ESS MODEL

The next major component is the ESS. The model was based on the physical pack received by A123. The ESS data was supplied by the manufacture to assist in modeling the ESS characteristics. The data provided includes the maximum current available at a given SOC, the resistance of each cell, and the open circuit voltage of each cell. The ESS SOC was

calculated using the current and determine the SOC based on the available capacity of each cell: (Matlab Documentation, R2015b)

$$SOC = \left(\int I * \left(\frac{-1}{3600} \right) \right) * \left(\frac{1}{(ESS_{capacity} * 100)} \right) \tag{2.5}$$

Where the I is current and $Cell_{Capacity}$ is the overall capacity of each cell. The SOC is than used with a look-up table to determine each cell's open circuit voltage and resistance. The voltages are then added together based on the configuration to determine the packs total voltages.

3.4 TRANSMISSION MODEL

The next component is the 8-speed transmission. This block was provided via the Mathworks library. Figure *6* shows the transmission design, and it resembles the design of the transmission used in the vehicle within a 5% variation for simulation and testing. Minor adjustments were made to the stock block. Inertial values provided by GM were added to mimic losses expected in the transmission. Also, the shift speeds and lag needed to change gears were adjusted to prevent the overspeed condition from occurring in the ICE and EM during shifts. The gear specifications of the planetary gears used in the transmission are also set to the actual value. This allows for the desired gear ratio to be achieved then the planetary gear is clutched to eh output shaft.

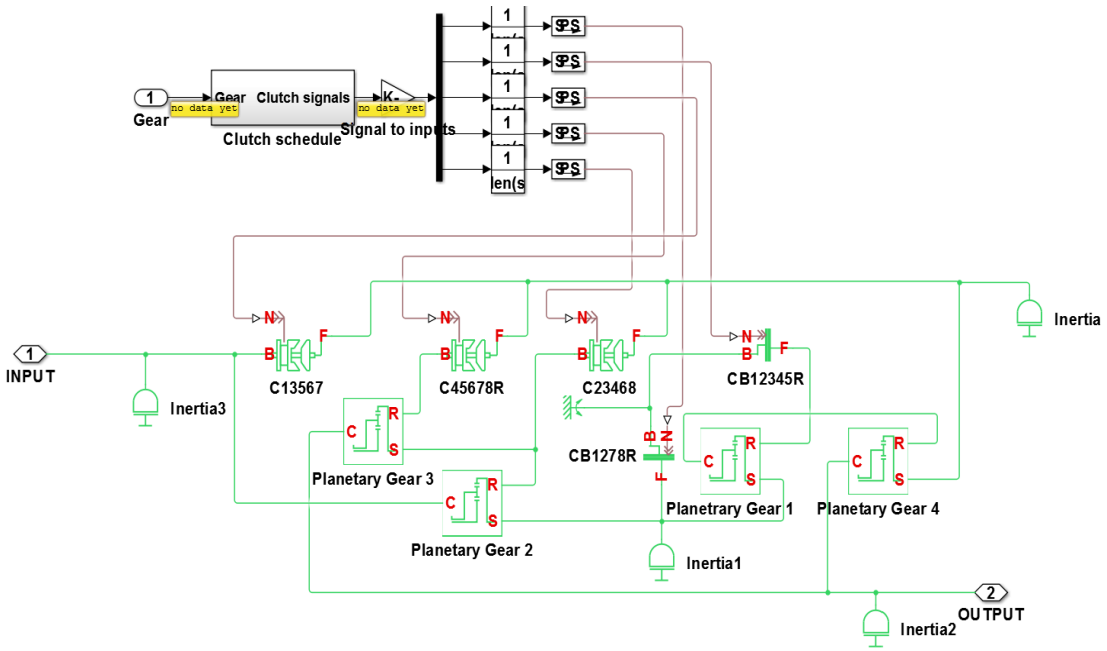

Figure 6. 8-Speed Transmission

3.5 VEHICLE DYNAMICS MODEL

Finally, the vehicle dynamics are split into multiple parts. First there is the vehicle body, then the wheels, and finally the brakes. The vehicle dynamics do not account for any lateral g forces only longitudinal. The wheel force is calculated using the Tire-road interaction formula, which determines the amount of tractive force produced at the contact surface as well as the wheel slip (Matlab Tire Doumentation, R2015b):

$$F_x = f(k, F_z) = F_z * D * \sin(C * \arctan[\{B_k - E * [B_k - \arctan(B_k)]\}]) \tag{2.6}$$

where F_x is the tractive force on the contact point, B, C, D, E are dimensionless coefficients for stiffness, shape, peak, and curvature, respectively. F_z is the vertical load of the tires and k is the wheel slip. Wheel slip is calculated

$$k = V_{sx} / |V_x| \tag{2.7}$$

$$V_{sx} = r_w - V_x \tag{2.8}$$

Where V_{sx} is wheel slip velocity, V_x is wheel hub longitudinal velocity, and r_w is the wheel radius. The wheel calculations allow the model to accurately determine the tractive force that can be produced with the wheels that are on the vehicle. This also will help when simulating vehicle acceleration and stop times based on the grip of certain tires.

The body of the vehicle is calculated using the following equations (Matlab Vehicle Body Documentation, R2015b):

$$m\dot{V}_x = F_x - F_d - mg * \sin\beta$$

(2.9)

$$F_x = n(F_{xf} + F_{xr}) \tag{2.10}$$

$$F_d = \frac{1}{2} C_d \rho A (V_x - V_w)^2 * \sin(V_x - V_w) \tag{2.11}$$

Where g is gravitational acceleration = 9.81m/s^2 , β is incline angle, m is the vehicle mass, V_x is longitudinal vehicle velocity, V_w is headwind speed, n is the number of wheels on each axle, F_{xf}, F_{xr} is the longitudinal forces on each wheel at the front and rear ground contact points, A is the effective vehicle frontal cross-sectional area, C_d aerodynamic drag coefficient, ρ is the density of air 1.18 kg/m^3, and F_d is he aerodynamic drag force. This vehicle block allows the vehicle to be simulated on a hill and provides a simple way to input all the parameters needed for the vehicle body being used on the actual vehicle.

The clutch between the ICE and EM is also based on a Simscape block that in-corporates slip and then frictional torque transfer. In the model, the clutch allows the ICE to disconnect from the rest of the drive-train, which allows the effective inertia from the ICE to be removed. When the ICE is turned on and connected to the drivetrain, the inertia of the ICE is added, as is the rate at which the clutch connects and produces shocks to the driveshaft

between the EM and ICE that causes torque spikes every time the ICE clutches in and out of the drive-train.

The overall torque power flow for this model is as the shown in equation 2.12:

$$T_E = \frac{((T_{Eng} + T_{Mot}) * gr_{trans}) * gr_{diff}}{R} \tag{2.12}$$

Where T_{Eng} is the ICE torque, T_{Mot} is EM torque, gr_{trans} transmission gear ratio, gr_{diff} final drive ratio, and R is the tire radius.

The objective of the model was set as high fidelity as possible so that as many aspects of the physical vehicle are used, and that the model accurately calculates the expected losses, shocks, and behavior that could be expected from the drive train. The vehicle parameters used in the model are given in Table Table *2* and 3, but there are some parameters that cannot be listed due to the NDA associated with the EcoCAR 3 program.

Table 2. Vehicle Model Specs

Vehicle Specs	
Mass	1900 kg
Wheel Radius	0.3 m
Engine Speed Range	0-6500 rpm
Engine Torque Range	0-231 Nm
Electric Motor Speed Range	0-8000 rpm
Electric Motor Torque Range	(-350)-350 Nm
Battery Energy	18.9 kWh
Battery Capacity	59.8 Ah
Improvised Gear Ratios	4.55, 2.95, 2.08, 1.67, 1.22, 1.00, .80, .62

Differential Ratio	3.27

Table 3. Specification Protected by NDA

Engine	Fuel Flow
	Internal Inertias
	Mass Air Flow
Transmission	Internal Inertias
	Actual Gear Ratios
Vehicle	Mass
	Frontal Vehicle Cross Section
	Drag Coefficiencies
ESS	Cell Configuration

The Driver model for the overall simulations uses a simple PI controller. The PI controller is manual tuned. The values from the PI controller are from -1 to 1. This would present positive torque request and negative torque request or braking. The values are continually until the vehicle trace is with in the 2% error of the reference vehicle velocity trace. The values use for the proportion gain is 0.25 and the integral gain is 0.075.

4. RULE-BASED CONTROLLER STRATEGY

4.1 VSC TORQUE PARAMETER CALCULATIONS

The VSC is broken down into two main sections. The first section is the torque delivery calculations and initial parameters. In this part of the code the VSC determines the torque request from the driver. It also looks at the amount of power available from the battery and

determines if the battery has enough power to deliver the torque request by the EM. These calculations are run at every cycle of the processor so that the lasts calculations based on the speed and torque request from the driver are available. The equation for determining the torque request is same to Equation 1.1. Once the torque request from the driver is determined, the EM torque request goes through the battery power check. The battery power check looks at the current available power of the battery based on the available current discharge allowed at the current SOC and the current voltage of the battery. It takes into account the parasitic loads on the HV system, such as the DC-DC converter power draw for the 12-volt system, theoretical losses of battery power from cables and the HV distribution box, and finally the amount of electrical power needed for the EM. A buffer is also subtracted from the amount of power of the ESS; this is an added precaution available from the actual battery, so that the battery power limits are not exceeded, thereby causing a fault or fuse to blow. If the EM toque request is greater than the actual power available from the ESS the torque is than calculated based on the power available, as shown in:

$$T_{Batt} = Pwr_{Available} / \omega_{Mot} \tag{3.1}$$

The additional torque needed is then provided by the ICE. Figure Figure *7* shows the Simulink layout of the ESS power calculations.

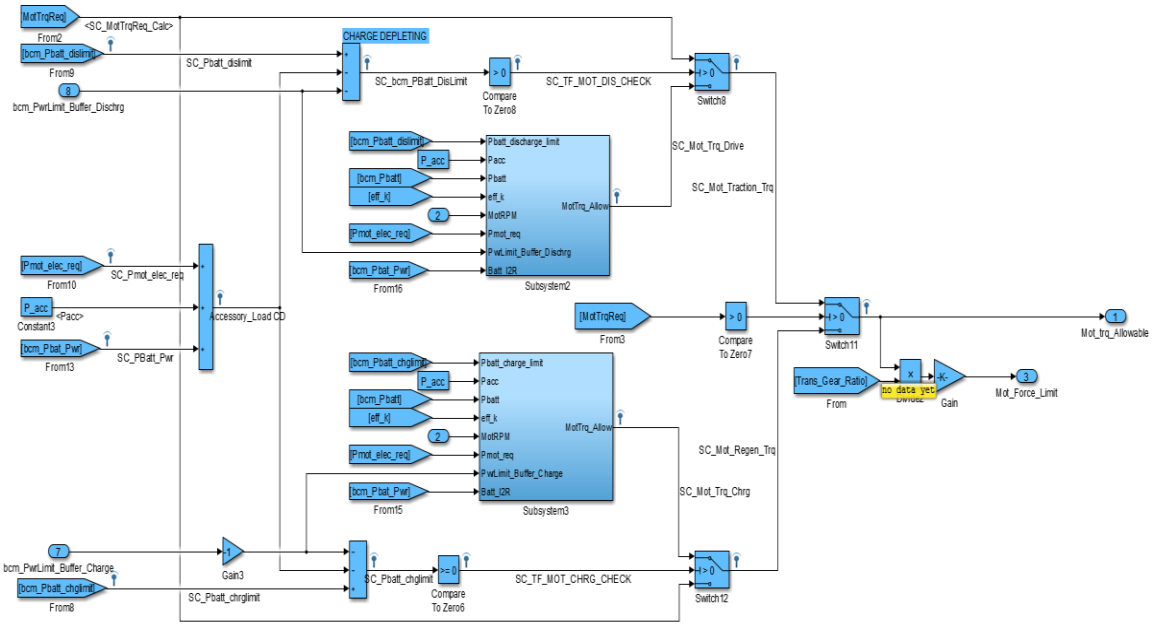

Figure 7. ESS Power Torque Check

The other half of the ESS torque calculation is to limit the amount of power when recharging the ESS. If the ESS is near its maximum voltage, the calculation will limit the regenerative torque request from the EM when the vehicle is decelerating. Due to the controller design requirements there is no regenerative brake blending. Meaning that when the brakes are applied physical braking is automatically applied using equation

$$F_{braking} = V_{Mass} * 1 * 9.81 \tag{3.2}$$

Where $F_{braking}$ is the force the physical brake applied to the vehicle and, V_{maxx} is the mass of the vehicle. This calculation assumes the maximum braking force will be 1g. The regenerative torque is then over laid on the mechanical braking. It is understood that the amount of energy that can be captured using brake blending will be less with the current approach but because of the physical vehicle restriction the controller was designed in this manner.

Other parameters that effect the torque requested are calculated in this section such as, the charge depleting (CD) torque calculations as well as charge sustaining (CS) as well as, the torque multiplication expected from the gear ratio and the differential is calculated as well.

4.2 VSC CD MODE SELECT

The second component of the VSC is to determine what mode the vehicle is in and takes the available torque values from the first component and determines the final torque output based on the vehicle mode. There are two main modes CD and CS. Within CD, there are additional two modes. The first mode is pure electric and the second mode is “blended”, meaning that the EM and ICE are both providing propulsion power but the ESS is being depleted. If the torque demand is less than or equal to the amount of torque available from the EM, then the vehicle will operate run in pure electric mode. If the torque demand is greater than the allowed EM torque, then the vehicle will enter blended mode. In this mode, the ICE will turn on and provide the supplement torque needed to match the torque request of the driver. A function call-out is used in each mode to limit the amount of computation needed for each process cycle. In pure electric mode, the available EM torque calculated from the torque calculation component of the VSC is passed through as the overall torque request. In blended mode, the equation used to determine the ICE torque is shown in equation 3.3 the torque output from the ICE is limited to the max torque available from the ICE. This determines the maximum torque available of the vehicle during CD mode.

$$T_{Eng} = (T_{dmd} - T_{Mot}) \in (T_{Eng\_\max}(\omega)) \tag{3.3}$$

8 shows the flow of the rule-based (RB) decision gateway to determine in what the mode the vehicle will operate.

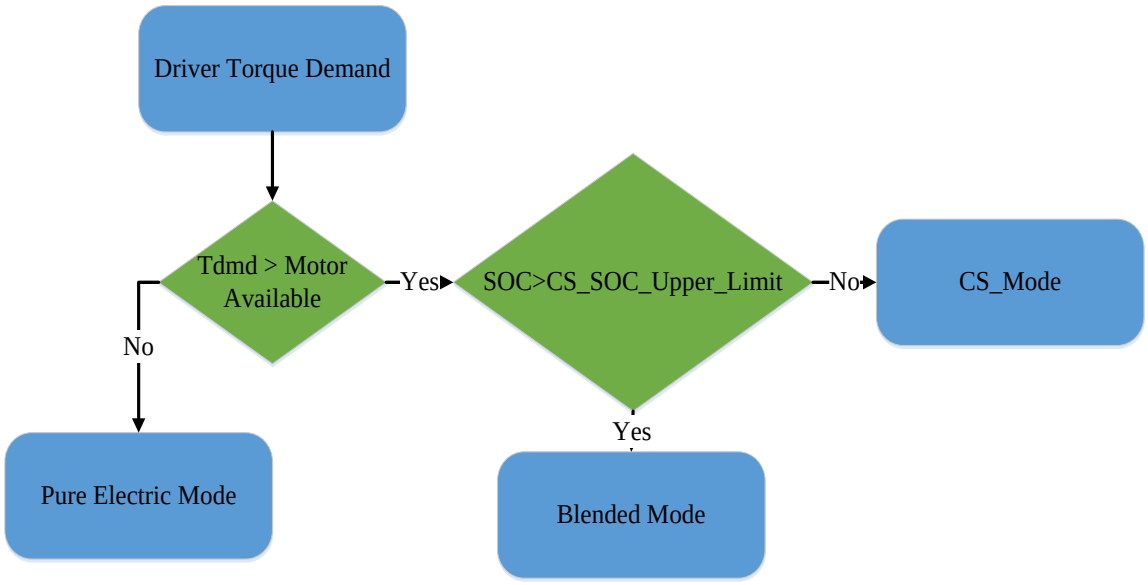

Figure *8.* CD Flow Chart

The CS_SOC_Upper_Limit value to determine if the VSC goes into CS mode is 0.20% SOC.

4.3 VSC CS MODE SELECTION

The decision making process for the CS mode is similar to CD. However, in CS mode, there are four modes to determine the optimal power and efficiency. Each mode is determine using the available SOC and the torque request by the driver. The first mode, is similar to CD. a pure electric mode; this mode is used at low speeds and low torque demands that the ESS can allow. The second mode, is the logic that the VSC will reside in the most when the vehicle is in CS mode during drives; called Motor Assist mode. In this mode, the ICE is the main torque provider and the EM provide the supplement torque needed. Similar to the process described in Chapter 2. the ICE is set to an ideal torque value at each given speed. The EM will then provide additional torque based on driver demand

or act as a generator to keep the ICE in its most optimal torque range, according to the following equations:

$$T_{Eng\_Opt} = T_{Eng} \in .22 < T_{Eff} < .30 \tag{3.4}$$

$$T_{Mot} = T_{dmd} - T_{Eng\_Opt} \in -T_{Mot\_Max}, T_{Mot\_Max} \tag{3.5}$$

The third mode, is activated when the EM has reached its torque limit and the driver has requested additional torque, this mode is called Power mode. In this mode, the ICE will be outside of its optimal torque region and provide additional torque, according to the following equations:

$$T_{Eng} = T_{Eng\_Opt} + (T_{dmd} - T_{Mot}) \tag{3.6}$$

$$T_{Mot} = T_{Mot\_Max} \tag{3.7}$$

The final mode, the regenerative mode, is when the SOC of the ESS is at its lowest allowable point and the ESS needs to be recharged. The ICE is set to maximum torque available and the EM will provide enough regenerative torque that will allow the net torque to be the torque demand. The ICE is set to the maximum value in order to allow the return of energy to the ESS as quickly as possible. If the torque demand is greater than or equal to the ICE torque than the EM will stop charging until the torque demand drops below the ICE torque maximum, according to:

$$T_{Mot} = \begin{cases} -T_{Eng\_Max} + T_{dmd} & T_{dmd} < T_{Eng\_Max} \\ 0 & T_{dmd} > T_{Eng\_Max} \end{cases} \tag{3.8}$$

$$T_{Eng} = T_{Eng\_\max} \tag{3.9}$$

Figure Figure *9* shows the flow chart for CS mode. A check on whether the SOC is within the allowed limits is performed at each processing. The VSC will not exit CS mode unless

the vehicle has returned to 80% SOC on the ESS. The primary process for this to occur is to charge the vehicle using electrical energy from the grid. The VSC will exit regenerative mode once the SOC returns to the CS Target SOC, as shown in Table 4. This table shows the SOC values that are used to determine the vehicle mode.

Table 4: CS SOC Limits

	CS Upper SOC Limit	CS Target SOC	CS Lower SOC Limit
ESS SOC	20%	16%	12.3%

The range was selected to allow the vehicle to run in CD mode as long as possible. It also allows the ESS to provide the majority of the power in the CS mode. If the SOC drops to below the vehicle will not be able to provide the power needed to safely operate the vehicle and will cause the voltage to drop below a safe level if the current draw is to large.

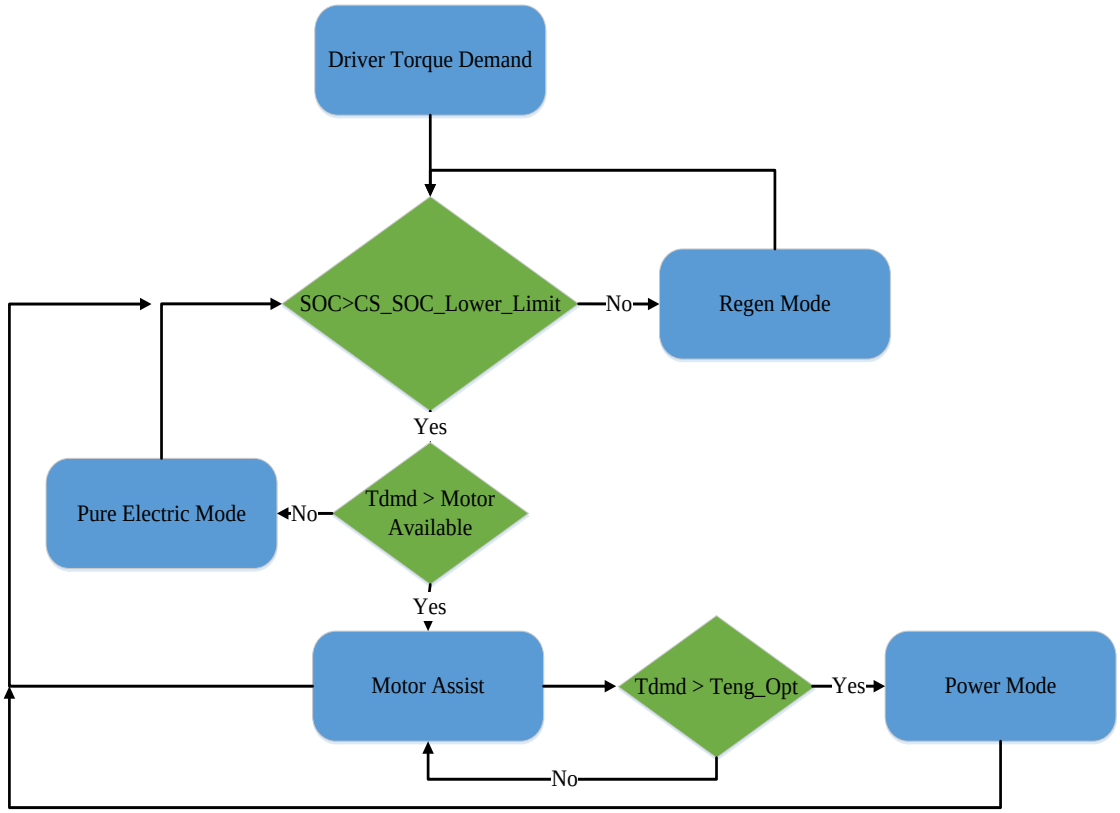

Figure 9. CS Mode Flow Chart

5. SQP IMPLEMENTATION

5.1 SQP IN RULE BASED HIERARCHY

The enhancement to the VSC to increase the efficiency of the vehicle included changing the process and calculation for the optimal ICE and EM torque split. Instead of using a pre-determined ideal torque, the amount of torque for each is determined using SQP. As stated in Chapter 2, this process is iterative and uses optimization to determines the minimum of an objective function. The SQP calculation is still implemented in the RB decision matrix in the second mode of the VSC. This calculation replaced the Motor Assist and Power modes because it could handle torque request greater than the ICE ideal torque points. Figure 10 shows the updated torque flow logic that the rule based logic uses.

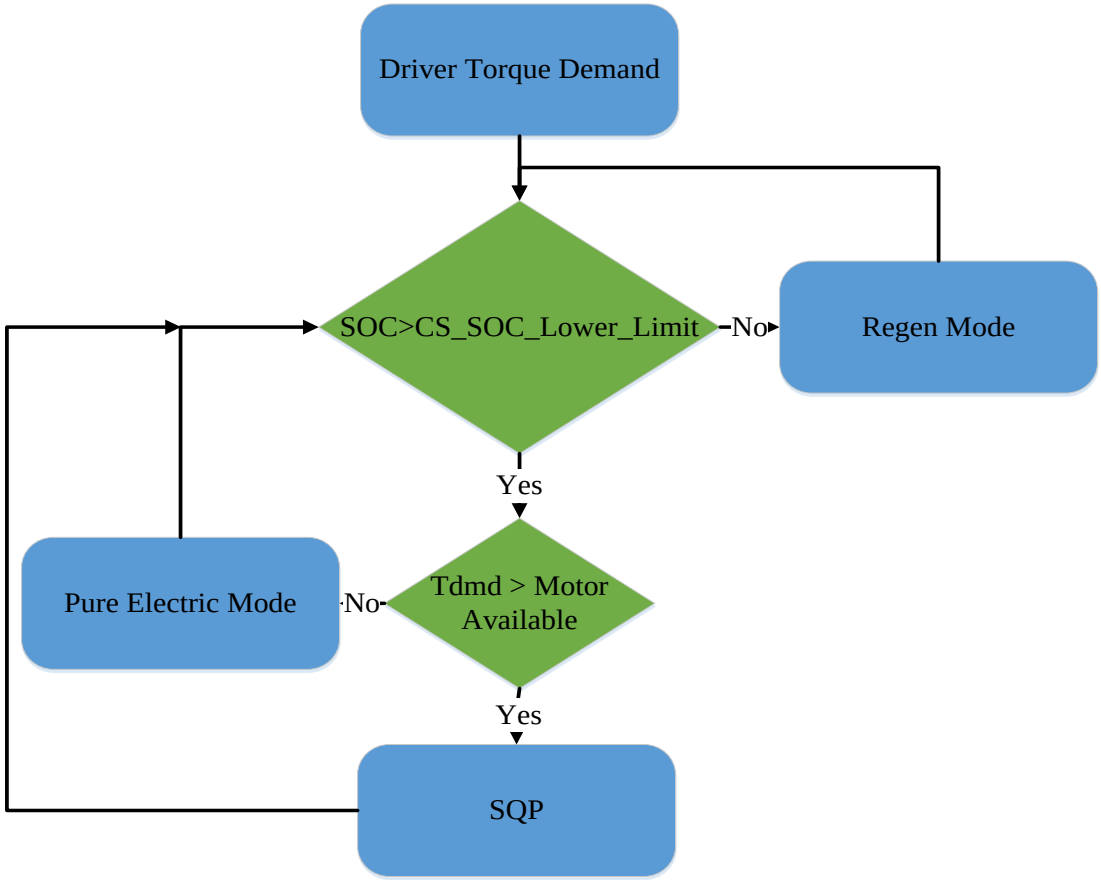

Figure 10: SQP Flow Logic

5.2 EFFICIENCY MAPS

The SQP algorithm uses the efficiency maps of both the ICE and EM to solve the global minimum of the objective function between the components. Figure *11* is the ICE efficiency map. This efficiency map was not originally provided by the manufactory or readily available through reliable sources. The efficiency map was calculated using secondary information, namely fuel flow maps and torque output maps developed through laboratory testing of the actual EcoCAR vehicle ICE and is rough estimation of the

expected ICE efficiency. The calculation used to create the ICE efficiency map are shown the following equations:

$$Eng_{Trq\_Pwr} = Eng_{Torque} * \left(\frac{2 * \pi}{60} \right) \tag{3.10}$$

$$Eng_{Fuel\_Pwr} = H_{LVH} * Eng_{Fuel\_Flow} \tag{3.11}$$

$$Eng_{Eff} = \left(\frac{Eng_{Trq\_Pwr}}{Eng_{Fuel\_Pwr}} \right) \tag{3.12}$$

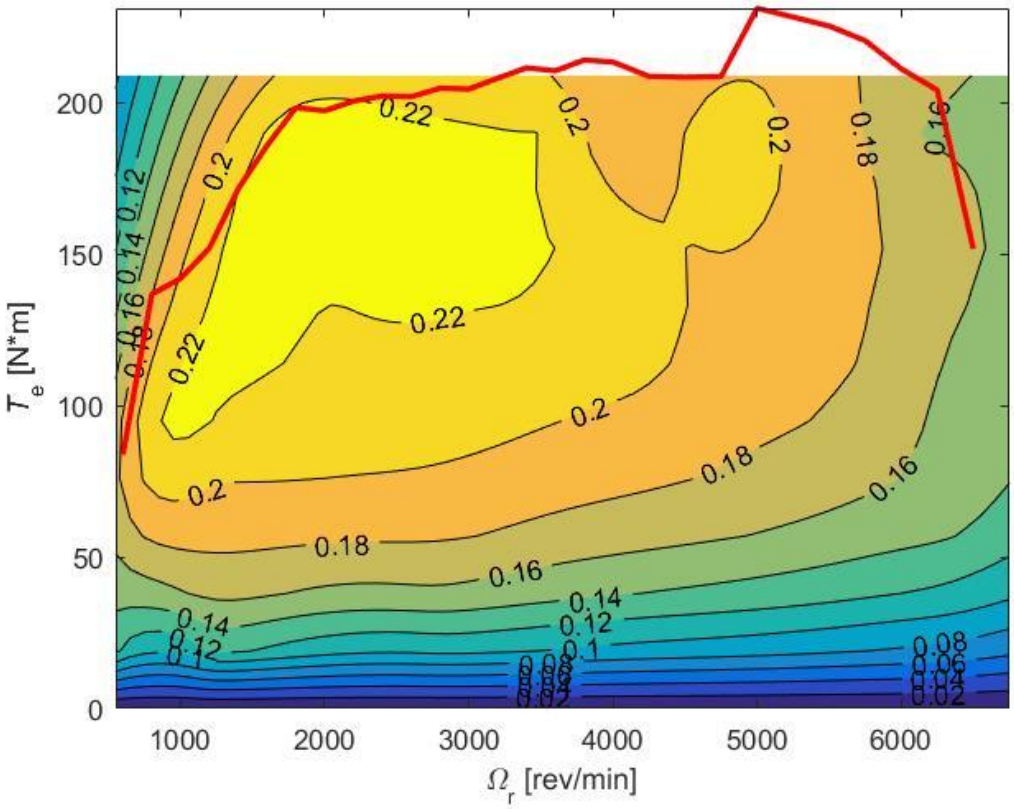

Figure 11. Engine Efficiency Map

The maximum torque curve is then plotted on top of the ICE efficiency map to indicate the max torque the ICE could achieve at any given rpm. There is a gap between the maximum torque and the efficiency map. This is due to the data available from the fuel flow maps

and the torque output maps. The data available from both were not the same sized matrices and therefore some of the data from the torque outputs had to be omitted in order to calculate the best efficiency map possible.

The EM efficiency map was provided by the manufacture, however only the positive torque efficiency values were available. The EM is able to provided positive and negative torque. In order to complete the efficiency map, the torque values were extended to show the maximum negative torque and the efficiency was than inverted and added to the original map. The results are shown in Figure 12.

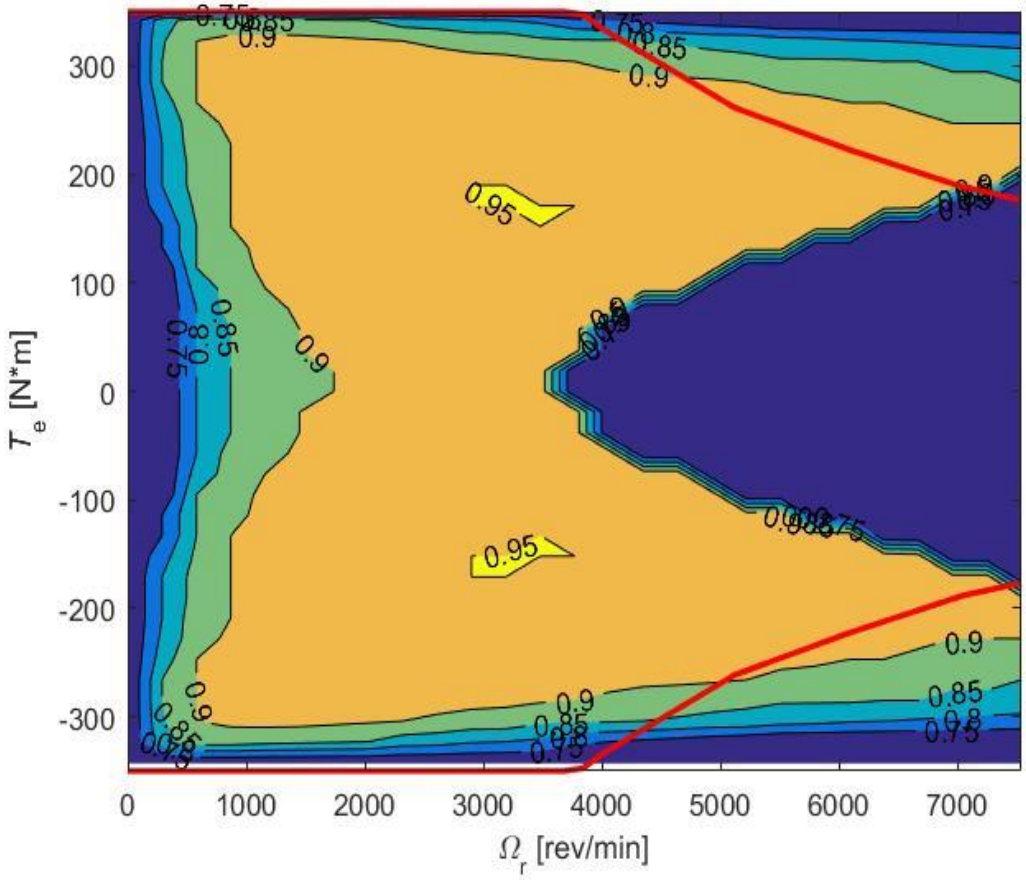

Figure 12: Motor Efficiency Map

The maximum positive and negative torque is over laid on the efficiency map to indicate the max torque available at each rpm. Comparing the EM efficiency map to other EM efficiency maps the negative torque efficiency usually has some slight variation in efficiency points. However, because the deviations are so small that inverting the positive torque efficiency points to represent the negative efficiency would suffice for the problem.

5.3 OBJECTIVE AND CONSTRAINTS EQUATIONS

In order to calculate the optimal point of the efficiency maps a surface curve was created and calculated using the surface curve fit tool from Matlab. This creates an equation that represents the efficiency maps that are later used in the objective function for the SQP algorithm. The ICE the surface curve fit is shown in Figure 13

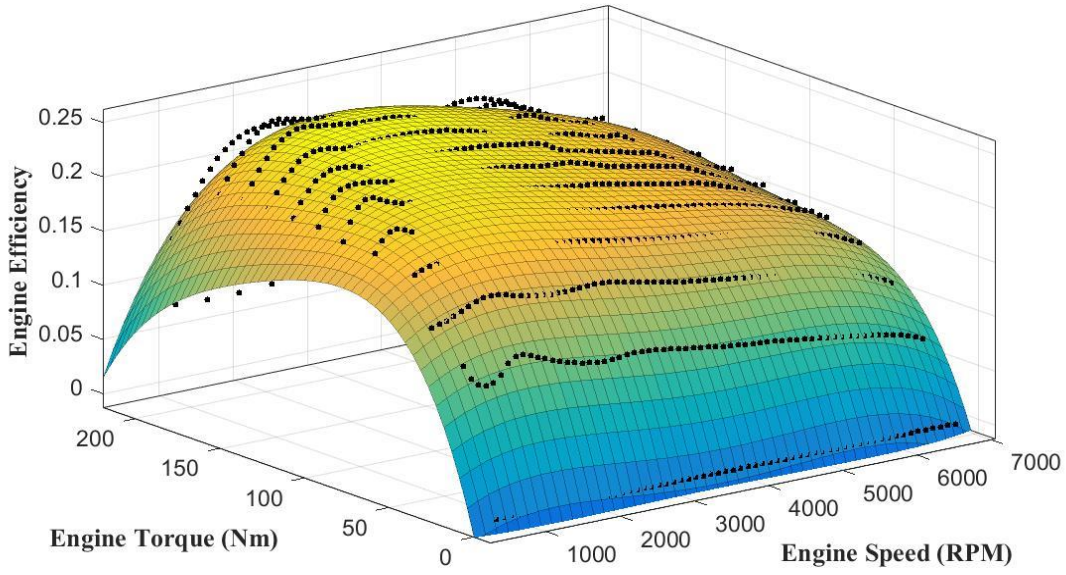

Figure 13: Surface Curve Fit for Engine Efficiency

The ICE efficiency equation generated is a 4^{th} order polynomial equation in the x and y direction:

$$\eta_{Eng} = P00 + (P10 * Eng\_S) + (P01 * X1) + (P20 * Eng\_S^2) + (P11 * Eng\_S * X1) + (P02 * X1^2)$$
$$+(P30 * Eng\_S^3) + (P21 * Eng\_S^2 * X1)$$
$$+(P12 * Eng\_S * X1^2) + (P03 * X1^3) \quad (4.1)$$
$$+(P40 * Eng\_S^4) + (P31 * Eng\_S^3 * X1)$$
$$+(P22 * Eng\_S^2 * X1^2) + (P13 * Eng\_S * X1^3)$$
$$+(P04 * X1^4)$$

The constants from the equation 4.1 for the ICE are shown in Table 5.

Table 5: Engine Constant Values

Constants for Engine Efficiency Equation	Values
P00	-0.01768
P10	4.81e-05
P01	0.006152
P20	-2.866e-08
P11	3.867e-07
P02	-7.454e-05
P30	6.22e12
P21	-1.551e-10
P12	2.061e-09
P03	3.653e-07
P40	-4.491e-16
P31	1.177e-14
P22	-8.463e-14
P13	-1.817e-12
P04	-7.057e-10

The curve fit is able to access most of the surface points; however, some of the higher-efficiency points, especially at low ICE speeds, are not accessible with the curve fit and make it difficult for the optimization to achieve the optimal efficiency point. A better curve fit is needed. The idea of dividing the map into multiple parts was developed. This allows the curve fit to the design equation that could better fit the efficiency map, without increasing the overall computation needed. Therefore, the ICE efficiency map was broken up into four different sections based on the rpm range. The first section is from 0 to 2000rpm, the second section is from 2000 to 3600rpm, the third section is from 3600 to 5200rpm, and the fourth section is from 5200 to 6800rpm. The other advantage of dividing

the efficiency map into multiple regions is that the equations created to determine the surface curve fit were lower-order polynomials than the previous surface curve fit. This also allowed the equation and the optimization strategy to achieve the higher efficiency points that were not accessible from the original curve fit. Figures 14-17 are the surface curve fits for the new regions of the ICE efficiency map. The equations and constant value can be seen in Appendix C.

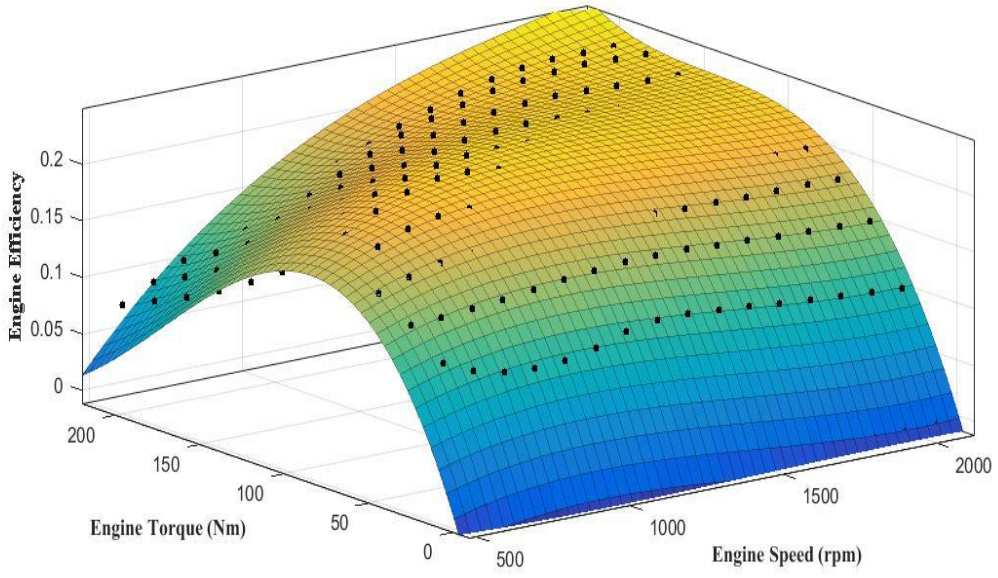

Figure 14: Engine Efficiency Region 1

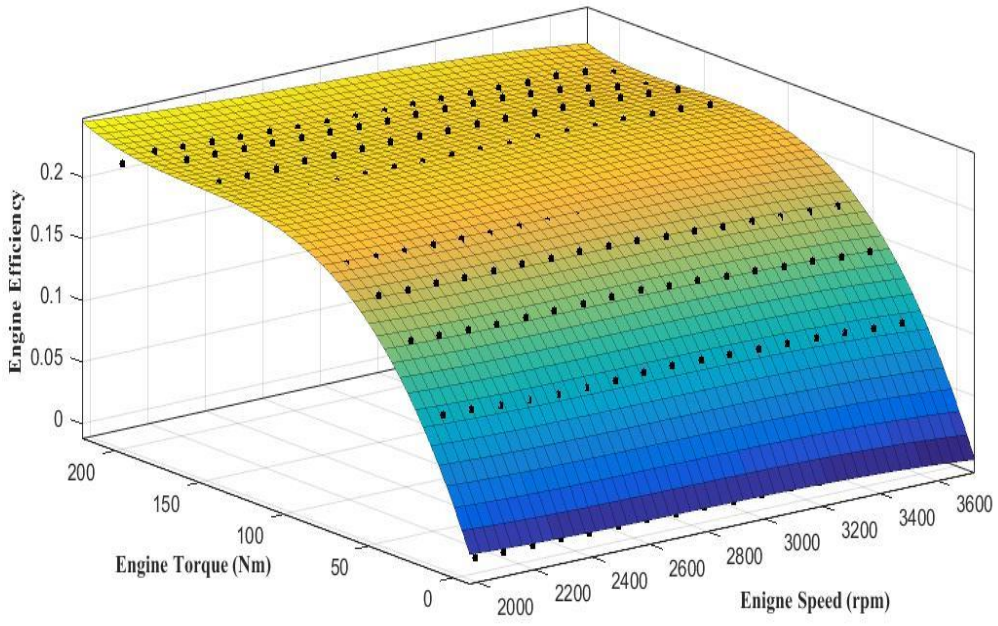

Figure 15: Engine Efficiency Region 2

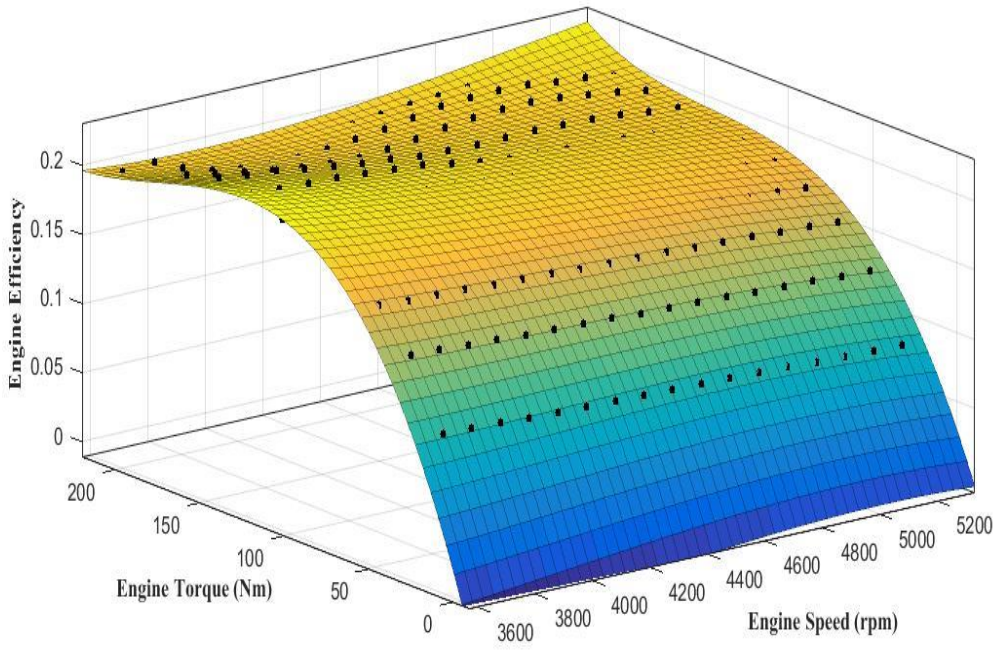

Figure 16: Engine Efficiency Region 3

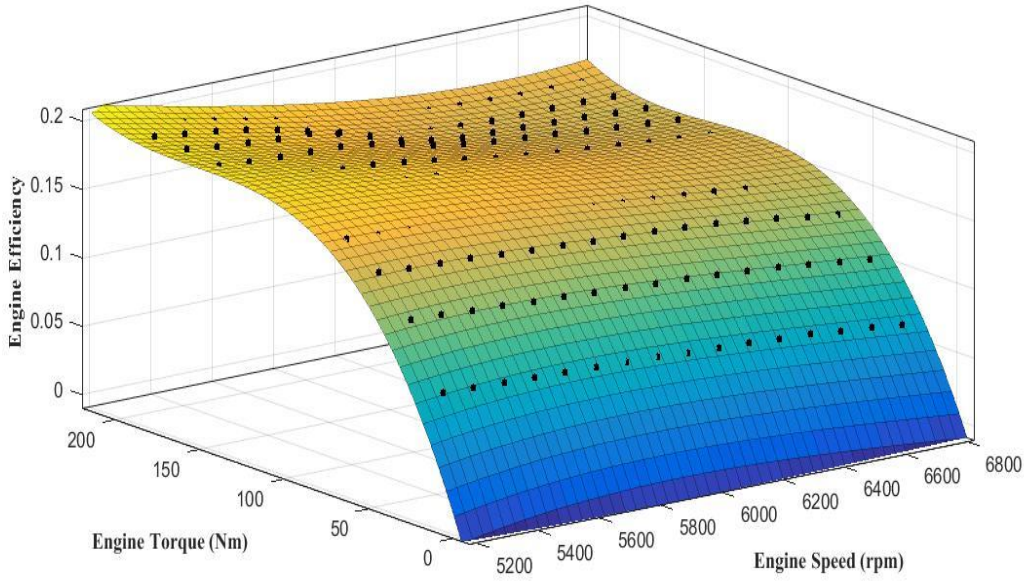

Figure 17: Engine Efficiency Region 4

The EM efficiency is shown in Figure 18.

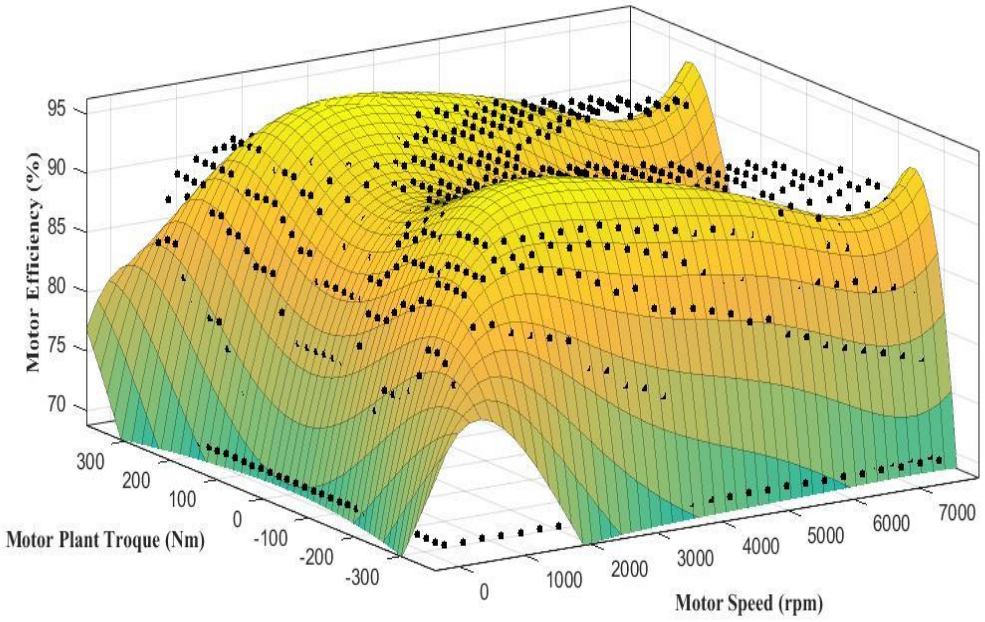

Figure 18: Surface Curve Fit for Motor Efficiency

The EM efficiency equation is similar to the ICE curve fit however, the equation is a 5th order polynomial in the x and y direction.

$$\begin{aligned}\eta_{Mot} &= P00+(P10*Mot\_S)+(P01*X2)+(P20*Mot\_S^2)+(P11*Mot\_S*X2)\\ &+(P02*(X2^2))+(P30*(Mot\_S^3))+(P21*Mot\_S^2*X2)+(P12*Mot\_S*X2^2)\\ &+(P03*X2^3)+(P40*Mot\_S^4)+(P31*Mot\_S^3*X2)+(P22*Mot\_S^2*X2^2)\\ &+(P13*Mot\_S*X2^3)+(P04*X2^4)+(P50*Mot\_S^5)+(P41*Mot\_S^4*X2)\\ &+(P32*Mot\_S^3*X2^2)+(P23*Mot\_S^2*X2^3)+(P14*Mot\_S*X2^4)+(P05*X2^5)\end{aligned} \quad (4.2)$$

The constants for the EM are shown in Table 6.

Table 6: Motor Efficiency Constants

Constant for Motor Efficiency Equation	Values
P00	86.23
P10	-11.52
P01	0.08342
P20	-4.143
P11	0.04229
P02	16.76
P30	0.6895
P21	-0.2942
P12	12.3
P03	0.323
P40	-1.357
P31	0.04853
P22	2.719
P13	-0.1041
P04	-7.847
P50	1.212
P41	0.03814
P32	-1.255
P23	0.1614
P14	-3.832
P05	-0.2013

The EM curve fit is similar to the ICE cure fit in that some areas of the curve fit do not reach some of the higher efficiency points. However, the multi-region approach did not alleviate the accessibility problem due to the large peaks in both positive torque and negative torque regions. Instead, the negative torque values were removed from the

efficiency map. This approach worked because the positive and negative efficiency values are mirrors of each other. Also, the algorithms did not have any change in value when the negative torque values were removed. Figure 19 shows the new surface curve fit for the positive torque values. The new constant values and equations are listed in Appendix D.

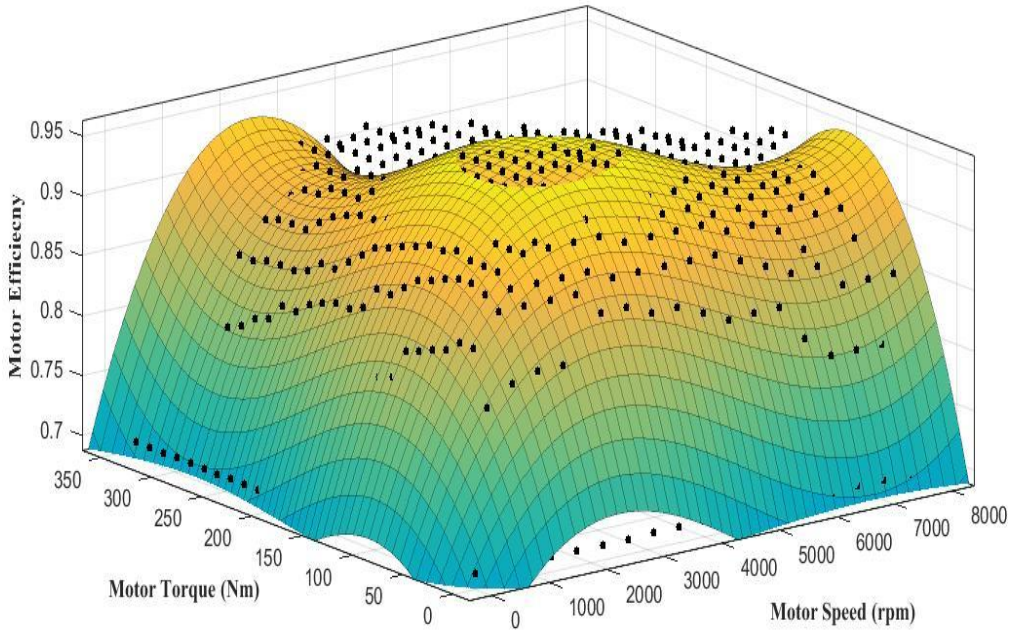

Figure 19: Motor Efficency Map Postive Torque

For all equations 4.1 and 4.2 the X1 and X2 values represent the torque values that need to be optimized. The *Mot_S* and *Eng_S* are the EM speed and EM speed in rpm.

The overall objective function becomes used in the SQP algorithm.

$$F_{obj} = (W_1 * (1 - \eta_{Eng}) + (W_2 * (1 - \eta_{Mot}) \tag{4.3}$$

Where W_1 and W_2 are the weights that are calculated from a separate equation using the ECMS function. The function utilizes the efficiency maps of the ICE and EM and minimized the efficiency losses of both components. The weighting will than adjust the

torque values and make the SQP algorithm put more torque to either the EM or ICE based on the weightings.

The behavior of ECMS determines how much the EM should be used to increase the efficiency of the system based on ECMS characteristics. However, because electrical energy and chemical energy are not the same, the two sources need to be changed to a common unit that could properly address the energy difference. The unit of energy is converted to dollar amount of the cost of each energy. As of this writing, E10 fuel is roughly $2.57/gallon nationally, and the cost of electrical grid energy is $0.10/kWh. The equation becomes

$$J = ((\dot{m}_{Eng} * \eta_{Eng}) * 2.57) + \left(\begin{array}{l} \gamma * S_{dis} * \left(\frac{1}{(\eta_{EM} * P_{EM} *)^2} \right) * \left(\frac{P_{EM}}{H_{LVH}} \right) \\ +(1-\gamma) * (S_{chg} * (\eta_{EM} * P_{EM})^2) * \left(\frac{P_{EM}}{H_{LVH}} \right) \end{array} \right) * 0.10 \quad (4.4)$$

Where $\dot{m}_{Eng}$ is the fuel flow rate of the ICE, η_{Eng} is the ICE efficiency, η_{Em} is the EM efficiency, P_{Em} is the power of the EM, and H_{LVH} is the lower heating value of fuel. For the EM the weight will be *J* value the ICE will be (1-J). The variables $\dot{m}_{Eng}$ and η_{Eng} are used to determine the weighting of the ICE. The values determine how much fuel is being used and amount of energy that could go back into the EM. The second half of the equation determines the amount of electrical energy used and determines the weight of the EM based on the parameters and if more or less electrical energy should be used. The weights are then multiplied by 10. This is because the weighting from 0-1 does not have enough effect to the amount of EM torque request produced by the SQP algorithm. To create greater variation to the final torque request from the SQP algorithm the weights were multiplied by 10.

After the objective function is determined, the constraints to the objective needed to be setup. Some of the constraints are based on the physical capabilities of each component and system parameters. There are seven different constraints that restrict the optimal point of the objective equation. The first is that the EM speed and ICE speed must be equal. This is because the ICE and EM are connected with an inline shaft with no gearing to allow for different rotation speed:

$$\omega_{EM} = \omega_{Eng} \tag{4.5}$$

The next is the total torque demand must be achieved by the combined torque produced by the EM and ICE:

$$T_{dmd} = T_{ENG} + T_{Mot} \tag{4.6}$$

The ESS power available will then determine the amount of torque the EM can provided based on the torque request. That inequality constraint is

$$T_{Mot} < \left(P_{ESS} * (Mot\_S * \frac{2\pi}{60}) \right) \tag{4.7}$$

Where P_{ESS} is the power available from the ESS, *Mot_S* is the speed of the EM in rpm. The ICE torque is constrained such that the ICE can-not provide more torque than its maximum and can-not provide negative torque to the system.

$$0 < T_{Eng} < T_{Eng\_Max} \tag{4.8}$$

The EM torque is the same as the ICE in that it can-not provides more torque than physically allowed, neither positive nor negative torque:

$$-T_{Mot\_Max} < T_{Mot} < T_{Mot\_Max} \tag{4.9}$$

Some of the research done with SQP the SOC is part of the constraints of the SQP algorithm. However, this usually increase the computation time and the termination values have to be adjusted to match both discharge and charging events. The work around was to make the SOC controlled by a separate algorithm; the ruble based algorithm. This allowed the SQP algorithm focus on providing the most efficiency torque value between the ICE and EM.

5.4 FMINCON VALIDATION

In order to validate the objective function and determine that the values coming from the equations and the constraints are being applied properly, a built in Matlab function known as “fmincon” was used. The fmincon function uses an objective function and constraints provided by the user to find the minimum value of that objective function. The function was built to be used in Matlab scripts and coding. The results from fmincon calculations can be seen in Appendix A. The code used with fmincon is available in Appendix B.

The fmincon function has a few issues for the other major part of these thesis which was the inability to run on hardware in real time. One of the function’s issues is it does not use any advanced equations to quickly find the optimal point. Each calculated result takes approximately 0.6 seconds of computation time for every computation cycle. The other issue is that fmincon is not supported in Simulink. This is a major issue because the hardware that was used for the testing only uses Simulink to create the C code that is loaded on the hardware.

In order to validate the fmincon function with in the vehicle model and achieve the results that are in appendix A, additional code was need to execute fmincon as an outside script that the variables get passed to and then are returned back to the model. The command to

do this was code.extrinsic, this command allowed fmincon to be run inside of the Simulink environment but drastically increased the simulation time. For a single 600-second cycle, it took approximately 6 hours to finish. The stock rule-based computation time using Simscape and Simulink environments takes approximately 25 minutes. Despite the long computation time, the fmincon function proved that the optimization could be solved with in the rule based controller and that values calculated are meeting the constraints.

5.5 SEQUENTIAL DYNAMIC PROGRAMMING ALGORITHM

In order to solve the problem of computation time, be able to be used in native Simulink without using extra code to run the model, and be able to be built to the hardware available, a new algorithm method was adopted. The best option was using SQP. It used advanced algebraic methods to solve the optimization problem quickly while keeping the same objective functions and constraints.

The SQP employs the Lagrange-Newton Equation to solve for the quadratic problem (QP) the subset of equation that find the minimal value is:

$$x_{k+1} = x_k - \partial_k H_k^{-1} g_k \tag{4.10}$$

where $H_k$and $g_k$are the Hessian and gradient at iteration k.

While solving the Lagrange-Newton equation, the equations are solved using Newton's method to update x and λ, and this is done by using the Taylor expansion to first order

$$[\nabla L(x_k + \partial x_k, \lambda_k + \partial \lambda_k)]^T = \nabla L_k^T + \nabla^2 L_k (\partial x_k, \partial \lambda_k)^T \tag{4.11}$$

and setting the left-hand side $\nabla L_{k+1}^T = 0$

$$\nabla^2 L_k \begin{pmatrix} \partial x_k \\ \partial \lambda_k \end{pmatrix} = -\nabla L_k^T \tag{4.12}$$

Next, the matrix to represent the Tyler expansion is computed:

$$\nabla^2 L_k = \begin{pmatrix} \nabla^2{}_x L & \nabla^2{}_{x\lambda} L \\ \nabla^2_{\lambda x} L & \nabla^2_{\lambda} L \end{pmatrix}_k = \begin{pmatrix} \nabla^2 f + \lambda^T \nabla^2 h & \nabla h^T \\ \nabla h & 0 \end{pmatrix}_k \tag{4.13}$$

Substituting values $W \triangleq \nabla^2 f + \lambda^T \nabla^2 h$ and $A \triangleq \nabla h$ the resulting equation becomes

$$\begin{pmatrix} W_k & A_k^T \\ A_k & 0 \end{pmatrix} \begin{pmatrix} S_k \\ \lambda_{k+1} \end{pmatrix} = \begin{pmatrix} -\nabla f_k^T \\ -h_k \end{pmatrix} \tag{4.14}$$

Solving for the above equation using iterations $x_{k+1} = x_k + s_k$ and λ_{k+1} eventually approached the optimal solutions for x and λ; x is the solution and λ is the LaGrange multiplier.

The algorithm then employs a sub-problem strategy with in solving for the Lagrange-Newton equation called an active-set strategy:

$$f(x_k + \alpha_k s_k) < f(x_k) \tag{4.15}$$

Where x_k is the solution if a_k, is the step within a region. If a_k, violates the constraints the algorithm adds the constraint that is violated to the active set and reduces a_k to the maximum feasible point. This strategy takes into account the constraints, removes any constraints that are not currently affecting the results of the solution, and then re-calculates the solution with constraints that are in the area of the optimal solution.

Once the first iteration of the QP problem is solved the algorithm preforms an active line search. The active line search is a merit function using a penalty function.

$$\emptyset(x, \lambda, \mu) = f(x) + \sum_{j=1}^{m_1} W_j \left|h_j\right| + \sum_{j=1}^{m_2} W_j \left|min\{0, -g\}\right| \tag{4.16}$$

The variables m_1 and m_2 are the number of equality and inequality constraints and w_j are the weights used to balance the infeasibilities. The process of using the variable w_j can be as the following

$$\begin{aligned} w_j &= |\lambda| \quad for \;\; k = 0 \\ w_{j,k} &= \max\{|\lambda_{j,k}|, 0.5(w_{j,k-1} + |\lambda_{j,k}|)\} \quad for \;\; k \geq 1 \end{aligned} \tag{4.17}$$

where μ_j would be used for the inequalities (j=1,….,m_2). The process is as follows: (1) solve the QP sub-problem to determine a search direction s_k (2) minimize the merit function along s_k to determine a step length a_k (3) set $x_{k+1} = x_k + a_k s_k$, (4) repeat process until the termination criteria are met.

The algorithm then updates the Hessian matrix with new active line search criteria using the Broyden-Fletcher-Goldfarb-Shanno algorithm (BFGS):

$$H_{k+1} = H_k + \left[\frac{\frac{\partial y}{\partial x} g \frac{\partial y}{\partial x} g^T}{\frac{\partial y}{\partial x} g^T \frac{\partial y}{\partial x} x}\right]_k - \left[\frac{\left(H\frac{\partial y}{\partial x} g\right)\left(H\frac{\partial y}{\partial x} g\right)^T}{\frac{\partial y}{\partial x} x^T H \frac{\partial y}{\partial x} x}\right]_k \tag{4.18}$$

where H_k is the current Hessian matrix, x_n is the current solution form the QP problem, and g_n is the partial derivative of the constraints for the updated active set problem. Another iteration of the QP algorithm is then performed to solve for the optimal solution once more. Once the termination criterion are met, the optimal solution has been achieved. (Wilde, 2000, p.315-350)

The code used in the model to collect results is shown in Appendix F and the results from testing the SQP for rotational speed and torque range are shown in Appendix E compared to RB torque and efficiency responses. With the SQP algorithm the torque split calculation is computed faster than with the Fmincon function. The SQP algorithm, with its advanced

algorithm tracking is able to find a solution in 0.06 seconds. This should allow the algorithm to be calculated in real world hardware.

6. RESULTS FROM SIMULATION AND HARDWARE TESTS

The simulation results from the RB and SQP controller methods compared for fuel economy (in units of mpg and mpgge (miles per gallong of gasoline equivalent), SOC and energy used. The mode each controller is in throughout the drive cycles is also shown, in order to illustrate how the different methods, affect the prevalence of the modes. The controller methods are compared using four different drive cycles to gain confidence that the methods are applicable to a wide variety of driving patterns. The first will consist of a simple acceleration, a short distance at constant speed, followed by an equal deceleration. The other three drive cycles will be based on EPA drive cycles: (1) The 505 drive cycle that is a simple urban drive cycle that has roughly a max speed of 22 mph. This cycle is the first 505 seconds of the FTP-75 cycle from the EPA. (2) the HWYFET cycle that simulates a highway drive with approximately 62 mph max speed and; (3) the US06 drive cycle that presents hard acceleration and aggressive driving, with maximum speed is roughly 75 mpg.

6.1 SIMPLE DRIVE CYCLE RESULTS

The testing began with the simple drive cycle that tested both controllers under acceleration, constant speed and braking dynamics. Figure 20 shows the drive cycle with the velocity trace for RB and SQP controllers. The RB has a little over-shoot when the vehicle stops accelerating and is slow to respond once the braking occurs. However, it still falls within the industry norms of 2% error. The SQP velocity also displays an overshoot. But is slightly over damped still within in the 2% error of industry norms. Neither approaches performed better when it came to following the trace.

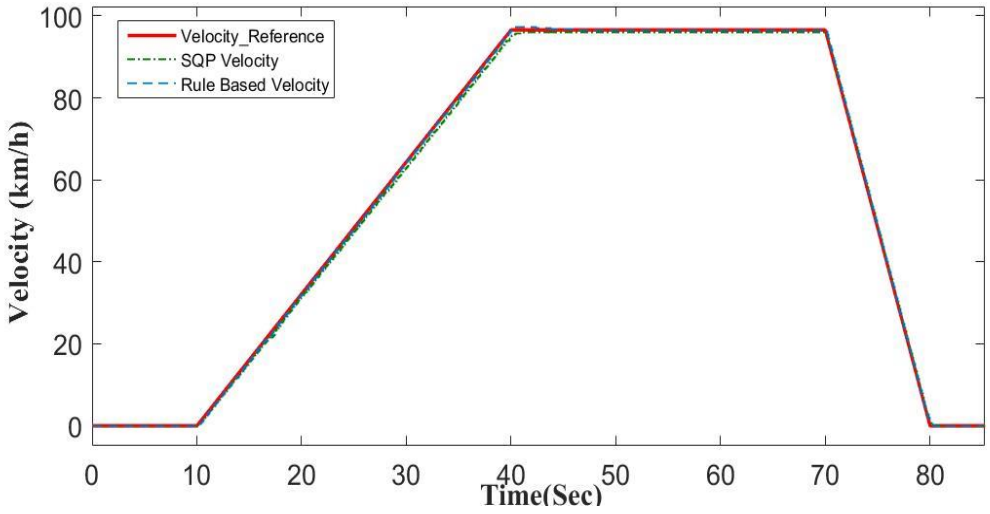

Figure 20: Simple Drive Cycle

Figure 21 shows the torque request from the accelerator pedal and the torque response using the SQP algorithm. It shows some points the ICE is disengaged and the EM handles the torque request. The spikes are from the transmission down shifting. In Figure 22 the RB controller torque response is shown. It can be seen that the ICE provides a particular torque and the EM adjusts to accomplish the torque request. The RB control strategy and the torque request calculation keeps the ICE on during low torque request and makes the EM regenerative mode. The results from both the control strategy are shown in Table 7.

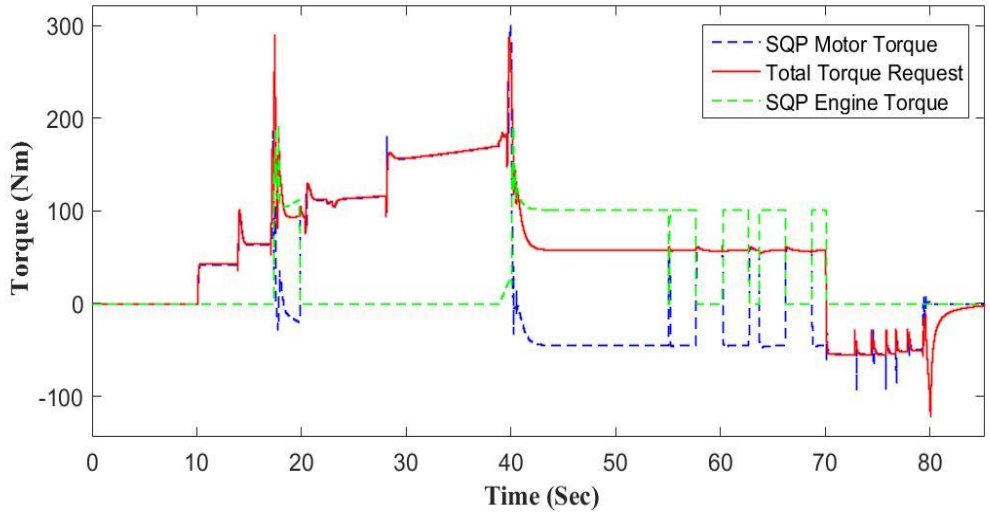

Figure 21: SQP Torque from Simple Drive Cycle

Table 7: Simple Drive Cycle Results

	Fuel Economy (mpg)	Fuel Economy (mpgge)	Final SOC	Fuel Used (Gal)	ESS Net Energy (kWh)	Total Energy (kWh)
RB	22.24	22.61	16.26	0.03748	-0.0196	0.8273
SQP	49.67	37.75	13.24	0.01667	0.17	0.5466

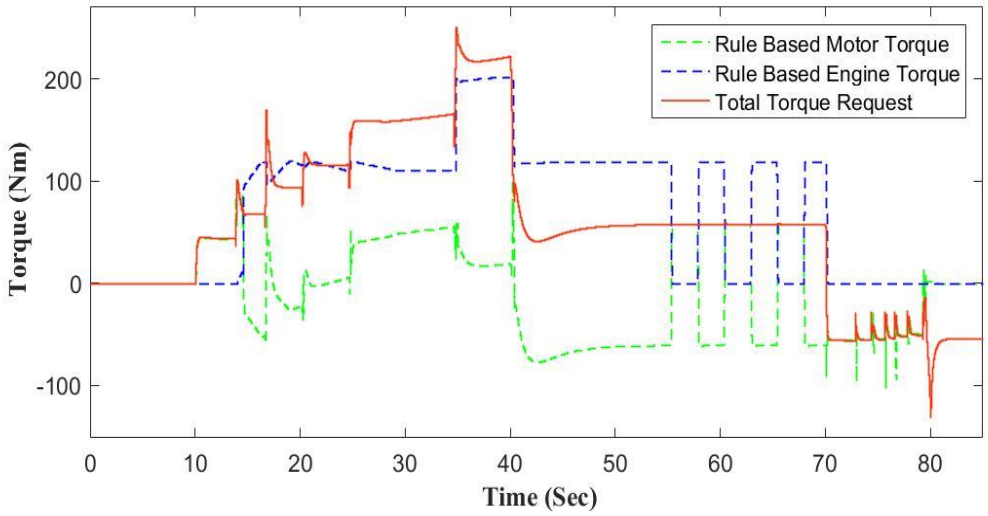

Figure 22: RB Torque from Simple Drive Cycle

The results show that for the case of the simple cycle simulation that the SQP algorithm caused more ESS energy to be used (and thus saw a larger change in SOC) than the RB method. Figure 23 shows the results for the SOC on the simple drive cycle. SQP utilized

the EM more to keep the ICE in its optimal torque range longer. This makes the controller force recharge of the ESS when the SOC drops a to lower level.

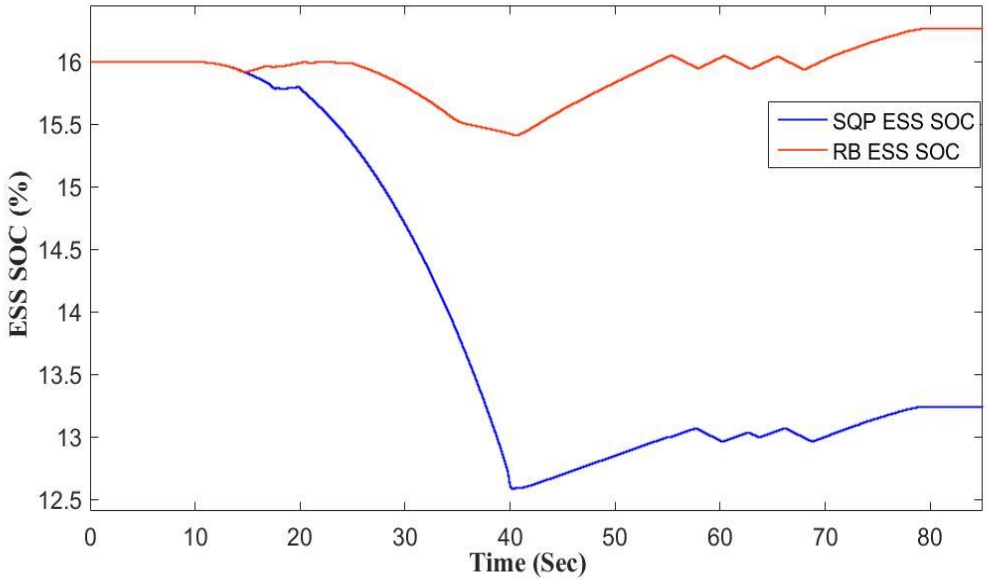

Figure 23: SQP and RB ESS SOC

6.2 US06 DRIVE CYCLE RESULTS

The next drive cycles tested was the US06. Figure 24 shows the complete drive cycle of the US06 cycle with both the RB and SQP velocity traces. The two controller's methods were able to handle the aggressive accelerations and decelerations of the drive cycle.

During the hard peaks the controllers handle torque requests as high as 450 Nm during high speed vehicle maneuvers.

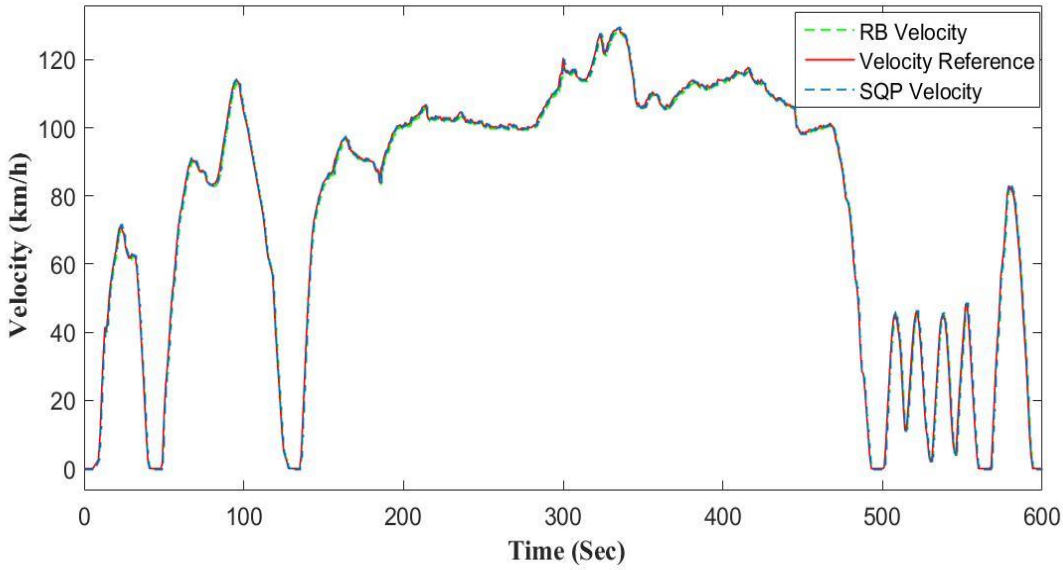

Figure 24: US06 Drive Cycle

However, a closer examination of a select portion of the drive trace of Figure 24, shown in Figure 25, illustrates that the SQP method velocity

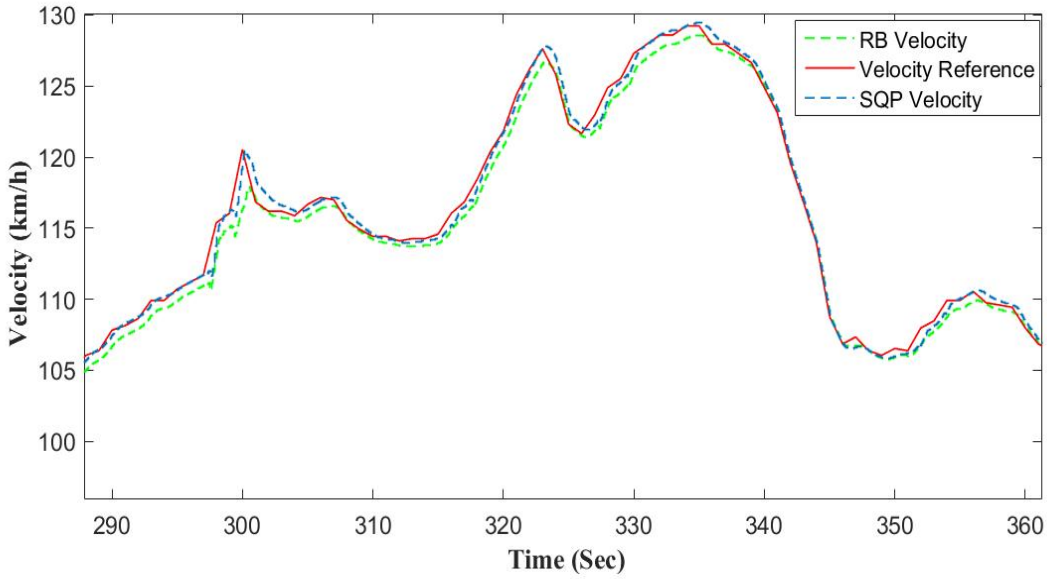

Figure 25: Section of US06 Drive Cycle

trace is able to follow the sudden changes at high speeds better than the RB method. This is because the RB need a stronger rate limiter in order to not let the EM and ICE over-speed. Also, some of the rapid changes are not achieved by either controller method because of the PI controller representing the accelerator positioning cannot adjust to rapid change quickly enough. In Figure 26, the torque request from the SQP algorithm is presented, and it shows that there are rapid changes that occur due to the aggressive nature of the drive cycle.

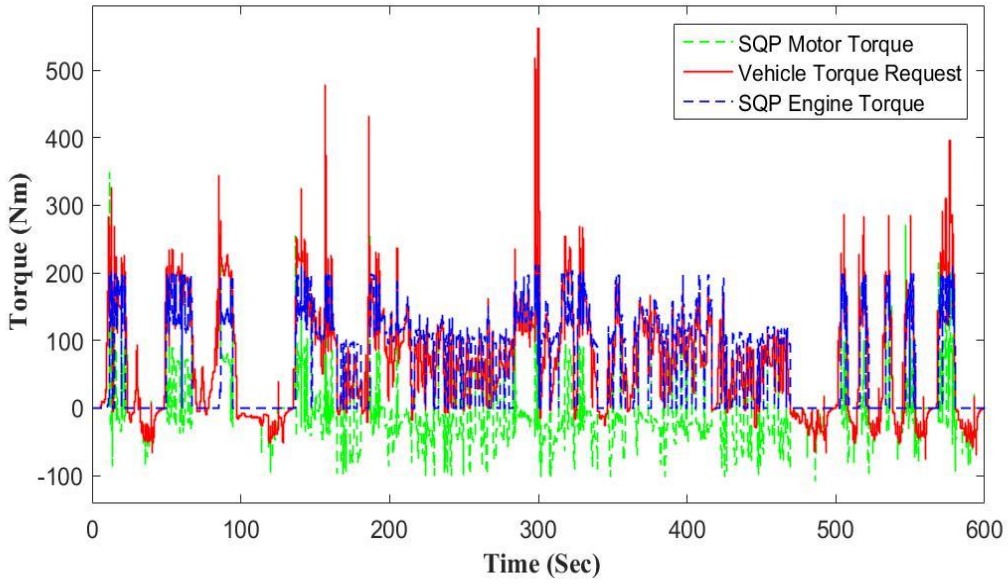

Figure 26: SQP Torque Request US06 Cycle

The large spike in torque request at the 300 second mark is the vehicle down shifting to get the torque up to meet the trace. This cause the PI controller to request max torque due to the rapid down shift and that no torque is passing during a shift. Figure 27 is a closer examination of a portion of the US06 results, and it shows the proper torque split between the ICE and EM to meet demand. At approximately the 11-second mark a large torque

spike is shown by the EM this is due to the ICE clutching onto the drive shaft and causing a torque spike to be seen by the EM.

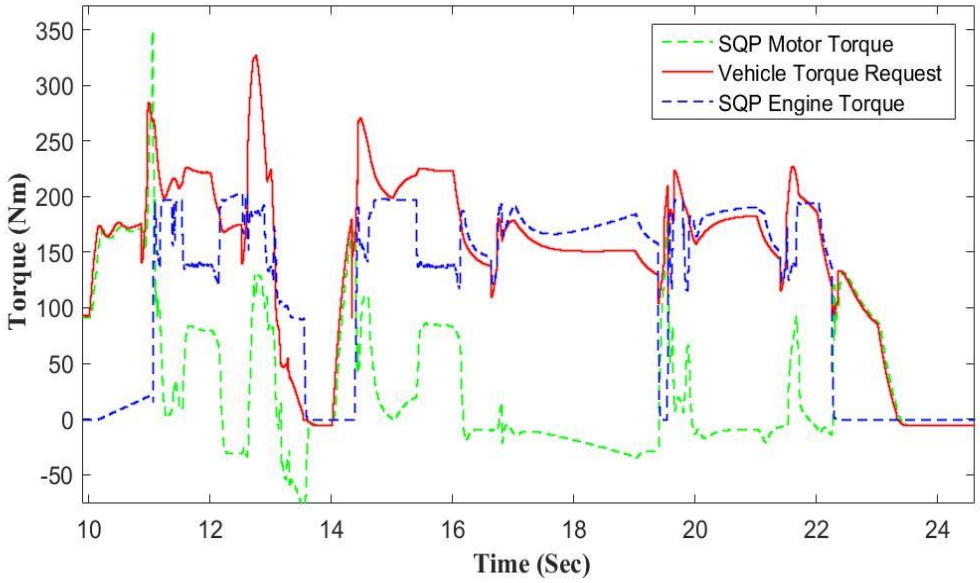

Figure 27: SQP Torque Request Closer Look US06

The RB method's torque request and behavior is seen in Figure 28 the torque behavior that was seen in the simple drive cycle, is also witnessed in the US06 cycle. The ICE holds torque demanded by the controller and the EM adjusts to meet the net torque requested.

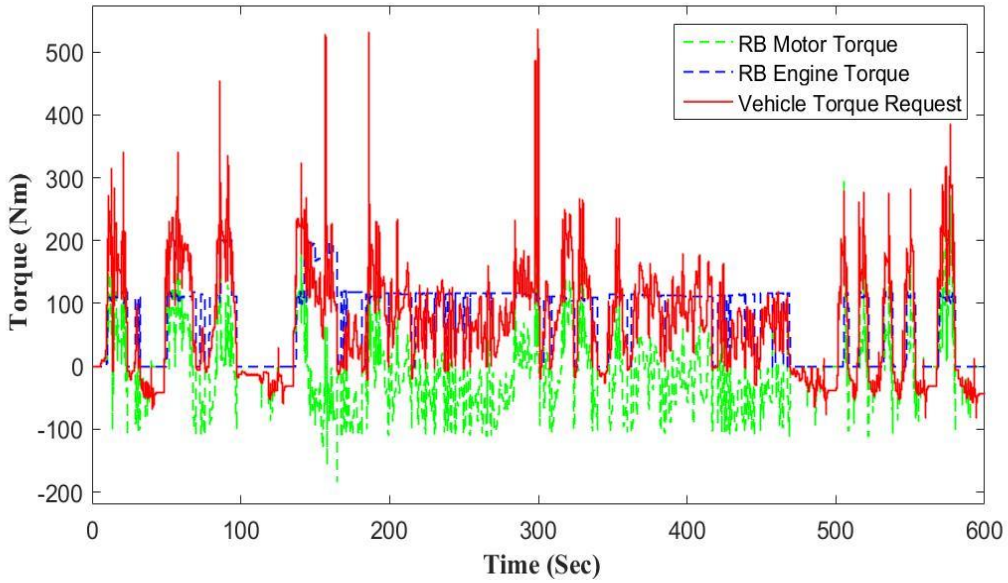

Figure 28: RB Torque Request US06

Figure 29 shows the ESS SOC of both control algorithms through the US06 drive cycle. During 200-300 second mark of the cycle the vehicle is driven at roughly 75 mph. The RB take the opportunity to recharge the ESS due to the low torque demand at high speeds because the ICE is supplying more torque than needed and causing the EM to operate as a generator longer, compared to the SQP method. The SQP method however, is already reaching the lower limit of the available SOC and tries to maintain the SOC above the lower limit but still executing the SQP algorithm. Near the end of the drive cycle, there is a large torque request and this can be noticed in the SOC value. The SOC for the RB

method drops about 2% compared to the SQP value of the SQP method. That because the SQP algorithm reached the lower limit of the CS SOC value.

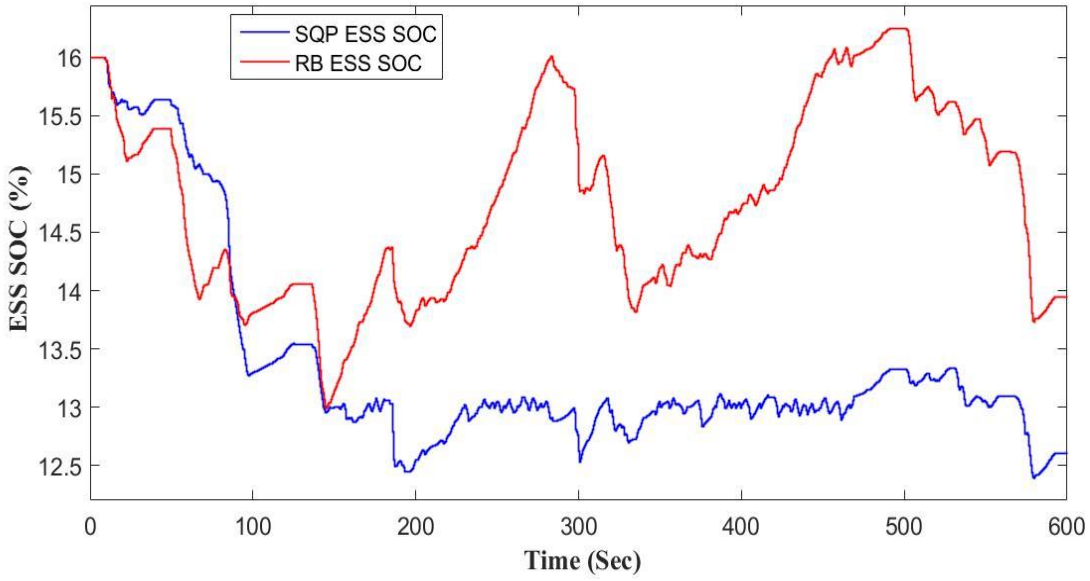

Figure 29: US06 ESS SOC

Table 8 shows the results from the SQP and RB controllers. The RB and SQP methods have different torque splits to handle the high torque demands. The notable details from table 7 are the amount of fuel used between the two methods, there is about 7.67% difference in fuel use as well as, 6.22% difference in total energy used.

Table 8: US06 Drive Cycle Results

	MPG	MPGE	SOC	Fuel Used (Gal)	ESS Net Energy (kWh)	Total Energy (kWh)
RB	26.73	26.47	13.94	0.299	0.093	6.863
SQP	28.77	28.17	12.6	0.2769	0.192	6.449

6.3 HWYFET DRIVE CYCLE RESULTS

The next drive cycle is the HWFET. This drive test how the controllers handle torque demands when the vehicle is already driving at high speeds. Figure 30 shows the drive

cycle compared with the SQP and RB methods' velocity trace. The SQP velocity trace is not readily seen due to overlapping with the original vehicle trace.

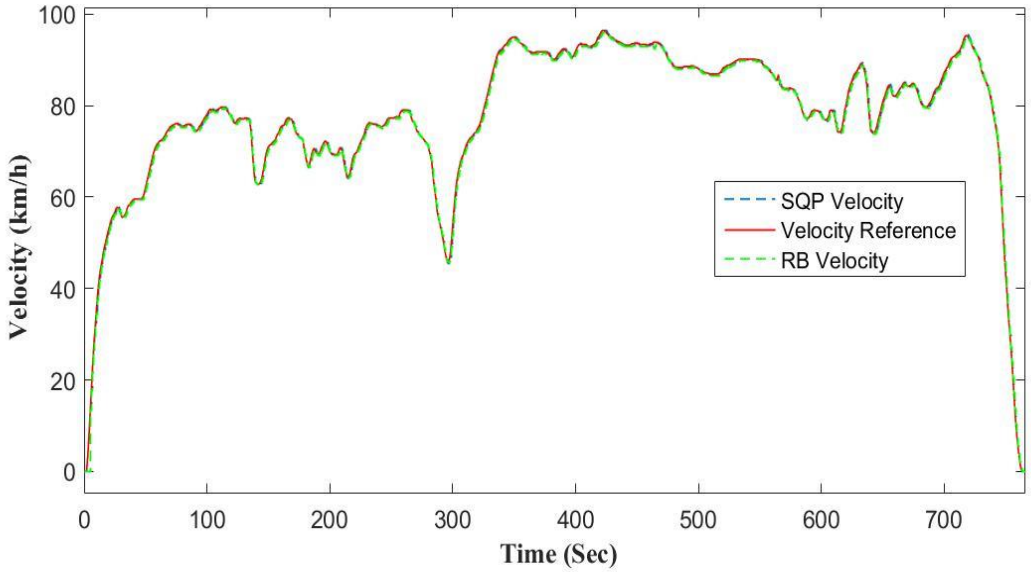

Figure 30: HWYFET Drive Cycle

Figure 31 shows the torque request for the SQP logic. The logic fluctuates between forced recharge of the battery mode and the SQP logic. This is because the ESS SOC is low enough that the rule-based logic in SQP will override the SQP logic and force the ICE into providing more torque than needed and make the EM operate as a generator. Once the SOC is above the lower limit it reverts to the SQP logic. Figure 32 shows the RB method torque request for the HWFET cycle. The two methods are very similar because the RB naturally holds a torque and adjust the EM torque, while the SQP method will only perform this task when the SOC is too low. Figure 33 shows the reason the SQP method causes the operation a majority of the time in the regenerative mode. Indicating that during the HWFET cycle, both controller methods are mainly are using the RB calculate for the torque split and to return energy to the ESS, however the SQP method is in regenerative mode, while RB is in motor assist mode when driving the cycle.

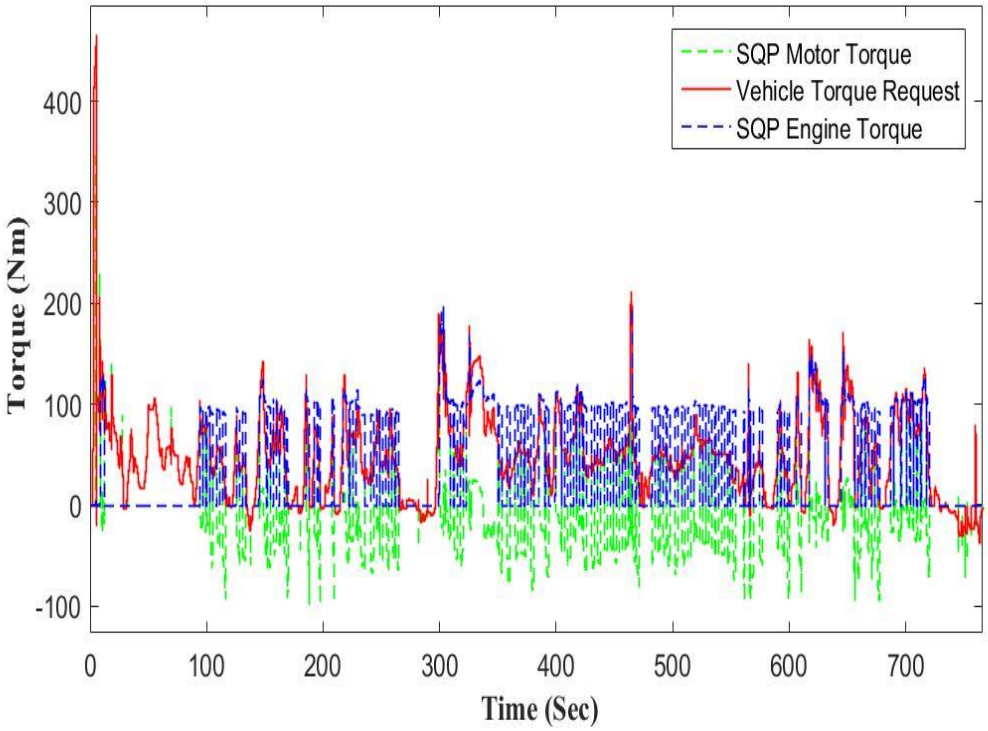

Figure 31: SQP Torque for HWYFET Cycle

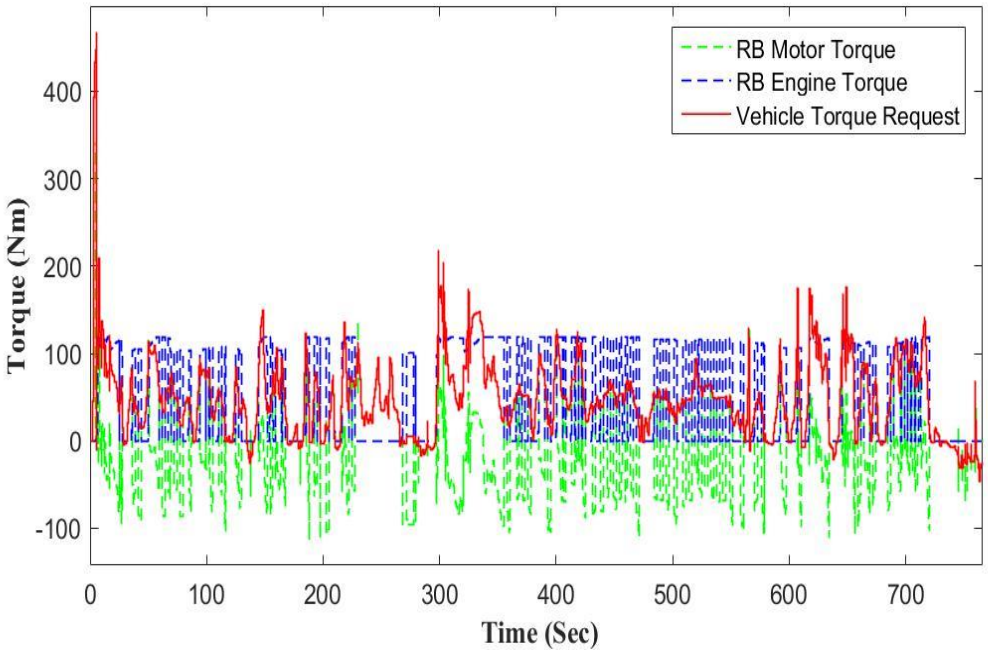

Figure 32: RB Torque for HWYFET Cycle

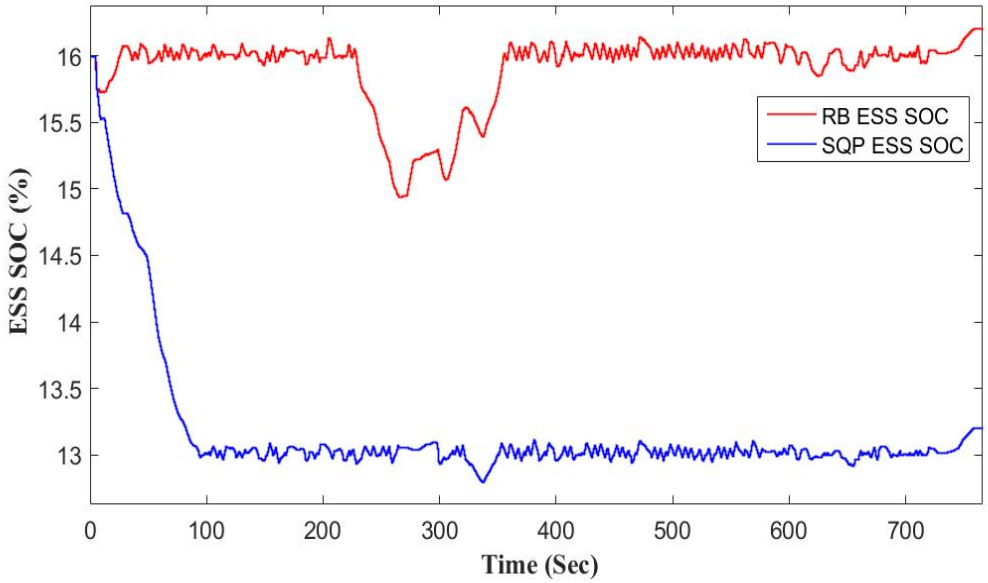

Figure 33: HWYFET ESS SOC

Table 9 shows the results of the HWFET drive cycle. There is a 20.4% difference in SOC and the net SOC for the RB method is negative because the ESS SOC was above the initial SOC value. However, the total energy used by the RB method is 9.81% greater than SQP method.

Table 9: HWYFET Cycle Results

	MPG	MPGE	SOC	Fuel Used (Gal)	ESS Net Energy (kWh)	Total Energy (kWh)
RB	48.61	48.38	16.20	0.212	-0.032	4.756
SQP	55.70	54.18	13.2	0.183	0.167	4.311

6.4 505 DRIVE CYCLE RESULTS

The final drive cycle test is the 505 drive cycle, this cycle simulates non-aggressive, urban driving. The drive traces for both methods are in Figure 34, and again the velocity traces are within 2% error trace from the reference trace.

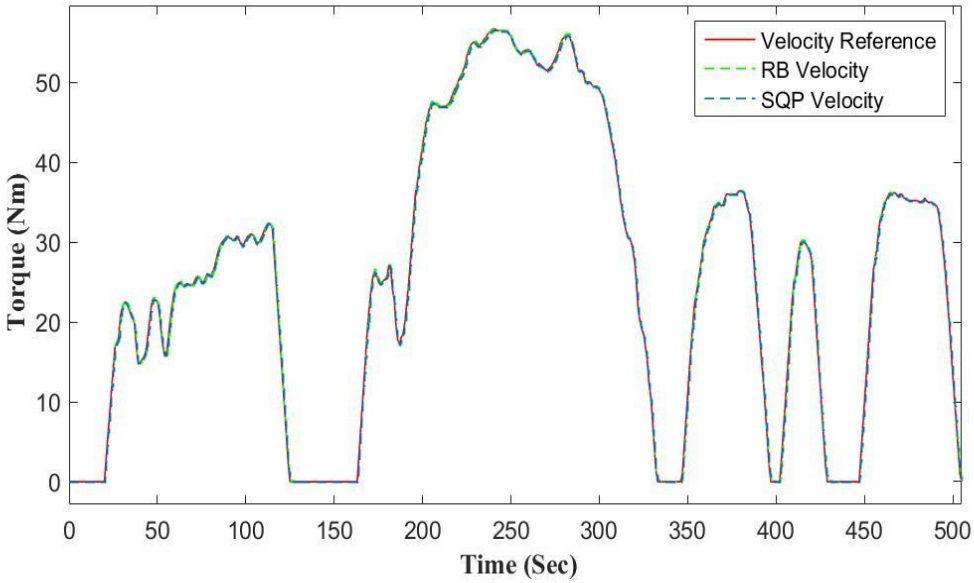

Figure 34: 505 Drive Cycle

Figure 35 illustrates how the SQP logic does not turn on the ICE unless the EM cannot accommodate the torque request. However, because the drive cycle is not really torque demanding, the EM provides the majority of the torque until the SOC needs to recharge.

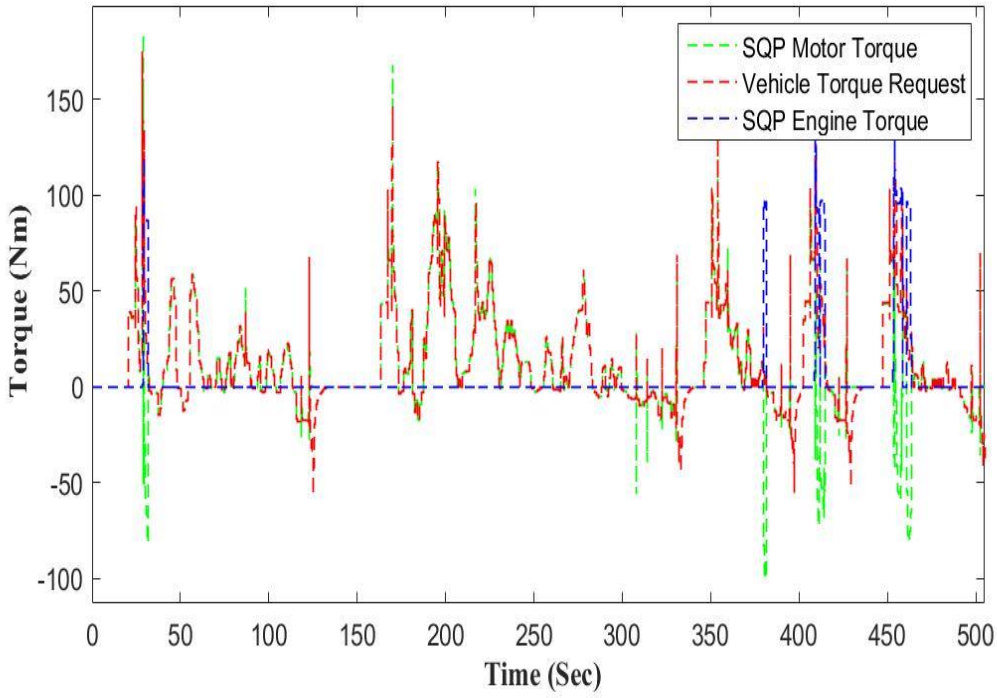

Figure 35: SQP Torque for 505 Cycle

Figure 36 illustrates that the ICE is on more for the RB method than the SQP method. This feature is the likely cause for why the efficiency lower in the RB method. However, the RB method has a higher ending SOC compared to the SQP method, partially offsetting the efficiency difference and a higher SOC could be beneficial if an aggressive acceleration or hill climb is subsequently performed.

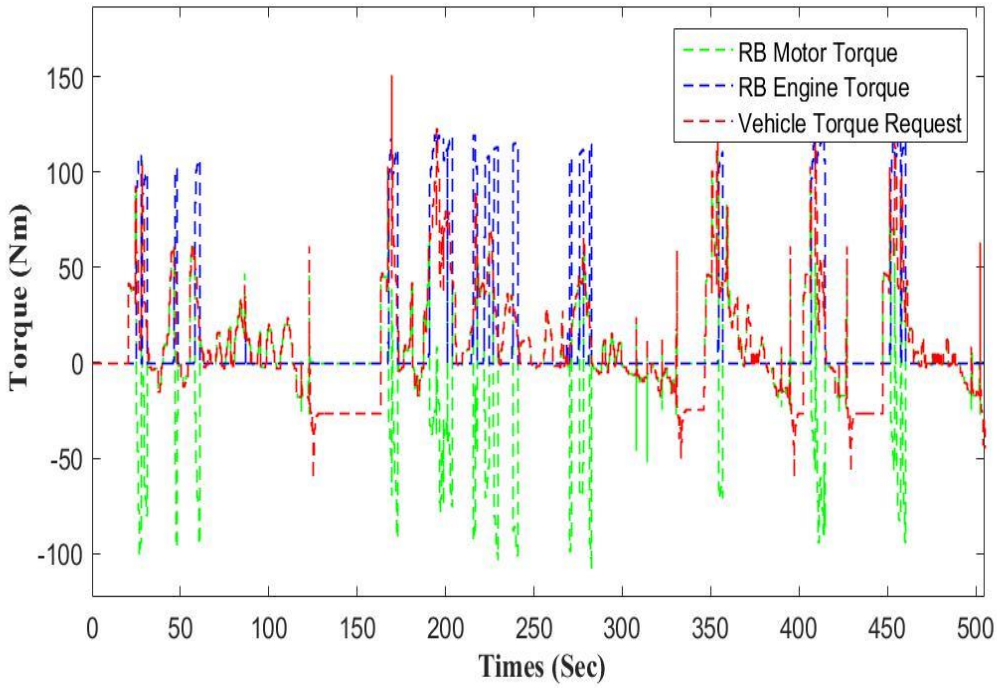

Figure 36: RB Torque for 505 Cycle

Figure 37 shows the ESS SOC during the "505" drive cycle for both methods, and illustrates that the RB method is indeed maintaining the ESS SOC at a higher SOC value because the ICE is on more of the time, and causing more motor regenerative operation. However, the SQP method maintains the SO within the allowable window as well. Table 10 shows the results from the 505 drive cycle. The fuel economy result namely the mpgge value which includes both fuel and electricity usage, demonstrates again that SQP method achieves higher efficiency consistently in comparison to the RB method. The total energy used by the SQP method is 70.07% less than the RB method as well.

Table 10: 505 Drive Cycle Results

	MPG	MPGE	SOC	Fuel Used (Gal)	ESS Net Energy (kWh)	Total Energy (kWh)
RB	60.89	60.52	16.01	0.0367	-0.007	0.8252
SQP	238.37	147.30	13.1	0.009	0.186	0.397

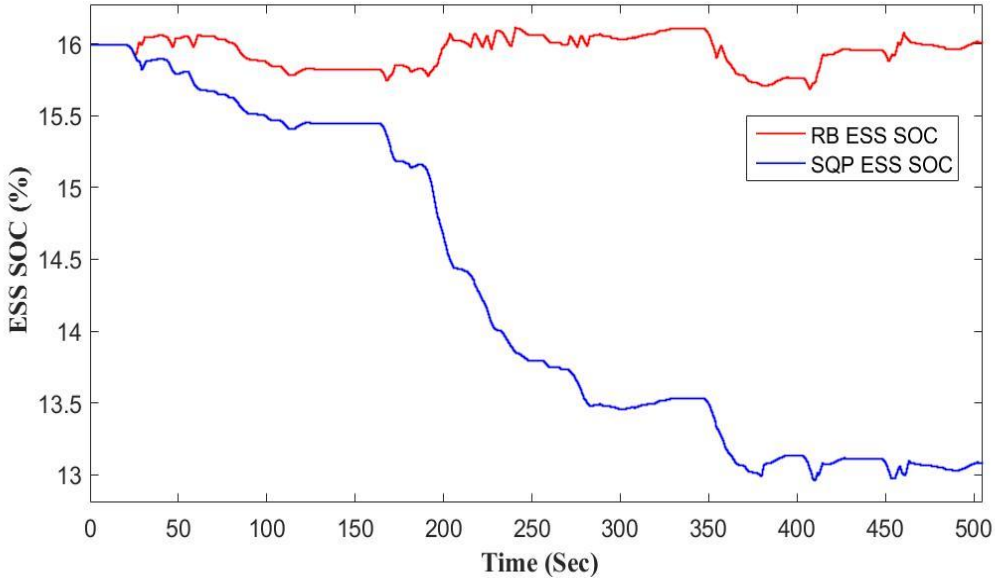

Figure 37: 505 ESS SOC

For the HWYFET and 505 cycles the RB method limited the amount of increase in SOC before the logic will shift all the torque back to the EM to reduce the amount of ICE use. This, in turn increase the overall efficiency of the vehicle.

Between all three drive cycles the RB method was able to maintain the SOC better than the SQP method. However, the SQP method performed better in the areas of energy savings. The SQP method also showed improvement in mpg and mpgge.

6.5 HARDWARE RESULTS

For the hardware results, the main aspect that was examined was the number of overruns that the processor had when it came to calculating the models. The SQP method must be able to complete its calculations before the process cycle to prove that it can be used on a

vehicle hardware system. The hardware used for this experiment is the Dspace DS1006 and the ETAS ES910. The Dspace is a platform device that can be programmed to simulate full models using any kind of hardware interface, such as, control area networks (CAN), analog and digital signals, and other forms of serial communication. The ETAS is a prototype controller; this device is for programming prototype software used for vehicle prototypes. The SQP method's logic is sectioned between the DSPACE and the ETAS hardware. The actual calculations and script are on the Dspace unit, and the calculated torque split will be sent to the ETAS through a CAN channel. The ETAS unit is programmed with the logic that determines whether to pass through the SQP calculated values or process the RB torque modes for the final torque split to the vehicle model. The reason that the SQP logic was not built on the ETAS hardware is because the ETAS software doesn't support MatLAB Function Block in Simulink. This makes it impossible to program the SQP logic on the ETAS hardware and do a complete side-by-side comparison of the two methods. Another discrepancy between the two hardware is the processor each has. The ETAS has a NXP PowerQUICC™ III MPC8548 with 800 MHz clock Double precision floating point unit. [ETAS website] Mean while the DSPACE hardware Quad-core AMD Opteron™ 2.8 GHz. processor. [DSPACE website] However, during the simulation for the SQP method, the logic was on one core of the processor only. This makes it difficult to see if the logic can be computed on lower-power hardware. However, these are the only two piece of hardware that were available to use for this experiment. The computation time and total simulation results for the US06 drive cycle are shown in Table 11.

The reason that the US06 was the only drive cycle selected for the hardware test, was because the US06 cycle is the most dynamic cycle and would show the amplify effects of communication latencies and issues with the SQP method. The computation time between RB and SQP is critical in determining if the SQP method could be implemented in actual vehicle hardware.

Table 11: Hardware Computation Time Results

Strategies	Average Computation Time (sec)	Simulation Time for US06 600 (Hours)
SQP	0.0030	0:32:25
RB	0.0027	0:27:53

Figure 38 shows the vehicle trace of the SQP and RB methods. The trace matches the results from model in loop (MIL) simulations.

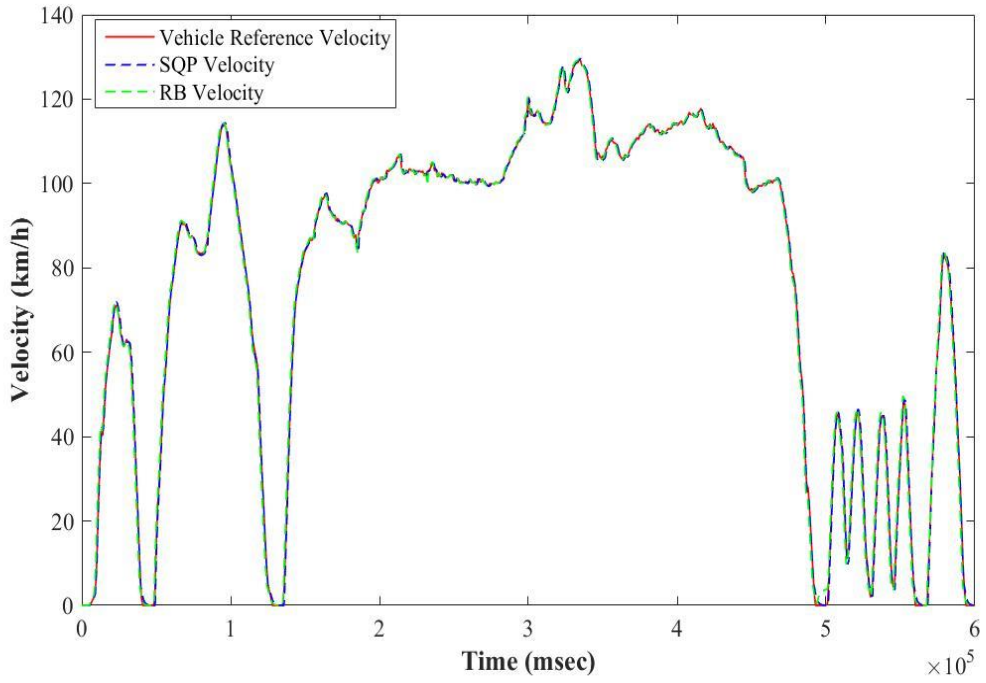

Figure 38: HIL Vehicle Trace For SQP and RB

Figure 39 shows the ESS SOC from both the SQP and RB methods. The results are not the same as the MIL results. The SQP method maintain the ESS SOC between the stated SOC limits like in the MIL results but the behavior of the SOC trace is different. One possible reaction could be due to the torque demand behaving differently from the MIL simulations. The behavior of the accelerator pedal could cause a larger torque demand and request more torque from the powertrain forcing the SQP method to respond to the demand.

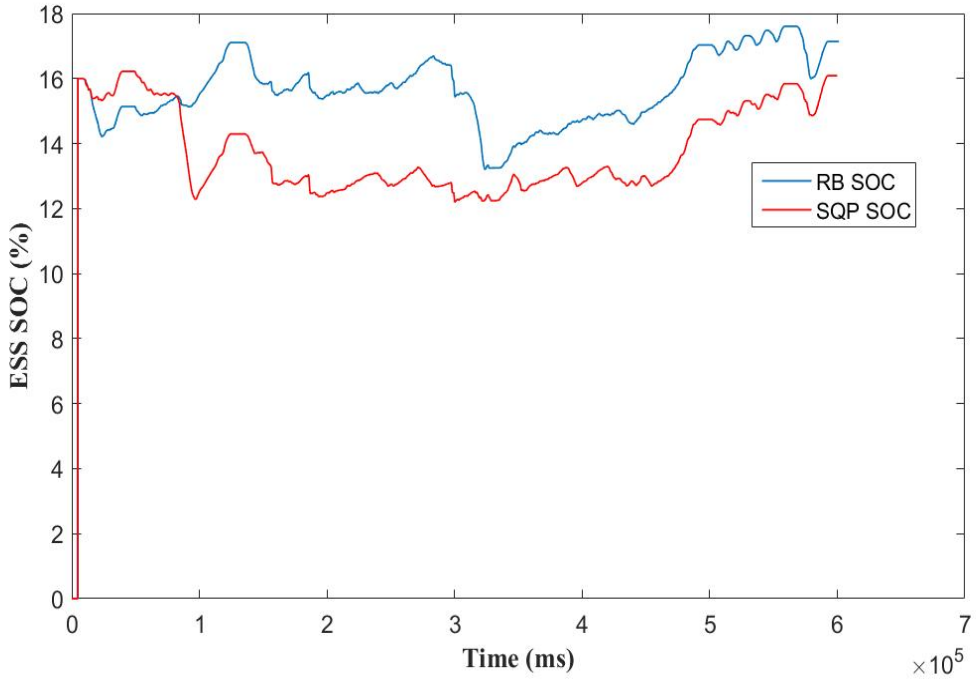

Figure 39: HIL ESS SOC for SQP and RB

Figure 40 and Figure 41 shows the torque split from the SQP method and the RB method there is no difference between the HIL and MIL results. Indicating that the program is stable during hardware testing and inherited latencies expected from CAN communication.

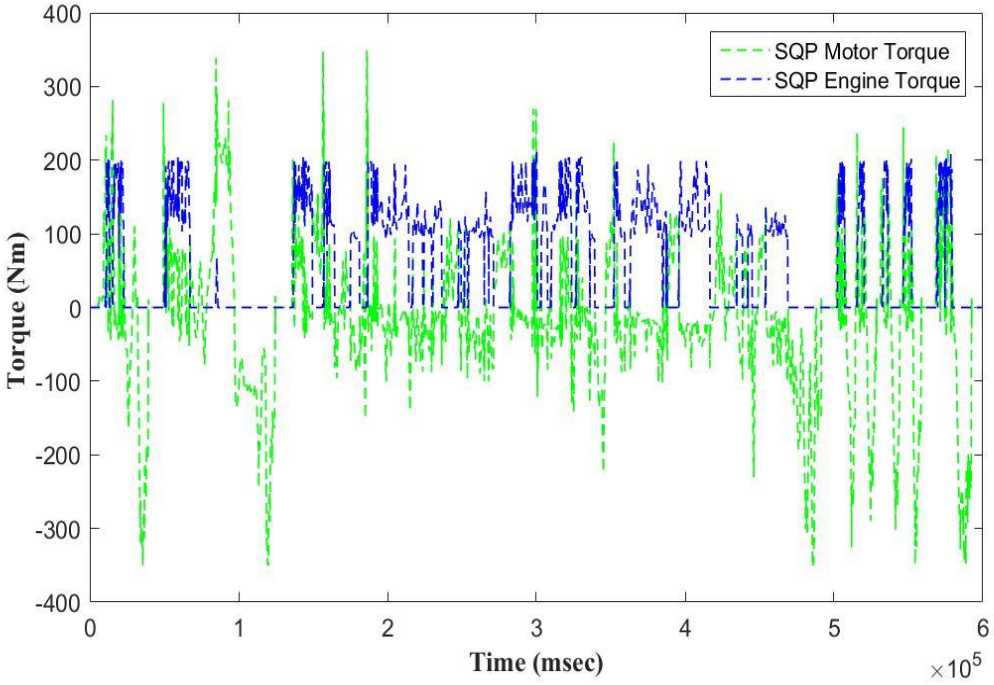

Figure 40: SQP HIL Torque Results

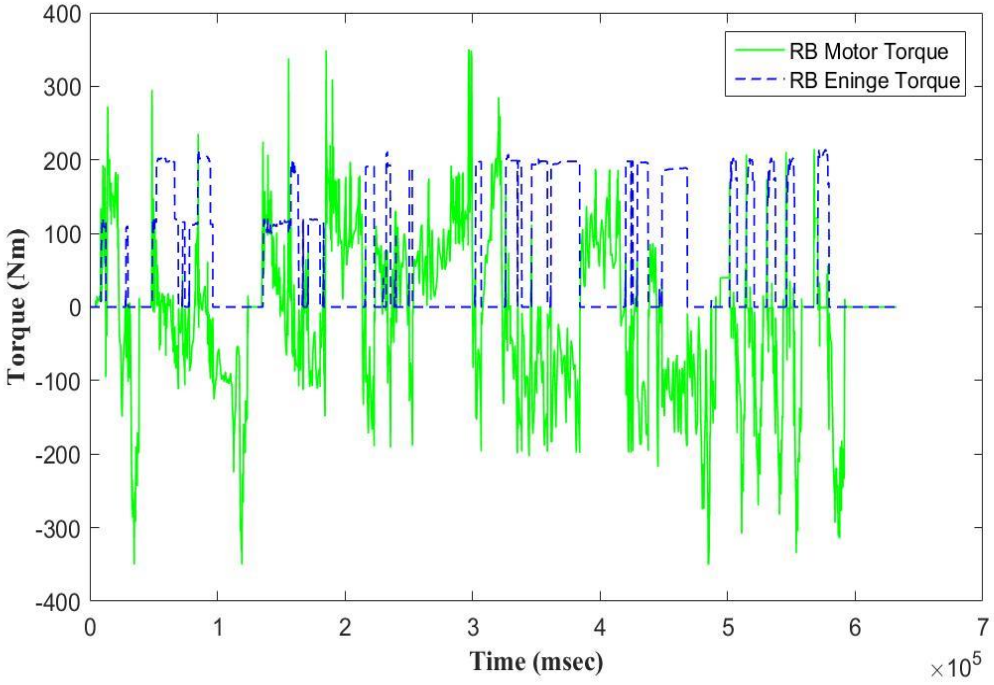

Figure 41: RB HIL Torque Results

7. CONCLUSION AND FUTURE WORK

7.1 CONCLUSION

From the results shown in Chapter 6, it can be determined that through software simulation, the SQP method can be used as a process to increase the efficiency of the overall system. It is a local optimization that can determine the proper torque split between the ICE and EM to reduce the overall energy consumption of the vehicle. The SQP method does deplete the ESS more than the RB method and it does require a separate logic that will return the ESS SOC to an acceptable level. The software results also show that SQP can accommodate rapid torque changes and different drive scenarios. The results from the SQP method compared to the RB method did show improvement; however, in some parts of the tested drive cycles, the SQP and RB did run similarly because of the SOC being too low and close to its lower limits. Table 12 shows the energy increase from switching from the RB method to SQP method logic for all four drive cycles used. It is noted that the SQP was able to increase efficiency and decrease energy usage for all three tested drive cycles, the greatest improvement was in the 505 due to the ICE not operating as often. This is due to the torque threshold for the SQP algorithm to calculate the torque for the EM and ICE was not reach as often as the RB torque threshold. The US06 had the lease increase in energy efficiency due to the high torque demands the cycle requests.

12. Energy Percent Difference

	Simple	US06	HWYFET	505
Energy Percent Increase	40.83%	6.22%	9.82%	70.07%

Also based on the results from the hardware, the SQP method can be used on standalone hardware. However, because the dSpace has such a powerful processor compared to the

ETAS hardware, the results could may be skewed. A way to accommodate this is proposed in future work.

7.2 FUTURE WORK

There is a lot of extra work that can be performed with SQP. First off the logic can be adapted so that it can maintain the SOC within its own logic instead of require a second logic. This would allow the SQP to be tested against the RB when the SOC is too low and still meet the torque requested by the driver. It would also see if the computation time increase due to that constant SQP calculation. Another improvement can be the objective function if it included the transmission gear ratios. This would allow control of what gear the transmission is in order to keep the EM and ICE speed at the most optimal points so that the torque split has even better efficiency. Secondly the SQP logic could be tested on hardware that is similar to the ETAS that supports the Matlab Function Block. This would allow a better comparison if the logic can be used in a real world vehicle with a weaker processor than the DSPACE.

REFERENCES

Administration, National Highway Traffic Safety. (2010). *Coperate Average Fuel Economy*. Retrieved 1 21, 2017, from https://www.nhtsa.gov/laws-regulations/corporate-average-fuel-economy

Alouani, B. A. (2009). A simple power based control strategy for hybrid electric vehicles . *Vehicle Power and Proulsion Conference* , 803-807.

Anthony M. Philips, M. J. (2000). Vehicle System Controller Desing for a Hybrid Electric Vehicle . *Control Applications 2000. Proceedings of the 200 IEEE International Conference on. IEEE, ,* 297-302.

Cristian, M. (2005). A-ECMS: An adaptive Algorithm for Hybrid Electric Vehicle Energy Management . *European Journal of Control* , 509-524.

Hofman Theo, v. D. (2008). Rule-Based equivalent fuel consmption minization strategies for hybrid vehicles. *IFAC Proceedings*, 5652-5657.

Jeff Waldner, J. W. (2011). Development and Testing of an Adacned Extedned Range Electric Vehcile. *Society of Automotive Engineers*.

Jianfeng Song, X. Z. (2011). Research on Control Stratgy of Mild Parallel Hybrid Electric Vehicles. *2011 International Conference on Electric Information and Control Engineering* .

Joao Albuquerque, V. G. (1999). Interior Point SQP Strategies for large-scale, strutured process optimiaztion problems . *Computers ad Chemical Engineering* , 543-554.

Kim Namwook, S. C. (2011). Optimal Control of Hybrid Electric Vehicles Based on Pontryagin's Minimum Principle. *IEEE Transactions on Control Systems Tehnology*, 1279-1287.

Kyoungcheol Oh, J. K. (2015). Optimal Power Distribution Control for Parallel Hybrid Eletric Vehicles . *Sociat of Automotive Engineering*, 50-62.

Laura V. Perez, G. R. (2006). Optimizatio of power management in an hybrid electric vehicle using dynamic programming. *ELSEVIER*, 293-303.

Liu Zhentong, H. H. (2012). Study on Control Strategy and Simulation for Parallel Hybrid Electric Vehicle. *Control Engineering and Communication Technology (ICCECT), 2012 International Conference on.* .

Martin Sivertsson, C. S. (2011). Adaptive Control of Hybrid Powertrain With Map-Based ECMS. *International Federation of Automatic Control World Congress*.

Mathworks. (n.d.). Matlab. *Matlab R2015b*. Boston, United States of America: Mathworks Inc.

Oh Kyoungcheol, J. K. (2005). Optimal Power Distribution Control for Parallel hybrid Electric Vehicles. *IEEE International Conference on Vehicular Electronics and Safety*, 79-85.

R. H. BYRD, R. A. (1992). AN SQP AUGMENTED LAGRAGIAN BFGS ALGORITHM FOR CONSTRAINED OPTIMIZATION. *Society for Industrial and Applied Mathematics*, 210-241.

Ren, M. Y. (2015). Sequantial Quadratic Programming. *Arizona State University* .

Salman, M. N. (2000). Control strategies for parallel hybrid vehicles. *American Control Conference 2000. Proceedings of the 2000*, 524-528.

Shankar R, M. J. (2012). The Novel Application of Optimization and Charge Blended Energy Management Control for Component Downsizing Within a Plug-In Hybrid Electric Vehicles. *Energies*, 4892-4923.

Tae Soo Kim, C. M. (2009). Model Predicitve Control of Velocity and Torqe Split in a Parallel Hybrid Vehicle. *IEEE*, 2014-2019.

Valeri H. Johnson, K. B. (2000). HEV Control Strategy for Real-Time Optimization for Fuel Economy and Emissions. *Society of Automotive Engieers*, 50-65.

Weng, C. (2011). Optimal Control of Hybrid Electric Vehicle with Power Split and Torque Split Strategies: A Comparative Case Study. *Proceedings of the 2011 American Control Conference* , 2131-2136.

Wilde, P. Y. (2000). *Principles of Optimal Design Modeling and Computation Second Edition.* Cambridge : Cambridge Univeristy Press.

Wishart, J. D. (2008). *Modelling, Simulation, Testing and Optimization of Advanced Hybrid Vehicle Powetrains.* Victoria : University of Victoria.

Yoshihiro TANAKA, M. F. (1986). A globally convergent SQP method for semi-infinite nonlinear optimization. *Jornal of Computational and Applied Mathematics* , 141-153.

Young-Bae Kim, D. X. (2011). Optimal Operation Strategy Development for Fuel Cell Hybrid Vehciles. *Journal of Mechanical Science and Technology*, 183-192.

Yuan Zhu, Y. C. (2004). A Four-Step Method to Desing an Energy Management Strategy for Hybrid Vehicles. *Amerian Control Conference*, 151-161.

Yuan Zhu, Y. C. (2006). Optimisation design of an energy management strategy for hybrid vehicles. *Interational Journal Alternative Propulsion*, 47-62.

Zhang Ham, Z. Y. (2004). Optimal Energy Management Strategy for Hybrid Electric Vehicles . *Society of Automotive Engineers*, 250-262.

Zhang Yi, a. H. (2011). Fuzzy Logic Torque Control Strategy for Parallel Hybrid Electric Vehicles. *Fuzzy Systems and Knowledge Discovery (FSKD) 2011 Eighth International Conference* , 640-644.

APPENDIX A

DATA COLLECTED FROM FMINCON SIMULATIONS

FMINCON ENG EFF											
	50	**100**	**150**	**200**	**250**	**300**	**350**	**400**	**450**	**500**	**550**
1000	23.07	23.07	23.07	23.07	23.07	23.07	23.07	23.07	23.07	22.67	22.67
1200	23.63	23.63	23.63	23.63	23.63	23.63	23.63	23.63	23.63	22.99	22.88
1400	24.15	24.15	24.15	24.15	24.15	24.15	24.15	24.15	24.15	23.65	22.19
1600	24.63	24.63	24.63	24.63	24.63	24.63	24.63	24.63	24.63	24.25	21.54
1800	25.08	25.08	25.08	25.08	25.08	25.08	25.08	25.08	25.08	24.8	20.85
2000	25.5	25.5	25.5	25.5	25.5	25.5	25.5	25.5	25.5	25.3	21.76
2200	25.88	25.88	25.88	25.88	25.88	25.88	25.88	25.88	25.88	25.76	22.18
2400	26.23	26.23	26.23	26.23	26.23	26.23	26.23	26.23	26.23	26.17	22.86
2600	26.55	26.55	26.55	2655	26.55	26.55	26.55	26.55	26.55	26.54	23.48
2800	27.35	27.35	27.35	27.35	27.35	27.35	27.35	27.35	27.35	27.35	25.03
3000	27.11	27.11	27.11	27.11	27.11	27.11	27.11	27.11	27.11	27.11	24.56
3200	27.35	27.35	27.35	27.35	27.35	27.35	27.35	27.35	27.35	27.35	25.03
3400	27.57	27.57	27.57	27.57	27.57	27.57	27.57	27.57	27.57	27.57	25.46
3600	27.77	27.77	27.77	27.77	27.77	27.77	27.77	27.77	27.77	27.77	25.84
3800	27.95	27.95	27.95	27.95	27.95	27.95	27.95	27.95	27.95	27.95	26.04
4000	28.11	28.11	28.11	28.11	28.11	28.11	28.11	28.11	28.11	28.11	25.38
4200	28.25	28.25	28.25	28.25	28.25	28.25	28.25	28.25	28.25	27.93	26.04
4400	28.38	28.38	28.38	28.38	28.38	28.38	28.38	28.38	28.38	27.54	26.41
4600	28.5	28.5	28.5	28.5	28.5	28.5	28.5	28.5	28.5	26.95	26.66
4800	28.61	28.61	28.61	28.61	28.61	28.61	28.61	28.61	28.61	26.53	26.53
5000	28.71	28.71	28.71	28.71	28.71	28.71	28.71	28.71	28.54	25.19	25.14
5200	28.81	28.81	28.81	28.81	28.81	28.81	28.81	28.81	28.24	25.61	25.61
5400	28.91	28.91	28.91	28.91	28.91	28.91	28.91	28.91	27.98	26.03	26.03
5600	29.01	29.01	29.01	29.01	29.01	29.01	29.01	29.01	27.64	26.49	26.49
5800	29.12	29.12	29.12	29.12	29.12	29.12	29.12	29.12	27.24	27.03	27.03
6000	29.23	29.23	29.23	29.23	29.23	29.23	29.23	29.23	27.7	27.7	27.7
6200	29.36	29.36	29.36	29.36	29.36	29.36	29.36	29.31	28.19	28.19	28.19
6400	29.71	29.71	29.71	29.71	29.71	29.71	29.71	29.71	29.71	29.71	29.71

FMINCON MOT EFF											
	50	100	150	200	250	300	350	400	450	500	550
1000	85.29	84.92	84.55	84.18	83.82	83.45	83.08	82.71	82.34	82.15	82.15
1200	87.04	86.63	86.21	85.8	85.38	84.97	84.55	84.14	83.73	83.49	83.49
1400	88.69	88.23	87.77	87.31	86.85	86.39	85.93	85.47	85.01	84.71	84.71
1600	90.24	89.73	89.22	88.72	88.21	87.71	87.2	86.69	86.19	85.83	85.83
1800	91.68	91.13	90.57	90.02	89.47	88.92	88.37	87.81	87.26	86.84	86.84
2000	93.02	92.42	91.82	91.22	90.62	90.03	89.43	88.83	88.23	87.75	87.75
2200	94.25	93.61	92.96	92.32	91.68	91.03	90.39	89.74	89.1	88.54	88.54
2400	95.38	94.69	94	93.31	92.62	91.93	91.24	90.55	89.86	89.23	89.22
2600	96.4	95.67	94.93	94.2	93.46	92.73	91.99	91.26	90.52	89.8	89.8
2800	98.85	97.98	97.11	96.23	95.36	94.49	9..2	92.74	91.87	91	90.88
3000	98.14	97.31	96.49	95.66	94.83	94.01	93.18	92.35	91.53	90.7	90.63
3200	98.85	97.98	97.11	96.23	95.36	94.49	93.62	92.74	91.87	91	90.88
3400	99.46	98.54	97.62	96.7	95.78	94.86	93.95	93.03	92.11	91.19	91.02
3600	99.95	98.99	98.03	97.06	96.1	95.13	94.17	93.21	92.24	91.28	91.06
3800	100.35	99.34	98.33	97.32	96.31	95.3	94.29	93.28	92.27	91.26	91.02
4000	100.6	99.58	98.52	97.47	96.41	95.36	94.3	93.24	92.19	91.13	91.12
4200	100.8	99.71	98.61	97.51	96.41	95.31	94.2	93.1	92	91.13	91.13
4400	100.8	99.74	98.59	97.4	96.3	95.15	94	92.85	91.71	91.06	91.06
4600	100.8	99.66	98.46	97.27	96.08	94.88	93.69	92.5	91.3	90.9	90.9
4800	100.71	99.47	98.23	96.99	95.75	94.51	93.27	92.03	90.79	90.65	90.65
5000	100.4	99.17	97.88	96.6	95.31	94.03	92.74	91.46	90.32	90.32	90.32
5200	100.09	98.76	97.42	96.09	94.76	93.43	92.1	90.77	89.86	89.86	89.86
5400	99.61	98.23	96.86	95.48	94.11	92.73	91.35	89.98	89.22	89.22	89.22
5600	99.02	97.6	96.18	94.76	93.34	91.91	90.49	89.07	88.49	88.49	88.49
5800	98.32	96.86	95.39	93.92	92.45	90.98	89.52	88.05	87.66	87.66	87.66
6000	97.51	96	94.48	92.97	91.46	89.94	88.43	86.92	86.74	86.74	86.74
6200	96.58	95.02	93.46	91.9	88.78	87.22	85.71	85.71	85.71	85.71	85.71
6400	95.31	93.7	92.1	90.49	88.88	87.28	85.67	84.57	84.57	84.57	84.57

FMINCON ENG TORQUE											
	50	**100**	**150**	**200**	**250**	**300**	**350**	**400**	**450**	**500**	**550**
1000	125.7	125.7	125.7	125.7	125.7	125.7	125.7	125.7	125.7	142	142
1200	128.78	128.78	128.78	128.78	128.78	128.78	128.78	128.78	128.78	150	152
1400	131.83	131.83	131.83	131.83	131.83	131.83	131.83	131.83	131.83	150	171.59
1600	134.79	134.82	134.82	134.82	134.82	134.82	134.82	134.82	134.82	150	185.9
1800	137.75	137.77	137.73	137.73	137.77	137.77	137.78	137.77	137.77	150	198.4
2000	140.64	140.66	140.66	140.66	140.66	140.66	140.67	140.66	140.66	150	197.36
2200	143.48	143.48	143.5	143.5	143.5	143.46	143.5	143.51	143.53	150	200
2400	146.26	146.26	146.26	146.28	146.25	146.28	146.28	146.28	146.3	150	200
2600	148.97	148.97	148.97	149	149	149	149	149	149	150	200
2800	151.62	151.62	151.65	151.65	151.63	151.65	151.65	151.64	151.66	151.67	200
3000	154.2	154.2	154.23	154.23	154.2	154.23	154.23	154.22	154.24	154.24	200
3200	156.7	1567	156.7	156.73	156.71	156.73	156.73	156.74	156.74	156.77	200
3400	159.12	159.12	159.12	159.13	159.16	15916	159.16	159.17	159.16	159.17	2000
3600	161.45	161.45	161.46	161.49	161.49	161.49	161.49	161.51	161.49	161.5	200
3800	163.69	163.69	163.7	163.73	163.73	163.73	163.73	163.74	163.73	163.74	201.87
4000	165.83	165.83	165.83	165.87	165.87	165.87	165.87	165.88	165.87	165.95	213.47
4200	167.85	167.85	167.86	167.91	167.91	167.91	167.91	167.91	167.91	178.53	209.67

4400	169.82	169.82	169.82	169.82	169.82	169.82	169.82	169.82	169.82	191.57	208.55
4600	171.61	171.61	171.61	171.61	171.61	171.61	171.61	171.61	171.61	204.62	208.52
4800	173.27	173.27	173.27	173.27	173.26	173.25	173.26	173.27	173.31	213.17	213.17
5000	174.72	174.73	174.73	174.74	174.77	174.76	174.77	174.77	180.68	230.67	231.2
5200	176.06	176.07	176.07	176.07	176.11	176.11	176.11	176.12	191.71	228.89	228.89
5400	177.22	177.22	177.23	177.24	177.28	177.27	177.28	177.29	199.73	226.52	226.52
5600	178.18	178.19	178.2	178.2	178.26	178.24	178.26	178.26	207.75	223.4	223.4
5800	179.02	179.02	179.02	179.01	179.02	179.01	179.02	179.1	215.77	218.65	218.65
6000	179.55	179.55	179.55	179.55	179.55	179.51	179.55	179.58	211.21	211.21	211.21
6200	179.82	179.82	179.81	179.81	179.81	179.82	179.82	181.33	205.59	205.59	205.59
6400	172.73	172.73	172.73	172.73	172.73	172.73	172.73	172.73	172.73	172.73	172.73

FMINCON MOT TORQUE											
	50	**100**	**150**	**200**	**250**	**300**	**350**	**400**	**450**	**500**	**550**
1000	-75.7	-25.7	24.3	74.3	124.3	174.3	224.3	274.3	324.3	350	350
1200	-78.78	-28.78	21.21	71.21	121.21	171.21	221.21	271.21	321.21	349.99	350
1400	-81.83	-31.83	18.16	68.16	118.17	168.16	218.16	268.16	318.16	349.99	350
1600	-84.79	-34.82	15.17	65.17	115.17	165.17	215.17	265.17	315.17	349.99	350
1800	-87.75	-37.77	12.26	62.26	1112.22	162.22	212.21	262.22	312.22	349.99	350
2000	-90.64	-40.66	9.33	59.33	109.33	159.33	29.32	259.33	309.33	349.99	350
2200	-93.48	-43.48	6.49	56.49	106.49	156.53	206.49	256.48	306.46	349.99	349.99
2400	-96.26	-46.26	3.73	53.71	13.74	153.71	203.71	153.71	303.69	349.9	350
2600	-98.97	-48.97	1	50.99	100.99	150.99	200.99	251	30.97	349.97	350
2800	-101.62	-51.62	-1.65	48.34	98.36	148.34	198.34	248.35	298.33	348.32	350
3000	-104.2	-54.2	-4.2	45.76	95.79	145.76	195.76	245.77	295.75	345.75	349.99
3200	-106.7	-56.7	-6.7	43.26	93.28	143.26	193.26	243.25	293.25	343.22	349.99
3400	-109.12	-59.12	-9.12	40.86	90.83	140.83	190.83	240.82	290.83	340.82	350
3600	-111.45	-61.45	-11.46	38.5	88.5	138.5	188.5	238.48	288.5	338.49	349.99
3800	-113.69	-63.69	-13.7	36.26	86.26	136.26	186.26	236.25	286.26	336.25	348.12

4000	-115.83	-65.83	-15.83	34.12	84.12	134.12	184.12	234.11	284.12	334.04	334.5
4200	-117.85	-67.85	-17.86	32.08	82.08	132.08	182.08	232.08	282.08	321.46	321.46
4400	-119.82	-69.82	-19.82	30.17	80.17	130.21	180.17	230.16	280.17	308.42	308.42
4600	-121.61	-71.61	-21.61	28.38	78.38	128.38	178.38	228.37	278.37	295.39	295.39
4800	-123.27	-73.27	-23.27	26.72	76.73	126.74	176.73	226.2	276.68	282.35	282.35
5000	-124.72	-74.73	-24.73	25.25	75.22	125.23	175.22	225.22	269.31	269.32	269.32
5200	-126.06	-76.07	-26.07	23.92	73.88	123.88	173.88	223.87	258.28	258.29	258.29
5400	-127.22	-77.23	-27.23	22.75	72.71	122.72	172.71	222.7	250.26	250.27	250.27
5600	-128.18	-78.19	-28.2	21.79	71.73	121.75	171.73	221.73	242.24	242.25	242.25
5800	-129.02	-79.02	-29.02	20.98	70.97	120.98	170.97	220.89	234.22	234.22	234.22
6000	-129.55	-79.55	-29.55	20.44	70.44	120.48	170.44	220.41	226.2	226.2	226.2
6200	-129.82	-79.82	-29.81	20.18	70.18	120.17	170.17	218.66	218.68	218.68	218.68
6400	-122.73	-72.73	-22.73	27.26	77.26	127.26	177.26	211.5	211.5	211.5	211.5

APPENDIX B

FMINCON CODE FOR SIMULATIONS

```
function x   =
ModelOptimization(Mot_S,Eng_S,T_dmd,bat_power,Pos_Mot_Trq_L
imit,Eng_Trq_Limit,Neg_Mot_Trq_Limit)
%Optimization Script
%% Optimization Equation Setup
%Initial Conditions for the optimization to start with
x0 = [0,0];   % The starting point.
%% Linear constraints
%First Constraint x1+x3=Torque currently assumed torque
%value is entered. Second constraint x2 = x4. due the
matrix rows need to
%equal the constrain looks like x2+0=x4.
A = [];
b = [];
Ae = [1 1];
Be = [T_dmd];
lb = [Neg_Mot_Trq_Limit,0];
ub = [Pos_Mot_Trq_Limit,Eng_Trq_Limit];

%% NonLinear Constraints
%nonlinconfun.m is the nonlinear function constraint for
battery power
F = @(x)TorqueOpti(x,Mot_S,Eng_S);
nonlinC = @(x)nonlinconfun(x,Mot_S,bat_power);

%% Options for Optimizing
% additional options for running the optimization
%options =
optimoptions(@fmincon,'Algorithm','sqp','MaxIter',50,'TolCo
n',1.0e-2,'TolFun',1.0e-2,'TolX',1.0e-2);
%% Optimization Equation
%using fmincon as the optimization operation it requries
the linear and non
%linear constraints. TorqueOpti is the objective function.
x = fmincon(F,[x0],[],[],Ae,Be,lb,ub,nonlinC);
end
function F= TorqueOpti(x,Mot_S,Eng_S)
%% Motor Curv Fit Equation
MotEffMax =.95;
p00 = .7482;
p10 = .0001127;
p01 = -2.801e-05;
p20 = -1.357e-08;
p11 = -4.579e-08;
MotCurvEqu =
p00+p10*Mot_S+p01*x(1)+p20*(Mot_S^2)+p11*Mot_S*x(1);
```

```
%% Engine Curve Fit Eqaution
EngEffMax = .22;
E00 = .06234;
E10 = 1.536e-05;
E01 = .002475;
E20 = -3.656e-09;
E11 = 1.214e-07;
E02 = -1.141e-05;

E30 = 6.464e-13;
E21 = -4.041e-11;
E12 = 9.496e-10;
EngCurvEqu =
E00+(E10*Eng_S)+(E01*x(2))+(E20*(Eng_S^2))+(E11*Eng_S*x(2))
+(E02*x(2)^2)+(E30*(Eng_S^3))+(E21*((Eng_S^2)*x(2)))+(E12*E
ng_S*(x(2)^2));
%% Objective Function
F = ((1)*(1-MotCurvEqu))+((1)*(1-EngCurvEqu));
end
function [c,ceq] = nonlinconfun(x,Mot_S,bat_power);
PackPower=bat_power;
c = [(Mot_S*x(1)) - PackPower];
ceq = [];
end
```

APPENDIX C

EQUATIONS FOR ENGINE EFFICIENCY MAPS

Region 1 of the engine efficiency map equation

$$EngEff_{R1} = P00 + (P10 * Eng\_S) + (P01 * X1) + (P20 * Eng\_s^2)$$
$$+(P11 * Eng\_S * X1) + (P02 * X2^2) + (P30 * Eng\_S^3)$$
$$+(P21 * Eng\_S^2 * X1) + (P12 * Eng\_S * X1^2) + (P03 * X1^3)$$

The constant values are

Equation Constants	Constant Values
P00	-0.05932
P10	0.0002065
P01	0.004109
P20	-1.604e-07
P11	8.005e-07
P02	-3.625e-05
P30	3.638e-11
P21	-3.003e-10
P12	2.983e-09
P03	6.759e-08

Table 13: Region 1 Constant Values

Region 2 of the engine efficiency map equation

$$EngEff_{R2} = P00 + (P10 * Eng\_S) + (P01 * X1) + (P20 * Eng\_s^2)$$
$$+(P11 * Eng\_S * X1) + (P02 * X2^2) + (P30 * Eng\_S^3)$$
$$+(P21 * Eng\_S^2 * X1) + (P12 * Eng\_S * X1^2) + (P03 * X1^3)$$

The constant values are

Equation Constants	Constant Values
P00	0.1971
0.P10	-0.0001629
P01	0.003235
P20	5.9e-08
P11	5.883e-08
P02	-2.059e-05
P30	-7.188e-12
P21	6.099e-13
P12	-4.57e-10
P03	4.671e-08

Table 14: Region 2 Constant Values

Region 3 of the engine efficiency map equation

$$EngEff_{R3} = P00 + (P10 * Eng\_S) + (P01 * X1) + (P20 * Eng\_s^2)$$
$$+(P11 * Eng\_S * X1) + (P02 * X2^2) + (P30 * Eng\_S^3)$$
$$+(P21 * Eng\_S^2 * X1) + (P12 * Eng\_S * X1^2) + (P03 * X1^3)$$

The constant values are

Equation Constants	Constant Values
P00	0.3535
0.P10	-0.0002876
P01	0.008113
P20	7.73e-08
P11	-1.521e-06
P02	-3.468e-05
P30	-6.524e-12
P21	1.115e-10
P12	2.519e-09
P03	4.903e-08

Table 15: Region 3 Constant Values

Region 4 of the engine efficiency map equation

$$EngEff_{R4} = P00 + (P10 * Eng\_S) + (P01 * X1) + (P20 * Eng\_s^2)$$
$$+(P11 * Eng\_S * X1) + (P02 * X2^2) + (P30 * Eng\_S^3)$$
$$+(P21 * Eng\_S^2 * X1) + (P12 * Eng\_S * X1^2) + (P03 * X1^3)$$

The constant values are

Equation Constants	Constant Values
P00	-2.429
0.P10	0.001123
P01	0.01159
P20	-1.688e-07
P11	-2.512e-06
P02	-2.715e-05
P30	8.294e-12
P21	1.877e-10
P12	6.619e-10
P03	5.191e-08

Table 16: Region 4 Constant Values

APPENDIX D

EQUATIONS FOR MOTOR EFFICIENCY MAPS

Motor efficiency map with just the positive torque efficiency values equation

$$MotEff_{Pos\_Trq} = P00 + (P10 * Mot\_S) + (P01 * X2) + (P20 * Mot\_s^2)$$
$$+(P11 * Mot\_S * X2) + (P02 * X2^2) + (P30 * Mot\_S^3)$$
$$+(P21 * Mot\_S^2 * X2) + (P12 * Mot\_S * X12) + (P03 * X2^3)$$

The constant values are

Equation Constants	Constant Values
P00	0.6812
P10	0.0001035
P01	0.002119
P20	-2.88e-08
P11	5.041e-07
P02	-1.519e-05
P30	2.057e-12
P21	-3.285e-11
P12	-9.011e-10
P03	2.678e-08

Table 17: Motor Positive Torque Efficiency Constant Values

APPENDIX E

DATA COLLECTED FROM SQP SIMULATIONS

SQP ENG EFF											
	50	**100**	**150**	**200**	**250**	**300**	**350**	**400**	**450**	**500**	**550**
1000	14	18.67	21.53	22.86	23.07	22.67	22.67	22.67	22.67	nan	nan
1200	22.88	22.88	22.88	22.88	22.88	22.88	22.88	22.88	22.88	22.88	nan
1400	22.19	22.19	22.19	22.19	22.19	22.19	22.19	22.19	22.19	22.19	nan
1600	21.54	21.54	21.54	21.54	21.54	21.54	21.54	21.54	21.54	21.54	nan
1800	20.85	20.85	20.85	20.85	20.85	20.85	20.85	20.85	20.85	20.85	nan
2000	21.76	21.76	21.76	21.76	21.76	21.76	21.76	21.76	21.76	21.76	nan
2200	22.11	22.11	22.11	22.11	22.11	22.11	22.11	22.11	22.11	22.11	22.11
2400	22.6	22.6	22.6	22.6	22.6	22.6	22.6	22.6	22.6	22.6	22.6
2600	23.26	23.26	23.26	23.26	23.26	23.26	23.26	23.26	23.26	23.26	23.26
2800	23.54	23.54	23.54	23.54	23.54	23.54	23.54	23.54	23.54	23.54	23.54
3000	24.12	24.12	24.12	24.12	24.12	24.12	24.12	24.12	24.12	24.12	24.12
3200	24.25	24.25	24.25	24.25	24.25	24.25	24.25	24.25	24.25	24.25	24.25
3400	24.37	24.37	24.37	24.37	24.37	24.37	24.37	24.37	24.37	24.37	24.37
3600	24.9	24.9	24.9	24.9	24.9	24.9	24.9	24.9	24.9	24.9	24.9
3800	24.95	24.95	24.95	24.95	24.95	24.95	24.95	24.95	24.95	24.95	24.95
4000	25.38	25.38	25.38	25.38	25.38	25.38	25.38	25.38	25.38	25.38	nan
4200	26.04	26.04	26.04	26.04	26.04	26.04	26.04	26.04	26.04	26.04	nan
4400	26.41	26.41	26.41	26.41	26.41	26.41	26.41	26.41	26.41	26.41	nan
4600	26.66	26.66	26.66	26.66	26.66	26.66	26.66	26.66	26.66	26.66	nan
4800	26.53	26.53	26.53	26.53	26.53	26.53	26.53	26.53	26.53	nan	nan
5000	25.14	25.14	25.14	25.14	25.14	25.14	25.14	25.14	25.14	25.14	
5200	25.61	25.61	25.61	25.61	25.61	25.61	25.61	25.61	25.61	nan	nan
5400	26.03	26.03	26.03	26.03	26.03	26.03	26.03	26.03	26.03	nan	nan
5600	26.49	26.49	26.49	26.49	26.49	26.49	26.49	26.49	26.49	nan	nan
5800	27.03	27.03	27.03	27.03	27.03	27.03	27.03	27.03	27.03	nan	nan
6000	27.7	27.7	27.7	27.7	27.7	27.7	27.7	27.7	nan	nan	nan
6200	28.19	28.19	28.19	28.19	28.19	28.19	28.19	28.19	nan	nan	nan
6400	29.71	29.71	29.71	29.71	29.71	29.71	29.71	nan	nan	nan	nan

SQP MOT EFF											
	50	**100**	**150**	**200**	**250**	**300**	**350**	**400**	**450**	**500**	**550**
1000	85.48	84.42	84.24	84.05	83.82	83.57	83.2	82.83	82.46	nan	nan
1200	87.24	86.82	86.41	85.58	85.58	85.16	84.75	84.33	83.92	83.5	nan
1400	89.06	88.6	88.14	87.68	87.22	86.76	86.29	85.83	85.83	84.91	nan
1600	90.75	90.25	89.74	89.24	88.73	88.22	87.72	87.21	86.7	86.2	nan
1800	92.35	91.8	91.24	90.69	90.14	89.59	89.04	88.48	87.93	87.38	nan
2000	93.69	93.1	92.5	92.5	91.3	90.7	90.1	90.11	88.91	88.31	nan
2200	94.98	94.34	93.7	93.05	92.4	91.77	91.12	90.48	89.83	89.19	88.55
2400	96.15	95.46	94.77	94.08	93.39	92.7	92.01	91.32	90.64	89.95	89.26
2600	97.18	96.45	95.71	94.98	94.24	93.51	92.77	92.04	91.3	90.57	89.83
2800	98.16	97.37	96.59	95.81	95.03	94.25	93.47	92.69	91.91	91.13	90.34
3000	98.97	98.15	97.32	96.49	95.66	94.84	94.01	93.18	92.36	91.53	90.7
3200	99.75	98.88	98	97.13	96.26	95.38	94.51	93.64	93.64	93.64	91.02
3400	100.4	99.5	98.58	97.66	96.74	95.83	94.91	93.99	93.07	92.15	91.23
3600	100.9	99.94	98.97	98.01	97.04	96.08	95.12	94.15	93.19	93.19	91.26
3800	100.14	100.4	99.35	98.34	97.33	96.32	95.31	95.31	93.29	92.28	91.27
4000	100.16	100.6	99.53	98.47	97.42	96.36	95.3	94.25	93.19	92.14	nan
4200	101.7	100.6	99.53	98.43	97.33	96.23	95.12	94.02	92.92	91.82	nan
4400	100.18	100.6	99.48	98.33	97.19	96.04	94.89	93.74	92.6	91.45	nan
4600	101.7	100.5	99.34	98.15	96.96	95.76	94.57	93.38	92.19	90.99	nan
4800	101.7	100.5	99.2	97.98	96.74	95.5	94.26	93.02	91.78	nan	nan
5000	101.9	100.6	99.33	98.05	96.76	95.48	94.19	92.91	91.62	90.34	nan

5200	101.5	100.2	98.83	97.5	96.17	94.84	93.51	92.18	90.85	nan	nan
5400	101	99.59	98.21	96.848	95.46	94.09	92.71	91.3	89.96	nan	nan
5600	100.3	98.89	97.46	96.04	94.62	93.2	91.78	90.35	88.93	nan	nan
5800	99.49	98.02	96.55	95.08	93.62	81.34	90.68	89.1	87.74	nan	nan
6000	98.47	96.95	95.44	93.93	92.41	90.9	89.39	87.87	nan	nan	nan
6200	97.38	95.82	94.26	92.71	91.15	89.59	350	86.47	nan	nan	nan
6400	95.31	93.7	92.1	90.49	88.88	87.28	85.67	nan	nan	nan	nan

SQP ENG TRQ											
	50	**100**	**150**	**200**	**250**	**300**	**350**	**400**	**450**	**500**	**550**
1000	142	142	142	142	142	142	142	142	142	142	142
1200	152	152	152	152	152	152	152	152	152	152	152
1400	171.6	171.6	171.6	171.6	171.6	171.6	171.6	171.6	171.6	171.6	171.6
1600	185.9	185.9	185.9	185.9	185.9	185.9	185.9	185.9	185.9	185.9	185.9
1800	198.4	198.4	198.4	198.4	198.4	198.4	198.4	198.4	198.4	198.4	198.4
2000	197.4	197.4	197.4	197.4	197.4	197.4	197.4	197.4	197.4	197.4	197.4
2200	200.6	200.6	200.6	200.6	200.6	200.6	200.6	200.6	200.6	200.6	200.6
2400	202.3	202.3	202.3	202.3	202.3	202.3	202.3	202.3	202.3	202.3	202.3
2600	202	202	202	202	202	202	202	202	202	202	202
2800	204.8	204.8	204.8	204.8	204.8	204.8	204.8	204.8	204.8	204.8	204.8
3000	204.5	204.5	204.5	204.5	204.5	204.5	204.5	204.5	204.5	204.5	204.5
3200	208.1	208.1	208.1	208.1	208.1	208.1	208.1	208.1	208.1	208.1	208.1
3400	211.5	211.5	211.5	211.5	211.5	211.5	211.5	211.5	211.5	211.5	211.5
3600	210.6	210.6	210.6	210.6	210.6	210.6	210.6	210.6	210.6	210.6	210.6
3800	214.1	214.1	214.1	214.1	214.1	214.1	214.1	214.1	214.1	214.1	214.1
4000	213.5	213.5	213.5	213.5	213.5	213.5	213.5	213.5	213.5	213.5	213.5
4200	209.7	209.7	209.7	209.7	209.7	209.7	209.7	209.7	209.7	209.7	209.7
4400	208.6	208.6	208.6	208.6	208.6	208.6	208.6	208.6	208.6	208.6	208.6
4600	208.5	208.5	208.5	208.5	208.5	208.5	208.5	208.5	208.5	208.5	208.5
4800	213.2	213.2	213.2	213.2	213.2	213.2	213.2	213.2	213.2	213.2	213.2
5000	231.2	231.2	231.2	231.2	231.2	231.2	231.2	231.2	231.2	231.2	231.2
5200	228.9	228.9	228.9	228.9	228.9	228.9	228.9	228.9	228.9	228.9	228.9
5400	226.5	226.5	226.5	226.5	226.5	226.5	226.5	226.5	226.5	226.5	226.5
5600	223.4	223.4	223.4	223.4	223.4	223.4	223.4	223.4	223.4	223.4	223.4
5800	218.7	218.7	218.7	218.7	218.7	218.7	218.7	218.7	218.7	218.7	218.7
6000	211.2	211.2	211.2	211.2	211.2	211.2	211.2	211.2	211.2	211.2	211.2
6200	205.6	205.6	205.6	205.6	205.6	205.6	205.6	205.6	205.6	205.6	205.6
6400	172.7	172.7	172.7	172.7	172.7	172.7	172.7	172.7	172.7	172.7	172.7

SQP MOT TRQ											
	50	**100**	**150**	**200**	**250**	**300**	**350**	**400**	**450**	**500**	**550**
1000	-92	-42	8	58	108	158	208	258	308	nan	nan
1200	-102	-52.01	-2.01	47.99	97.99	147.99	197.99	247.99	297.99	347.99	nan
1400	-121.59	-71.59	-21.59	28.41	78.41	128.41	178.41	228.41	278.41	328.41	nan
1600	-135.9	-85.9	-35.9	14.1	4.1	114.1	164.1	214.1	264.1	314.1	nan
1800	-148.4	-98.4	-48.4	1.6	51.6	101.6	151.6	201.6	251.6	301.6	nan
2000	-147.36	-97.6	-47.6	2.64	52.64	102.64	152.64	202.64	252.64	302.64	nan
2200	-150.55	-100.55	50.5	-0.55	49.45	99.45	149.45	199.45	249.45	299.45	349.45
2400	-152.6	-102.26	-52.26	-2.26	47.74	97.74	147.74	197.74	247.74	297.74	347.74
2600	-152	-102	-52	-2	48	98	148	198	248	298	348
2800	-154.81	-104.81	-54.81	-4.81	45.19	95.19	145.19	195.19	245.19	295.19	345.19
3000	-154.52	-104.52	-54.52	-4.52	45.48	95.48	145.48	195.48	245.48	295.48	345.48
3200	-158.1	-108.1	-58.1	-8.1	41.9	91.9	141.9	191.9	241.9	291.9	341.9
3400	-161.52	-111.52	-61.52	-11.52	38.48	88.48	138.48	188.48	238.48	288.48	338.48
3600	-160.56	-110.56	-60.56	-10.56	39.44	89.44	139.44	189.44	239.44	289.44	339.44
3800	-164.1	-114.1	-64.1	-14.1	35.9	85.9	135.9	185.9	235.9	285.9	335.9
4000	-163.47	-113.47	-63.47	-13.47	36.53	86.53	136.53	186.53	236.53	286.53	nan
4200	-159.67	-109.67	-59.67	-9.67	40.32	90.32	140.32	190.32	240.32	290.32	nan

4400	-158.55	-10.55	-58.55	-8.55	41.45	91.45	141.45	191.45	241.45	291.45	nan
4600	-158.52	-108.52	-58.52	-8.52	41.47	91.47	141.47	19.4	241.47	291.47	nan
4800	-163.17	-113.17	-63.17	-13.17	36.82	86.82	136.82	186.82	236.82	nan	nan
5000	-181.21	-131.21	-81.21	-31.21	18.79	68.79	118.79	168.79	218.79	268.79	nan
5200	-178.89	-128.89	-78.89	-28.89	21.11	71.11	121.11	171.11	221.11	nan	nan
5400	-176.52	-126.52	-765 2	-26.52	23.47	73.47	123.47	173.47	223.47	nan	nan
5600	-173.4	-123.4	-73.4	-23.4	26.59	76.59	126.59	176.59	226.59	nan	nan
5800	-168.65	-118.65	-68.65	-18.65	31.34	81.34	131.34	181.34	231.34	nan	nan
6000	-161.21	-111.21	-61.21	-11.21	38.79	88.79	138.79	188.79	nan	nan	nan
6200	-155.59	-105.59	-55.59	-5.59	44.4	94.4	144.4	194.4	nan	nan	nan
6400	-122.73	-72.73	-22.73	27.26	77.26	127.26	177.26	nan	nan	nan	nan

APPENDIX F

SQP CODE FOR SIMULATIONS

```
function [x1,x2] =
VEH(Mot_S,Eng_S,T_dmd,bat_power,Pos_Mot_Trq_Limit,Eng_Trq_L
imit,Neg_Mot_Trq_Limit,J1,J2)
opt.linesearch = true; % false or true
% Set the tolerance to be used as a termination criterion:
opt.eps = 6.5e-5;

% Set the initial guess: (column vector, i.e. x0 = [x1; x2]
)
x0 = [0; 0];

% Feasibility check for the initial point.
%  if
max(g1(x0,Mot_S,Eng_S,T_dmd,bat_power,Pos_Mot_Trq_Limit,Eng
_Trq_Limit,Neg_Mot_Trq_Limit)>0)
%      errordlg('Infeasible intial point! You need to start
from a feasible one!');
%      return
%  end
%% Run optimization
% Run your implementation of SQP algorithm. See mysqp.m

solution = mysqp1(x0,
opt,Mot_S,Eng_S,T_dmd,bat_power,Pos_Mot_Trq_Limit,Eng_Trq_L
imit,Neg_Mot_Trq_Limit,J1,J2);
x1 = solution(1);
x2 = solution(2);
end


%%%%%%%%%%%%%% Sequential Quadratic Programming
Implementation %%%%%%%%%%%%%%%%
function solution = mysqp1(x0,
opt,Mot_S,Eng_S,T_dmd,bat_power,Pos_Mot_Trq_Limit,Eng_Trq_L
imit,Neg_Mot_Trq_Limit,J1,J2)
    % Set initial conditions

    x = x0; % Set current solution to the initial guess

    % Initialize a structure to record search process
%     solution = struct('x',[]);
%     solution.x = [solution.x, x]; % save current solution
to solution.x

    % Initialization of the Hessian matrix
```

```
    W = eye(2,2);               % Start with an identity
Hessian matrix

    % Initialization of the Lagrange multipliers
    mu_old = [0;0;0;0;0;0;0];    % Start with zero Lagrange
multiplier estimates

    % Initialization of the weights in merit function
    w = [0;0;0;0;0;0;0];         % Start with zero weights


    % Set the termination criterion
%   gnorm = norm(df1(x,Mot_S,Eng_S,J1,J2) +
(mu_old'*dg1(x,Mot_S,Eng_S,T_dmd,bat_power,Pos_Mot_Trq_Limi
t,Eng_Trq_Limit,Neg_Mot_Trq_Limit))); % norm of Largangian
gradient
    gnorm = 1;
    coder.varsize('solution',[2 1]);
    solution = zeros(2,1);
    ii = 1;
    while gnorm>opt.eps % if not terminated

        % Implement QP problem and solve
%         if strcmp(opt.alg, 'myqp')
            % Solve the QP subproblem to find s and mu
(using your own method)
            [s, mu_new,i] = solveqp1(x,
W,Mot_S,Eng_S,T_dmd,bat_power,Pos_Mot_Trq_Limit,Eng_Trq_Lim
it,Neg_Mot_Trq_Limit,J1,J2);
%         else

            if i==1
                solution=s;
                return
            end
%             % Solve the QP subproblem to find s and mu
(using MATLAB's solver)
%             qpalg = optimset('Algorithm', 'active-set',
'Display', 'off');
%             [s,~,~,~,lambda] =
quadprog(W,[df1(x)]',dg1(x),-g1(x),[], [], [], [], [],
qpalg);
%             mu_new = lambda.ineqlin;
%         end
%
        % opt.linesearch switches line search on or off.
```

```
        % You can first set the variable "a" to different
constant values and see how it
        % affects the convergence.
         if opt.linesearch
            [a, w] = lineSearch1(x, s, mu_old,
w,Mot_S,Eng_S,T_dmd,bat_power,Pos_Mot_Trq_Limit,Eng_Trq_Lim
it,Neg_Mot_Trq_Limit,J1,J2);
        else
            a = 0.1;
         end

        % Update the current solution using the step
        dx = a*s;                % Step for x
        x = x + dx;              % Update x using the step

        % Update Hessian using BFGS. Use equations (7.36),
(7.73) and (7.74)
        % Compute y_k
        xkp=x-dx; %previous xk
        y_k = [df1(x,Mot_S,Eng_S,J1,J2) +
mu_new'*dg1(x,Mot_S,Eng_S,T_dmd,bat_power,Pos_Mot_Trq_Limit
,Eng_Trq_Limit,Neg_Mot_Trq_Limit)]' -
[(df1(xkp,Mot_S,Eng_S,J1,J2) +
mu_new'*dg1(xkp,Mot_S,Eng_S,T_dmd,bat_power,Pos_Mot_Trq_Lim
it,Eng_Trq_Limit,Neg_Mot_Trq_Limit))]';

        % Compute theta

         if max((dx'*y_k)-(0.2*dx'*W*dx)) >= 0
            theta = 1;
         else
             theta = (0.8*dx'*W*dx)/((dx'*W*dx) -
(dx'*y_k));
         end

        % Compute  dg_k using y_k, theta, W and dx
         dg_k = [theta;theta].*y_k+ [(1 - theta);(1 -
theta)].*(W*dx);

        % Compute new Hessian using BFGS update formula
         W = W +
((dg_k([1:2],1)*dg_k([1:2],1)')/(dg_k([1:2],1)'*dx([1:2],1)
)) -
(((W*dx([1:2],1))*(W*dx([1:2],1))')/(dx([1:2],1)'*W*dx([1:2
],1)));    % from (6.36)
```

```
        % Update termination criterion:
         gnorm = norm(df1(x,Mot_S,Eng_S,J1,J2) +
(mu_new'*dg1(x,Mot_S,Eng_S,T_dmd,bat_power,Pos_Mot_Trq_Limi
t,Eng_Trq_Limit,Neg_Mot_Trq_Limit))); % norm of Largangian
gradient

        mu_old = mu_new;  % Update mu_old by setting it to
mu_new

        % save current solution to solution.x
%         solution.x = [solution.x, x];
        solution = x;
        ii = ii+1;
        if ii>10
            break
        end
    end
end


function [s, mu0,i] = solveqp1(x,
W,Mot_S,Eng_S,T_dmd,bat_power,Pos_Mot_Trq_Limit,Eng_Trq_Lim
it,Neg_Mot_Trq_Limit,J1,J2)
    % Implement an Active-Set strategy to solve the QP
problem given by
    % min     (1/2)*s'*W*s + c'*s
    % s.t.    A*s-b <= 0
    %
    % where As-b is the linearized active contraint set

    % Strategy should be as follows:
    % 1-) Start with empty working-set
    % 2-) Solve the problem using the working-set
    % 3-) Check the constraints and Lagrange multipliers
    % 4-) If all constraints are staisfied and Lagrange
multipliers are positive, terminate!
    % 5-) If some Lagrange multipliers are negative or
zero, find the most negative one
    %     and remove it from the active set
    % 6-) If some constraints are violated, add the most
violated one to the working set
    % 7-) Go to step 2
    i = 0;

    % Compute c in the QP problem formulation
    c = [df1(x,Mot_S,Eng_S,J1,J2)]';
```

```
    % Compute A in the QP problem formulation using all
constraints
     A0 =
dg1(x,Mot_S,Eng_S,T_dmd,bat_power,Pos_Mot_Trq_Limit,Eng_Trq
_Limit,Neg_Mot_Trq_Limit);

    % Compute b in the QP problem formulation using all
constraints
     b0 = -
g1(x,Mot_S,Eng_S,T_dmd,bat_power,Pos_Mot_Trq_Limit,Eng_Trq_
Limit,Neg_Mot_Trq_Limit);
    mu0 = zeros(size(b0));
    % Initialize variables for active-set strategy
    stop = 0;           % Start with stop = 0
    % Start with empty working-set
    coder.varsize('A',[10 10]);
    coder.varsize('b',[4 4]);
    A = [];          % A for empty working-set
    b = [];          % b for empty working-set
    % Indices of the constraints in the working-set
    coder.varsize('active',[1 4]);
    active = [];     % Indices for empty-working set
    coder.varsize('s',[10 10]);
    s = [];
    kk=1;
    while ~stop  % Continue until stop = 1
        % Initialize all mu as zero and update the mu in
the working set
    mu0 = zeros(size(b0));

        % Extact A corresponding to the working-set from A0
        if ~isempty(active)
         A = A0(active,:);
        % Extract b corresponding to the working-set from
b0
         b = b0(active,:);

        end
        coder.varsize('mu',[10 10]);
        coder.varsize('s',[10 10]);
        % Solve the QP problem given A and b
        [s, mu] = solve_activeset(x, W, c, A, b);
        % Round mu to prevent numerical errors (Keep this)
        mu = round(mu*1e12)/1e12;
```

```
        % Update mu values for the working-set using the
solved mu values
         mu0(active) = mu;

        % Calculate the constraint values using the solved
s values
         gcheck = A0*s - b0;

        % Round constraint values to prevent numerical
errors (Keep this)
        gcheck = round(gcheck*1e12)/1e12;

        % Variable to check if all mu values make sense.
        mucheck = 0;        % Initially set to 0
        coder.varsize('Iadd',[1 4]);
        coder.varsize('Iremove',[1 4]);
        % Indices of the constraints to be added to the
working set
        Iadd = [];              % Initialize as empty
vector
        % Indices of the constraints to be added to the
working set
        Iremove = [];           % Initialize as empty
vector

        if (numel(mu) == 0)
            % When there no mu values in the set
             mucheck = 1;          % OK
        elseif all(min(mu)) > 0
            % When all mu values in the set positive
             mucheck = 1;          % OK
        else
            % When some of the mu are negative
            % Find the most negaitve mu and remove it from
active set
             [~,Iremove] = min(mu);  % Use Iremove to
remove the constraint
             mu(Iremove)=[];
        end
        if gcheck(1)>1e4||gcheck(7)== 0.0289
            i = 1;
            return
        end
        % Check if constraints are satisfied
        if max(gcheck) <= 1e-3
            % If all constraints are satisfied
```

```
            if mucheck == 1
                % If all mu values are OK, terminate by
setting stop = 1
                 stop = 1;
            end
        else
            % If some constraints are violated
            % Find the most violated one and add it to the
working set
             [~,Iadd] = max(gcheck); % Use Iadd to add the
constraint
        end
        % Remove the index Iremove from the working-set
       % Remove the index Iremove from the working-set
        active(Iremove) = [];
        % Add the index Iadd to the working-set
        active = [active,Iadd];

        % Make sure there are no duplications in the
working-set (Keep this)
        active = unique(active);
        kk = kk+1;
        if kk>=4
            break
        end
    end

end


function [s, mu] = solve_activeset(x, W, c, A, b)
    % Given an active set, solve QP

    % Create the linear set of equations given in equation
(7.79)
    if isempty(A)
        M = [W];
        U = [-c];
    else
        M = [W, A'; A zeros(size(A,1))];
        U = [-c ; b];
    end
%     if det(M)==0
%         msg = 'This problem cannot be solved; Matrix is
singular';
%         error(msg)
```

```
%     end
      sol = M\U;           % Solve for s and mu

      s = sol(1:numel(x));                % Extract s from
the solution
      mu = sol(numel(x) + 1:numel(sol));    % Extract mu
from the solution

end

function f1 = f1(x,Mot_S,Eng_S,J1,J2)
    %%% Calculate the objective function f(x)
    x1 = x(1);
    x2 = x(2);
    MotCurvEqu = zeros(1,1);
    EngCurvEqu = zeros(1,1);
%% Motor Curv Fit Equation
if Mot_S<2000
 p00 = .7482;
 p10 = .0001127;
 p01 = -2.801e-05;
 p20 = -1.357e-08;
 p11 = -4.579e-08;
 MotCurvEqu =
p00+p10*Mot_S+p01*x1+p20*(Mot_S^2)+p11*Mot_S*x1;

 E00 = -0.03945;
 E10 = 0.0001066;
 E01 = 0.005797;
 E20 = -7.506e-08;
 E11 = 1.53e-06;
 E02 = -8.147e-05;
 E30 = 1.022e-12;
 E21 = -5.213e-10;
 E12 = 2.042e-09;
 E03 = 4.016e-07;
 E40 = 8.783e-15;
 E31 = -1.687e-13;
 E22 = 3.561e-12;
 E13 = -2.665e-11;
 E04 = -7.142e-10;
 EngCurvEqu =
E00+E10*Eng_S+E01*x2+E20*Eng_S^2+E11*Eng_S*x2+E02*x2^2+E30*
Eng_S^3+E21*Eng_S^2*x2+E12*Eng_S*x2^2+E03*x2^3+E40*Eng_S^4+
E31*Eng_S^3*x2+E22*Eng_S^2*x2^2+E13*Eng_S*x2^3+E04*x2^4;
```

```
elseif (2000<Mot_S)&&(Mot_S<3600)
 p00 = .7482;
 p10 = .0001127;
 p01 = -2.801e-05;
 p20 = -1.357e-08;
 p11 = -4.579e-08;
 MotCurvEqu =
p00+p10*Mot_S+p01*x1+p20*(Mot_S^2)+p11*Mot_S*x1;

 E00 = -0.005662;
 E10 = 8.336e-06;
 E01 = 0.006959;
 E20 = -4.536e-09;
 E11 = -3.399e-07;
 E02 = -8.24e-05;
 E30 = 1.082e-12;
 E21 = 8.313e-11;
 E12 = 2.316e-09;
 E03 = 4.387e-07;
 E40 = -9.548e-17;
 E31 = -1.531e-14;
 E22 = 6.464e-14;
 E13 = -8.357e-12;
 E04 = -8.432e-10;
 EngCurvEqu =
E00+E10*Eng_S+E01*x2+E20*Eng_S^2+E11*Eng_S*x2+E02*x2^2+E30*
Eng_S^3+E21*Eng_S^2*x2+E12*Eng_S*x2^2+E03*x2^3+E40*Eng_S^4+
E31*Eng_S^3*x2+E22*Eng_S^2*x2^2+E13*Eng_S*x2^3+E04*x2^4;

elseif (3600<Mot_S)&&(Mot_S<5200)
 p00 = .7482;
 p10 = .0001127;
 p01 = -2.801e-05;
 p20 = -1.357e-08;
 p11 = -4.579e-08;
 MotCurvEqu =
p00+p10*Mot_S+p01*x1+p20*(Mot_S^2)+p11*Mot_S*x1;

 E00 = 3.136;
 E10 = -0.003027;
 E01 = 0.02692;
 E20 = 1.087e-06;
 E11 = -1.395e-05;
 E02 = -6.623e-05;
 E30 = -1.721e-10;
 E21 = 3.061e-09;
```

```
 E12 = 1.626e-09;
 E03 = 2.934e-07;
 E40 = 1.013e-14;
 E31 = -2.27e-13;
 E22 = -4.334e-14;
 E13 = 1.688e-12;
 E04 = -5.679e-10;
 EngCurvEqu =
E00+E10*Eng_S+E01*x2+E20*Eng_S^2+E11*Eng_S*x2+E02*x2^2+E30*
Eng_S^3+E21*Eng_S^2*x2+E12*Eng_S*x2^2+E03*x2^3+E40*Eng_S^4+
E31*Eng_S^3*x2+E22*Eng_S^2*x2^2+E13*Eng_S*x2^3+E04*x2^4;

elseif (5200<Mot_S)&&(Mot_S<6800)
 p00 = .7482;
 p10 = .0001127;
 p01 = -2.801e-05;
 p20 = -1.357e-08;
 p11 = -4.579e-08;
 MotCurvEqu =
p00+p10*Mot_S+p01*x1+p20*(Mot_S^2)+p11*Mot_S*x1;

 E00 = -10.54;
 E10 = 0.007327;
 E01 = -0.03075;
 E20 = -1.904e-06;
 E11 = 1.783e-05;
 E02 = -2.206e-05;
 E30 = 2.193e-10;
 E21 = -2.906e-09;
 E12 = -1.18e-08;
 E03 = 2.696e-07;
 E40 = -9.442e-15;
 E31 = 1.535e-13;
 E22 = 1.154e-12;
 E13 = -3.512e-12;
 E04 = -4.494e-10;
 EngCurvEqu =
E00+E10*Eng_S+E01*x2+E20*Eng_S^2+E11*Eng_S*x2+E02*x2^2+E30*
Eng_S^3+E21*Eng_S^2*x2+E12*Eng_S*x2^2+E03*x2^3+E40*Eng_S^4+
E31*Eng_S^3*x2+E22*Eng_S^2*x2^2+E13*Eng_S*x2^3+E04*x2^4;

end
%% Objective Function

f1= ((J1)*(1-MotCurvEqu))+((J2)*(1-EngCurvEqu));
```

```
%f1 = -
((MotCurvEqu/MotEffMax)*(MotCurvEqu))+((EngCurvEqu/EngEffMa
x)*(EngCurvEqu));

end

function df1 = df1(x,Mot_S,Eng_S,J1,J2)
    %%% Calculate the gradient of the objective (row
vector)
    %%% df(x)/dx = [df/dx1, df/dx2, ..., df/xn]
    x1 = x(1);
    x2 = x(2);
    df12 = zeros(1,1);

if (Mot_S<2000)

 E00 = -0.03945;
 E10 = 0.0001066;
 E01 = 0.005797;
 E20 = -7.506e-08;
 E11 = 1.53e-06;
 E02 = -8.147e-05;
 E30 = 1.022e-12;
 E21 = -5.213e-10;
 E12 = 2.042e-09;
 E03 = 4.016e-07;
 E40 = 8.783e-15;
 E31 = -1.687e-13;
 E22 = 3.561e-12;
 E13 = -2.665e-11;
 E04 = -7.142e-10;
df12 = - 1.0*E31*Eng_S^3 - 2.0*E22*Eng_S^2*x2 -
1.0*E21*Eng_S^2 - 3.0*E13*Eng_S*x2^2 - 2.0*E12*Eng_S*x2 -
1.0*E11*Eng_S - 4.0*E04*x2^3 - 3.0*E03*x2^2 - 2.0*E02*x2 -
1.0*E01;

elseif (2000<Mot_S)&&(Mot_S<3600)

 E00 = -0.005662;
 E10 = 8.336e-06;
 E01 = 0.006959;
 E20 = -4.536e-09;
 E11 = -3.399e-07;
 E02 = -8.24e-05;
 E30 = 1.082e-12;
 E21 = 8.313e-11;
```

```
 E12 = 2.316e-09;
 E03 = 4.387e-07;
 E40 = -9.548e-17;
 E31 = -1.531e-14;
 E22 = 6.464e-14;
 E13 = -8.357e-12;
 E04 = -8.432e-10;
df12 = - 1.0*E31*Eng_S^3 - 2.0*E22*Eng_S^2*x2 -
1.0*E21*Eng_S^2 - 3.0*E13*Eng_S*x2^2 - 2.0*E12*Eng_S*x2 -
1.0*E11*Eng_S - 4.0*E04*x2^3 - 3.0*E03*x2^2 - 2.0*E02*x2 -
1.0*E01;

elseif (3600<Mot_S)&&(Mot_S<5200)

 E00 = 3.136;
 E10 = -0.003027;
 E01 = 0.02692;
 E20 = 1.087e-06;
 E11 = -1.395e-05;
 E02 = -6.623e-05;
 E30 = -1.721e-10;
 E21 = 3.061e-09;
 E12 = 1.626e-09;
 E03 = 2.934e-07;
 E40 = 1.013e-14;
 E31 = -2.27e-13;
 E22 = -4.334e-14;
 E13 = 1.688e-12;
 E04 = -5.679e-10;
df12 = - 1.0*E31*Eng_S^3 - 2.0*E22*Eng_S^2*x2 -
1.0*E21*Eng_S^2 - 3.0*E13*Eng_S*x2^2 - 2.0*E12*Eng_S*x2 -
1.0*E11*Eng_S - 4.0*E04*x2^3 - 3.0*E03*x2^2 - 2.0*E02*x2 -
1.0*E01;

elseif (5200<Mot_S)&&(Mot_S<6800)

 E00 = -10.54;
 E10 = 0.007327;
 E01 = -0.03075;
 E20 = -1.904e-06;
 E11 = 1.783e-05;
 E02 = -2.206e-05;
 E30 = 2.193e-10;
 E21 = -2.906e-09;
 E12 = -1.18e-08;
 E03 = 2.696e-07;
```

```
 E40 = -9.442e-15;
 E31 = 1.535e-13;
 E22 = 1.154e-12;
 E13 = -3.512e-12;
 E04 = -4.494e-10;
df12 = - 1.0*E31*Eng_S^3 - 2.0*E22*Eng_S^2*x2 -
1.0*E21*Eng_S^2 - 3.0*E13*Eng_S*x2^2 - 2.0*E12*Eng_S*x2 -
1.0*E11*Eng_S - 4.0*E04*x2^3 - 3.0*E03*x2^2 - 2.0*E02*x2 -
1.0*E01;
end

p00 = .7482;
p10 = .0001127;
p01 = -2.801e-05;
p20 = -1.357e-08;
p11 = -4.579e-08;

df11 = - 1.0*p01 - 1.0*Mot_S*p11;

df1 = [df11,df12];

end
function g1 =
g1(x,Mot_S,Eng_S,T_dmd,bat_power,Pos_Mot_Trq_Limit,Eng_Trq_
Limit,Neg_Mot_Trq_Limit)
    %%% Calculate the constraints (column vector)
    %%% g(x) = [g1(x); g2(x); ... ; gm(x)]
    x1 = x(1);
    x2 = x(2);

    g11= (x1*(Mot_S*(2*pi/60)))-bat_power;
    g12= -x1+Neg_Mot_Trq_Limit;
    g13= x1+x2-T_dmd;
    g14= x1-Pos_Mot_Trq_Limit;
    g15= x2-Eng_Trq_Limit;
    g16= -x2;
    g17= -x1-x2+T_dmd;

    g1 =[g11;g12;g13;g14;g15;g16;g17];
end
function dg1 =
dg1(x,Mot_S,Eng_S,T_dmd,bat_power,Pos_Mot_Trq_Limit,Eng_Trq
_Limit,Neg_Mot_Trq_Limit)
    %%% Calculate the gradient of the constraints
    %%% dg(x)/dx = [dg1/dx1, dg1/dx2, ... , dg1/dxn;
    %%%             dg2/dx1, dg2/dx2, ... , dg2/dxn;
```

```
    %%%              ...
    %%%              dgm/dx1, dgm/dx2, ... , dgm/dxn]
    x1 = x(1);
    x2 = x(2);

    dg1 = [(Mot_S*(2*pi/60)) 0;-1 0;1 1;1 0;0 1;0 -1;-1 -
1];

end
% Armijo line search
function [a, w] = lineSearch1(x, s, mu_old,
w_old,Mot_S,Eng_S,T_dmd,bat_power,Pos_Mot_Trq_Limit,Eng_Trq
_Limit,Neg_Mot_Trq_Limit,J1,J2)
    t = 0.1; % scale factor on current gradient: [0.01,
0.3]
    b = 0.8; % scale factor on backtracking: [0.1, 0.8]
    a = 1; % maximum step length

    D = s;                    % direction for x

    % Calculate weights in the merit function using eaution
(7.77)
     w = max(abs(mu_old) , 0.5*(w_old+abs(mu_old)));
    % terminate if line search takes too long
    count = 0;
    while count<100
        % Calculate phi(alpha) using merit function in
(7.76)
        x_updated=x + a*D;
         phi_a = f1(x_updated,Mot_S,Eng_S,J1,J2)+
w'*abs(min(0 , -
g1(x_updated,Mot_S,Eng_S,T_dmd,bat_power,Pos_Mot_Trq_Limit,
Eng_Trq_Limit,Neg_Mot_Trq_Limit)));

        % Caluclate psi(alpha) in the line search using
phi(alpha)
        phi0 = f1(x,Mot_S,Eng_S,J1,J2)+ w'*abs(min(0 , -
g1(x,Mot_S,Eng_S,T_dmd,bat_power,Pos_Mot_Trq_Limit,Eng_Trq_
Limit,Neg_Mot_Trq_Limit)));                    % phi(0)
        dphi0 = df1(x,Mot_S,Eng_S,J1,J2)*s +
w'*((dg1(x,Mot_S,Eng_S,T_dmd,bat_power,Pos_Mot_Trq_Limit,En
g_Trq_Limit,Neg_Mot_Trq_Limit)*s).*(g1(x,Mot_S,Eng_S,T_dmd,
bat_power,Pos_Mot_Trq_Limit,Eng_Trq_Limit,Neg_Mot_Trq_Limit
)>0));                    % phi'(0)
        psi_a = phi0 + t*a*dphi0;                    %
psi(alpha) = phi(0)+t*alpha*phi'(0)
```

```
        % stop if condition satisfied
         stop = phi_a <= psi_a; %linear convergince of the
penalty function
        if all(stop);
            break;
        else
            % backtracking
            a = a*b;
            count = count + 1;
        end
    end
end
```