Neuro-Dynamic Programming and Reinforcement Learning for Optimal Energy Management of a Series Hydraulic Hybrid Vehicle Considering Engine Transient Emissions

by

Rajit Johri

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Mechanical Engineering)
in The University of Michigan
2011

Doctoral Committee:

Research Professor Zoran S. Filipi, Chair
Professor Jessy W. Grizzle
Professor Huei Peng
Associate Research Scientist John W. Hoard
Assistant Professor Domitilla Del Vecchio, Massachusetts Institute of Technology

कर्मण्येवाधिकारस्ते मा फलेषु कदाचन ।
मा कर्मफलहेतुर्भूर् मा ते संगोऽस्त्वकर्मणि ॥४७॥

श्रीकृष्ण
श्रीमद्भगवद्गीता

“You have a right to perform your prescribed duty, but you are not entitled to the fruits of action. Never consider yourself the cause of the results of your activities, and never be attached to not doing your duty.”

Shri Krishna
Bhagavad Gita (Translation)

To my parents

ACKNOWLEDGMENTS

This dissertation was made possible due to the support, guidance and encouragement of many people. Their contributions, whether direct or indirect, are most appreciated. First and foremost, I would like to thank Professor Zoran Filipi for being an outstanding advisor. His unique ability to "see the big picture," connect concepts and contextualize results has been instrumental in helping me grow as a student, as a researcher and as a human being. I appreciate him for affording me a tremendous amount of freedom and responsibility. More than anything else, Prof Filipi always believed in me.

I am thankful of my committee members Professor Jessy Grizzle, Professor Huei Peng, Research Scientist John Hoard and Assistant Professor Domitilla Del Vecchio for providing their critical feedback. I truly appreciate their time and effort in reviewing this work.

I would like to acknowledge my funding agencies, State of Michigan 21st Century Jobs Fund and Bosch Rexroth for their generous support of my work over the years. In particular, I would like to thank Ed Grief and Simon Baseley, both at Bosch Rexroth, for their technical help and ensuring this research is relevant to the industry.

The W. E. Lay Automotive Laboratory at the University of Michigan was a pleasant and intellectually stimulating workplace, owing to the high caliber and intellect of my peers. Though it is impossible to thank everybody individually, Fernando Tavares and Ashwin Salvi deserve a special mention. Without their help, constant feedback and constructive criticism, this work would not have been possible. The long nights in the test cell, friendly arguments on topics ranging from research to sports to culture will be fondly missed. Thank you for making my experience in the group so memorable and rewarding. Finally, I would like to thank all my lab mates and staff, past and present.

When I first arrived in Ann Arbor, I never imagined I would end up staying here as long as I did. But my time here has been nothing short of extraordinary, not only due to

all the knowledge I have gained over the years but also because of the number of outstanding people I have met. Several friends deserve special recognition for their friendship and support during my doctoral studies. I thank Rahul Ahlawat for being an exceptional friend since the days we were in undergraduate studies together and for encouraging me to pursue higher studies. I thank Natasha Sant for continuing to be a great friend irrespective of our geographical distances. I thank Neha Kaul for always being there when I needed help. I thank Sidharth Misra and Awlok Josan with whom I have shared many wonderful moments and developed lasting friendships over my years in Ann Arbor.

I express deep gratitude towards my uncle Navin Sinha, my aunt Ira Sinha, and my cousins Bharat and Arjun. Without their constant support and encouragement, I doubt I would have completed even my first semester of graduate studies. They were with me when I needed my family the most. Bharat was instrumental in me choosing the University of Michigan over other schools. I am fortunate I heeded his advice.

Finally, I would like to thank my parents, Gp. Capt. Arun Kumar Johri and Ila Johri, for their unconditional love, support and unwavering belief in me. They gave me all the opportunities in the world, the freedom to pursue the ones I desired, and the encouragement to follow through. Last but not the least, I thank my brother Rohit and my sister-in-law Payal who I'm fortunate to have in my life.

GO BLUE!

Rajit Johri
Ann Arbor

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF APPENDICES

ABSTRACT

Neuro-Dynamic Programming and Reinforcement Learning for Optimal Energy Management of a Series Hydraulic Hybrid Vehicle Considering Engine Transient Emissions

by

Rajit Johri

Chair: Zoran S. Filipi

Sequential decision problems under uncertainty are encountered in various fields such as optimal control and operations research. In this dissertation, a framework for solving policy optimization problems with multiple objectives and large design state space is developed based on Neuro-Dynamic Programming (NDP) and Reinforcement Learning (RL). The new algorithms are then used to create an intelligent supervisory controller for hybrid vehicle with an emphasis on bridging the gap between theoretical and applied work. Dynamic Programming (DP) is well suited for determining an optimal solution for constrained nonlinear model based systems. However, DP suffers from curse of dimensionality i.e. computational effort grows exponentially with state space. The new algorithms address this problem and enable practical application of DP to a much broader range of problems.

The power management problem for a hybrid vehicle can be formulated as an infinite time horizon stochastic sequential decision-making problem. In the past, policy optimization has been applied successfully to design optimal supervisory controller for best fuel economy. Static emissions have been considered too but engine research has shown that transient operation can have significant impact on real-world emissions. Modeling transient emissions results in addition of more states. Therefore, the problem with multiple objectives i.e. minimize fuel consumption and transient particulate and

NO_X emissions, becomes computationally intractable by conventional DP. Availability of predictive but fast emissions model is another challenge.

In this dissertation, a self-learning approach to develop optimal power management with multiple objectives, e.g. to minimize fuel consumption and transient engine-out particulate matter and NO_X emission, for a series hydraulic hybrid vehicle is proposed. The self-learning neural controller is based on the fundamental principles of NDP and RL. The other significant contribution is development of fast emission sensors, which are capable of predicting in real-time engine-out transient particulate and NO_X emissions.

The self-learning controller starts "naïve" i.e. with no knowledge on how to control the onboard power sources. The controller learns from its interactions with the environment and improves its performance over time. The controller tries to minimize multiple objectives and continues to evolve until a global solution is achieved. This novel approach enables real time implementation of controller for problems with large state space.

Diesel engine combustion and emission formation is highly nonlinear and thus creates a challenge related to engine control with emission feedback. The emission models developed in this dissertation belong to the family of hierarchical models, namely "neuro-fuzzy model tree". The approach is based on divide-and-conquer strategy i.e. to divide a complex problem into multiple simpler subproblems, which can then be identified using simpler class of models. Advanced experimental setup incorporating a medium duty diesel engine is used to generate training data. The engine is characterized using specifically design perturbation signal, and fast emission analyzers for particulate matter and NO_X provide instantaneous engine-out emissions.

Finally, the supervisory controller along with virtual emission sensors for particulate and NO_X is implemented and evaluated using the Engine-In-the-Loop (EIL) setup. EIL is a unique facility to systematically evaluate control methodologies through concurrent running of real engine and a virtual hybrid powertrain. The EIL facility uses fast instruments and emission analyzers to investigate how critical in-vehicle transients affect engine and powertrain response. Given the nonlinear system dynamics, it is critical for considering real-world conditions and developing implementable strategies.

The series hydraulic hybrid vehicle with power management controller designed using stochastic dynamic programming (SDP) results in 65% improvement in fuel economy over conventional vehicle during federal urban driving schedule (FUDS). The NDP based controller results in additional 16% reduction in particulate and 38% reduction in NO_X, with a 0.7% fuel economy penalty compared to SDP as a baseline.

Chapter 1

INTRODUCTION

1.1 Energy Crisis and Climate Change

Dwindling oil reserves and ever increasing demand for energy has resulted in a push for alternate energy resources and efficient utilization of fossil resources. The world energy consumption is steadily increasing with estimated increase of 49% from 495 quadrillion Btu in 2007 to 739 quadrillion Btu in 2035 [1] (Figure 1.1). Developing countries like China and India are driving the energy demand growth as these economics develop further, putting even more strain on natural resources (Figure 1.1 – Non-OECD countries). One of the biggest consumers of energy is the transportation sector. The economic activity and population are fueling the increased demand for energy in the transportation sector. Transportation sector accounts for 27% of the world energy consumption and will continue to grow by projected 1.3% per year from 2007 to 2035. The energy outlook of the US is not that different from the world with the transportation sector accounting similar 27% of energy consumption [2], Figure 1.2.

Transportation sector rely almost exclusively on liquid petroleum fuels as the energy source due to their higher volumetric energy density and convenience of use. To reduce the dependence of US on imported fuel and dependence of transportation sector on fossil fuels, US Congress enacted Corporate Average Fuel Economy (CAFE) in 1975. The National Highway Traffic Safety Administration (NHTSA) oversees CAFE regulations and sets fuel economy standards for passenger cars and light trucks in US. The CAFE standards, however, have changed very slightly in last two decades and have not kept pace with growth in transportation sector. To overhaul CAFE, Energy Independence and Security Act (EISA) was introduced and will result in fuel economy increase by 40% by 2020.

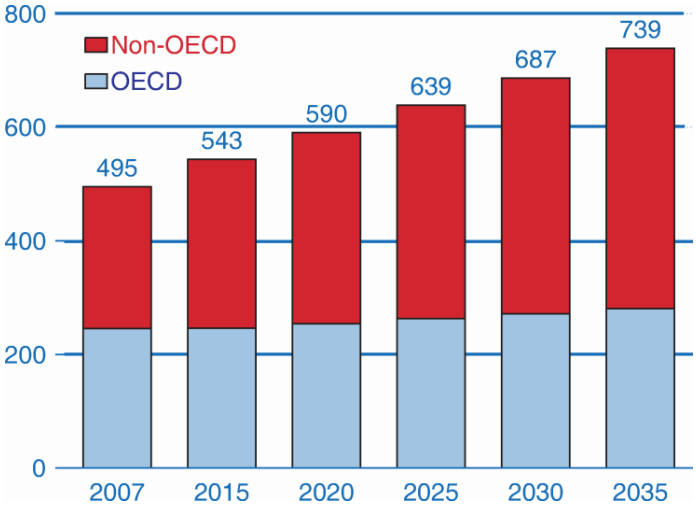


Figure 1.1: World marketed energy consumption (quadrillion BTU) [1].

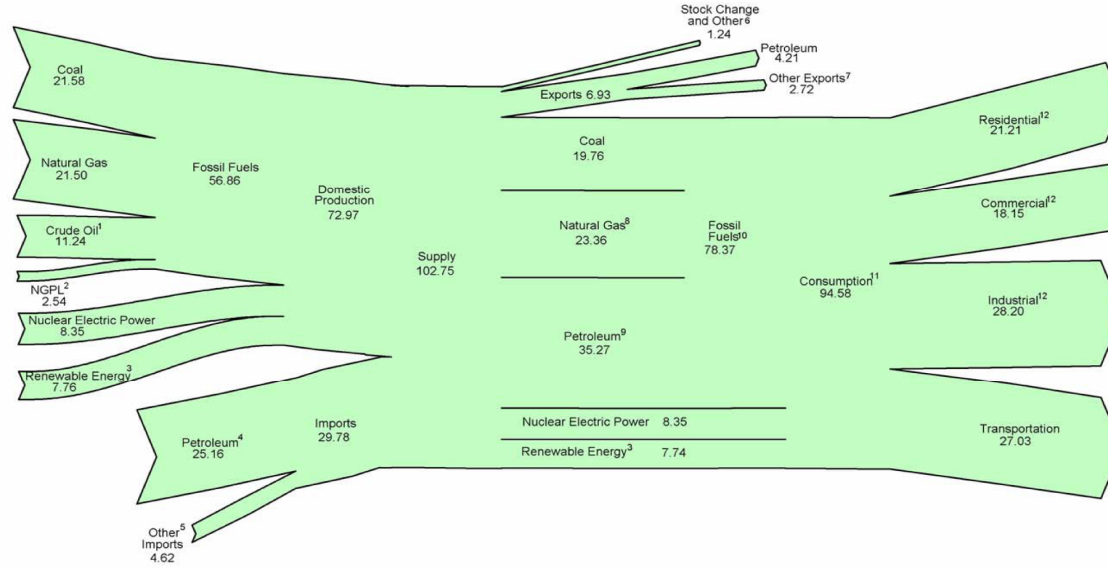


Figure 1.2: Energy flow, 2009 (Quadrillion BTU) [2].

The transportation sector can be further divided into air, marine, rail, light-duty and heavy-duty vehicles. It can be seen from Figure 1.3 that the projected increase in energy consumption in the future will be from the heavy-duty vehicles [3] and novel efficient technologies will be required to reduce the overall fossil fuel consumption and meet newer regulations. Until recently, the fuel economy of heavy-duty trucks was unregulated and left up to the market forces. This is set to change with the recent Presidential memorandum launching a joint Environmental Protection Agency (EPA) and NHTSA

effort to establish the fuel efficiency and greenhouse gas emissions standards for medium and heavy-duty vehicles beginning with model year 2014 [4].

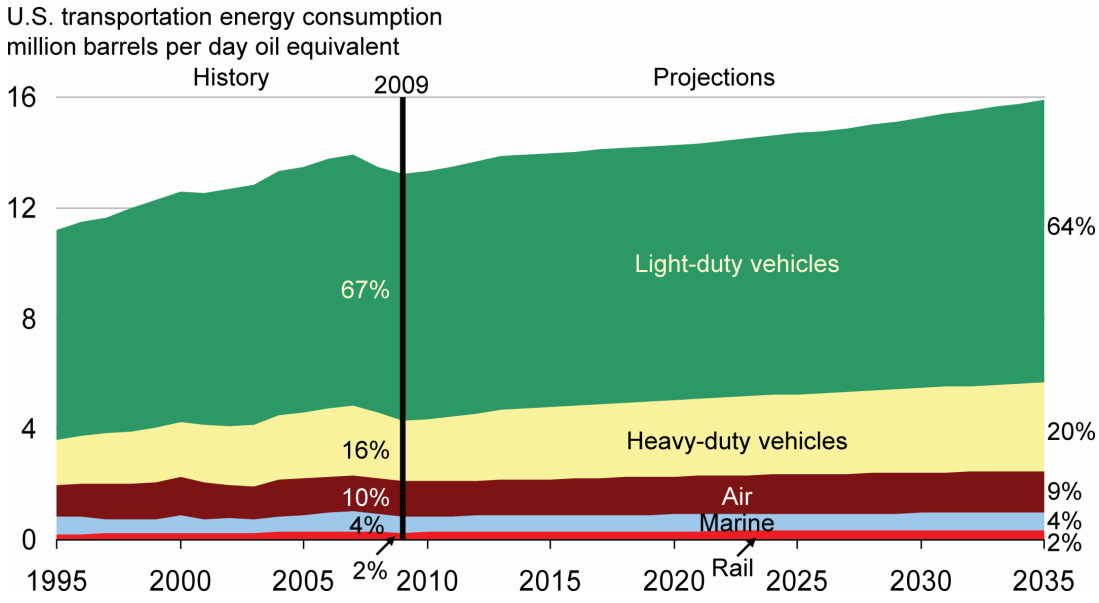


Figure 1.3: Projected increase of energy consumption in transportation sector [3].

Concurrent with implementation of increasingly stringent fuel economy regulations is the adoption of the emission standards. Transportation sector is the single biggest contributing factor to air pollution. To reduce the impact of transportation sector on air quality, tailpipe emission standards are becoming ever more stringent. EPA regulates emission standards in US and was founded in 1970 under the Clean Air Act to set statutory limits on emissions. The Clean Air Act Amendments of 1990 imposed two sets of standards, Tier 1 and Tier 2 for light-duty non-commercial vehicles. The Tier 1 regulations were phased in progressively between 1994 and 1997 and, limited the allowable levels of unburnt hydrocarbon (HC), carbon monoxide (CO), nitrogen oxides (NO_X) and particulate matter (PM). The Tier 2 regulations were introduced in 2004 and phased in through 2009. In addition to the pollutants regulated by Tier 1, Tier 2 also defined limits on aldehydes and non-methane organic gases. The Tier 2 regulations were significantly stricter to Tier 1, decreasing the tail pipe emissions drastically as seen from Figure 1.4. Similar to US standards, other countries like Europe, Japan, and India have their own emission standards, which regulate the amount of harmful species in exhaust. In order for newer vehicles to meet these ever-tightening regulations, multi-pronged

approach with newer carbon-neutral energy sources, better powertrain design and use of lighter materials for construction of vehicles will be required.

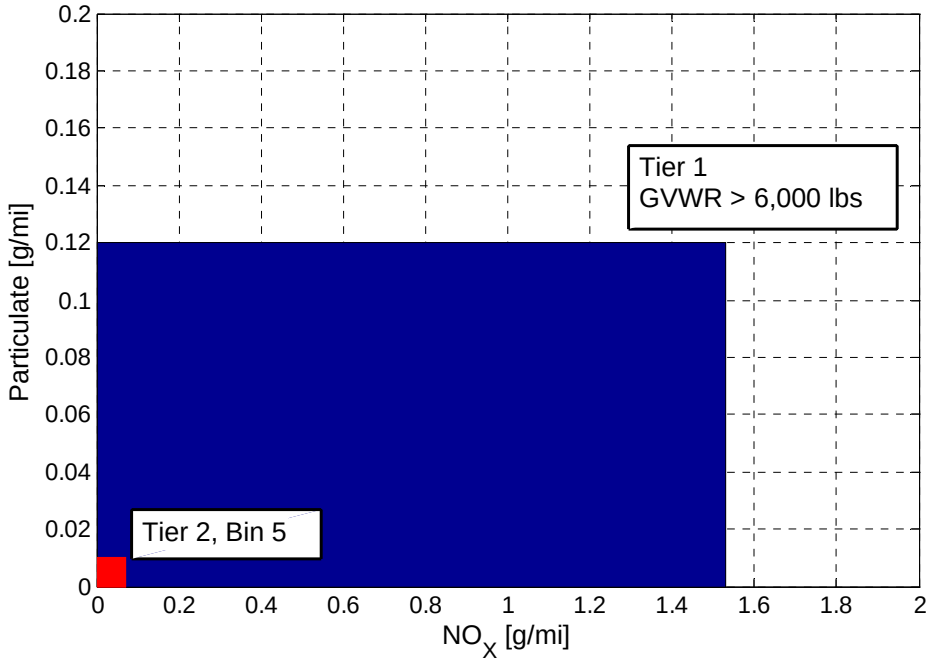


Figure 1.4: EPA NO_X and particulate matter regulation trends.

1.2 Pathways to Better Fuel Economy and Low Emissions

Development of modern vehicles is driven by the need to address the energy security and climate change with increased fuel economy, while simultaneously meeting strict exhaust emission regulations. Vehicles in future will use renewable and clean source of energy but in near future, petroleum fuels will still power vehicles.

Various engine based technologies like variable valve timing, cylinder deactivation, turbocharging etc. are being introduced to increase the efficiency of internal combustion engine and simultaneously decrease emissions. Fuel economy has a very strong dependency on vehicle mass and aerodynamic drag [5]. Significant breakthroughs in cost-effective lightweight materials and manufacturing processes have led to reduction in vehicle weight without compromising performance, cost and safety. Substituting lightweight, high-strength materials such as aluminum, magnesium, advanced high-strength steels, and fiber-reinforced composites for mild steel in vehicle applications have

a positive impact on fuel economy and emissions. Application of computer-aided engineering (CAE) have ushered in newer cars with improved aerodynamics thereby further improving fuel economy. These efforts nonetheless have been effective but are not enough to meet stringent regulations. One solution to meet such intense challenges is through hybridization. Hybridization enables significant leaps in fuel economy improvements, Figure 1.5 by optimizing engine operation and additionally providing opportunity to recuperate kinetic energy during braking.

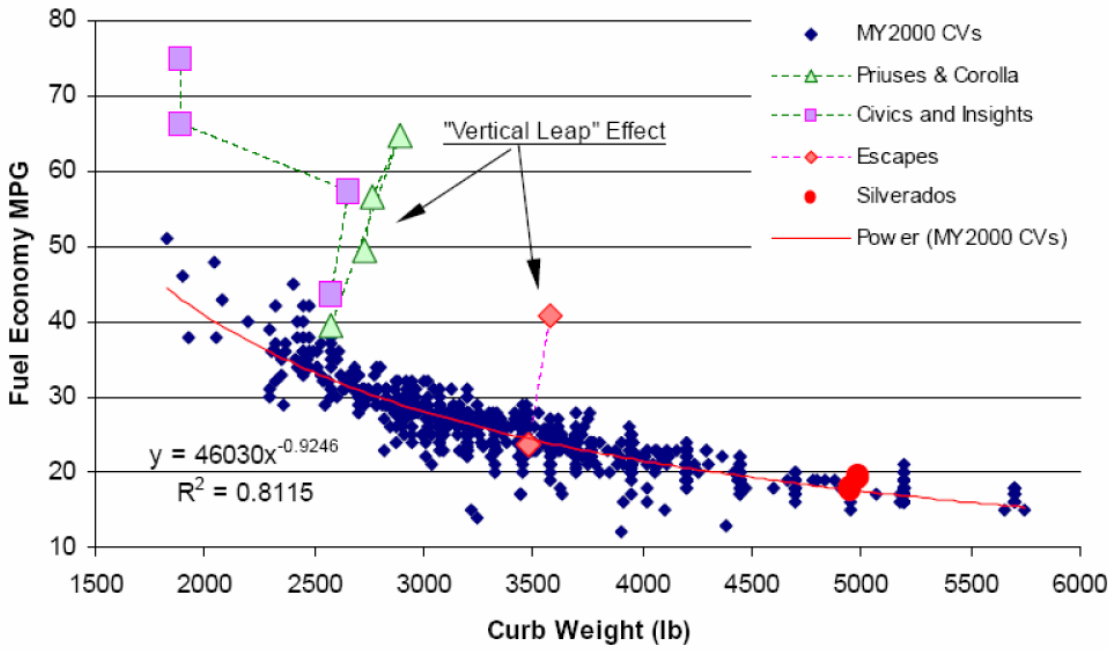


Figure 1.5: Vertical leap in fuel economy improvement with hybridization [5].

Most of the industry efforts are presently devoted to passenger vehicle market. However, the light-duty and heavy-duty vehicles account for 23 percent [4] of the total greenhouse gas production from the transportation sector. Furthermore, based on projections for energy consumption (Figure 1.3), trucks' consumption of petroleum fuel will continue to increase. Therefore, any proposed plan for addressing fuel economy and emission regulation need to include trucks as a part of the overall strategy.

Addressing fuel economy and emissions for trucks has more profound impact on global energy crisis than passenger vehicles. Trucks spend a lot more time on road compared to the passenger vehicles and their annual fuel consumption per vehicle is very large. However, scope for improving fuel economy is very limited in trucks. They already

employ very efficient diesel engines. Other methods like lightweight structures for truck body design are not usually considered to improve fuel economy as the payload dictates truck weight. There is also a limit to reducing aerodynamic losses in a truck. The only way to get vertical leap in fuel economy benefits is through hybridization. The large mass of trucks implies large amount of kinetic energy can be recuperated from regenerative braking. Many carmakers are already adding hybrid vehicles to their existing line-ups of conventional vehicle. To the contrary, the truck market is largely untapped and yet it offers a chance for huge impacts.

1.2.1 Hybrid Powertrain

Hybrid powertrains have an additional source of energy onboard the vehicle like battery, accumulator or ultra-capacitor and a secondary prime mover like electric motor/generator or hydraulic pump/motor. Efficient management of this secondary source of energy provides additional degree of freedom in operation of the engine, and the whole powertrain can be designed to improve fuel economy by the possibility of (i) downsizing the engine, (ii) recovering energy during braking event by regeneration, (iii) optimizing engine operation and, (iv) engine shutdowns.

Hybrids can be classified either on the basis of powertrain architecture i.e. connection of primary and secondary source of power to wheel, Figure 1.6 or the secondary power source onboard vehicle.

1.2.1.1 Classification based on energy source

There are many different energy storage concepts being researched for hybrid application like electric (battery, ultra-capacitor), hydraulic, pneumatic and inertial (flywheel). Electric hybrids have received most of the attention due to their successful application in passenger vehicles (Toyota Prius, Honda Insight, and Chevrolet Volt). Hydraulic hybrids hold a key advantage for future truck powertrain design. Utilization of hydraulic propulsion and energy storage components can offer significant advantage. Previous work done by Filipi et al. [6], [7], [8] have shown hydraulics to be well suited for truck application due to their higher power density and higher energy conversion efficiency.

1.2.1.2 Classification based on powertrain configuration

There are multiple hybrid powertrain configurations in literature based on the location of secondary power source and the underlying powertrain architecture. However, they all can be categorized under 3 broad classification, namely parallel, series and powersplit.

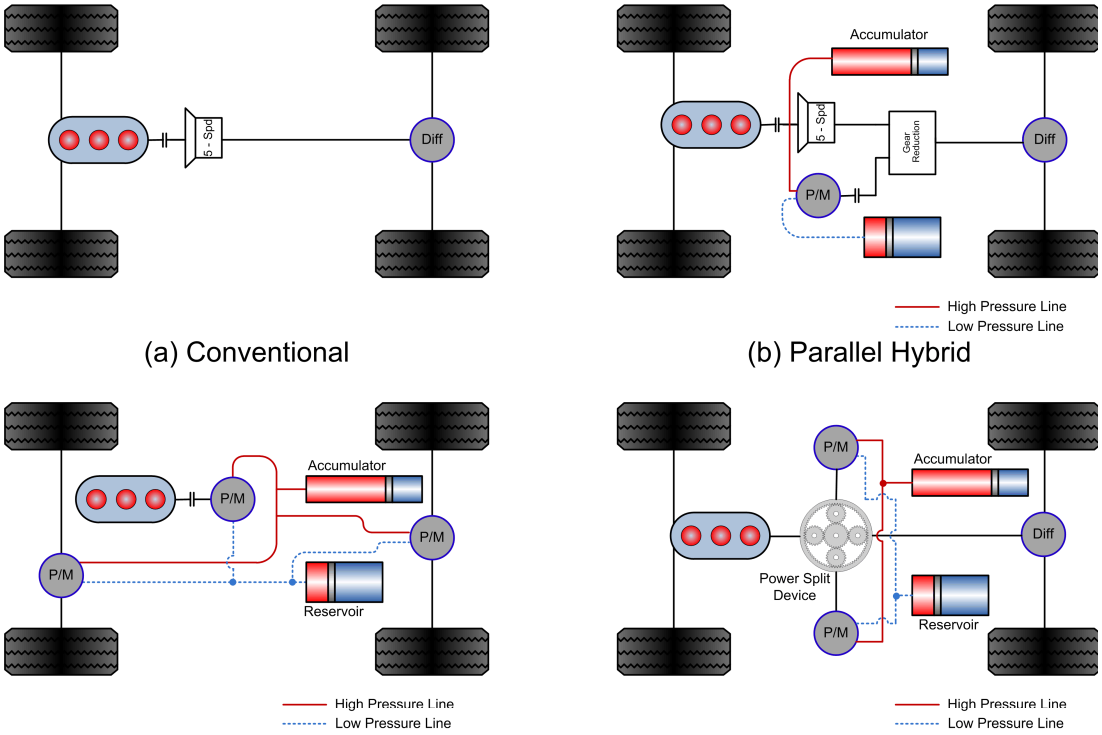


Figure 1.6: Hydraulic hybrid powertrain architectures.

Parallel hybrid is also known as mild hybrid or power assist hybrid due to small power capacity of secondary energy source. They can further be classified as post or pre transmission based on the location of pump/motor system. The secondary power source can only drive the vehicle for a very short distance and is primarily used for launch assist, optimizing engine transient operation and regenerative braking. The engine operating point optimization is limited due to mechanical coupling between wheels and engine through transmission. Parallel systems have an advantage that they can be retrofitted to existing vehicles.

Series hybrid is also known as full hybrid because all the propulsion is provided by the pump/motor. There exists no direct mechanical linkage between the engine and the wheels. Series hybrid provides greater flexibility in operating the engine, as vehicle states do not directly affect the engine. However, the soft connection between primary source of power i.e. engine and wheels means the mechanical energy at engine is always converted to electrical/hydraulic energy at generator/pump and then converted back to mechanical energy by motor. Thus, the overall efficiency of the system depends on the conversion efficiency of individual devices.

Powersplit hybrid has the characteristics of both parallel and series systems. The power can be routed to the wheels from both the engine and the secondary power source. At heart of powersplit hybrids is a power split (planetary gear set) device. The overall nature of powertrain depends upon where pump, engine, motor and vehicle are connected to different nodes of powersplit device.

1.2.2 Diesel Engine

Diesel engines are proven and available technology that is inherently more efficient than their counterpart gasoline engines by 20-40%. Diesel engine lower fuel consumption is attributed to its overall lean combustion, low pumping loses and higher compression ratios. The diesel engine combustion process is heterogeneous in nature with parts of combustion taking place in oxygen deficit zone and other parts in oxygen rich zones. The complex process involved, such as turbulent mixing, three-dimensional heterogeneity coupled with presence of high localized temperatures and high in-cylinder pressures, leads to conditions that are favorable for formation of nitrogen oxides and particulate matter. To meet EPA norms, modern diesel engines employ many actuators like variable geometry turbochargers (VGT), exhaust gas recirculation (EGR) etc. along with exhaust aftertreatment systems. This increases the complexity, cost and controls challenge for a modern diesel engine. However, emission aftertreatment is still a necessity.

The aftertreatment technologies for diesel are costlier compared to gasoline engine. Some components need regeneration e.g. lean NO_X trap and diesel particulate filter. This causes additional fuel economy penalty. Novel methods are being researched but none holds promise in near future. Significant research is being undertaken to reduce the

engine-out emissions from the diesel engine. These efforts include in-cylinder combustion, use of alternate fuels, development of newer aftertreatment technologies, and the development of novel combustion modes. Typically these experimental and simulation studies are done at steady state conditions. These efforts undoubtedly are useful, but since engine operation is transient most of the time in actual real world driving, transient emission understanding is of great importance.

Recently researchers [9], [10], [11], [12], [13] have evaluated the effect of transients on diesel engine emissions. Samulski and Jackson [9] showed particulate emissions from diesel engines are very sensitive to transient operation and reported an average increase of 47% over steady state. Rakopoulos et al. [13] used two-zone transient diesel engine thermodynamic model to study the effect of load and engine parameters on transient emissions and noted a significant increase in soot production with step change in load. Based on advanced experimentation for testing under highly dynamics conditions, Hagena et al. [10], [14] concluded that transient soot emissions can account for almost half of the total soot emission when engine is operated over a realistic driving schedule. Steady state map based models fail to capture the transient nature of emission when engine is operated transiently and underestimate soot production. This can be seen from Figure 1.7 as the integrated area under the transient trace is much higher compared to the quasi-steady state curve. The transient spike is higher and precedes the prediction by quasi-steady state.

To explain the incapability of steady state models to predict transients correctly, we need to look closely into engine operation during transients. During change in engine load, the engine command changes nearly instantaneously and the fuel injected follows the demand. The intake manifold pressure lags due to turbocharger inertia and the delay in boost pressure results in lower in-cylinder air-to-fuel ratio, limiting the amount of fuel that can be burned. The engine ECU monitors the rise in intake manifold pressure and limits the fuel injection to prevent “smoke”, however the instantaneous air-fuel ratio can still display excursion below the steady state values. Other actuator dynamics such as EGR valve also play an important role in deviation of in-cylinder chemistry from steady state and transient emission formation. The residual dynamics have slower time scales and it takes time to purge the intake manifold after the EGR command is changed to

"zero". Also, Filipi et al. [10] showed that step change of load results in increased exhaust backpressure to inlet manifold pressure thereby increasing the internal residual. The presence of residual helps in reduction of NO_X but results in higher particulate matter production. The steady state emission model is a function of engine load and hence cannot capture the transient effects.

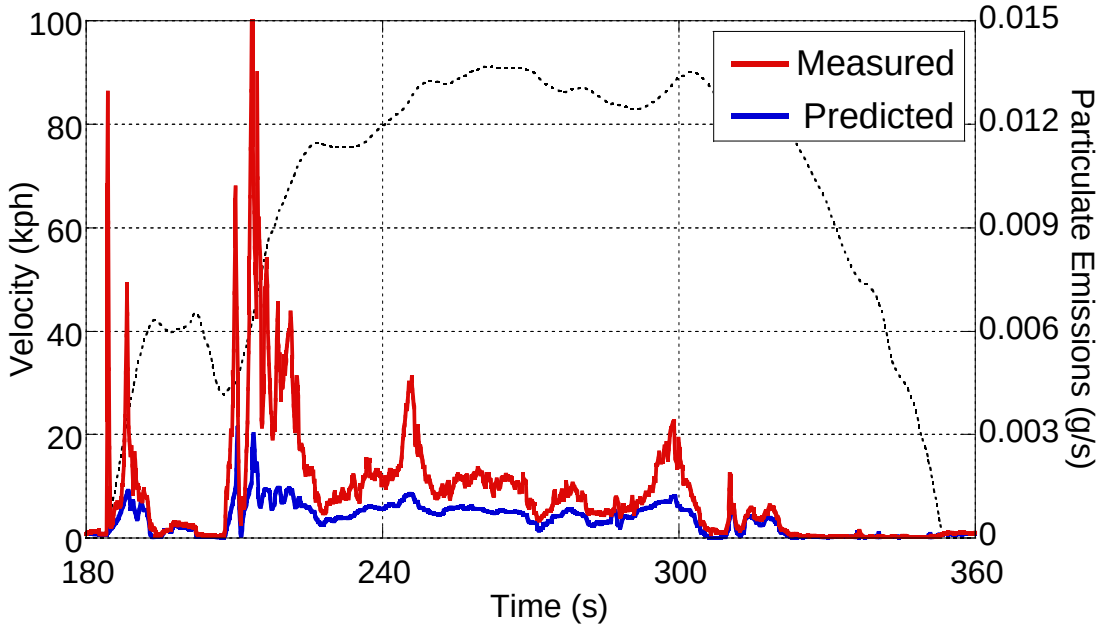


Figure 1.7: Quasi-steady state model prediction compared with measured particulate matter emissions [14].

The quasi-steady state predictions deviate considerably at the initiation of transient when conditions are irregular. Transient conditions easily dominate the emission trends for a heavy-duty vehicle, particularly over an aggressive driving schedule like federal urban driving schedule (FUDS). Consequently, dealing with transients needs to be part of the overall low-emissions strategy, as more than half of the particulate matter can be attributed to rapid increase in load.

1.3 Research Objectives

The flexibility enabled with hybridization creates chances for a synergistic approach, in which the hybrid supervisory control can be augmented to address both emissions and efficiency. The reward will be a possibility to reduce engine-out emissions and thereby

reducing the size and complexity of the aftertreatment system. The overarching goal of this dissertation is to design an intelligent supervisory controller with consideration for multiple objectives like fuel economy and low transient in-vehicle exhaust and, realize cleaner vehicles with smaller or no aftertreatment systems.

Dynamic Programming (DP) is an effective tool for stochastic problems with sequential decision-making. DP framework allows determination of optimal solution for constrained non-linear model based systems. This motivates the use of DP for calculating the optimal control policy for hybrid vehicle power management. However, DP framework suffers from well-known curse of dimensionality i.e. the computational and memory cost to solve problems grows exponentially with increase in the state space. This makes practical applicability of DP to real-life problems somewhat limited and puts an upper constraint on the type of problems that can be effectively tackled. A hybrid power management problem can only be solved with reduced order models. An inherent problem with including transient emission models in policy optimization is the resulting increase in state space. The combined problem in its entirety, policy optimization of a supervisory controller with transient emission objectives, is computationally intractable with present DP algorithms and computational power. Therein lies the requirement for the development of new generation of algorithms capable of breaking the curse of dimensionality in policy optimization. A large part of this dissertation is geared towards the development of new generation of algorithms capable of alleviating the curse of dimensionality in policy optimization. These algorithms will then be used to create a supervisory controller for hybrid vehicles with an emphasis on bridging the gap between theoretical and applied work.

Transient emission formation is a highly nonlinear and complex process. The problem is compounded by the fact that NO_X and particulate matter emission demonstrate a nonlinear change with change in engine operating conditions. Quasi steady state emission models do not capture the nonlinearities in emission formation and hence perform poorly in estimating transient emissions. Physics based models for exhaust gas concentration can predict transient emissions. These models take into account complex thermodynamical and chemical equations as well as side effects like swirl, tumble, quenching, and local temperatures. However, these models are not suitable for control

design as they are too complex and have large number of states, making them computationally slow. Stated controller development objective requires transient emission models, which are fast enough to be employed for control-oriented problems, and yet detailed enough to capture system dynamics accurately. Therein lies the requirement of transient emission models, which are fast yet can capture the emission dynamics accurately.

Simulation based vehicle design has its own limitations. Vehicle level studies using state of art simulations can be done within the constraint of reasonable computational time frame. However, phenomenon like soot formation in diesel engine is prohibitively slow and require sophisticated computational fluid dynamics (CFD) and chemical kinetics models. In addition, CFD models cannot be used for vehicle level studies to assess vehicle performance and fuel consumption. To get a better insight and emission trends with very high confidence require actual engine with real actuators. Therefore, to assess and evaluate using performance of new control strategies with emission objectives, they will be evaluated in Engine-In-the-Loop (EIL) setup. EIL couples physical engine with virtual simulation models and allows realistic transient engine operation with advanced emission analyzers giving insight into real time emissions. Integration and simulation of such a test cell poses unique challenges and they will be addressed systematically to provide reliable methodology supported with real time control models and strategy.

The above research goals can be summarized into four objectives:

1. Develop novel numerical techniques capable of alleviating the curse of dimensionality in policy optimization. This will be accomplished by leveraging neuro-dynamic programming algorithm.
2. Apply new algorithms for design of optimal supervisory control strategies with multiple objectives including transient emissions.
3. Characterize transient emission formation in a diesel engine and develop virtual sensors to enable achieving objective 2. A neuro-fuzzy tree based models are developed which involves multiple models linked with a fuzzy framework.
4. Demonstrate effectiveness of new strategies implemented on real hardware using Engine-In-the-Loop (EIL) and generate insight for model development.

In summary, this dissertation pushes the frontiers of dynamic programming and develops novel numerical techniques for design of optimal power management controller with multiple objectives. The dissertation also develops advanced transient emission virtual sensors suitable for optimization studies and on-board engine application. Finally, the influence of powertrain design and power management strategies on fuel economy and transient emissions is comprehensively and accurately characterized by combining real physical engine with advanced real-time models for powertrain in the EIL setup. The proposed algorithms are not limited to a particular hybrid configuration with above proposed objective. The objective function can be easily modified to include other objectives like battery health, drivability and thermal management.

1.4 Literature Review

Hybrid powertrain allows greater flexibility in controlling the engine and provides an opportunity for reducing engine-out emissions. An intelligent supervisory controller; designed with multiple objectives like fuel economy and low transient in-vehicle exhaust, can help in realizing cleaner vehicles with smaller or no aftertreatment systems. The problem requires addressing two key fundamental issues, optimal supervisory control and diesel emission modeling. The goal of this dissertation is to connect these two previously separate bodies of literature, Figure 1.8.

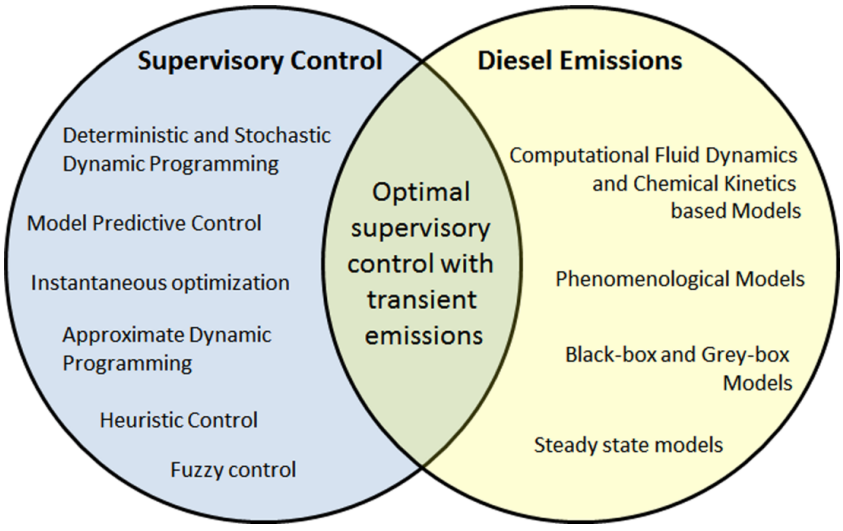


Figure 1.8: Area of contribution of this dissertation.

1.4.1 Hybrid Power Management

The supervisory control has a profound impact on hybrid vehicle system behavior and its ultimate benefits. Hybridization adds additional degree of freedom in controlling vehicle power generation unit. The power management affects both the transient performance and drivability of vehicle. The primary task is to maximize the fuel economy, while ensuring safe operation regardless of the driver demand and vehicle states. The gains with the hybrid vehicle are expected from effective regenerative braking and optimization of engine operation. Since the duration of zero-power intervals can be significant, engine shutdowns are a third factor potentially contributing to the fuel savings. However, the latter requires special measures to ensure continuous operation of accessories and safe vehicle operation [15].

The control of a hybrid vehicle is an instantaneous management of the power flow from engine and secondary power source. The objective, minimize fuel consumption and emissions, are global i.e. the quantity to be minimized is integral over the whole trip; however, the control actions are local in time. Furthermore, the control is subject to integral constraint, such as nominally maintain SOC and local constraints like driver demand and individual component limitations. The problem can be formulated mathematically as

$$\begin{aligned} J_t &= \int_0^{T_f} \dot{m}_f(t)dt \\ \left\{P_{eng}^{opt}(t), P_{\sec}^{opt}(t)\right\} &= \arg \min_{\left\{P_{eng}(t), P_{\sec}(t)\right\}} J_t \end{aligned} \tag{1.1}$$

subject to:

$$\begin{aligned} &P_{des}(t) = P_{eng}(t) + P_{\sec}(t) \\ &SOC_{\min} < SOC(t) < SOC_{\max} \\ &0 < P_{eng}(t) < P_{eng,\max}(t) \\ &P_{\sec,\min}(t) < P_{\sec}(t) < P_{\sec,\max}(t) \end{aligned} \tag{1.2}$$

where P_{des} is desired power, P_{eng} is engine power, P_{sec} is power from secondary power source like hydraulic pump/motor or electrical generator/motor, and SOC is state of charge of secondary storage unit like hydraulic accumulator or electrical battery.

Power management of hybrid vehicles has been researched extensively and there are many different strategies for controlling the engine and secondary power source. The strategies can be broadly classified into three categories: (i) heuristics based like rule-based, (ii) instantaneous optimization like equivalent consumption minimization strategy, and (iii) global optimization like horizon optimization.

Most of the power management strategies available in literature pertain to electric hybrids. With high power density and low energy density, power management in series hydraulic hybrid is very different from hybrid electric vehicles (HEV). Nevertheless, a brief summary of different power management strategies applied to HEV and HHV is presented next.

Jalil et al. [16] and Caratozzolo et al. [17] introduced the rule based supervisory control strategies for HEV with thermostatic "on/off" type engine control. The premise of these strategies is to operate engine at the fuel-efficient “sweet spot”. Liang et al. [18] proposed rule based strategy with predetermined engine power compensation function to maintain SOC at reasonable level. Barsali et al. [19], [20] proposed couple of strategies for series HEV with forecast algorithm for calculating average engine fuel consumption. The engine operation is thermostatic to sustain SOC at predetermined level. Wu et al. [21] showed the potential of hydraulic hybrid in passenger vehicle with a rule based controller. Subsequently, Kapellen et al. [22] showed significant fuel improvement with on/off strategy on heavy refuse truck. Kim et al. [8], [23] recently proposed a modulated control strategy for series hydraulic hybrid based on the insight gained from Engine-In-the-Loop testing with thermostatic control. The engine power demand is estimated as the output of a high gain PI controller. The engine is ramped slowly to the desired power demand based on SOC deviation from a desired value. Tavares et al. [24] applied modulated strategy to powersplit hydraulic hybrid with a variable displacement engine. Strategies based on fuzzy logic [25], [26] and state machines [27] have also been proposed.

Pisu et al [28], Paganelli et al. [29], [30] and Sciarretta et al. [31] treated power management problem as an instantaneous optimization problem. The proposed strategy, Equivalent Consumption Minimization Strategy (ECMS), is based on the concept of instantaneous equivalent fuel consumption. Given a driver power demand, the optimal

engine power is determined from pre-calculated semi-optimal instantaneous power split ratio based on minimization of equivalent fuel consumption. The idea of ECMS is to minimize the instantaneous cost function obtained from the sum of the fuel consumption and an equivalent fuel consumption related to the SOC variation. ECMS reduces the global optimization problem into an instantaneous optimization problem with a cost function dependent only on the system variables at that current time. ECMS can be viewed as a physical interpretation to instantaneous optimization problem using Pontryagin's Minimum Principle [32], [33]. Recently Borhan et al. [34], [35] have also applied model predictive control for hybrid power management problem.

Brahma et al. [36], Sciarretta et al. [37], Lin et al. [38], Liu et al. [39] and Wu et al. [6] applied dynamic programming to find optimal energy management strategy for HEV and HHV with various different architectures. Deterministic dynamic programming (DDP) enables significant improvements beyond what can be achieved with simple intuitive rules. However, the optimal benchmark obtained by the DDP process is not implementable and subsequent rule extraction sacrifices some of the fuel economy potential [7]. The Stochastic Dynamic Programming (SDP) eliminates the rule extractions step and allows direct development of an implementable control strategy for vehicle supervisory control. SDP is not based on a particular driving cycle (time signal), but rather the statistical characteristics of many driving cycles and hence it is non-cycle-beating. Its application to parallel HEV was pioneered by Lin et al. [40], Tate et al. [41] and Liu et al. [42], and a first attempt at addressing the hydraulic configuration was pursued recently by Kim [8]. Johri et al. [43] applied SDP with naturalistic driving cycles for an ultra-high efficiency passenger vehicle with series hydraulic hybrid powertrain.

Most of these works have focused on minimizing the fuel consumption with some researchers quantifying the impact of hybrids on emissions [44]. Recently researchers [40], [45], [46], [47], [48], [49] have developed control strategies with both fuel consumption and emissions as objective, however, the models used for emissions were fairly simple and steady state in nature. Minimizing fuel consumption along with transient emissions is still a challenge and is worth investigating further. The nature of the series hybrid system, with the engine decoupled from the wheels, allows significant freedom in designing the supervisory control strategy. This also creates a special

challenge when it comes to application of advanced algorithms like DP. DP algorithms suffer from curse of dimensionality and obtaining solutions to problems with large degree of freedom is very numerically intensive. The problem of minimizing fuel consumption along with transient emissions for a series hydraulic hybrid will require novel numerical methods to address these dimensionality and computational challenges.

1.4.2 Emission Modeling

Accurately modeling particulate matter and NO_X formation requires knowledge of in-cylinder conditions like exact local concentrations and local temperatures. The challenge in dynamical emission modeling consists in finding models that represent the highly nonlinear process of emission formation with sufficient accuracy, but are parameterized using data available for engine. Emission modeling can be broadly classified into two categories: (i) physics based and, (ii) data driven empirical models. Physics based models use chemical kinetics and fluid dynamics principles to model the underlying emission formation phenomenon whereas the empirical models are generally experimental data driven.

Physics based models can further be classified based on single or multi-zone combustion model as well as zero-dimensional or multi-dimensional modeling for geometrical features. Simplest physics based models combine zero-dimensional models with phenomenological models for emissions. Phenomenological models for emission formation are usually semi-empirical, i.e. they relate emission formation in a way consistent with fundamental theory, but not necessarily derived from theory. Hiroyasu et al. [50] proposed a two-step phenomenological model for soot. The model described the net rate of change of the soot mass as the difference between soot formation and oxidation processes. Bayer and Foster [51] developed a crank angle zero-dimensional model for soot formation and oxidation based on the Hiroyasu model. Another variation of Hiroyasu model in literature is given by Khan et al. [52]. Ericson et al. [53], Andersson et al. [54] and Lyons et al. [55] have proposed NO_X model for diesel engine with zero-dimensional dynamics and Zeldovich mechanism. Recently Seykens et al. [56] developed a zero-dimensional model based on Bayer and Foster model for soot prediction and extended NO dynamics based NO_X model.

Greater accuracy in predicting diesel combustion and emission can be obtained by employing quasi and multi-dimensional models. Pitsch et al. [57] developed a predictive model for 3-D calculations of diesel engine combustion. A flamelet model for non-premixed combustion capable of describing autoignition, the following burnout of the partially premixed phase, and the transition to diffusive burning is derived. It included the description of soot and NO_X formation based on detailed chemical reaction kinetics. Bazari [58] combined a nonlinear transient engine cycle software simulation with a versatile quasi 2-D multi-zone combustion-emission model in order to predict exhaust emissions under transient operating conditions. Jung and Assanis [59] developed a quasi-dimensional, multi-zone, direct injection diesel combustion model and implemented in a full cycle simulation of a turbocharged engine. The combustion model accounts for transient fuel spray evolution, fuel-air mixing, ignition, combustion and NO and soot pollutant formation. Some of the other computational fluid dynamics (CFD) based models in literature are [60], [61], [62]. The CFD based models are suitable only for studying detailed in-cylinder cycle resolved phenomenon and are extremely computationally intensive. The calculation covers only the closed part of cycle and requires initial boundary conditions like cylinder pressures, temperature and density. This severely limits the applicability of these models for transient studies coupled with powertrain simulation over a driving schedule and makes them unsuitable for control development and optimization studies.

Physics based models with CFD and chemical kinetics represents one end of modeling spectrum. On the other end lie the quasi steady state models. Quasi steady state models can be considered as simplest empirical models and have very small computational load. However, these models do not capture transients accurately and cannot be used for design studies involving transient engine behavior [10], [12], [63], [13]. Giakoumis et al. [64] suggested to correct for discrepancy between steady state model and actual results with a correction factor based on load increase. A similar approach was applied by Shilling et al. [65] for NO_X modeling. The detailed combustion model is used to extrapolate the sensitivities of the NO_X emissions to the various engine inputs and store the results in maps as a function of the operating conditions. The real-time model uses these maps for the computation of the NO_X emissions.

Some researchers have proposed empirical models that capture emission dynamics based on certain engine parameters. Kirchen [66], [12] developed a mean value model for soot and showed the effectiveness with tip-in operations. The model included empirical correlations relating engine out emission with engine operating conditions. Brahma et al. [67] proposed a mean value model for NO_X emissions. Warth et al. [68] combined various phenomenological models with genetic based algorithms for parameter estimation for steady state estimation of NO and soot. Brahma et al. [69] used a combined approach where multiple phenomenological models were linked with neural network and weights were trained to predict soot. A major drawback of the above proposed models is the required knowledge of in-cylinder pressures and available fuel mass as model inputs. This makes these models unsuitable for simulation studies over complete driving cycle.

Other empirical methods include black box modeling using system identification techniques. Neural network based models have been developed in past to predict NO_X and soot due to good universal function approximation capability of multi-layer perceptron. Ouladsine et al. [70] developed neural network based model for soot based on the opacimeter data. Opacimeter measures the percentage loss of light intensity between a light source and a receiver to calculate the opacity of the exhaust gas. Opacity of exhaust gases only captures a subset of the particulate matter i.e. the visible component of particulate matter i.e. “smoke”. Some of the NO_X models with artificial neural network have been proposed by Krijnsen et al. [12], [71] Toth-Nagy et al. [72], and Wang et al. [73]. Wang et al. required in-cylinder pressures to calculate crank angle at 50% mass burnt fraction (CA50). Alberer et al. [74] and Re et al. [75] proposed a model for NO_X and particulate matter based on structure identification and genetic algorithm. These models are designed using experimental data recorded over a section of driving schedule and hence are valid over small operating region of the engine. A dynamic programming based algorithm for selecting best engine operation to minimize transient emissions will rely on exploring the engine operating space and the models need to be valid over entire operating regime.

Another common issue with all the above proposed empirical models is that they all require large number of inputs including the previous time histories. This poses a

significant challenge in control optimization using DP based framework. The DP treats time history as independent state of the model, which in turn exponentially increases computational time and memory required to solve the problem. In this dissertation, we are concerned with models that capture transient emissions with high accuracy and can be employed within DP framework. This will require novel modeling approach to develop transient emission models, which are computationally fast and valid over entire engine operating space.

1.5 Technical Challenges

Design of optimal power management controller for series hydraulic hybrid powertrain with fuel economy and transient emission objectives is particularly challenging for the following reasons:

1. Optimal power management is a non-trivial problem that requires solution of an optimal control problem with multiple inputs, stochastic dynamics, nonlinear constraints, and multiple objectives. A fundamental framework is required.
2. Dynamic programming framework suffers from "curse of dimensionality". Inclusion of transient emission objective in power management controller design will result in considerable increase in number the system states. Innovative and newer class of algorithms is required to alleviate curse of dimensionality and obtain optimal solution.
3. Diesel engine combustion and emission formation is extremely nonlinear and stochastical in nature. Development of fast control oriented transient emission models is a challenge. Novel methods are required to capture transient soot and NOX characteristics in a diesel engine.
4. The input signals are stochastic, i.e. there is no a priori knowledge of driver power demand. Stochastic modeling and control techniques are required.
5. Hydraulic hybrids have very small energy storage capacity and pose unique challenges while designing supervisory controller especially for series configuration. Optimizing the powertrain only for fuel consumption with no consideration for transients can lead to harsh engine operation.

6. Engine-In-the-Loop (EIL) techniques are required for evaluating the effect of different power management strategies on physical engine performance and transient engine-out emissions. Implementation and evaluation of controller with physical hardware require consideration for controller robustness, signal degradation and real system dynamics often ignored in simulation studies.

1.6 Contributions

This dissertation develops novel techniques to design optimal power management controller with multiple objectives for a series hydraulic hybrid vehicle with diesel engine. Nonetheless, the approach is fundamental and is applicable beyond series hydraulic hybrid powertrain. These techniques are applicable to other problems involving complex physical systems, stochastic dynamics, multiple design objectives and specially problems with very large state space. The proposed algorithms can be modified to address other hybrid powertrain configurations or include other objectives like battery health, drivability and thermal management. Another major contribution of this dissertation is in the modeling of transient diesel soot and NO_X emissions. The models are derived from experimental data but the approach is generic and applicable to other engines. These models are capable of running concurrently with real engine as virtual sensors or integrating with dynamic programming based framework for optimization and control development studies. The research reported throughout this dissertation is based primarily of publications by the author [43], [24], [76], [77], [78], [79], [80], [81]. The main contributions of this dissertation in the field of optimal control for hybrids and emission modeling are:

1.6.1 Neuro-Dynamic Programming

A new formulation for developing supervisory controller for hybrid vehicles with multiple objectives like transient emissions and fuel consumption is introduced. The formulation is based on the concept of neuro-dynamic programming (NDP). This algorithm allows for near-optimal solution to problems with very large state-action space, which are computationally intractable with conventional techniques.

1. Numerical techniques to address very large state space – Mathematical techniques like functional approximation of cost-to-go function and incremental training using temporal difference learning are introduced for the first time in the context of designing hybrid power management controller. The algorithm is first demonstration of design of supervisory controller with 8 states and state-action space cardinality of 10^{13}.
2. Multi-objective power management – This power management formulation considers multiple objectives i.e. minimization of combined fuel consumption and transient emissions, for the first time.
3. Self-Learning Controller – A novel neural network based controller is introduced which learns by interacting with the environment.

1.6.2 Power Management via Stochastic Optimal Control

The power management problem of series hydraulic hybrid powertrain is formulated and solved using stochastic dynamic programming (SDP).

1. Model development – Two different powertrain models, namely control oriented and vehicle simulation models are developed for series hydraulic hybrid architecture. The models are augmented with high fidelity transient soot and NO_X emission models.
2. System conscious vs. engine centric approach – The SDP problem is formulated to produce state feedback of desired set point for both engine speed and torque i.e. the engine is not restricted to operate along the best BSFC line [43] unlike previous work where the sole controller output is engine power demand i.e. the engine is operated along best BSFC line [40], [8].
3. Numerical techniques – Multiple numerical techniques are introduced to reduce the computational time of SDP.

1.6.3 Engine-In-the-Loop Validation

The power management controller is simulated along with virtual hybrid powertrain concurrently with real engine in the Engine-In-the-Loop (EIL) facility. This allows for realistic evaluation of proposed power management controllers.

1. Integration issues and challenges – A systematic approach for transitioning from simulation to embedded controller for real world application is presented. The integration challenges, particularly issues related to causality and multi-rate controller are addressed.
2. Transient emission – Fast transient emission analyzers are used to develop deep insight into engine system behavior and quantifying the emission reduction with new proposed controller.

1.6.4 Transient Particulate Matter and NO_X Emission Model

Transient emission models for NO_X and soot are developed. The models are capable of predicting transient NO_X and soot emissions over complete range of engine operation with input parameters available from standard ECU and sensors.

1. Neuro-fuzzy model tree – A hierarchical model with Gaussian validity functions and local neural networks for transient NO_X and soot prediction is constructed using experimental data. The modeling techniques are motivated by the idea of divide and conquer the input-output space.
2. Perturbation signal – A multi-Pseudo Random Signal (m-PRS) perturbation signal is designed specifically for characterizing the diesel engine. The test signal is designed to excite all the engine operating frequencies to ensure the training data obtained captures all the operating regimes of the engine, i.e. the data are “rich”.
3. Regressor selection techniques – Two algorithms, namely Orthogonal Least Squares (OLS) and Automatic Relevance Determination (ARD), are introduced for automatic selection of regressors for local models.
4. Real-time models – The transient models are implemented in real time and are validated by running concurrently with physical engine.

1.7 Dissertation Overview

This dissertation is arranged into two major sections. The first section (Chapter 3 to Chapter 4) introduces the energy management problem in hybrid vehicles. The power management controller decisions or control actions influences the evolution of states in a hybrid vehicle. The decision made at any given time depends on the state of the system. The controller objective is to select a decision-making rule i.e. feedback policy, that optimizes a certain performance criterion. Chapter 3 applies classical dynamic programming to solve hybrid power management problem with single objective. However, the applicability of dynamic programming is limited if the underlying state space is very large. This includes power management problems with multiple objectives and complex system dynamics like emissions. Chapter 4 introduces a new class of algorithms, namely neuro-dynamic programming, to solve energy management problem. The algorithm advances the present knowledge in the field of optimal controller design. The new approach allows for considering multiple objective problems with detailed system dynamics. The methodology allows hybrid system to learn about its behavior through simulation, and to improve the controller performance through iterative reinforcement. The section concludes with experimental validation of proposed controllers with real engine. The second section (Chapter 5) focuses on transient diesel engine emission characterization and modeling. Novel techniques for characterizing engine tailpipe emissions are introduced, followed by an innovative approach to modeling complex nonlinear process. Detailed overview of each chapter is given next.

Chapter 2 introduces the series hydraulic hybrid powertrain configuration. The chapter gives a detailed overview of vehicle simulation models as well as reduced dynamics control oriented models followed by brief introduction of Engine-In-the-Loop (EIL) facility. Merging of real and virtual worlds, by combining real physical engine with advanced real-time models for powertrain is an effective way of accurately and comprehensively characterizing the influence of powertrain design and power management strategies on fuel economy and transient emissions. Finally, to evaluate and assess the performance of different policy optimization based power management controllers, a baseline power management controller is introduced.

Chapter 3 introduces Stochastic Dynamic Programming (SDP) as a tool for policy optimization for hybrid vehicles. An infinite horizon discounted cost stochastic dynamic programming problem is setup using a set of naturalistic driving schedules. An additional degree of freedom is included in SDP analysis, with the engine power demand being split into two variables, namely engine torque and speed. The optimal controller obtained using SDP is compared with baseline thermostatic controller. The results represent a significant departure from the conventional wisdom of operating the engine near its "sweet spot" and indicate what is preferred from the system standpoint.

The classical dynamic programming based algorithms are constrained by the curse of dimensionality, i.e. exponential increase in computational effort with increase in system state space. Chapter 4 proposes a novel approach capable of overcoming the curse of dimensionality and solving policy optimization for a system with very large design state space. A self-learning supervisory controller for hybrid vehicles based on the principles of neuro-dynamic programming is proposed. The controller learns and improves its performance over time by interacting with the environment. Initially the problem is setup with single objective of minimizing the fuel cost and solved using NDP, which is then compared with results from SDP to demonstrate the effectiveness of the proposed technique. The power management problem is then reformulated with additional objective of minimizing transient emissions. The NDP framework is extended with models from Chapter 5 to include transient NO_X and soot emissions. Inclusion of the transient NO_X and soot emission models results in introduction of five additional states and the problems' state-action space becomes extremely large. The NDP based self-learning controller is designed with multiple objectives and the power management controller is evaluated using EIL. The resulting tradeoff between multiple objectives i.e. fuel consumption, NO_X and soot is evaluated.

Diesel engine combustion and emission formation is highly nonlinear and thus creates a challenge related to engine diagnostics and engine control with emission feedback. Chapter 5 presents a novel methodology to address the challenge and develop virtual sensing models for engine exhaust emission. These models are capable of predicting transient emissions accurately and are computationally efficient for control and optimization studies. Chapter 5 describes development of neuro-fuzzy models for

prediction of transient soot and NO_X emission from a diesel engine. The modeling techniques are motivated by the idea of divide and conquer the input-output space i.e. to divide a complex problem into multiple simpler sub problems, which then can be identified using simpler class of models. A specially designed training procedure and experimental tests are proposed to excite all the operating frequencies of the engine. The diesel engine is tested using integrated hardware and software tools for automated testing with high-speed data recording. The transient soot and NO_X emission is recorded using fast emission analyzers. Chapter 5 then introduces two different input regressor selection techniques, Orthogonal Least Square (OLS) and Automatic Relevance Determination (ARD). The OLS is a linear subset selection technique and involves selecting orthogonal regressors, which result in maximum error reduction in output variance. ARD provides a Bayesian framework for ranking the inputs and thereby provides a systematic procedure for the selection of relevant inputs from all the input variables. The experimental data is finally used to construct three different transient emission models. The difference lies in the structure of local models, the validity function and regressor selection technique. Finally, the model is validated with transient emission recorded during EIL testing of engine coupled to virtual hybrid

Chapter 6 summarizes the main results of this dissertation and proposes possible future research directions.

Chapter 2

SERIES HYDRAULIC HYBRID VEHICLE

2.1 Introduction

Development of alternative powertrains is driven by the need to address the energy security and climate change with increased fuel economy. Hybridization provides a significant leap in fuel economy improvements. Hybrid vehicles use secondary storage of energy like battery, accumulator and a secondary source of power like electric motor/generator or hydraulic pump/motor. Efficient management of the secondary source of energy provides additional degree of freedom in operation of engine and the whole powertrain can be designed to improve fuel economy and lower emissions. However, the vehicle system becomes more complicated and requires sophisticated control strategy to maximize the benefits.

Hybrid system architecture is a key factor in maximizing the potential benefits from a hybrid vehicle. The parallel systems are easier to implement but are limited with regards to flexibility offered in controlling the engine. Previous work by Wu et al. [6] and Matheson et al. [82] showed the parallel hybrids are effective in regeneration, but the fuel economy advantages diminish for duty cycles with high speed cruising. In contrast, a series hybrid provides a full flexibility in operating the engine under any driving conditions. There is no direct coupling between the vehicle speed and engine speed, and engine can be operated to maximize fuel economy and minimize emissions. Hence, a series hybrid configuration is chosen for this dissertation as it allows exploring unique opportunities in operating engine.

The vehicle considered in this dissertation is a medium-duty truck. Power flows through hybrid subsystems during launch and braking can be very high in a truck due to their large mass. Hydraulic hybrids store energy in hydraulic accumulator by

compressing nitrogen gas, as the hydraulic fluid itself is theoretically incompressible. Since there is practically no limit on how fast the gas is compressed, hydraulic accumulators can be used where the rate of change of energy is very high unlike batteries where energy storage depends on slow chemical reactions. This makes hydraulics well suited for truck applications as they can be used for aggressively recuperating braking energy and large power assist during launch. Figure 2.1 shows accumulator along with other energy storage devices on energy vs. power density plot [8]. It is evident from the Figure 2.1 that accumulators have over 100+ times more power density though they have very small energy density. In contrast, low energy density is a drawback compared to batteries as high rate of power can only be sustained for short time intervals. This is of critical importance for series hydraulic hybrid vehicle (S-HHV) analysis since the low storage capacity of the hydraulic accumulator creates a special challenge for controller development, very different from electric system and will require special consideration while designing power management controller. Increasing the size of accumulator results in bigger and heavier accumulator and may offset advantages with weight penalty and packaging constraints.

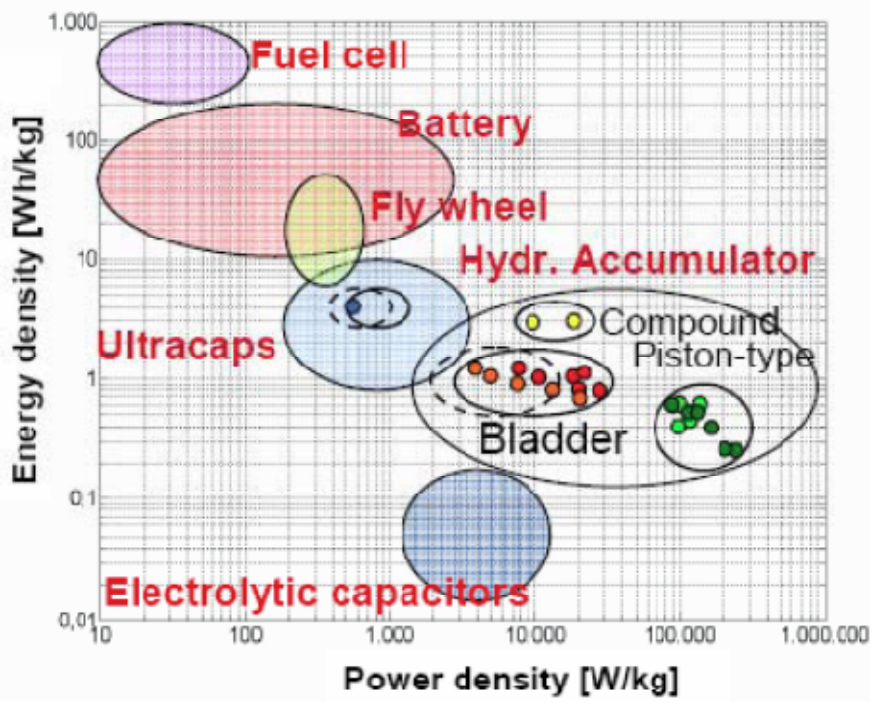


Figure 2.1: Energy vs. power density for various energy storage systems.

Another advantage of hydraulic devices is their high efficiencies. Accumulators designed for mobile application have efficiency in the region of 98% [83] independent of charging rate whereas battery efficiency is a function of its charging rate. Williamson et al. [84] showed the average efficiency of about 90% during driving cycle simulations. The efficiency for hydraulic pump/motors can be as high as 95% with 90%+ efficiency over their most operating region for mobile application devices. Also hydraulic pump/motors are more compact [85], Figure 2.2 and hence have smaller inertias resulting in faster bandwidth. Since there are multiple energy conversions in a series hybrid powertrain configuration, the overall efficiency of the powertrain is dependent on the conversion efficiency of individual devices making hydraulic a more suitable choice.

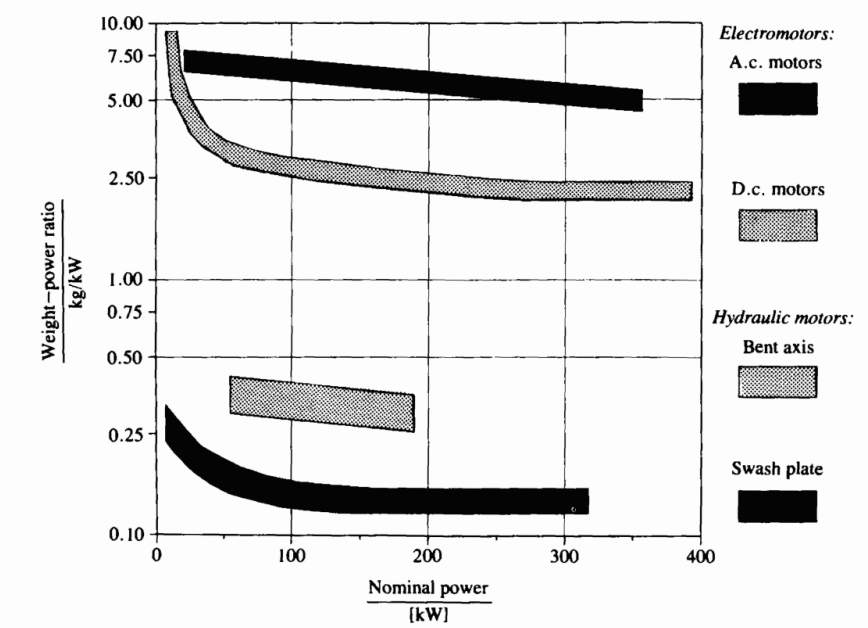


Figure 2.2: Comparison of weight-to-power ratio of electric and hydro machinery [85].

Maximum power of hydraulic machinery is proportional to the potential difference of the pressure head at its port. With present accumulator technologies this is limited to maximum of 420 bar [85] but will go up as the technology for accumulator design progresses. Higher head pressures would result in even smaller hydraulic machinery for similar power requirement. One drawback of hydraulic machinery compared to electrical counterpart is their maximum speed limitations but it is still not serious enough to limit their application for mobile applications.

This chapter begins with a high-level overview of the S-HHV and the discussion of the modeling approach. Two sets of simulation models are developed, one for vehicle simulation studies and other for control development. This is followed by an overview of the Engine-In-the-Loop (EIL) setup at the University of Michigan and real-time models required for simulating virtual series hydraulic hybrid powertrain including the integration challenges in combining real engine and virtual models. Finally, a heuristic controller is developed to serve as baseline and fuel economy results for different drive schedules is presented. This controller will be used later in the dissertation to assess the performance of policy optimization based controllers.

2.2 Vehicle

A 4X4 military vehicle intended for on-road and off-road purpose is considered in this dissertation. The baseline vehicle specifications correspond to High Mobility Multipurpose Wheeled Vehicle (HMMWV). The vehicle is modeled to have series hydraulic hybrid powertrain, shown in Figure 2.3 with the design similar to that presented by Kim et al. [86] and Filipi et al. [23]. The engine is connected to pump to create a power generation subsystem capable of charging the accumulator. The motors propel the vehicle with hydraulic energy stored in accumulator. There is no mechanical linkage between engine and the wheels giving flexibility in operating the engine. The motors can be operated as pumps for regenerative braking. Table 2.1 gives the specification of the vehicle and powertrain.

The vehicle is configured to have one drive motor per axle. Previous work on a series hydraulic hybrid [86] showed the advantage of this particular configuration over single motor design with transfer case for a 4X4 mid-size truck. It was also shown that sequential operation of two motor design can result in better fuel economy over simultaneous operation. The concept is to operate motors sequentially resulting in higher loads per motor and hence higher efficiency. Rear motor is used primarily for propulsion and front motor augments the torque in extreme cases. While braking, front motor is used primarily for regeneration. The choice between front and rear motor operation is based on

weight transfer during acceleration and braking. The results shown in the dissertation are obtained with the sequential operation of motors.

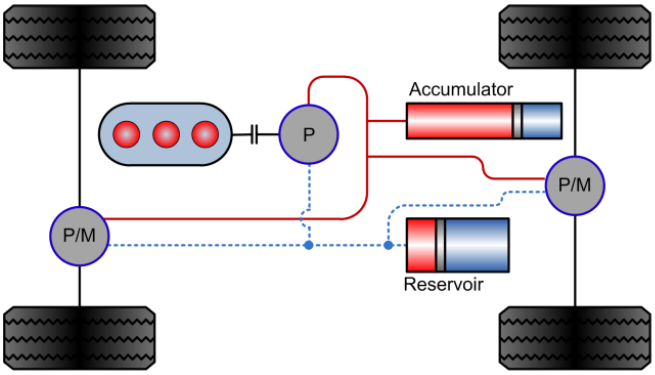


Figure 2.3: Series hydraulic hybrid configuration.

Table 2.1: Series hydraulic hybrid specifications

Engine	Description	6.4L International V8
	Max. Power	261 kW @ 3000 RPM
	Max. Torque	881 Nm @ 2000 RPM
Pump	Design	Axial Piston Variable Displacement
	Size	300 cc/rev
	Max Power	700 kW @ 350 bar @ 4000 RPM
Propulsion Motor	Design	Axial Piston Variable Displacement
	Size	180 X 2 cc/rev
	Max Power	420 kW @ 350 bar @ 4000 RPM
Accumulator	Capacity	98 Liter (Max. Gas Volume)
	Max Pressure	350 bar
	Min Pressure	120 bar
Vehicle	Type	HMMWV
	Weight	5112 kg
	Coefficient of Drag	0.7
	Frontal Area	3.58 m^2
	Tire Radius	0.4412 m
	Final Drive Ratio	4.086
Transmission	Design	2 speed automatic
	1^{st} Gear Ratio	3 : 1
	2^{nd} Gear Ratio	1 : 1

2.3 Vehicle Models

The work presented in this dissertation uses two different dynamic models. The first model is a high fidelity model with vehicle and powertrain components modeled in Simulink, based on the vehicle/powertrain simulation platform developed at the University of Michigan [87]. The vehicle/powertrain simulation platform has been validated with vehicle test data from proving ground [87] and subsequently thoroughly updated to represent HMMWV [88]. The hydraulic components were added to the vehicle simulation in context of parallel [6] and series hybrid systems [86]. This vehicle/powertrain simulation platform is used for validation of supervisory controller. The second model is a control-oriented powertrain model with dynamics faster than 1 Hz ignored. The model has very few system states to keep computational cost low and is used for design of supervisory controller. This arrangement of two separate models, one for control design and another for validation allows for application of advanced mathematical techniques while keeping the problem feasible during controller design phase and providing accurate real-world results during validation phase.

2.3.1 Vehicle Simulation Models

2.3.1.1 Engine

The engine model takes driver command and external load torque as input and calculates the engine speed and fuel consumption. The model includes an engine torque map, a fuel controller with speed governing functions and engine dynamics as shown in Figure 2.4. The engine torque, T_e is calculated using a lookup table with fuel injected, m_f and engine speed, ω_e as inputs.

$$T_e = f\left(\omega_e, m_f\right) \tag{2.1}$$

$$\omega_e = \frac{1}{I_e}\int\left(T_e - T_p\right)dt \tag{2.2}$$

where I_e is engine inertia, ω_e and T_e are engine speed and torque respectively and T_p is the load torque from the hydraulic pump. A diesel engine fuel injector controller provides the mass of fuel injected to the lookup table based on throttle command and engine speed. A

turbo-lag is simulated by including time delay in injection with time constant calibrated based on data obtained from engine testing [7].

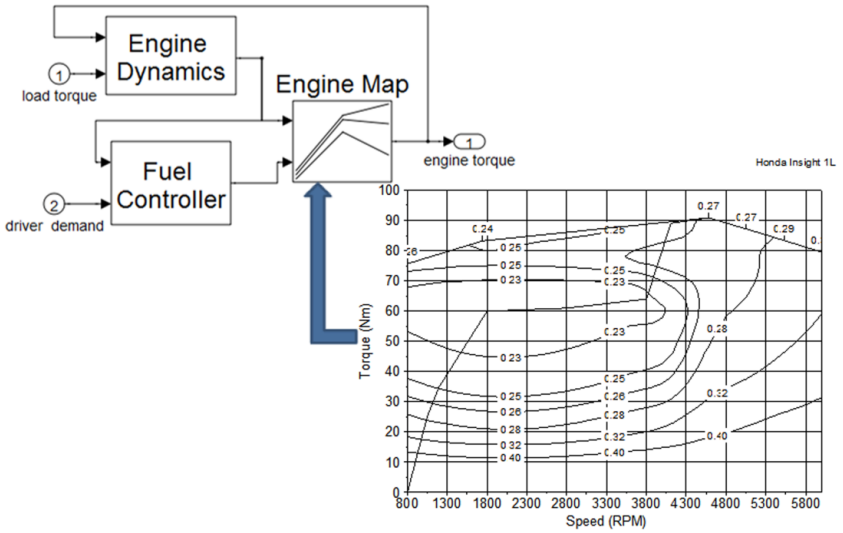


Figure 2.4: Engine model.

The engine model is based on test results of 6.4L medium-duty turbocharged diesel engine at the University of Michigan. The engine specifications along with steady state BSFC map from 6.4L engine are given in Appendix A.

2.3.1.2 Hydraulic pump/motor

The hydraulic pump/motor (P/M) model is based on updated version of Wilson's pump/motor theory [89]. The P/M is modeled as an axial piston variable displacement type with the torque and flow controlled by the displacement command. The theoretical flow and torque output is corrected based on the losses estimated using physics-based expressions. The flow losses encompass the laminar, compressibility and turbulent leakage (or "slip"), and the torque losses comprise viscous, hydrodynamic and mechanical. The loss expressions include constants that are calibrated using available experimental data [90], and after calibration the model is capable of capturing effects of all operating parameters [7] and [86].

The ideal leak-free volumetric flow rate, Q_i of oil from P/M is given by

$$Q_i = x\omega D \tag{2.3}$$

and ideal frictionless torque, T_i is given by

$$T_i = x\Delta pD \tag{2.4}$$

where x is the displacement command, ω is the pump/motor angular velocity, D is the displacement of pump/motor, and Δp is the pressure difference across pump/motor.

The actual and ideal volumetric flow determines the volumetric efficiency:

$$\eta_{v,pump} = \frac{Q_a}{Q_i} = 1 - \frac{C_s}{xS} - \frac{\Delta p}{\beta_o} - \frac{C_{st}}{x\sigma} \tag{2.5}$$

$$\eta_{v,motor} = \frac{Q_i}{Q_a} = \frac{1}{1 + \frac{C_s}{xS} + \frac{\Delta p}{\beta_o} + \frac{C_{st}}{x\sigma}} \tag{2.6}$$

where C_s and C_{st} is the laminar and turbulent leakage coefficients respectively, and β_o is the fluid bulk modulus.

$$S = \frac{\mu\omega}{\Delta p} \tag{2.7}$$

$$\sigma = \frac{\omega D^{\frac{1}{2}}}{\left(\frac{2\Delta p}{\rho_o}\right)^{\frac{1}{2}}} \tag{2.8}$$

Torque efficiency relates actual and ideal torque:

$$\eta_{t,pump} = \frac{T_i}{T_a} = \frac{1}{1 + \frac{C_v S}{x} + \frac{C_f}{x} + C_h x^2 \sigma^2} \tag{2.9}$$

$$\eta_{t,motor} = \frac{T_a}{T_i} = 1 - \frac{C_v S}{x} - \frac{C_f}{x} - C_h x^2 \sigma^2 \tag{2.10}$$

where C_v, C_f and C_h are viscous, frictional and hydrodynamic loss coefficients respectively.

Figure 2.5 and Figure 2.6 show trends of P/M mechanical efficiencies and volumetric efficiencies with pressure, displacement and speed. The figures illustrate an important difference compared to electrical machines, namely an added dimension due to the pressure dependency. The pressure difference across the machine will vary within a wide range during vehicle operation; hence, both the absolute level of peak torque and the efficiencies will vary with it. This creates an additional challenge in controller development and requires care in assessing performance constraints.

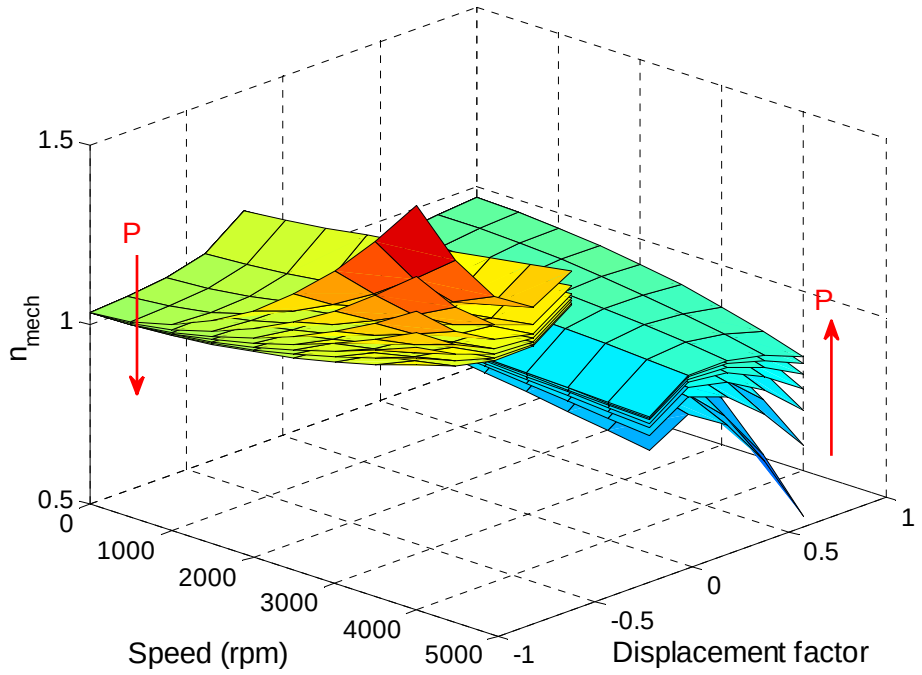


Figure 2.5: Pump/Motor mechanical efficiency as a function of load (i.e. displacement factor), speed and pressure difference across the machine.

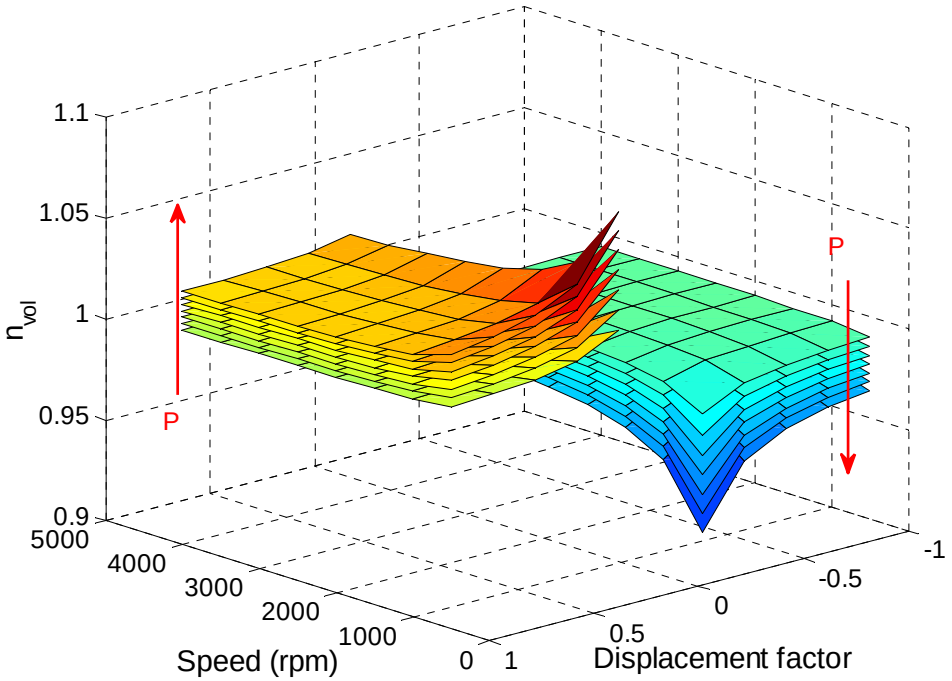


Figure 2.6: Pump/Motor volumetric efficiency as a function of load (i.e. displacement factor), speed and pressure difference across the machine.

2.3.1.3 Accumulator

A hydro-pneumatic accumulator is used for energy storage in hydraulic hybrids. A positive fluid flow rate into the accumulator compresses the nitrogen gas stored in the bladder, thus storing energy. A low pressure reservoir is used in the system to prevent cavitation of hydraulic devices. The net head pressure on hydraulic devices is the difference between accumulator and reservoir pressure.

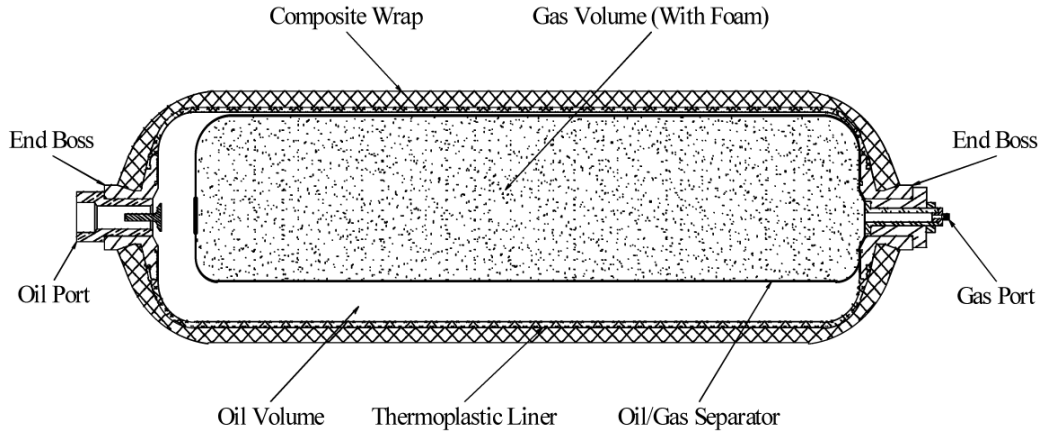


Figure 2.7: Schematic of bladder type accumulator [90].

Figure 2.7 shows a schematic of a bladder type accumulator with all its components. Bladder keeps the gas separate from the oil and hence gas inside accumulator can be treated as closed system. It is assumed that the accumulator gas exchanges work with hydraulic fluid and heat with both accumulator shell and elastomeric foam. In order to correctly predict the accumulator dynamic performance and efficiency, a full thermodynamic model is used. It is derived from considerations for energy conservation [89], [7], [91], [92], and includes the effects of heat transfer and the real gas properties.

$$m_g \frac{du}{dt} = -P_g \frac{dv}{dt} - m_f c_f \frac{dT}{dt} - hA_w\left(T - T_w\right) \tag{2.11}$$

where m_g is mass of gas, m_f is mass of foam, c_f is foam specific heat, h is convection heat transfer coefficient, A_w is wall area and T_w is wall temperature

Using expression for internal energy

$$du = c_v dT + \left[T\left(\frac{\partial P_g}{\partial T}\right)_v - P_g \right] dv \tag{2.12}$$

in the energy balance equation results in

$$\left[1+\frac{m_f c_f}{m_g c_v}\right]\frac{dT}{dt}=\frac{T_w-T}{\tau}-\frac{T}{c_v}\left[\left(\frac{\partial P_g}{\partial T}\right)_v\right]\frac{dv}{dt} \tag{2.13}$$

with thermal constant defined as

$$\tau=\frac{m_g c_v}{hA_w} \tag{2.14}$$

where m_g is the mass of gas, c_v is specific heat, h is the heat transfer coefficient and A_w is the area of the wall exposed to the gas.

To account for real gas behavior, the Benedict-Webb-Rubin (BWR) state equation is used to correlate the gas temperature, pressure and specific volume:

$$P_g=\frac{RT}{v}+\frac{\left(B_0RT-A_0-\frac{C_0}{T^2}\right)}{v^2}+\frac{bRT-a}{v^3}+\frac{a\alpha}{v^6}+\frac{\left(C\left(1+\frac{\gamma}{v^2}\right)e^{-\frac{\gamma}{v^2}}\right)}{v^3T^2} \tag{2.15}$$

where P_g, T and v is gas pressure, temperature and specific volume respectively and, A_0, B_0, C_0, a, b, c, α and γ are BWR gas constants.

The efficiency of charging-discharging is a strong non-linear function of the thermal time-constant, therefore increasing the heat capacity ($m_g c_v$) and reducing the heat loss (hA_w) is beneficial. Addition of elastomeric foam has several advantages [91] [93] like (i) elastomeric foam insulates the gas and hence reduces temperature variations thereby increasing storage capacity, and (ii) gas pressure becomes a good indicator of state of charge (SOC) with constant temperature. This insight led to now a common practice of adding the elastomeric foam to the gas side in order to enhance the thermal capacity and elevate the conversion efficiencies to the mid-nineties.

The SOC for a hydraulic device is defined as the ratio of instantaneous gas volume to accumulator gas capacity:

$$SOC=\frac{V-V_{\min}}{V_{\max}-V_{\min}} \tag{2.16}$$

In real application, pressure can be used as indicator of *SOC*, provided the temperature variations are kept low. This is tied to the accumulator design, e.g. the

advanced carbon fiber accumulator with foam can minimize the temperature fluctuations significantly enough to satisfy this assumption. The mass of gas is directly related to the pressure range for a given accumulator size. In this dissertation, the mass of gas is chosen to provide a precharge pressure of 12 MPa and a maximum pressure of 35 MPa.

2.3.1.4 Transmission

The transmission is modeled as a finite state machine with different gears being the different states of the system. A blending function is added to simulate inertia and torque phase during gearshift. Blending function provides a fast and accurate way of representing off-going and on-coming clutch. The vehicle is equipped with 2-speed gearbox between propulsion motor and final drive. The gear shifting logic based on motor speed and motor displacement command is shown in Figure 2.8.

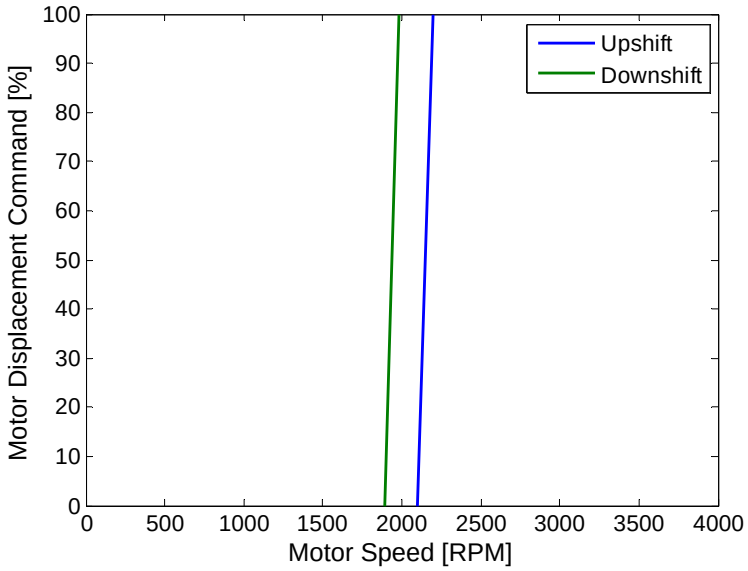


Figure 2.8: Gearshift logic based on motor speed and motor displacement command.

2.3.1.5 Vehicle

The vehicle is a modeled as a point mass system. This is deemed sufficient for the fuel economy studies. The model includes rolling resistance and aerodynamic drag. The vehicle model also contains a brake model, which acts as a coulombic friction device. The vehicle model equations are given below.

$$T_{rollres} = \mu_f \cdot m \cdot g \cdot R_{tire} \tag{2.17}$$

$$T_{drag} = 0.5 \cdot A_f \cdot C_d \cdot \rho \cdot \omega_{wh}^2 \cdot R_{tire}^3 \tag{2.18}$$

$$T_{brake} = (R_{vis} \cdot \omega_{wh} + \mathrm{sgn}(\omega_{wh}) \cdot F_{stat}) \cdot x_{brake} \quad if\ \omega_{wh} > \omega_{\min} \tag{2.19}$$

$$T_{brake} = (R_{vis} \cdot \omega_{wh} + \mathrm{sgn}(\omega_{wh}) \cdot F_{stat} \cdot \omega_{wh} / \omega_{\min}) \cdot x_{brake} \quad if\ \omega_{wh} < \omega_{\min} \tag{2.20}$$

$$v_{act} = \int (T_{drive} - T_{drag} - T_{rollres} - T_{brake}) / (m \cdot R_{tire}) \cdot dt \tag{2.21}$$

where T_{drive}, T_{brake}, T_{drag} and $T_{rollres}$ is drive torque, brake torque, drag torque and rolling resistance torque. μ_f is rolling friction coefficient, C_d is coefficient of drag, A_f is frontal area, ρ is air density, g is gravitational constant, R_{tire} is the tire radius, m is the vehicle mass and ω_{wh} is wheel velocity. The brake model is represented by equations (2.19) and (2.20) where F_{stat} is static friction, x_{brake} is the brake command from driver, R_{vis} is viscous friction coefficient and ω_{min} is the minimum wheel speed.

2.3.1.6 Driver

The vehicle simulation platform employs a cyber-driver for fair comparison between different architecture and power management studies. The driver is modeled as a proportional integral (PI) controller acting on the error between the actual vehicle velocity and the desired vehicle velocity defined by the selected driving schedule.

$$\alpha = K_p (v_{des} - v_{act}) + K_i \int (v_{des} - v_{act}) dt \tag{2.22}$$

where α is driver command, v_{des} and v_{act} is the desired and actual velocity respectively, and K_p, K_i are PI controller gains. A 3-second preview is added to the driver model to represent actual driver with anticipation.

2.3.2 Control Oriented Models

2.3.2.1 Engine

The engine dynamics is modeled as a quasi-steady map. The fuel consumption, m_f is modeled as a static function of engine speed, ω_e and engine torque, T_e.

$$m_f = f(\omega_e, T_e) \tag{2.23}$$

2.3.2.2 Transient particulate matter and NO_X emission

The emission model is a neuro-fuzzy model tree and is capable of predicting transient diesel emissions. A brief description of the model is given in this section for completeness sake and detailed model can be found in Chapter 5. The model consists of multiple local neural networks which are locally valid and the contribution of each model is weighted according to their validity function, Φ. The validity function defines the region of influence of a particular local model and is a fuzzy framework with triangular membership functions. The neuro-fuzzy model tree based particulate matter and NO_X transient emission models have small computation footprint and are well suited for DP framework.

$$\begin{aligned} PM^k &= \sum_{i=1}^{M} f_{NN\_PM}\left(\bar{w}_{PM}, \bar{u}_{PM}\right) \cdot \Phi_i(\omega_e^k) \\ \bar{u}_{PM} &= \left\{\omega_e^k, T_e^k, P_{im}, m_f, \dot{m}_f, PM^{k-1}\right\} \end{aligned} \tag{2.24}$$

$$\begin{aligned} NOx^k &= \sum_{i=1}^{M} f_{NN\_NOx}\left(\bar{w}_{NOx}, \bar{u}_{NOx}\right) \cdot \Phi_i(\omega_e^k) \\ \bar{u}_{NOx} &= \left\{\omega_e^k, T_e^k, P_{im}, m_f, NOx_{ss}, NOx^{k-1}\right\} \end{aligned} \tag{2.25}$$

where k is the time index, $\bar{w}$ is the set of neuron weights, $\bar{u}$ is the set of input regressors, ω_e^k is current engine speed, T_e^k is current engine torque, $P_{im} = f(\omega_e^k, T_e^k)$ is steady state inlet manifold pressure, m_f is current mass of fuel injected, $\dot{m}_f$ is the rate of change of fuel injection, PM^{k-1} is the previous predicted particulate matter, $NO_X = f(\omega_e^k, T_e^k)$ is the steady state NO_X and NO_X^{k-1} is the previous predicted NO_X.

2.3.2.3 Hydraulic accumulator

The accumulator is modeled as a polytropic process to reduce the number of model states and computational cost.

$$P_g v^n = const \tag{2.26}$$

where P_g is the gas pressure, v is the specific gas volume, n is the polytropic coefficient and *const* is the constant. The parameters n and c are tuned using high fidelity thermodynamics based accumulator model described in section 2.3.1.3. The discrete-time specific volume dynamics is given by

$$v_{k+1} = v_k + (Q_m + Q_p) \tag{2.27}$$

where Q_m is the motor fluid flow and Q_p is the pump fluid flow. Q_m and Q_p are positive when the fluid flows into the accumulator and negative when it flows out of the accumulator. The SOC is defined similar to high fidelity thermodynamics based accumulator model.

2.3.2.4 Hydraulic pump/motor

The hydraulic pump/motor (P/M) is modeled based on updated version of Wilson's P/M theory [89] and the governing equations are given in section 2.3.1.2. The control-oriented model employs an inverted P/M dynamics. At any given instant, P/M torque and speed is known and is used to calculate the displacement command. The displacement command is then used to calculate pump/motor flow. For pumping mode

$$\begin{aligned} \eta_{T,pump} &= \frac{T_i}{T_a} = \frac{|x|\Delta pD}{T_a} = \left(1 + \frac{C_v S}{|x|} + \frac{C_f}{|x|} + C_h x^2 \sigma^2\right)^{-1} \\ T_a &= \Delta pD|x| + \left(C_v S + C_f\right)\Delta pD + C_h \sigma^2 \Delta pD|x|^3 \end{aligned} \tag{2.28}$$

where C_v is the viscous loss coefficient, C_f is the frictional loss coefficient, C_h is the hydrodynamic loss coefficient, and S and σ are dimensionless coefficients. The constants in the loss expressions are calibrated using experimental data [90]. The equation (2.28) can be rearranged to $ax^3 + bx + c = 0$ and can be solved algebraically to calculate the roots.

$$\begin{aligned} a &= C_h \sigma^2 \Delta pD \\ b &= \Delta pD \\ c &= \left(C_v S + C_f\right)\Delta pD - T_a \end{aligned} \tag{2.29}$$

The equation has two imaginary roots and one real root. The real solution is used for calculating the pump displacement and subsequently pump flow. Similarly, for motor mode

$$\begin{aligned} \eta_{T,pump} &= \frac{T_a}{T_i} = \frac{T_a}{|x|\Delta pD} = 1 - \frac{C_v S}{|x|} - \frac{C_f}{|x|} - C_h x^2 \sigma^2 \\ T_a &= \Delta pD|x| - \left(C_v S + C_f\right)\Delta pD - C_h \sigma^2 \Delta pD|x|^3 \end{aligned} \tag{2.30}$$

and

$$\begin{aligned} a &= C_h \sigma^2 \Delta p D \\ b &= -\Delta p D \\ c &= \left(C_v S + C_f \right) \Delta p D + T_a \end{aligned} \tag{2.31}$$

The real root of the above equation gives the motor displacement and can be used for calculating the motor flow.

2.3.2.5 Transmission and differential

The transmission is modeled as a discrete state system and the gear change takes place instantly with no dynamics. The output torque is obtained using following expression

$$T_t = GR \cdot \eta_g \cdot T_m \tag{2.32}$$

where GR is the transmission gear ratio, η_g is the gear efficiency (varies with gear) and T_m is the motor torque. The final propulsion torque after the differential is calculated using

$$T_d = FD \cdot \eta_{FD} \cdot T_t \tag{2.33}$$

where FD is the final drive ratio, η_{FD} is the final drive efficiency and T_d is the drive torque at wheels.

2.3.2.6 Vehicle

The vehicle is modeled as a point mass system and the wheel speed, ω_{wh} is calculated using

$$\omega_{wh}^{k+1} = \omega_{wh}^{k} + \frac{1}{m R_{tire}^{\ 2}} \left(T_{wh} - \left(F_r + F_a \right) \cdot R_{tire} \right) \cdot \Delta t \tag{2.34}$$

where $T_{wh} = T_d - T_b$ is the net wheel torque with T_b is the braking torque, R_{tire} is the dynamic tire radius, F_r and F_a is the rolling resistance and aerodynamic drag force, and m is the mass of the vehicle. Δt is the sampling time.

2.4 Engine-In-the-Loop

Development of modern propulsion concepts and validation of power management strategies invariably needs some degree of prototyping. Simulation tools are indispensable for prediction of system behavior under test scenario and evaluating different concepts and methodologies. Simulation tools also lend themselves to optimization framework and can be used for optimizing component size as well as control strategies. However, simulation based vehicle design tools have some limitations. A vehicle level simulation study over complete driving cycle is limited in fidelity by the computation time frame and complex phenomena like transient emission formation cannot be accurately predicted. Simulating soot formation is prohibitively slow as it involves coupling sophisticated computational fluid dynamic and chemical kinetic models. Therefore, the state-of-art simulation models are limited to predicting the fuel economy, vehicle performance and system dynamics. With advancement in technology, synergistic combination of physical and virtual prototypes has become possible and a closed loop interaction between physical and virtual components is possible with little loss of fidelity. This provides a powerful "middle ground" between virtual simulation and full physical prototyping. Having certain key components in physical allow application of sophisticated diagnostics like fast emission analyzers in case of engine-in-loop tests. Engine-in-the-Loop (EIL) is frequently used in industry for design and calibration of engine and powertrain controllers [94], [95] but it is being increasingly utilized to explore future powertrain configurations. Argonne National Labs developed and utilized HIL setup to investigate the tradeoff between fuel efficiency and NO_X emissions [96]. Filipi et al. [97], [23] demonstrated the effectiveness of integrating virtual components with real engine for investigation of clean diesel technologies along with advanced propulsion concepts. Hagena et al. [11], [14] used the EIL facility to characterize the impact of transients on soot emissions. Simon et al. [98] further expanded the realm of HIL tests by distributing the physical and virtual systems across different cities and connecting the whole system over internet.

Figure 2.9 gives the overview of the EIL facility at the University of Michigan. The EIL setup features a state-of-the-art engine (in the forefront, Figure 2.9) and a highly dynamic AC dynamometer (in the back, Figure 2.9) with the accompanying control

system. This setup is different from conventional hardware-in-the-loop systems that typically feature control unit in hardware integrated with virtual systems being controlled. This setup features a major piece of hardware integrated with virtual systems emulating realistic operating conditions. Using the virtual driveline/vehicle simulation enables rapid prototyping of different powertrain configurations and different power management strategies. The engine is fully instrumented and test cell is equipped with fast emission analyzers and high speed data acquisition systems. This allows for advanced diagnostics and generating insights for developing low-emission concepts.

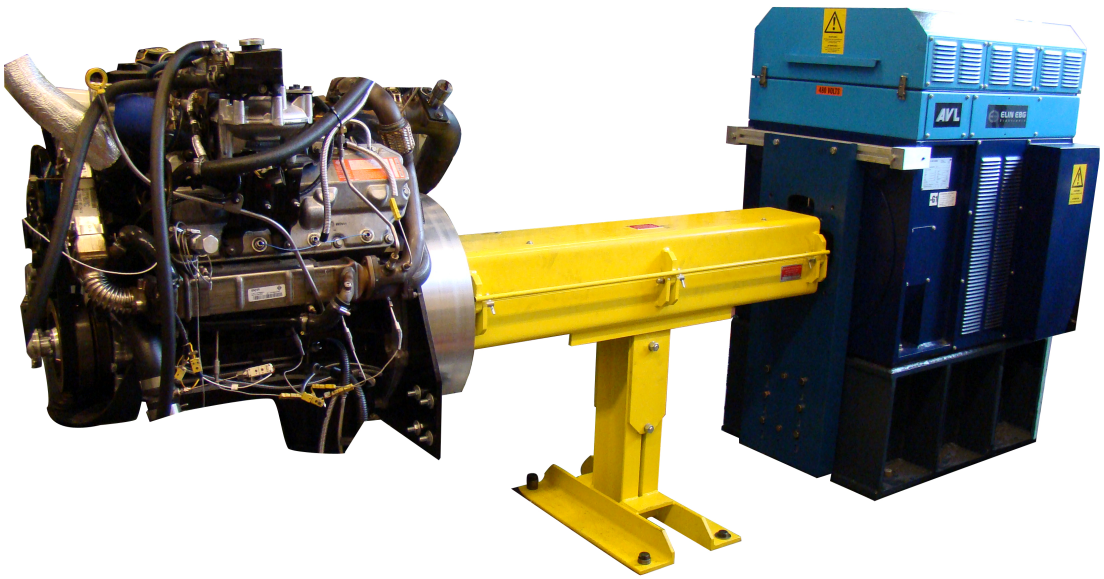


Figure 2.9: Engine-In-the-Loop test cell.

One of the key advantages of EIL is the rapid prototyping of the power management controller and real-world evaluation. However, the EIL setup is not limited to evaluating different concepts but is a tool to systematically investigate and generate insights into engine operation in a complex hybrid powertrain. Previous work done by Liu et al. [42] showed that EIL facility is a valuable tool in evaluating the real-life performance of the supervisory controller and demonstrated that a supervisory controller which performed well in their simulation failed to perform adequately with physical engine. Similarly based on the learning from EIL and performance of thermostatic controller, Filipi et al. [23] proposed a modulated controller that outperformed thermostatic controller with better fuel economy and lower engine-out emissions.

2.4.1 Engine-In-the-Loop Setup Overview

This section gives a brief overview of the EIL setup at the University of Michigan, Figure 2.10. The Figure 2.10 shows the experimental block diagram. The engine is coupled with highly dynamic AC dynamometer emulating the load on the engine and is controlled via AVL PUMA Open system. The virtual systems are simulated on dSPACE real-time simulation platform. The dSPACE system exchanges system variables with AVL PUMA system and together they orchestrate the EIL test. The test cell is equipped with fast emission analyzers for particulate matter and NO_X measurement, which communicates with the AVL system. Transient emissions along with other sensor signals and are recorded using high speed data acquisition system. The AVL uses an internal clock to sync the incoming data from different sensors and align them appropriately. The complete details and specifications of test cell equipment are given in Appendix A.

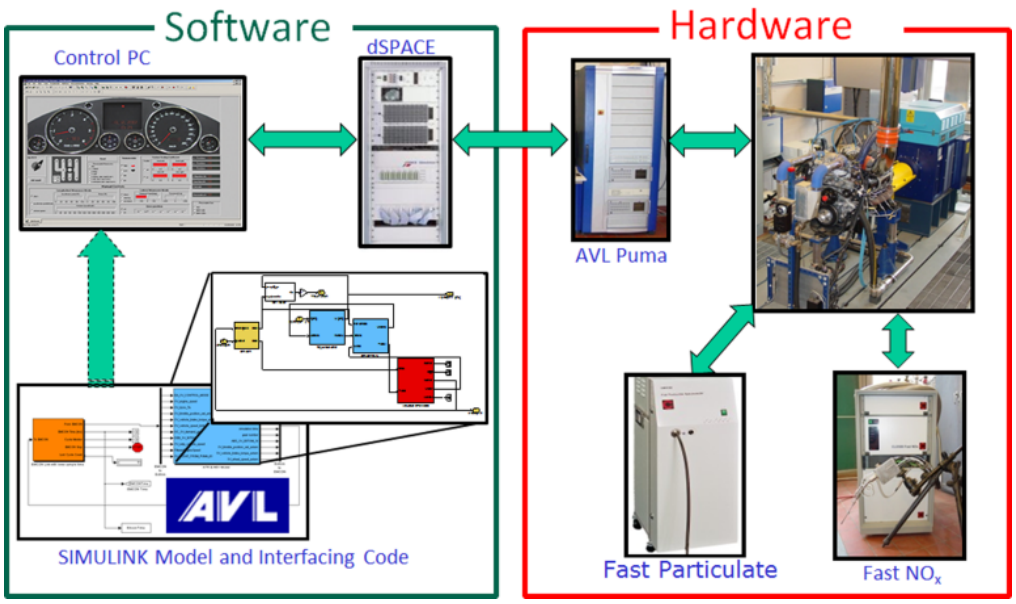


Figure 2.10: Engine-In-the-Loop test cell configuration.

To illustrate the communication flow in the EIL setup, assume a scenario where a conventional vehicle is simulated over driving schedule. Based on the present demanded velocity, the virtual driver applies appropriate acceleration/deceleration command depending on the difference between the present and demanded vehicle velocity. This engine command is sent to AVL PUMA and is processed into corresponding voltage signal. This signal is then commanded to engine control unit (ECU) which then controls

different actuators and servo-level loops onboard engine to regulate injection timing, injection amount, VGT vane position and EGR valve position. The engine responds by producing torque which is then measured by the dynamometer. This torque value is then relayed back to dSPACE via AVL PUMA and the virtual model calculates the change in vehicle speed due to the supplied engine torque. The updated vehicle velocity is subsequently translated to engine speed, based on current transmission states and this speed setpoint is sent back to dynamometer through AVL PUMA and dSPACE. The updated vehicle velocity is also provided to the driver for subsequent adjustment of the pedal command. The EIL architecture is flexible to allow evaluation of different powertrain architectures, like conventional and hybrids. Previous work done at the University of Michigan [97], [23], [42] allowed leveraging the EIL setup for investigations of a variety of powertrain architectures. In the context of hybrids, the driver command is sent to supervisory controller which then calculates the appropriate commands for the engine and the other components.

This dissertation focuses on the series hydraulic hybrid powertrain architecture. In this case, the dynamometer emulates the hydraulic pump. The real-time virtual powertrain models generate the desired speed/torque set points based on present system states and the supervisory controller outputs. The dSPACE gives the information to AVL PUMA which further relays the desired engine torque demand to engine and desired pump speed demand to dynamometer. This casualty and structure of signal commands is based on the previous work done by Kim et al. [8] and Filipi et al. [23]. An inverse model of engine, based on experiment, is used for translating the desired torque to nominal engine throttle command at any given engine speed. The desired engine speed is maintained by adjusting the load on the engine, i.e. by changing the dynamometer speed set point. The built-in dynamometer controller adjusts the load to achieve the desired speed set point. The response of dynamometer is close to real life hydraulic pump as the dynamometer uses model based dynamometer-inertia compensation to removes 70% of the inertia effect.

2.4.2 Real-Time Vehicle/Powertrain Models

The accuracy of EIL simulations is dependent on the fidelity of the real-time models. The models should capture the effect of engine transients on the vehicle while running in real-time on dSPACE platform. The real-time models used in EIL are high fidelity models used for vehicle simulation (refer section 2.3.1) with the exception of engine and hydraulic pump. The engine is replaced with physical engine and the pump is simulated by dynamometer. Replacing the hydraulic pump with dynamometer creates an interesting challenge of calculating the fluid flow into virtual accumulator based on measurements of real variables on EIL setup. The torque and speed measured by the dynamometer needs to be translated into fluid flow and fluid flow cannot be calculated without the information about displacement factor of pump. To overcome this model, the EIL setup uses an inverted model of pump, similar to one used in control oriented model for design of supervisory controller (refer section 2.3.2).

The cyber driver in EIL is augmented with driver preview for accurate tracking of speed profile and a lead compensator to compensate for communication lag in the EIL setup. The lead compensator is designed using loop shaping and details are available in work done by Filipi et al. [10].

2.4.3 EIL Integration Challenges

Concurrent running of virtual powertrain/vehicle models and physical engine requires addressing integration issues. The initial main challenges faced during EIL setup are connection causality, signal and communication delays and virtual driver response. Filipi et al. [97] gave a detailed description of these issues and steps taken to mitigate these challenges. Filipi and Kim [23] first emulated the series hydraulic hybrid configuration in EIL. They addressed two additional integration issues namely, power-generation control and calculation of fluid flow into the virtual accumulator based on measurements of real variables in EIL setup.

Implementing the policy optimization based controllers developed in this dissertation required dealing with additional issues. One of the major issues required dealing with different simulation time step between virtual models, power management controller and feedback signals from sensors. The power management controller is designed to be

updated at 1 Hz. This is done to reduce the computational time during design (refer section 2.3.2). The 1 Hz captures the relevant system dynamics and is similar to other energy management controllers in literature [43], [40], [8], [99]. The real time powertrain/vehicle simulation models, on the other hand, are simulated at 10^4 Hz on the dSPACE platform. The virtual transient emission sensors (refer Chapter 5) run at 10 Hz. Finally, the dSPACE communicates with AVL PUMA Open system at 10^3 Hz.

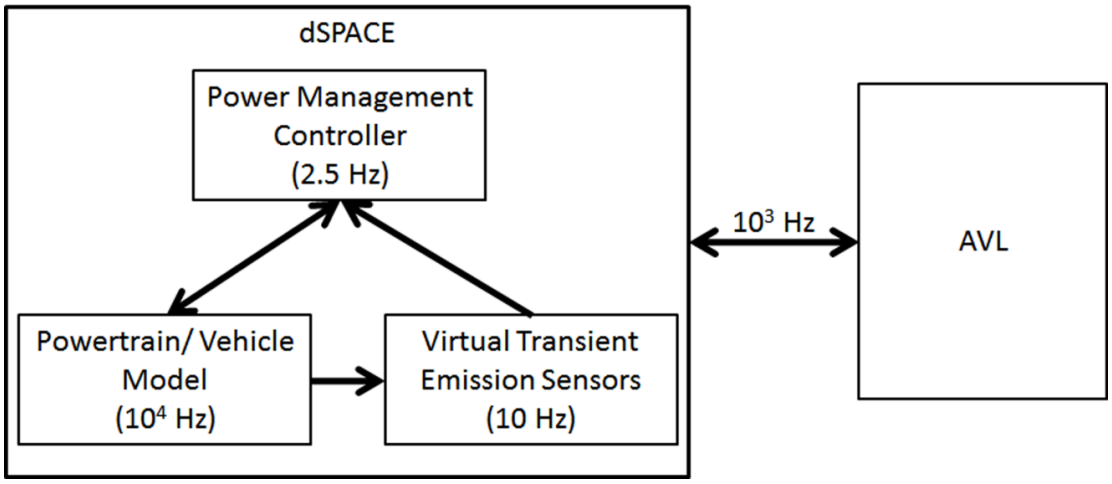


Figure 2.11: Different sampling rates.

A multi-rate updating scheme is required to handle the different time steps involved. The power management controller controls the engine torque and 1 Hz update rate is slow and noticable by driver. To give more pedal responsiveness, the power management controller is updated at 2.5 Hz. The propulsion motors are updated at default rate of 10^3 Hz to yield very fast pedal response. Figure 2.11 shows an overview of different update rates involved. The implementation of this multi-rate scheme is challenging. Care needs to be taken to maintain the data integrity during transfer between section of codes with different update rate.

Another issue is the presence of noise in the measured signals. Any measured signal in real world is usually corrupted with high frequency noise and requires signal conditioning before it can be used for feedback. Power management controllers based on policy optimization require state feedback. Filtering different state signals through same low pass filter will experience different amount of phase shifts based on the instantaneous frequency of the signal. This will result in power management controller receiving

modified state information and the state feedback gains from the controller will not be optimal. Special care is taken while designing the controller to increase the robustness of the controller to noisy signal. Final implementation in the EIL setup did not use any signal conditioning.

2.5 Supervisory Power Management

The power management controller in a hybrid vehicle supervises the engine and additional auxiliary power to improve efficiency and performance. This dissertation proposes different optimal power management controllers designed with stochastic driver model and a choice of objective criterion(s). To quantify the benefits of advanced policy optimization algorithms for controller design, a baseline rule based controller is presented next, i.e. the thermostatic controller [100] for comparison. The thermostatic controller is an intuitive engine-centric approach and resembles a "bang-bang" control. The thermostatic strategy is designed around the conventional wisdom of operating engine at the "sweet spot". The expectation is that running the engine at the most efficient point will be beneficial from the fuel economy standpoint. The next section briefly describes the thermostatic controller and gives the simulation results over different EPA driving schedules.

2.5.1 Thermostatic Controller

The thermostatic strategy is an engine-centric charging strategy designed around the SOC. Driver command is sent directly to propulsion motors and ensures that vehicle follows the desired velocity profile. The SOC is a sole variable used to control the engine.

Figure 2.12 shows the basics of SOC based thermostatic control. When SOC is above a threshold value e.g 0.4, the engine power demand is 0. As the SOC falls below this threshold value, the engine is brought online and asked to charge the accumulator. The dead band is implemented to prevent frequent switching between engine on/off states. Therefore, engine continues re-charging until SOC crosses the upper limit, i.e. $SOC_{threshold} + 0.15$. The engine idles if the SOC is above this upper limit. In the extreme

case of hard acceleration or hill climb, engine is operated at progressively higher power levels as the SOC keeps falling below the threshold. The maximum engine power is requested before the SOC falls below the absolute minimum for a given system.

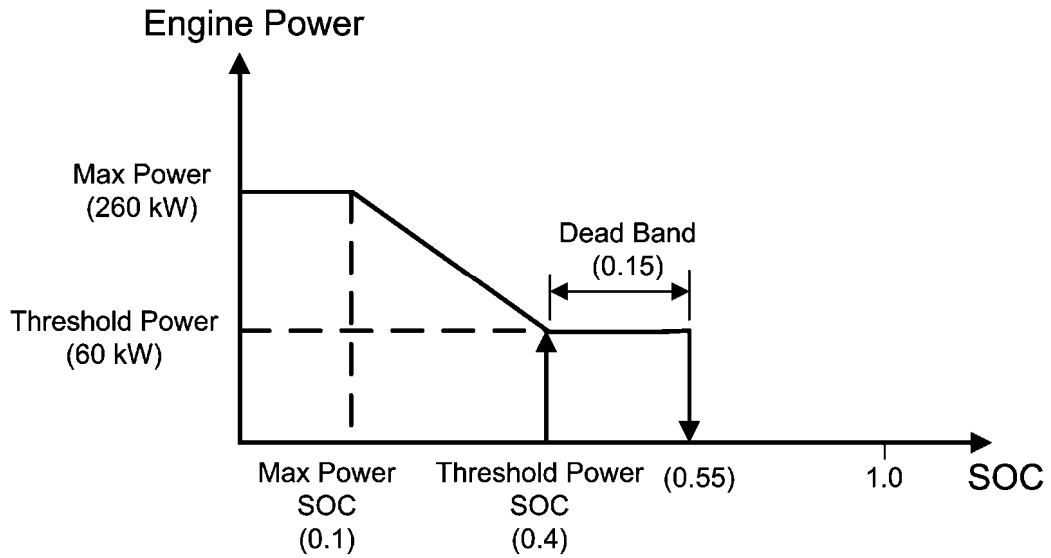


Figure 2.12 : Schematic illustrating the thermostatic power management concept.

An integral part of thermostatic controller design is the decision about engine operation during recharging, i.e. the threshold power ($P_{threshold}$) and the combination of engine speed/torque for a given power demand. The conventional wisdom suggests keeping the engine at the fuel efficient "sweet spot". The expectation is that running the engine at the most efficient point when charging the accumulator will be the best since the fuel energy conversion comes with small relative losses. This neglects the impact of pump efficiency. In addition, this leads to relatively aggressive charging since the "sweet spot" is close to peak torque. Keeping in mind a comparatively small accumulator storage capacity, the resulting system-level effects are short and frequent engine transients [100]. Very rapid engine accelerations consume energy, and engine trajectory passes through sub-optimal regions. At the same time, short and frequent recharging increases the engine idling time in between and magnifies the resulting penalty. Previous work done by Kim et al. [100] showed that the best threshold power from the system efficiency standpoint is not the "sweet spot". A systematic study involving parametric sweep of different threshold powers is done to determine the threshold power and is chosen to be 60 kW which is lower than the "sweet spot" as suggested by Kim et al.

[100]. The remaining issue is the impact of frequent engine ramp-ups on the exhaust emissions and this is exactly what this research aims to address.

2.5.2 Simulation Results for Thermostatic Controller

The series hydraulic hybrid with thermostatic controller is simulated over different EPA driving schedules. Figure 2.13 shows the system behavior over a section of federal urban driving schedule (FUDS). The thermostatic controller operates engine completely independent of vehicle power demand and engine operation is based on present SOC. The engine is switched on when SOC falls below 0.4 and is ramped to 60 kW of power. When SOC reaches 0.55, the engine returns to idling. During harsh acceleration, the engine power is progressively increased to maintain SOC from falling. During braking events, the motor is used to store braking energy in accumulator.

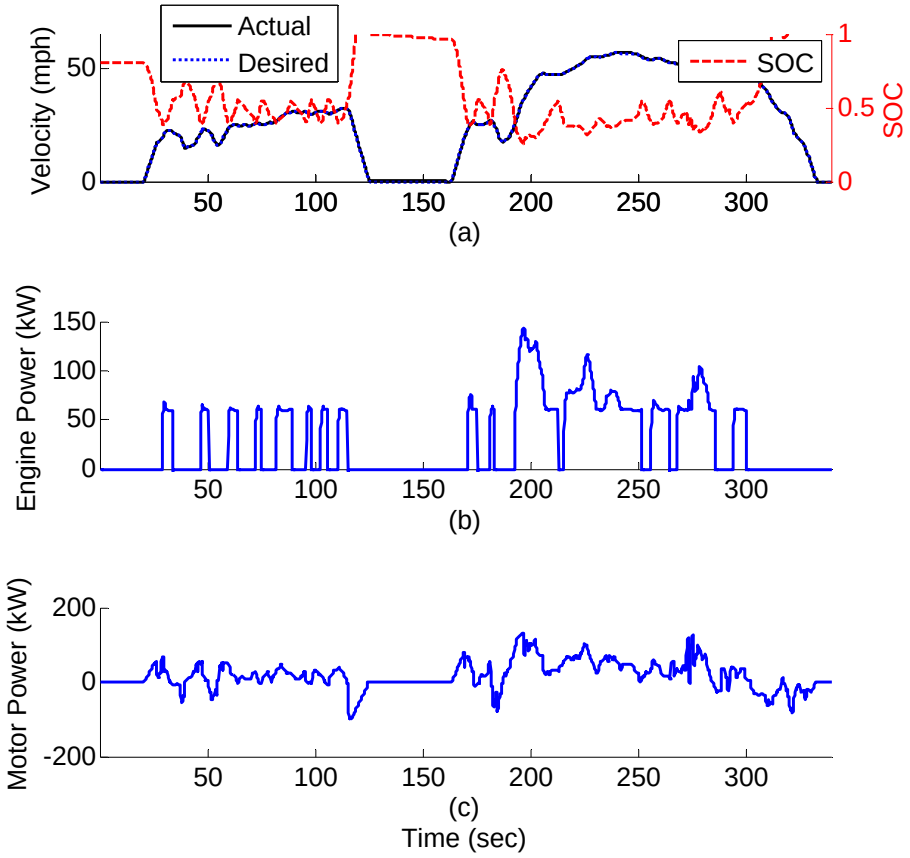


Figure 2.13: Simulated series hybrid powertrain behavior with the thermostatic control: a) vehicle speed and SOC during first 350 sec of FUDS, b) engine power demand for $P_{threshold}$ = 60 kW, and c) propulsion motor power.

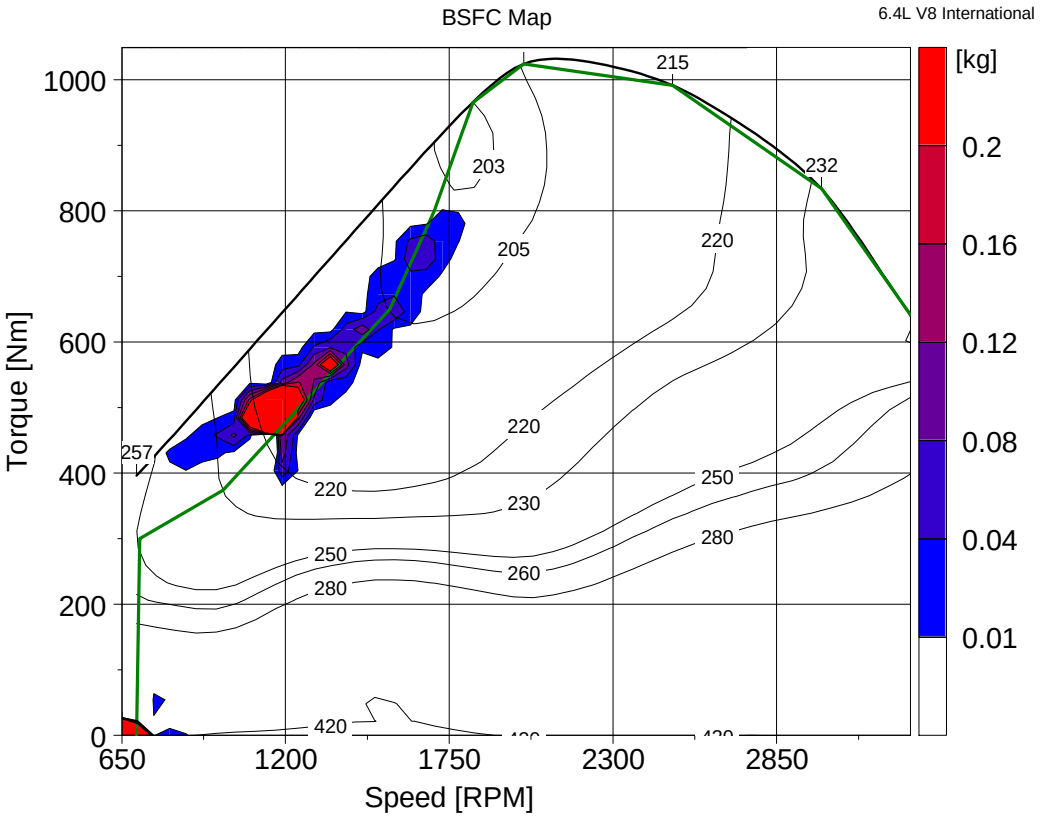


Figure 2.14 : Engine visitation points on the BSFC map, with a color scale indicating the relative amount of fuel consumed in a given zone during simulation over FUDS for thermostatic controller in S-HHV.

Figure 2.14 shows the engine visitation points for FUDS and indicates success in operating the engine along the best BSFC line. While the most frequently visited region is not the true "sweet spot", it is still in the zone of very low BSFC. Obviously, the loss of engine BSFC is relatively small and thus far outweighed by the system-level benefits. The vehicle fuel economy over the FUDS is more than 14.7 mpg, Table 2.2, around 40% improvement in fuel economy over conventional vehicle. The conventional vehicle is equipped with a 4-speed automatic gearbox and details can be found in [86] with the exception of the engine. The engine used by Kim et al. was a 6.0L medium-duty International with HEUI fuel injection system. The improvements predicted for highway driving cycle (HWFET) are not nearly as high as other driving cycles, but still tangible. This is due to lack of opportunities for regenerative braking. While this is certainly impressive, even the most careful parametric studies do not guarantee the optimum. The system-level effects are too complex for intuitive reasoning, hence the motivation for

investigating the horizon-based policy optimization algorithm presented in the next chapter.

Table 2.2: Fuel economy comparison for a conventional vehicle and S-HHV with thermostatic power management over EPA driving schedules

		FUDS	**HWFET**	**LA92**
4-Speed Conventional	MPG	10.59	12.55	9.27
	% Improvement	-	-	-
S-HHV with thermostatic controller	MPG	14.77	14.27	12.07
	% Improvement	39.4 %	13.7 %	30.2 %

Chapter 3

OPTIMAL POWER MANAGEMENT FOR A HYBRID VEHICLE

3.1 Introduction

The supervisory power management controller's main task is to orchestrate the engine and secondary power to meet the driver power demand. The supervisory controller has profound impact on system operation and the ultimate benefits of hybridization. Numerous strategies have been proposed for design of supervisory controller. These approaches can be categorized as heuristic, optimal and suboptimal. Heuristic strategies involve rule based [101], [16], [102] and fuzzy based [103], [25], [82] controllers which often rely on researchers' knowledge about individual system efficiencies. These strategies are intuitive and easy to implement. However, heuristic strategies strongly depend on choice of thresholds and cannot capture complex system level effects.

Optimal strategies aim to minimize an objective function, typically fuel consumption, over a given time horizon. Dynamic programming (DP) has been applied to numerically solve the optimization problem [6], [38], [39], [104], [37]. Optimal controllers, however, are inherently non-causal i.e. they require knowledge about the future driving conditions. This limits their practical applicability and requires rule extraction, which in turn sacrifices some of the fuel economy [7]. A suboptimal controller based on stochastic dynamic programming (SDP) eliminates the rule extractions step and allows direct development of an implementable control strategy for vehicle supervisory control. SDP is not dependent on a particular driving cycle but the statistical characteristics of many driving cycles and hence it is non-cycle-beating. It was previously applied to a parallel hybrid electric vehicle by Lin et al. [40] and Liu et al. [42], and a first attempt at addressing the series hydraulic configuration was pursued

recently by Kim [8] and Johri et al. [43]. The series hydraulic hybrid with its small energy storage creates difficult yet relevant problem for application of SDP.

In this chapter, SDP based algorithm will be employed for design of power management for series hydraulic hybrid. The controller will optimize the given objective over an infinite horizon. The essential part of the supervisory policy in any hybrid is a decision about splitting the vehicle power demand between the engine and the alternative power source. Previous implementation of SDP based controllers [40], [42], [8] for hybrid were designed with emphasis on system optimization i.e. SDP was used to solve for engine power demand and then this engine power demand was mapped to engine operating point based on best BSFC line. This approach though easier does not achieve the best possible results. The work presented in this chapter applies SDP to both engine and system optimization i.e. the decision about the threshold engine power is split into two decisions, about the engine speed and torque. The hope is that the algorithm may uncover a possible hidden "reserve".

This chapter deals with the development of supervisory controller using SDP for a series hydraulic hybrid vehicle. An infinite horizon discounted future power management problem with fuel economy objective is formulated. SDP is then applied to solve for optimal state feedback controller. Finally, results from SDP strategy are presented and is compared with baseline thermostatic controller. The SDP based supervisory controller establishes baseline for evaluating the future power management controllers designed using neuro-dynamic programming.

3.2 Stochastic Dynamic Programming

3.2.1 Problem Formulation

Given the vehicle, engine and powertrain configuration, an infinite horizon power management problem for a series hybrid vehicle can be formulated as

$$\begin{aligned} \text{Minimize:} \quad & J = \lim_{N\to\infty} \underset{w_k}{E}\left\{\sum_{k=0}^{N-1} \alpha^{\kappa} g\left(x_k, u_k, w_k\right)\right\} \\ \text{subject to:} \quad & x_{k+1} = f\left(x_k, u_k, w_k\right) \\ & \mathrm{x} \in \mathrm{X} \\ & \mathrm{u} \in \mathrm{U} \end{aligned} \tag{3.1}$$

where x is the state vector, u is the control input vector, w is the disturbance vector, g is the instantaneous cost function and $0 < \alpha < 1$ is the discount factor. Discount factor implies cost incurred at present is more important than incurred in the future. This also ensures that the J is finite over an infinite horizon.

3.2.2 Cost Function

The minimization problem of combined transient particulate matter and NO_X emissions with fuel consumption is a multi-objective optimal control problem. A single objective function can be defined by combining different objectives with weighting factor. The instantaneous cost function, g is function of states, control input and driver demand and is given by

$$\begin{aligned} g = w_{FC} \cdot FC(x_k, u_k) + w_{NOx} \cdot NO_x(x_k, u_k) + w_{PM} \cdot PM(x_k, u_k) \\ + \mu \cdot (SOC - SOC_{ref})^2 \cdot (SOC < SOC_{ref}) \end{aligned} \tag{3.2}$$

where FC is the normalized fuel consumption, NO_X is the normalized transient NO_X emission, PM is the normalized transient particulate matter emission. The w_{FC}, w_{NOx} and w_{PM} are the normalized weighting parameter and $\sum(w_{FC} + w_{NOx} + w_{PM}) = 1$. The weights in equation (3.2) can be varied to create a set of optimal control policies and generate a pareto optimality set.

The latter term in the above formulation penalizes the deviation of SOC below a threshold value. This penalty factor is different from the one used by Lin et al. [40] for HEV. In HEV, a penalty factor was added to the cost function to satisfy charge sustaining constraint and limit the operation of SOC within a narrow window due to battery health and operating considerations. Hydraulic accumulator does not suffer from similar constraints and SOC can vary over complete range. However, a lower bound on SOC is imposed to maintain vehicle drivability at all conditions. The above penalty function tries

to maintain low SOC reference value, e.g. 0.2 in this dissertation, to allow enough energy storage capacity for regeneration during braking.

The addition of transient emission objectives increases the system states considerably and requires numerical techniques to deal with the curse of dimensionality associated with dynamic programming based algorithms. In this chapter, only the fuel consumption is minimized. The problem can be treated as a subset of equation (3.2) with weight w_{FC} is set to 1 and weights w_{PM} and w_{NOx}set to 0. The complete problem with multi objective is solved in Chapter 4.

3.2.3 Constraints

The power management controller needs to satisfy certain bounds on states and control inputs. These bounds ensure that vehicle, engine and hydraulic devices do not operate in regimes, which are unfavorable from performance standpoint or are detrimental to health of these devices.

$$
\begin{aligned}
&\max(\omega_{e,\min}, \omega_{p,\min}) \le \omega_{e,k} \le \min(\omega_{e,\max}, \omega_{p,\min}) \\
&\omega_{p,k} = \omega_{e,k} \\
&SOC_{\min} \le SOC_k \le SOC_{\max} \\
&\omega_{wh,k} = \omega_{wh,des} \\
&\omega_{m,k} = \frac{\omega_{wh,k}}{FD \cdot GR} \\
&T_{e,\min}(\omega_{e,k}) \le T_{e,k}(\omega_{e,k}) \le T_{e,k}(\omega_{e,k}) \\
&T_{m,\min}(\omega_{m,k}, SOC_k) \le T_{m,k} \le T_{m,\max}(\omega_{m,k}, SOC_k) \\
&T_{p,\min}(\omega_{p,k}, SOC_k) \le T_{p,k} \le T_{p,\max}(\omega_{p,k}, SOC_k)
\end{aligned}
\tag{3.3}
$$

Lower engine speed is constrained between the maximum of minimum engine speed and minimum pump speed, and upper engine speed is constrained between minimum of maximum engine speed and maximum pump speed. Both engine and pump have same minimum speed of 0 and engine has lower maximum speed. However, the minimum engine speed is not set to 0 even though series architecture allows for shutting down engine during idling. The EIL facility does not have the capability of shutting down engine during experiments and to keep consistency between controllers evaluated in test cell and simulation, minimum engine speed is set to idling speed. Hence, the engine

speed is constrained by idling speed from bottom and maximum rated speed from top. The pump and engine are physically connected in series architecture, which enforces the speed of pump equal to engine speed. The motor speed is constrained to be proportional to vehicle speed.

3.2.4 Driver Model

The driver is modeled as a discrete Markov process and is used to generate future power demands, given present states. The driver acceleration and braking command is interpreted as power demand to be satisfied by the powertrain. The driver demand, P_{dem} is modeled as a stationary discrete-time stochastic process and a stationary Markov chain is used to generate driver demand, which is discretized as

$$P_{dem} = \left\{ P_{dem}^{1}, P_{dem}^{2}, ..., P_{dem}^{N_p} \right\} \tag{3.4}$$

The dynamics of driver power demand is assumed to be

$$P_{dem,k+1} = w_k \tag{3.5}$$

where the probability distribution of w_k is assumed to be

$$\begin{aligned} & p_{ij,l} = \Pr\left\{ w = P_{dem}^{j} \mid P_{dem} = P_{dem}^{i}, \omega_{wh} = \omega_{wh}^{l} \right\} \\ & i, j = 1, 2, ..., N_p \qquad l = 1, 2, ..., N_\omega \end{aligned} \tag{3.6}$$

where $p_{il,j}$ represent the one-step transition probability, N_p is the cardinality of driver power demand and N_ω is the cardinality of ω_{wh} vector.

Using vehicle/powertrain model and the sampled driving cycles, sequence of observation (P_{dem}, ω_{wh}) are calculated which are then mapped on to a sequence of quantized states. The transition probability is then calculated using maximum likelihood generator.

$$\hat{p}_{ij,l} = \frac{m_{ij,l}}{m_{il}} \tag{3.7}$$

where $m_{ij,l}$ is the number of times P_{dem}^{i} to P_{dem}^{j} transition occurs for given wheel speed, ω_{wh}^{l} and $m_{il} = \sum_{j=1}^{n} m_{ij,l}$ is the number of times that state i has occurred. The summation $\sum_{l=1}^{n} \hat{p}_{ij,l} = 1$ should always be satisfied.

In this dissertation, naturalistic driving cycles (Figure 3.1), based on actual driving behavior of randomly selected drivers in South East Michigan [105], [106], are used for generating transition probability matrix, Figure 3.2.

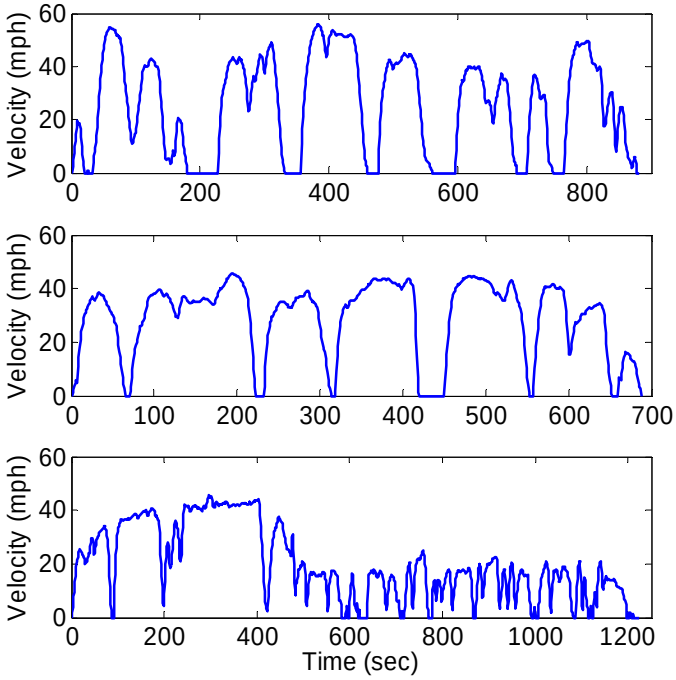


Figure 3.1: Naturalistic driving cycles recorded during typical commutes in SE Michigan.

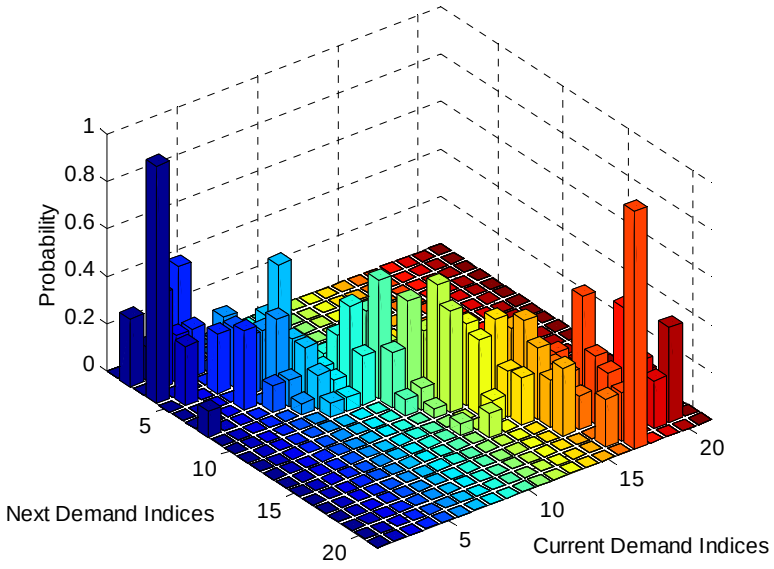


Figure 3.2: Transition probability of driver power demand for a particular wheel speed, ω_v = 54 rad/s derived from naturalistic driving schedules.

3.2.5 Stochastic Dynamic Programming Algorithms

Stochastic Dynamic Programming (SDP) provides a framework to solve for optimal value function in equation (3.1) and from which an optimal policy can be derived. The *Bellman's equation* or *Hamiltonian-Jacobi-Bellman equation* is a special consistency condition that the optimal value function has to satisfy.

$$J^*(i) = \min_{u \in U} E\left[g(i,u,j) + \alpha J^*(j) \mid i,u \right], \ \forall i, \ \ i,j \in X \tag{3.8}$$

where *J** is the *optimal value function* or *cost-to-go function*, *i* is the present state, *j* is the subsequent state to *i*, *g(.)* is the cost incurred to go to state *j* from state *i* under control *u*, and *E[·|i,u]* is the expected cost with respect to *j*, given *i* and *u*.

The goal of SDP algorithm is to numerically calculate the optimal cost-to-go function *J**. The Bellman's equation can be solved using classical dynamic programming techniques; value iteration and policy iteration algorithms.

Value Iteration Algorithm is a principal method for calculating the optimal cost-to-go vector *J**. The algorithm starts with some initial *J* and iterates over following equation until J^k converges.

$$J^{k+1}(i) = \min_{u \in U(i)} \left(g(i,u,j) + E\left[\alpha J^k(j) \mid i,u \right] \right) \quad \forall i \tag{3.9}$$

where *k* is the iteration number, and *j* is the next state subsequent to *i*. Generally, the value iteration algorithm takes infinite iterations to converge. The value iteration method updates the estimate of cost-to-go vector for all states simultaneously. An alternative is to use *Gauss Seidel iteration* i.e. to update cost-to-go at one state at a time and incorporating this computation for calculation of cost-to-go for next subsequent states.

Policy Iteration Algorithm is an alternative to value iteration algorithm, which is guaranteed to converge within finite steps. The algorithm starts with an initial stationary policy π_0 and generates a sequence of updated policies π_1, π_2 … with every iteration. The policy iteration algorithm iterates between a policy evaluation step and a policy improvement step until the cost-to-go function converges to *J**. In policy evaluation step, given a policy π_k, J^{π}_{k+1} is calculated by solving linear set of equations

$$J^{\pi}_{k+1}(i) = g(i,\pi(i),j) + E\left[\alpha J^{\pi}_k(j) \mid i,u \right] \qquad \forall i \tag{3.10}$$

where k is the iteration number, and j is the next state subsequent to i given the control input u. The policy improvement step is evaluated next and updated policy π_{k+1} is calculated.

$$\pi_{k+1}(i) = \arg\min_{u \in U(x^i)} \left[g(i,u,j) + E\left[\alpha J_{k+1}^{\pi}(j) \mid i,u \right] \right] \qquad \forall i \tag{3.11}$$

where J^{π} is the approximate cost function obtained from the policy evaluation step.

For problems with large number of states, solving linear set of equations by either inversion or *Gauss elimination method* is very computational and time consuming. To alleviate this problem, the linear set of equations can be solved using value iteration. This modified algorithm is known as ***Hybrid Policy/Value Iteration Algorithm***. This chapter employs hybrid policy/value iteration algorithm to derive optimal cost function and corresponding optimal control policy.

3.2.6 Actor-Critic System

An interesting interpretation of policy iteration method is to view the algorithm as an *actor-critic* system, Figure 3.3. The policy evaluation step is viewed as the work of *critic*, who evaluates the performance of current policy, i.e. given a proper policy π, the corresponding cost function $J^{\pi}(x)$ is calculated by iteratively updating the Bellman equation

$$J_{k+1}^{\pi}(x^i) = g(x^i, \pi(x^i), w^i) + \underset{w}{E}\left\{ \alpha J_k^{\pi}(x^j) \right\} \qquad \forall i \tag{3.12}$$

where k is the iteration number, and x^j is the new state, i.e., $x^j = f(x^i, \pi(x^i), w^i)$. The policy improvement step is viewed as the work of an actor, who takes into account the latest critique from critic and acts on the improved policy, i.e. based on present cost function J_{k+1}^{π} the improved policy π_{k+1} is found through the following equation

$$\pi_{k+1}(x^i) = \arg\min_{u \in U(x^i)} \left[g(x^i, u, w^i) + \underset{w}{E}\left\{ \alpha J_{k+1}^{\pi}(x^j) \right\} \right] \qquad \forall i \tag{3.13}$$

where J_{k+1}^{π} is the approximate cost function obtained from the policy evaluation step.

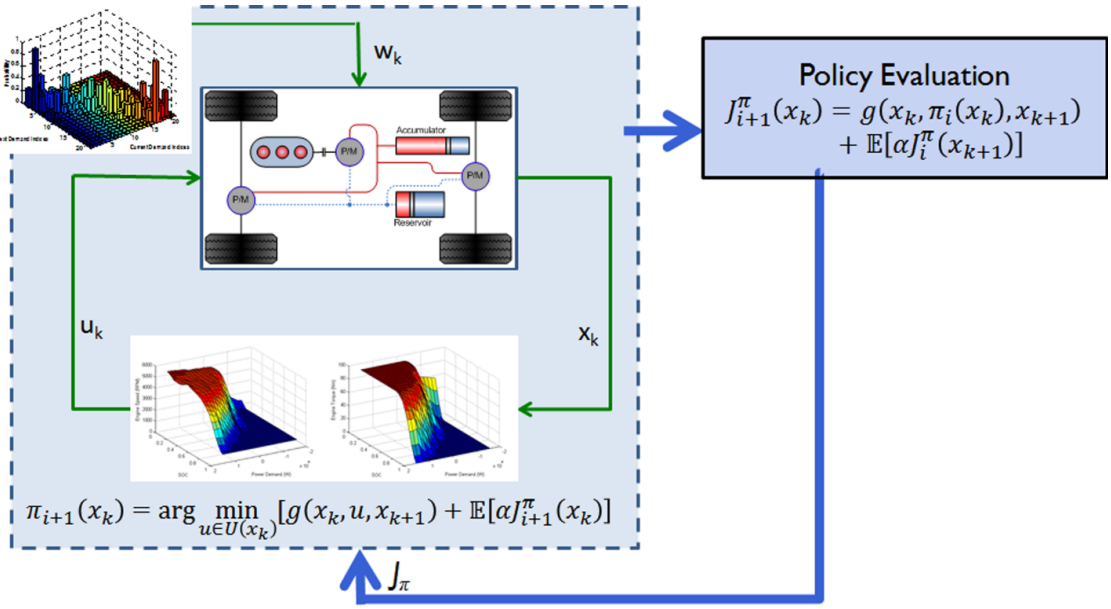


Figure 3.3: Interpretation of hybrid policy iteration as an actor-critic system.

The policy evaluation step is repeated with new policy to update the cost function. This iterative process is repeated, until J^{π} converges within a selected tolerance level. The resulting control policy, J^{π} is stationary i.e. does not change from one stage (time) to the next and can be implemented in controller as a time invariant state feedback law.

3.2.7 Numerical Techniques to Reduce Computational Effort

Dynamic programming based algorithms are computationally intensive. The computational cost and memory requirement grows exponentially with increase in number of states. This section gives a brief overview of steps taken to reduce the computational effort and make the complex problem numerically tractable. These efforts are not enough to solve the complete multi-objective problem with transient emissions and fuel consumption. A new algorithm will be proposed in Chapter 4 to deal with problems that are more complex. However, these techniques are still indispensable and will still be required with more efficient algorithm in Chapter 4.

- **Compiled code:** The Matlab is an interpreted language with Just-in-Time (JIT) compiler for speeding up certain calculations. To further speedup calculations, custom codes written in C and C++ and compiled as MEX files are interfaced

with Matlab. This gave huge reduction in computation time. Specifically, sections of code that required multiple for-loops and are frequently called through SDP routine are coded in C++.

- **Profiler:** Profiler is a handy tool, which allows calculating the amount of resources used by each function call and section of code. Profiler is used extensively to figure out bottlenecks in code and code is re-written in more efficient manner.
- **Error checking:** The code is written with single purpose to solve the above problem and, sacrifices error checking and generalization features for speed.
- **BLAS/LAPACK routines:** The C and C++ code is written to use BLAS/LAPACK math libraries. These libraries are platform specific and optimized for speed.
- **Cluster computer:** The code is written with parallel structure to effectively use multiple computing nodes available through Center of Applied Computing at the University of Michigan.
- **Vectorization:** The model equations are written with vector algebra. The code accepts vector of control inputs and gives vector of outputs, as opposed to using for-loops in Matlab. Matlab code is highly optimized to run faster with vector and matrix operations. All the user-defined functions in SDP algorithm are written with this capability in mind.
- **Preallocation:** The memory space is preallocated to reduce computational overhead with memory initialization. Care is taken to prevent vector grow dynamically inside a for-loop.
- **Overhead reduction:** Simulink is a popular tool for solving dynamical systems. However, Simulink has an overhead computational cost with every function call to Simulink from Matlab. Since the dynamical model is called multiple times during SDP, the entire model is coded in Matlab to reduce this overhead cost.

3.3 SDP with Fuel Economy Objective

This section focuses on the design of optimal power management for a series hydraulic hybrid with only fuel economy as the objective. The problem is formulated to find optimal control law, which minimizes the fuel consumption over infinite horizon. The instantaneous cost function is given by

$$g = FC(x_k, u_k) + \mu \cdot (SOC - SOC_{ref})^2 \cdot (SOC < SOC_{ref}) \tag{3.14}$$

where FC is the fuel consumption at given state x_k and control input u_k. The instantaneous cost is based on the overall system efficiency. The power generation unit includes engine and pump subsystem. The pump efficiency is a function of SOC and hence the overall system efficiency changes with SOC. This is a key difference between approach in this work and previous approaches. The latter term penalizes the SOC below a set value and is explained in section 3.2.2.

The system is modeled with 3 states x_k (with little abuse of notation, the disturbance vector w_k is included in the state vector) and 2 control inputs u_k.

$$x_k = \begin{cases} SOC \\ \omega_{wh} \\ P_{dem} \end{cases} \tag{3.15}$$

$$u_k = \begin{cases} \omega_e^{dem} \\ T_e^{dem} \end{cases} \tag{3.16}$$

Previous work on SDP based supervisory controller for hybrids treated engine power demand as the sole control decision [40], [8]. The engine is then asked to provide the power by operating along the best BSFC line. In this study, engine is not restricted to operate along the best BSFC line, since the insight from the previous thermostatic SOC study [100] clearly indicates that system-level gains may offset a small loss of BSFC. The intention is to provide an additional degree of freedom and allow the algorithm to discover the best way to run the power generation subsystem. Hence, the controller produces desired set point for both engine speed and torque, based on given states of the vehicle; driver power demand, SOC and vehicle speed.

The SDP results are captured in time invariant state feedback lookup tables, which can be directly implemented in the vehicle simulation platform. Figure 3.4 and Figure 3.5

show a set of representative lookup tables for desired engine speed and desired engine torque for a particular wheel speed. The set points guarantee optimal operation of the whole system, namely engine and pump, rather than just engine. This is a key distinction between this work and the previous attempt [8]. To differentiate between two approaches, this approach is referred to as system centric SDP while the previous work is referred to as engine centric SDP.

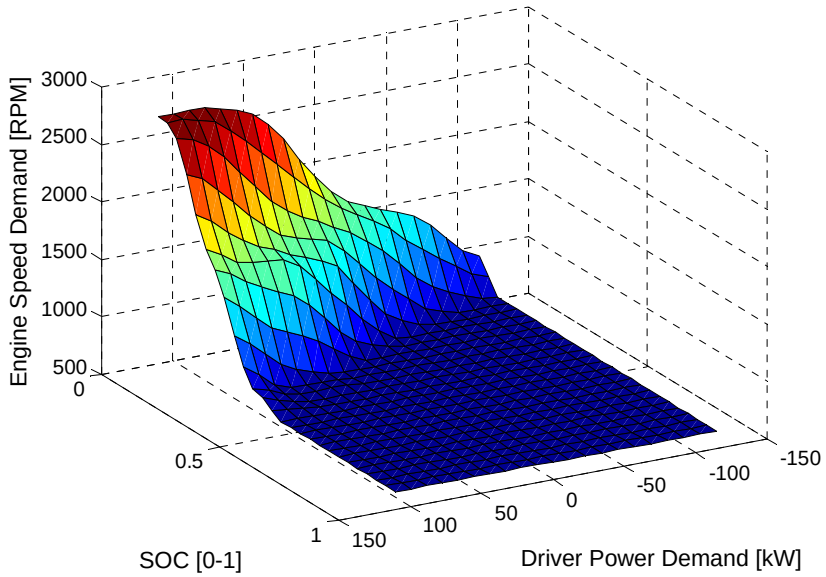


Figure 3.4 : Optimal SDP engine speed policy (ω_{wh} = 54 rad/s).

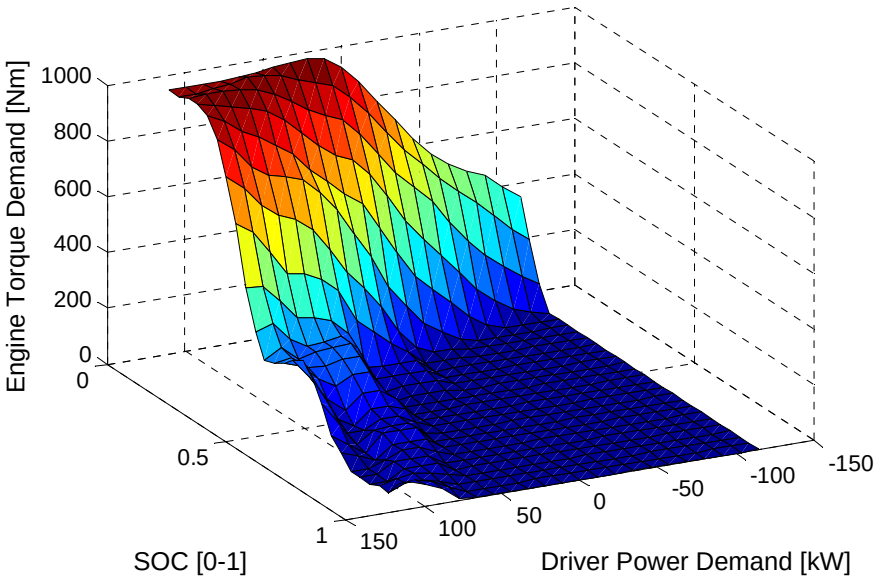


Figure 3.5 : Optimal SDP engine torque policy (ω_{wh} = 54 rad/s).

3.3.1 Simulation Results for SDP Controller with Fuel Economy Objective

Figure 3.6 and Figure 3.7 show the engine operation over FUDS driving cycle. It can be seen from Figure 3.6 that engine operation departs from the best BSFC line. The proposed system centric SDP policy operates the engine to maximize the system efficiency i.e. combined engine and pump efficiency rather than just the engine operation.

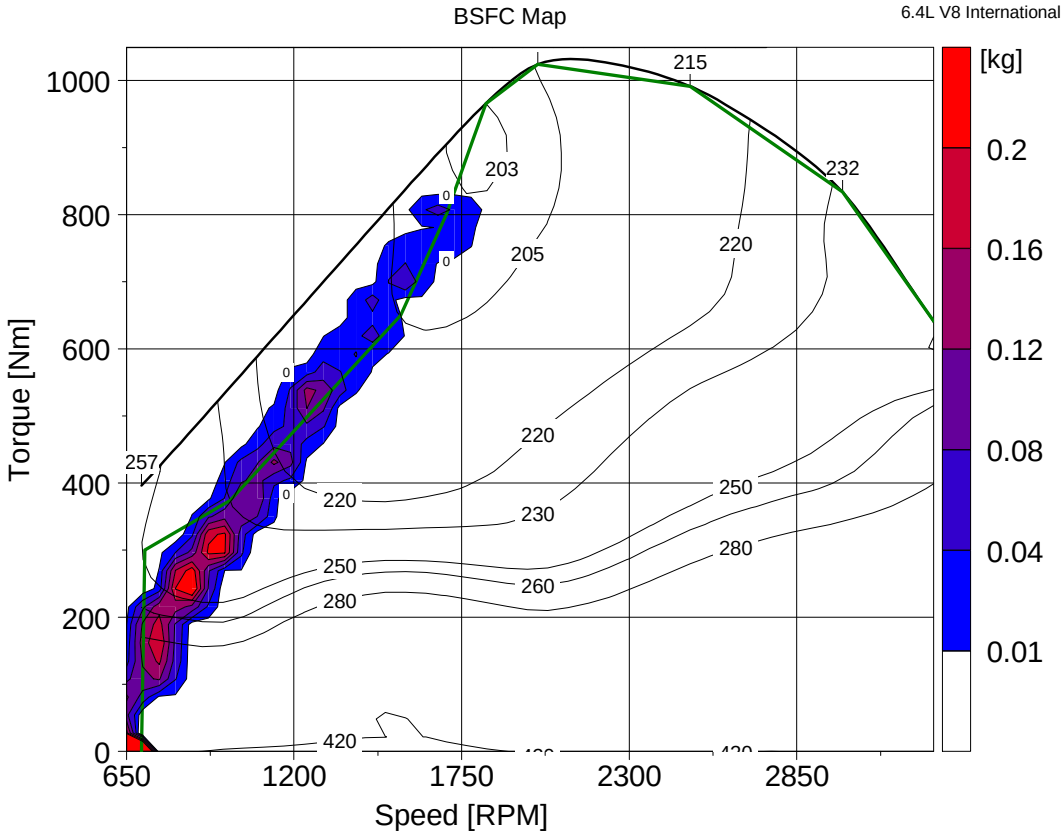


Figure 3.6: Engine visitation points on the BSFC map, with a color scale indicating the relative amount of fuel consumed in a given zone during simulation over FUDS for system centric SDP controller in S-HHV. The controller output is demanded engine speed and demanded engine torque.

It can be seen from Figure 3.7 that SDP based controller does a good job in maintaining low SOC throughout the driving cycle. The controller uses hydraulic power for vehicle propulsion at high SOC values (engine demand is zero, Figure 3.7b). As SOC drops, engine is ramped up and produces enough power to maintain the desired value, 0.2 in this case. This allows maximum regeneration capability during braking events. The engine operation resembles load-following mode, except the speeds are much lower than

in the case of a mechanical transmission, hence pushing the loads up into the high-efficiency region.

Table 3.1 shows the fuel economy of series hydraulic hybrid over different EPA driving schedules along with percent improvement over the conventional vehicle. It includes predictions obtained with SDP based supervisory controller designed with engine centric and with system centric strategies. The fuel economy of the S-HHV is much better than the baseline conventional with either control strategy. The advantage of the S-HHV over the highway cycle is smaller, but still tangible.

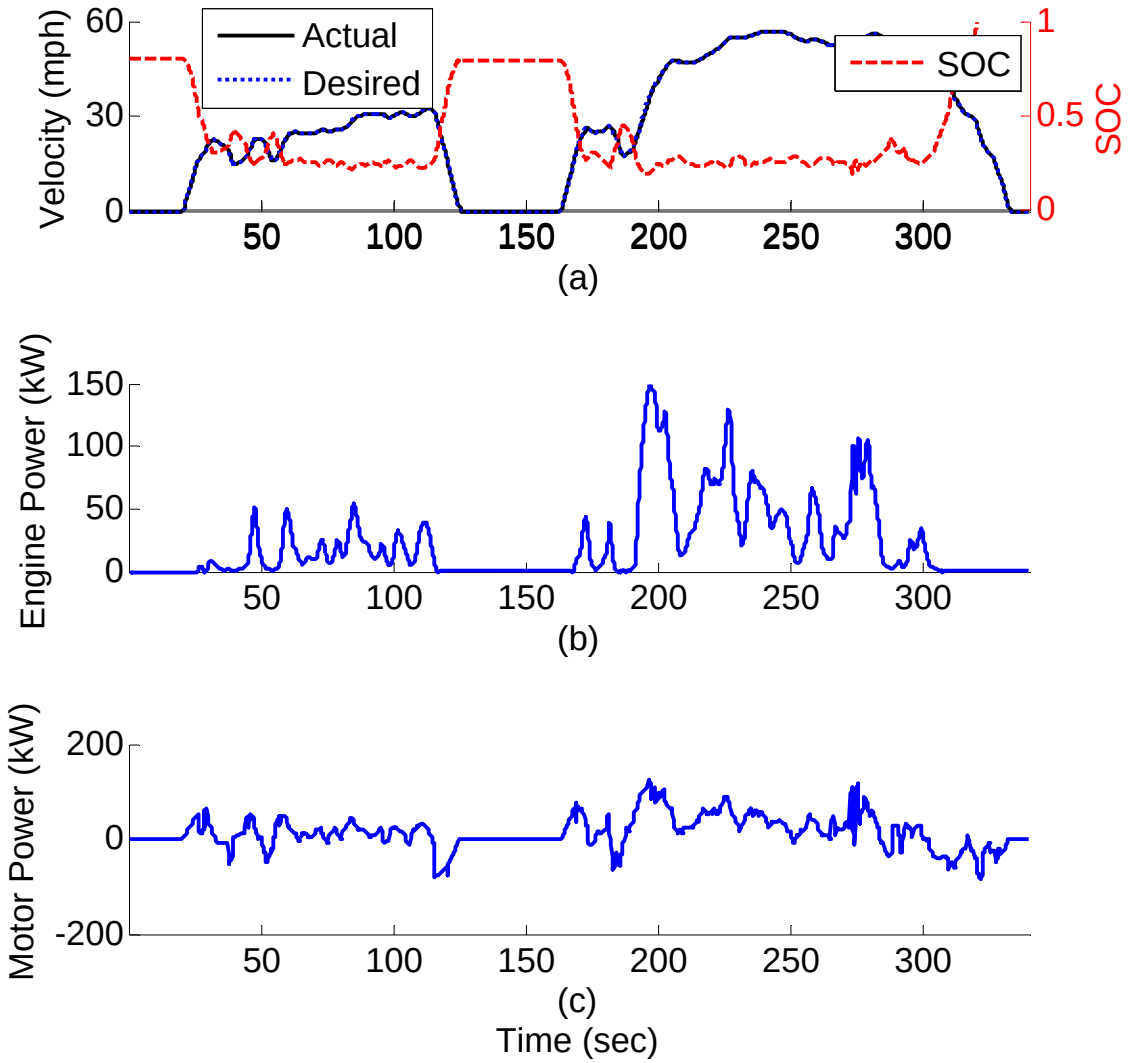


Figure 3.7: Simulated series hybrid powertrain behavior with the system centric SDP control: a) vehicle speed and SOC during first 350 sec of FUDS, b) engine power, and c) propulsion motor power.

To highlight the better fuel economy with SDP based controllers, Table 3.1 includes the fuel economy results for thermostatic controller. The SDP controller is 20% more efficient by effectively managing the engine and hydraulics. The SDP strategy differs from the thermostatic in a very important way that would give it an edge in a practical application. The engine operation is closer to load-following, but with higher load and lower speed operation than in the case of the vehicle with mechanical transmission. In other words, the transients are relatively mild, but related to vehicle performance. This would result in a much better driver feel than occasional bursts of power experienced with the thermostatic controller. In addition, the likelihood of demonstrating the same results with real hardware is higher in case of the SDP than thermostatic control. Previous work on engine-in-the-loop testing by Filipi et al. [23] has shown excellent agreement between predictions and experiments in case of milder engine operation, and tangible discrepancies in case of the bang-bang control. The penalties associated with rapid transients (energy for acceleration, excursions of operating parameters) are obviously not fully captured with a system-level simulation. This, together with the drivability considerations and easier management of the aftertreatment system makes the SDP a preferred option. This would also be a very robust controller, since SDP produces directly implementable state-feedback lookup tables.

Table 3.1: Fuel economy and % improvement for a S-HHV with different power management strategies compared to conventional vehicle as baseline over different EPA driving schedules

		FUDS	**HWFET**	**LA92**
S-HHV with thermostatic controller	MPG	14.77	14.27	12.07
	% Improvement	39.4 %	13.7 %	30.2 %
S-HHV with SDP based controller (engine centric)	MPG	17	17.1	13.9
	% Improvement	60 %	36.2 %	49 %
S-HHV with SDP based controller (system centric)	MPG	17.47	17.4	14.2
	% Improvement	65 %	38.6 %	53.2 %

The proposed SDP controller combines the higher-level power management strategy with lower level control decision to divide demanded power from engine into demanded engine speed and demanded engine torque. The engine operation with proposed SDP controller is different from the SDP controller proposed by Lin et al. [40]. The SDP controller proposed by Lin et al. [40] relied on the best BSFC trajectory to divide demanded power demand into demanded engine speed and demanded engine torque. Figure 3.8 shows the engine visitation points over FUDS for series hydraulic hybrid vehicle with engine centric SDP controller with demanded engine power output. Combining higher-level power management strategy for power demand from engine with lower level controller for calculating demanded engine speed and demanded engine torque results in additional 3-5 % gain in fuel economy (Table 3.1).

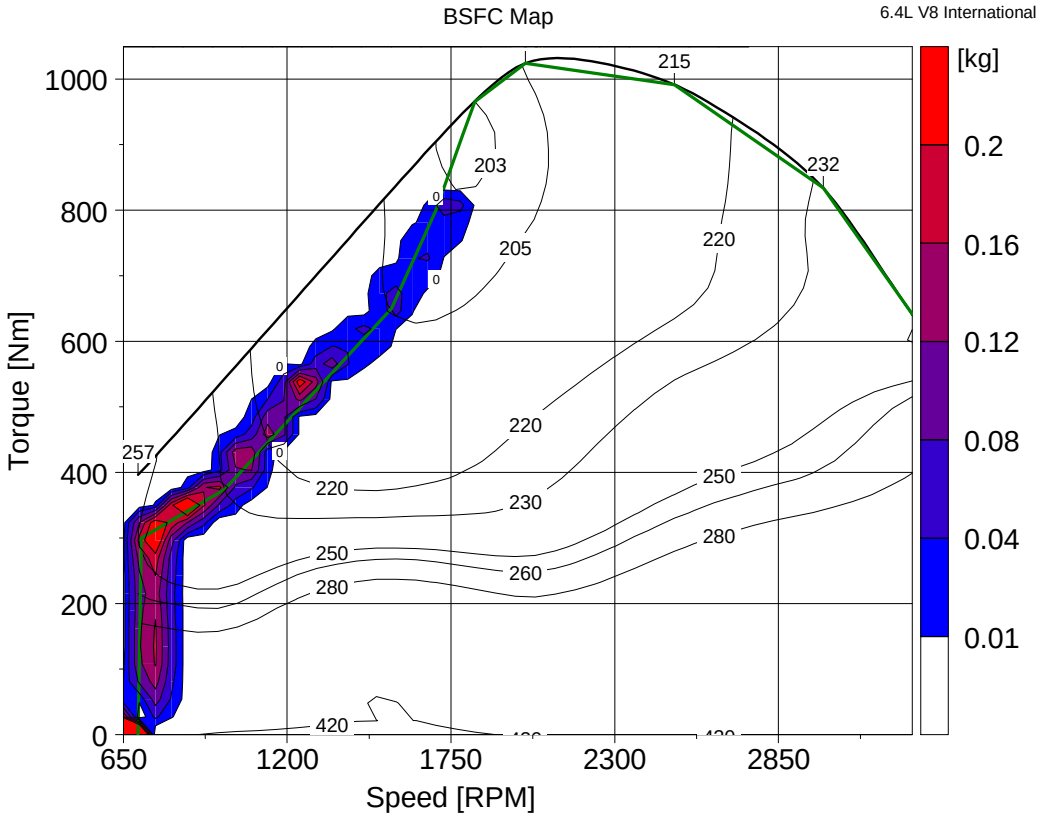


Figure 3.8: Engine visitation points on the BSFC map, with a color scale indicating the relative amount of fuel consumed in a given zone during simulation over FUDS for engine centric SDP controller in S-HHV. The controller output is demanded engine power.

The SDP algorithm presented in this chapter is an effective tool for generating optimized power management policy. However, the multi-objective problem of minimizing combined fuel consumption and transient emissions with cost function outlined in section 3.2.2, is computationally intractable with the proposed SDP algorithm. Even with the sophisticated numerical techniques outlined in section 3.2.7, the computational and memory requirements of such a problem far outweighs the resources available in a cluster computer. A new class of algorithms is required to tackle such a problem.

Chapter 4

NEURO-DYNAMIC PROGRAMMING

4.1 Introduction

Problems of sequential decision making under uncertainty (stochastic control) are encountered in wide array of fields, like optimal control design, operations research, and machine learning. Dynamic Programming (DP) is an effective tool for such problems and allows determination of optimal solution using constrained nonlinear model based systems. In addition, it allows for inclusion of multiple objectives during policy optimization. Theoretically, the hybrid power management controller can be designed using DP with multiple objectives in consideration, like reduced transient emissions, reduced noise vibration harshness (NVH) along with the original fuel economy objective. A detailed modeling of multiple phenomena involves increase in system/plant states and the space spanned by the states grows exponentially. This in turn results in exponential growth in computational/memory resources required to calculate optimal solution. This is widely known as *curse of dimensionality* of dynamic programming. The curse of dimensionality is not only restricted to state space but can also arise from action and decision spaces [107]. Figure 4.1 shows the exponential rise in computational demand with increase in space spanned by states. Therefore, classical dynamic programming algorithms are only applicable to problems with few thousand state counts with the present computational resources and this effectively limits the number of states to around 3 with discretization level of approximately 20 each.

This chapter focuses on development of algorithm that produces near-optimal policy with reasonable amount of computational resources. The idea centers on evaluation and approximation of optimal cost-to-go function with neural networks. At the center of this approach is a self-learning neural network, which adapts over time to reflect the optimal

cost-to-go function. The approach is hence called *Neuro-Dynamic Programming (NDP)*. Two critical ideas in NDP approach are

1. **Compact functional representation of cost-to-go.** The main idea is to represent cost-to-go values with a functional representation. This reduces the amount of memory required to store cost-to-go values. If the cost-to-go values have underlying characteristic, the functional approximation can exploit the behavior and predict cost-to-go values for complete state space without being explicitly trained for each state.
2. **Recursive method for updating the cost-to-go functional approximation based on successive observation of state transition and associated cost.** The other important concept of NDP is of recursively updating functional approximation. Sutton et al. [108] proposed temporal difference learning as a method for approximating long-term future cost as a function of present state. The algorithm improves the approximation of the long-term cost as more and more state transitions are observed in an incremental fashion. This approach reduces the computational burden of the DP algorithm.

The proposed algorithm is a type of reinforcement learning (RL). RL is learning by an agent to accomplish a particular task through trial-and-error interactions with environment based on reinforcement signals from the environment [108]. Reinforcement learning is different from supervised learning which is widely used in many fields, such as artificial neural networks, and statistical pattern recognition. Supervised learning involves agent learning to perform a certain task after training from a knowledgeable supervisor. In supervised learning, the agent does not learn from its interaction with the environment. In contrast, a general RL model includes an agent (controller) that interacts with environment (system) over a sequence of discrete steps. The agent, based on the state of environment, selects and takes an action, u (controller output) according to a given policy, π and incurs an instantaneous cost, g. The goal of the agent is to minimize (maximize) the cost over time (objective function). A policy's value function gives the expected return if a given agent uses that policy.

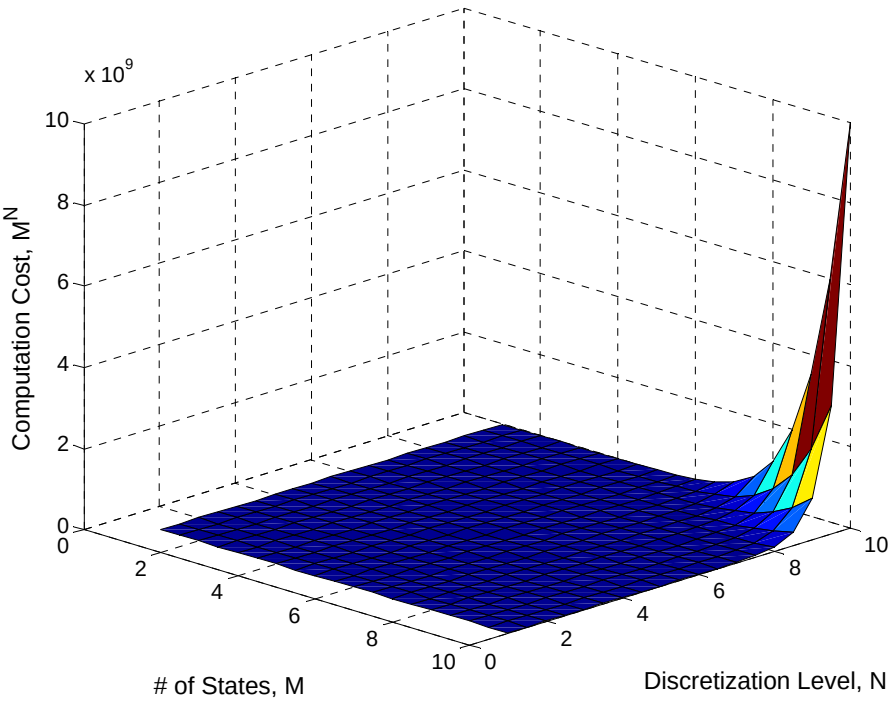


Figure 4.1: Curse of dimensionality associated with dynamic programming.

The chapter primarily focuses on neuro-dynamic programming and temporal difference learning techniques and their application to solve power management problem in hybrid vehicles. The chapter begins with detailed description of NDP and the algorithm for designing power management controller. To validate the NDP and demonstrate the effectiveness of the algorithm to learn optimal policy, the power management problem defined in section 3.3 is solved using NDP. Next, the original multi objective problem with fuel economy and transient emission is reformulated. This problem has a very large state-action space and NDP is successfully applied to design a self-learning neural controller. Finally, the controller is evaluated over different EPA driving schedules and the simulation results are discussed.

4.2 Neuro-Dynamic Programming

Neuro-Dynamic Programming (NDP) is also known as Approximate Dynamic Programming (ADP) by many researchers. The term “approximate” in ADP comes from the fact that the method centers on approximation of optimal cost-to-go function. In

particular, the optimal cost-to-go function $J^*(\cdot)$ is replaced with suitable approximation $\tilde{J}(\cdot, r)$ where r is a vector of parameters. Hence, the representation can be considered as mapping of higher dimensional cost-to-go vector, $J \in \mathbb{R}^n$ using a lower dimensional parameter vector, $r \in \mathbb{R}^m$ $(m \ll n)$.

$$\pi^*(i) = \arg\min_u E\left[g(i,u,j) + \alpha \tilde{J}(j,r) \mid i,u \right] \tag{4.1}$$

The above equation is modified Bellman's equation with J replaced by mapping $\tilde{J}: \mathbb{R}^m \rightarrow \mathbb{R}^n$. The methods for solving modified Bellman's equation are mostly derived from policy iteration algorithm. The algorithm generates a sequence of policies, $\pi = \{\pi_1, \ldots, \pi_k\}$ with every iteration and the corresponding estimate of cost-to-go is calculated using compact representation, $\tilde{J}(\cdot, r)$.

The function $\tilde{J}$ is called the scoring function and the value $\tilde{J}(j,r)$ is called the score of state j [109]. In most problems, optimal cost-to-go is a highly complicated function of states. A compact representation by a scoring function attempts to breaks this complexity. However, an important issue is the selection of compact representation with a tradeoff between complexity and size. Some of the architectures in literature [109], [110], [111] are:

1. **Feature mapping:** In a feature based compact representation, each component $\tilde{J}_i$ of the scoring function, $\tilde{J}$ is a function of some feature vector, $f(i)$ and parameter vector r but not an explicit function of state i

   $$f(i) = \left(f_1(i), \ldots, f_m(i) \right) \tag{4.2}$$

   where m is the cardinality of feature space. The features can be constructed heuristically or through optimization.

   - **Lookup table:** The feature space is represented by lookup table and the parameter vector, r contains one component for each possible feature vector. With effective feature extraction, many states can be associated with one feature vector. The feature space will be a smaller than total number of states. In an extreme case, each feature vector corresponds to single state and there are as many parameters as states.

- **Linear architecture:** The scoring function is a linear combination of features and the cost approximation is given by

$$\tilde{J}(i,r) = r_0 + \sum_{n=1}^{m} r_n f_n(i) \tag{4.3}$$

where $r = \{r_0, r_1, \ldots, r_m\}$ is the parameter vector. The feature vector $f_n(i)$ can be considered the basis vector (though they do not need to form the basis of a vector space in strict sense).

2. **Multilayer perceptron neural network:** A neural network is a universal function approximator and has been shown capable of fitting any nonlinear function to an arbitrary degree of precision [112]. This makes neural network an excellent choice for scoring function. The score of a state, $\tilde{J}(j,r)$ is represented by a multilayer perceptron network with the layer weights being the parameter vector. The feature extraction mapping can either be absent or explicitly included.

This dissertation uses neural networks for approximating the cost-to-go function. Neural networks have been used successfully in many fields like pattern recognition, virtual sensing and nonlinear system identification. The neural network, in traditional sense, is trained by minimizing the error between input-output mapping and the desired nonlinear function in least square sense. The training is normally performed by using a training data set comprising of input-output combination, i.e. *{i, F(i)}* which is representative of mapping *F* to be approximated.

However, in contrast to supervised learning of neural network, there is no data pair of input-output combination for the scoring function to be approximated in NDP. A least square optimization with pair *{i, J*(i)}* to approximate $\tilde{J}$ is not possible as the optimal cost-to-go value *J*(i)* for a given state *i* is unknown. The only possibility is to simulate for the cost-to-go estimate, *J(i)* for a given policy (suboptimal usually) and to iteratively improve the policy based on simulation outcome. The target for the neural network training changes with every iteration as the simulation finds a better cost-to-go estimate, *J(i)*. This creates computational difficulties that do not arise in traditional neural network applications. The network needs to be trained incrementally under stochastic environment

and a non-stationary target. In addition, the network is trained in a feedback fashion where the output of network from previous instance is used to calculate the training target for the current step. These challenges are systematically handled in this chapter with novel numerical techniques.

4.2.1 Policy Evaluation with Monte Carlo

Consider a stochastic shortest path problem with termination state cost 0. Let us perform Monte Carlo simulations with each simulation ending with termination state 0. Consider $g(i,j)$ be the instantaneous cost incurred with transition from state i to state j under policy π. Also consider for m^{th} time a give state i_0 is encountered and let $\{i_0, i_1, \ldots, i_N\}$ be the remainder of the state trajectory and $c(i,m)$ be the cumulative cost up to terminal state 0, i.e.

$$c(i_0, m) = g(i_0, i_1) + \cdots + g(i_{N-1}, i_N) \tag{4.4}$$

We assume that the different simulated trajectories are statistically independent and each trajectory is generated using Markov process determined by policy π. The cost-to-go, J^π is then given by

$$J^\pi(i) = E\left[c(i,m)\right] \qquad \forall i \tag{4.5}$$

The estimate of J^π can be calculated by

$$J^\pi(i) = \frac{1}{N}\sum_{m=1}^{N} c(i,m) \qquad \forall i \tag{4.6}$$

or iteratively by using the update formula

$$J(i) := J(i) + \gamma_m \left(c(i,m) - J(i)\right) \tag{4.7}$$

where $\gamma_m = 1/m$ is the step size.

At the end of simulation run, the estimate of $J(i_k)$ can be updated by using the subtrajectory $\{i_k, i_{k+1}, \ldots, i_N\}$ with initial state i_k embedded inside the original trajectory.

$$J(i_k) := J(i_k) + \gamma_k \left(g(i_k, i_{k+1}) + g(i_{k+1}, i_{k+2}) + \cdots + g(i_{N-1}, i_N) - J(i_k)\right) \quad \forall k = 0, \ldots, N-1 \tag{4.8}$$

where γ_k is the step size and can change between iterations.

The policy iteration with Monte Carlo algorithm can be modified for approximate policy iteration that combines Monte Carlo simulations and functional approximations.

Consider a functional approximation $\tilde{J}(\cdot, r)$ of cost-to-go J^{π} for a stationary policy π. The $\tilde{J}(\cdot, r)$ can be calculated by solving the least squares optimization problem

$$\min_{r} \sum_{i \in S} f = \frac{1}{2} \sum_{i \in S} \sum_{m=1}^{M(i)} \left(\tilde{J}(i,r) - c(i,m) \right)^2 \tag{4.9}$$

where S is the set of representative states and for each $i \in S$, there are $M(i)$ samples of the cost $c(i,\cdot)$. The r is the parameter vector to be determined. The above least square problem can be solved iteratively using incremental gradient method. By differentiating f

$$\begin{aligned} \nabla f &= \nabla \tilde{J}(i,r) \sum_{m=1}^{M(i)} \left(\tilde{J}(i,r) - c(i,m) \right) \\ &= \nabla \tilde{J}(i,r) \left(\tilde{J}(i,r) - \sum_{m=1}^{M(i)} c(i,m) \right) \end{aligned} \tag{4.10}$$

The incremental gradient method will result in following update formula evaluated over all states $i \in S$

$$r := r - \gamma \nabla \tilde{J}(i,r) \left(\tilde{J}(i,r) - \sum_{m=1}^{M(i)} c(i_k, m) \right) \tag{4.11}$$

Now consider a series of simulated sequence $i = \{i_1, \ldots, i_n\}$ of states generated by Monte Carlo simulation. At a typical iteration, the system is at state i_k and a control $u_k \in U$ based on current policy π is applied. The next state i_{k+1} is generated by simulating transition probability $p_{i_k j}(u)$. The incremental update is then given by

$$r := r - \gamma \sum_{k=0}^{N-1} \nabla \tilde{J}(i_k, r) \left(\tilde{J}(i_k, r) - \sum_{m=k}^{N-1} g(i_m, u_m, i_{m+1}) \right) \tag{4.12}$$

The above update formula only considers some of the terms in the sum of squares in equation (4.9), namely, the cost samples $c(i_k,\cdot)$ associated with the states i_k visited by the trajectory under consideration. The incremental update of the parameter vector r is carried out once the trajectory has been simulated. The parameter vector r is updated incrementally, with each iteration corresponding to a new simulated trajectory. For faster convergence extended Kalman filter can be used at the expense of more computation. A drawback with the above approach is that the update of the parameter vector r needs to

wait for the end of simulated trajectory. This drawback is mitigated with temporal difference learning, described in next section.

4.2.2 Temporal Difference (TD) Methods

Temporal Difference (TD) learning method [108] is a reinforcement learning method with great intuitive appeal and has received considerable interest by robotics and machine learning community. The major breakthrough in implementation of TD methods came with TD-Gammon program that learned to play Backgammon at grandmaster level [113]. The TD methods have since then become a powerful choice for Markovian environments like game playing.

Define temporal difference d_k by

$$d_k = g(i_k, u_k, i_{k+1}) + \tilde{J}(i_{k+1}, r) - \tilde{J}(i_k, r) \tag{4.13}$$

The temporal difference d_k represents the difference between the estimate

$$g(i_k, u_k, i_{k+1}) + \tilde{J}(i_{k+1}, r) \tag{4.14}$$

of cost-to-go based on simulated outcome of the current stage and the current estimate $\tilde{J}(i_k, r)$. The temporal difference provides an indication about how far our present estimate of $\tilde{J}(\cdot, r)$ is from the optimal value. For Bellman equation to hold and $\tilde{J}(\cdot, r) \approx J^*$, the TD error should be zero. Therefore, for a given control policy, π, the equation $d_k = 0$ can be solved for $\tilde{J}(., r)$ in least square sense.

The iteration in equation (4.12) then becomes

$$r := r + \gamma \sum_{k=0}^{N-1} \nabla \tilde{J}(i_k, r) \sum_{m=k}^{N-1} d_k \tag{4.15}$$

The parameter vector r can be updated as soon as d_k becomes available rather than waiting for the end of simulated trajectory. The update equation then can be implemented online and is given by

$$r := r + \gamma d_k \sum_{m=0}^{k} \nabla \tilde{J}(i_m, r) \qquad k = 0, \ldots, N-1 \tag{4.16}$$

The online implementation of above update formula requires calculation of gradient $\nabla \tilde{J}(i_m, r)$ at each step, at the current value of r. This is extremely cumbersome and

increases the computation overhead of the update formula. A practical alternative is to calculate the gradient $\nabla\tilde{J}(i_m, r)$ as soon as i_k is generated with the present value of r and use that value of the gradient in subsequent updates, even though value of r is changing at every step.

The temporal difference is family of algorithms and can be generalized from TD(1) to TD(λ), where λ is a parameter in [0,1]. Detailed discussion is given by Bertsekas et al. [109] and Sutton et al. [108]. The offline version of TD(λ) is given by

$$r := r + \gamma \sum_{k=0}^{N-1} \nabla\tilde{J}(i_k, r) \sum_{m=k}^{N-1} d_k \lambda^{k-m} \tag{4.17}$$

and the corresponding online version of update formula is given by

$$r := r + \gamma d_k \sum_{m=0}^{k} \lambda^{k-m} \nabla\tilde{J}(i_m, r) \qquad k = 0, \ldots, N-1 \tag{4.18}$$

4.2.2.1 Discounted infinite horizon problems

For a discounted problem, the temporal difference is defined by

$$d_k = g(i_k, u_k, i_{k+1}) + \alpha\tilde{J}(i_{k+1}, r) - \tilde{J}(i_k, r) \tag{4.19}$$

and the online update rule following transition from state i_k to i_{k+1} is

$$r := r + \gamma d_k \sum_{m=0}^{k} (\alpha\lambda)^{k-m} \nabla\tilde{J}(i_m, r) \qquad k = 0, \ldots, N-1 \tag{4.20}$$

For infinite horizon problems, there is no terminal state and the trajectory may never terminate. A few changes are incorporated in the algorithm to address these issues. First, the step size diminishes as the algorithm progresses for convergence. Second, the algorithm is frequently reinitialized and restarted if some of the important states of the Markov chain are transient (e.g. a fixed initial state is never revisited). Without reinitialization, there will be very little training of these states and the approximation of the cost-to-go will be very poor.

4.2.2.2 Algorithm for training functional approximation

The NDP algorithm calculates the functional representation of optimal cost-to-go based on random samples. The estimation problem can be stated as optimizing the expectation of a function

$$\min_{r \in R} \mathrm{E}\left\{F\left(r\right)\right\} \tag{4.21}$$

where typically F is the mean square error (MSE). In the context of cost-to-go function, the optimal parameter vector r can be obtained by using the following equation.

$$r = \arg\min_{r \in R} \mathrm{E}\frac{1}{2}\left(\hat{J} - \tilde{J}(r)\right)^2 = \min_{r \in R} \frac{1}{2}\sum_{i}^{m}\left(\hat{J}_i - \tilde{J}(r_i)\right)^2 \tag{4.22}$$

where $\hat{J}$ is the estimate of J^* at present instance and $\tilde{J}(r)$ is the functional representation of cost-to-go. The above problem can be solved using stochastic gradient methods. Some of the algorithms used in this dissertation are reviewed briefly in Appendix B. The least square problem in equation (4.9) is solved using all the incremental gradient methods described in Appendix B. Extended Kalman Filter (EKF) gave the fastest convergence rate with very small additional computational overhead of all the methods listed in Appendix B. Hence, EKF is used in this dissertation to train critic and actor neural networks.

4.2.2.3 Step size

The step size, γ plays an important role in the performance of incremental gradient algorithms employed in this dissertation to train actor and critic networks. The direction of incremental gradient method differs from the gradient direction by an error proportional to the step size. To ensure convergence in stochastic gradient algorithms following rules must be met [107], [109].

$$\begin{aligned} &\gamma_n \geq 0, \qquad n = 1, 2, \ldots \\ &\sum_{n=1}^{\infty} \gamma_{n-1} = \infty, \\ &\sum_{n=1}^{\infty} (\gamma_{n-1})^2 < \infty \end{aligned} \tag{4.23}$$

The second condition is required to guarantee that step size is not too small and the algorithm stalls prematurely. It ensures that steps are large enough to overcome any initial conditions or random fluctuations. The last condition guarantees that the step size diminishes with iteration and eventually become small enough to assure convergence.

The step size selection is even more important for neuro-dynamic programming due to the moving target value $\tilde{J}(r)$, which at the start of the algorithm can be very far from optimal J^*. In TD(λ) algorithm, the cost-to-go, J of being in particular state is estimated from a sequence of random observations, $\hat{J}$. The estimated cost depends on the approximation of cost-to-go function, $\tilde{J}(\cdot, r)$ which itself is non-stationary and steadily increasing. The step size needs to strike a balance between minimizing error (small step size) and responding to non-stationary data (large step size). Appendix C gives some of the step size recipes used in this dissertation. Bias adjusted Kalman filter (BAKF) and Sompolinsky-Barkai-Seung (SBS) step size rules gave satisfactory performance and are employed for calculating step size.

4.2.3 Actor-Critic System

An interesting and intuitive way to describe the NDP algorithm, similar to policy iteration algorithm presented in Chapter 3, is to view it as an *actor-critic* system [109], [108]. The general schematic of the NDP is shown in Figure 4.2.

The NDP structure in Figure 4.2 includes two networks; actor and critic. The critic network is trained to estimate the optimal cost-to-go function. The actor network is trained to produce optimal control inputs, which are greedy with respect to cost-to-go function. The objective is to optimize the desired performance by learning to choose appropriate control actions through interaction with environment. During the learning process, the critic criticizes the actor's actions and the actor incorporates the latest evaluation by critic for next control action. The critic learns about whatever the policy is being followed by actor and critiques it. The critique (reinforcement signal) drives the learning of both the actor and the critic networks.

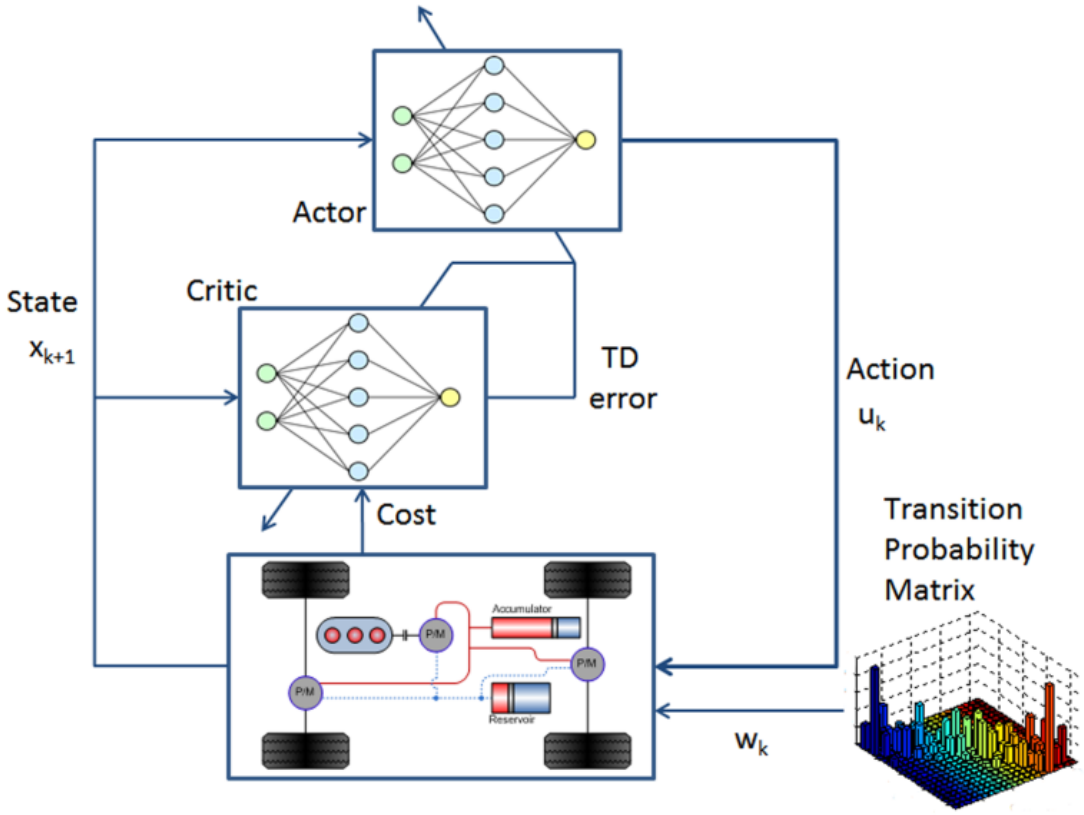


Figure 4.2: Schematic diagram illustrating neuro-dynamic programming as an actor-critic system.

The controllers are "naïve" at the starting of the algorithm. Both the actor and the critic networks are initialized with random weights. Based on present system states, the actor network produces control action and the system moves to a newer state. The critic evaluates the new state and calculates the reinforcement signal to tune the parameters in actor as well as critic network. With time, the actor learns to produce "favorable" control actions.

The critic network is implemented as a standard multilayer feedforward neural network. The input to critic is the states of system and the output is the approximate cost-to-go function $\tilde{J}$. The network is trained by backpropogating the TD error signal and the weights, r of the critic network are updated using equation (4.20). The actor network is similar to the critic network. The actor network is required because it would require a lot of computation and memory to compute improved policy, $\bar{\pi}$, given in equation (4.1) online. The algorithm computes $\bar{\pi}(i)$ only at a set $\hat{S}$ of sample states and generates an

approximation architecture $\tilde{\pi}(\cdot, v)$ where v is a vector of tunable parameters. The neural network is trained by minimizing the least square problem

$$\min_{v} \sum_{i \in \hat{S}} \left\| \tilde{\pi}(i, v) - \bar{\pi}(i) \right\|^2 \tag{4.24}$$

4.2.4 Policy Update

A standard actor-critic method evaluates the associated cost-to-go J^{π} using TD(λ) for a fixed policy π and then performs the policy update. In other words, the actor controls the system using policy π while the critic observes the consequences and computes J^{π}. The approximation $\tilde{J}(\cdot, r)$ of J^{π} is constructed in least square sense. For a standard algorithm, the policy π is kept constant for a long time till the critic's computations converge to J^{π}. This new converged value of J^{π} is then used by critic to calculate a new policy. The new policy $\bar{\pi}$ is then calculated which is greedy with respect to $\tilde{J}(\cdot, r)$. This may not be suitable for problems with large state space as evaluating over all state combinations would mean long computational time between policy update. Alternatively, the policy π can be updated more frequently with new policy $\bar{\pi}$ without waiting for policy evaluation algorithm to converge J^{π}, i.e. there is more frequent communication between actor and critic. This is known as ***asynchronous policy iteration***. An extreme possibility is to replace policy π with new policy $\bar{\pi}$ subsequent to every state transition, i.e. new control u_k is calculated after every iteration

$$u_k = \arg\min_{u \in U(i_k)} \left(g(i_k, u, j) + E\left[\alpha \tilde{J}(j, r_k) \right] \right) \tag{4.25}$$

where α is the discount factor, r_k is the current parameter vector and j is the possible next states. The actor carries out new policy after every simulated transition. This class of algorithm is known as ***optimistic policy iteration***. The convergence behavior of this algorithm is quite complex and not fully understood [109]. However, optimistic policy iteration with TD(λ) update is one of the most effective NDP methods and is used in this dissertation.

4.2.5 Exploration vs. Exploitation

Classical SDP algorithms require evaluating cost-to-go function at every state. This is known as exploration of state space and is only possible for problems with small state space. It is computationally impossible to evaluate cost-to-go function at every state for a problem with very large or infinite state space. On the other hand, exploitation involves utilizing the present information about cost-to-go function and making decisions that are greedy with respect to present cost-to-go function approximation. An exploitation strategy is good from computation consideration for problems with large state space. However, the algorithm with pure exploitation strategy is susceptible to being stuck in local optimum because of the poor estimate of certain states.

The strategy used in this dissertation is a mix of exploration and exploitation strategies and is known as ***ε-greedy*** strategy [108]. The algorithm chooses a greedy policy i.e. exploitation strategy for most of the times but reverts to exploration strategy with small probability ε and select action at random, independently of cost-to-go function. An advantage of such a strategy is that on limit, as number of iteration increases, every control action will be sampled infinitely and the control policy π will converge to optimal π*. Sutton et al. [108] showed the effectiveness of ε-greedy strategy compared to greedy policy.

4.2.6 Neuro-Dynamic Programming Algorithm

Consider a series of simulated sequence $i = \{i_1, \ldots, i_n\}$ of states generated by Monte Carlo simulation. At a typical iteration, the system is at state i_k and a control $u_k \in U$ based on the current policy π is applied. The next state i_{k+1} is generated by simulating transition probability $p_{i_k j}(u)$.

The parameter vector is then updated by running a TD(λ) update [109] (refer section 4.2.2)

$$r_{k+1} = r_k + \gamma_k d_k \sum_{m=0}^{k} (\alpha\lambda)^{k-m} \nabla \tilde{J}(i_m, r_m) \tag{4.26}$$

where k is the iteration number, γ is the step size, λ is the TD parameter, α is the discount factor, d_k is the temporal difference and gradient $\nabla \tilde{J}(i, r)$ is the vector of partial

derivatives with respect to parameter vector r. The equation (4.26) is an incremental gradient update. The step size γ is updated using Bias Adjusted Kalman Filter (BAKF) described in Appendix C.

Define eligibility vector, z_k

$$z_k = \sum_{m=0}^{k} (\alpha\lambda)^{k-m} \nabla \tilde{J}(i_m, r_m) \tag{4.27}$$

The TD update is then written as

$$r_{k+1} = r_k + \gamma_k d_k z_k \tag{4.28}$$

where z_k is updated by

$$z_{k+1} = \alpha\lambda z_k + \nabla \tilde{J}(i_{k+1}, r_{k+1}) \tag{4.29}$$

The critic neural network update is accelerated by including nonlinear learning rates and the update equation (4.26) becomes

$$r_{k+1} = r_k + \gamma_k d_k \sum_{m=0}^{k} (\alpha\lambda)^{k-m} H_m^{-1} \nabla \tilde{J}(i_m, r_m) \tag{4.30}$$

Direct computing of the matrix H^{-1} is very computationally costly. Kalman theory is efficiently applied for calculating the inverse of the Hessian (refer Appendix B).

4.3 Power Management Problem with Multiple Objectives

The power management problem explored in this section is similar to the problem defined in Chapter 3. The problem is redefined for completeness. Given the vehicle, engine and powertrain configuration, this section examines the infinite horizon power management problem, which can be formulated as

$$\begin{aligned} \text{Minimize:} \quad & J = \lim_{N\to\infty} \underset{w_k}{E} \left\{ \sum_{k=0}^{N-1} \alpha^{\kappa} g\left(x_k, u_k, w_k\right) \right\} \\ \text{subject to:} \quad & x_{k+1} = f\left(x_k, u_k, w_k\right) \\ & \mathrm{x} \in \mathrm{X} \\ & \mathrm{u} \in \mathrm{U} \end{aligned} \tag{4.31}$$

where x is the state vector, u is the control input, w is the disturbance vector, g is the instantaneous cost function and $0<\alpha<1$ is the discount factor.

The next two sections apply NDP to solve the above problem with a different objective function. First, the problem with fuel economy objective is addressed and the results are compared with SDP to validate the effectiveness of NDP. Next, the problem with multiple objectives i.e. to minimize transient emissions along with fuel economy is solved using NDP. This problem is computationally intractable with classical SDP methods because it involves a state-action space of 10^{13} and shows the strength of NDP.

4.4 NDP with Fuel Economy Objective

The system is modeled with 3 states x_k (with little abuse of notation, the disturbance vector w_k is included in the state vector) and 2 control inputs u_k.

$$x_k = \begin{cases} SOC \\ \omega_{wh} \\ P_{dem} \end{cases} \tag{4.32}$$

$$u_k = \begin{cases} \omega_e^{dem} \\ T_e^{dem} \end{cases} \tag{4.33}$$

where k is the time index, SOC is the present state of charge, ω_{wh} is the present wheel speed, P_{dem} is the driver power demand, ω_e^{dem} is the desired engine speed and T_e^{dem} is the desired engine torque.

The cost function is given by

$$g = FC(x_k, u_k) + \mu \cdot (SOC - SOC_{ref})^2 \cdot (SOC < SOC_{ref}) \tag{4.34}$$

where FC is the fuel consumption by engine.

The state space spanned by the problem is not large and is computationally tractable with classical SDP [43]. This provides an opportunity to compare the result obtained with NDP to the strategy previously generated using SDP (refer Chapter 3). This effectively allows validating the new approach before embarking on studies of larger problems with larger state space.

The self-learning controller has three neural networks, one critic and two actor networks, which are trained simultaneously by interacting with the environment. All three neural networks are multilayer perceptron networks with *hyperbolic tangent* as the activation function. The networks take three inputs, namely the state vector and have one

hidden layer with 30 neurons. The output of critic network is the approximate value of cost-to-go and the two actor networks output desired engine speed and desired engine torque.

The networks are initialized with random weights i.e. they start naïve. The networks are trained incrementally using TD signal and learn to control the hybrid powertrain as the algorithm progresses. The algorithm performs Monte Carlo simulations to generate sample trajectories. At any given state, the actor network is evaluated and the control input is applied to the system. The algorithm calculates the temporal difference and updates the critic network using equation (4.26). The system moves to newer states based on the applied input. The actor network is then updated to produce control actions which are ***ε-greedy*** [108] (refer section 4.2.5) with respect to latest cost-to-go function approximation. The algorithm chooses a greedy policy based on the present knowledge of cost-to-go function most of the times but reverts to exploration strategy with small probability ε. The algorithm steps are repeated until the cost-to-go function approximation converges. The control action chosen by algorithm and the states visited depend on approximation of cost-to-go function, which in turn depends on states visited thus far by algorithm. This can lead to algorithm being stuck in the local optima where poor approximation of cost-to-go function for certain states can prevent algorithm from visiting those states. To overcome this problem, the algorithm is frequently restarted from random states.

4.4.1 Simulation Results for NDP Controller with Fuel Economy Objective

In this section, simulation results for hybrid powertrain with self-learning neural controller over EPA driving cycles are presented. Figure 4.3 gives the block diagram implementation of self-learning controller with 3 inputs. The results are compared against the system centric SDP based controller presented in Chapter 3.

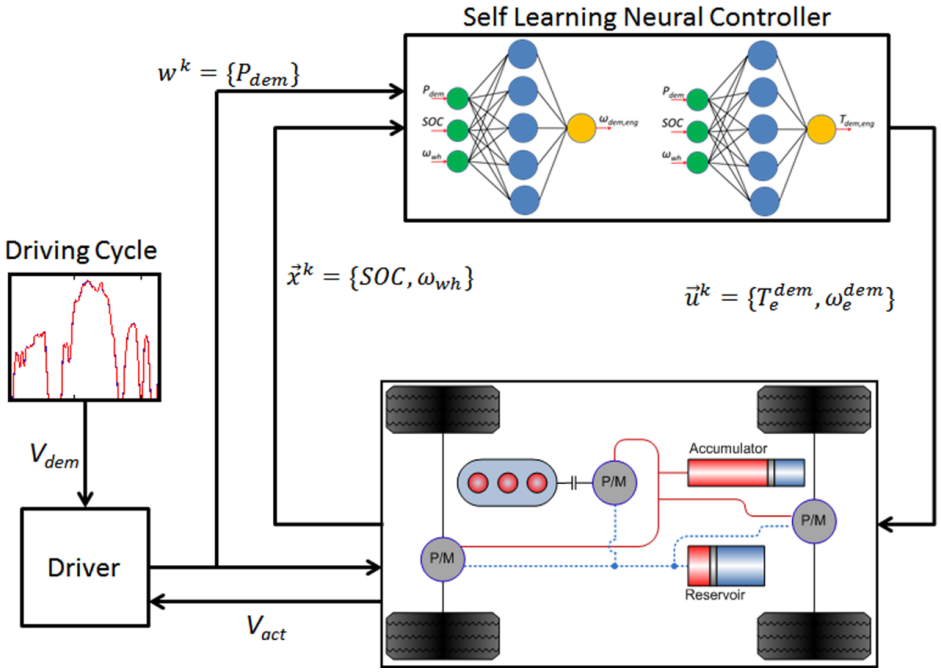


Figure 4.3: Schematic representation of self-learning neural controller with 3 inputs along with a S-HHV.

Figure 4.4 shows time plot for SOC, engine power and motor power for NDP based controller over a section of FUDS. It can be seen from the Figure 4.4 that the NDP based controller learns how to actively manage two power sources, namely engine and hydraulics, and is able to follow driving cycle. In addition, the controller does a good job in maintaining low SOC throughout the driving cycle (Figure 4.4a). This allows maximum energy to be recuperated during braking event. The NDP controller learns to use hydraulic energy at launch when SOC is high (engine power demand is zero). As SOC drops, engine is ramped up and produces enough power to maintain the desired value, 0.2 in this case. The NDP based controller operates the engine in a milder fashion by slowly ramping up the engine power (Figure 4.4b) and significantly reducing fluctuations of SOC (Figure 4.4a). It appears that the engine is almost in a "load following" mode but without sharp changes of load or high frequency fluctuations. At the same time the algorithm selects the best combination of engine speed and torque to produce desired power as discussed in the next paragraph. Overall, NDP identifies the best strategy from the system point of view rather than engine-centric approach.

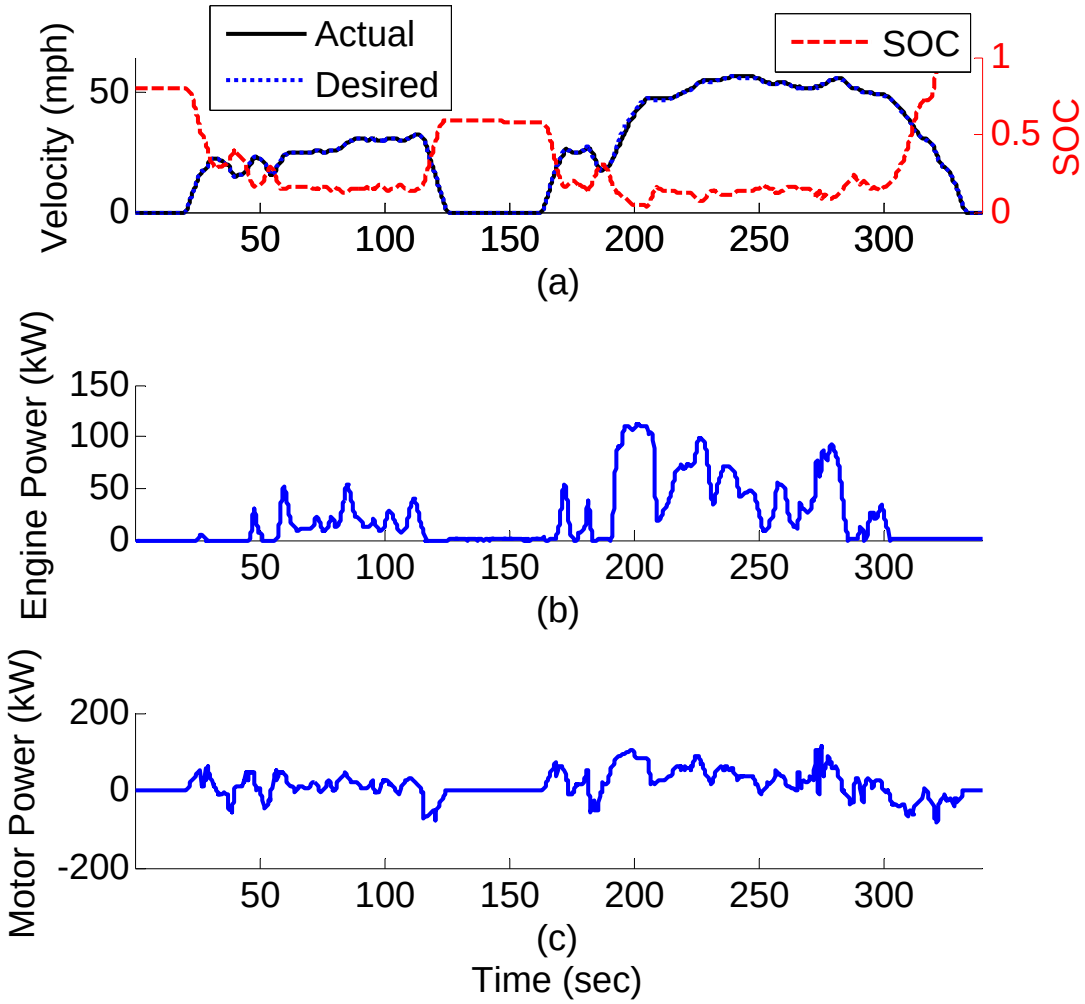


Figure 4.4: Simulated series hybrid powertrain behavior with the NDP controller with 3 inputs and designed with fuel economy consideration: a) vehicle speed and SOC during first 350 sec of FUDS, b) engine power, and c) propulsion motor power.

Figure 4.5 shows the engine operating points superimposed on the BSFC map. The color scale indicates the amount of fuel consumed by the engine at a given operating region during FUDS. The NDP controller operates the engine over a wider region. It can be seen that the NDP based controller intelligently controls engine in the low BSFC regions for any power level without any prior information about the best BSFC trajectory. However, the engine operation deviates slightly from best BSFC line. This is attributed to the fact that controller is trying to maximize the system efficiency i.e. combined engine and pump subsystem efficiency rather than engine efficiency alone.

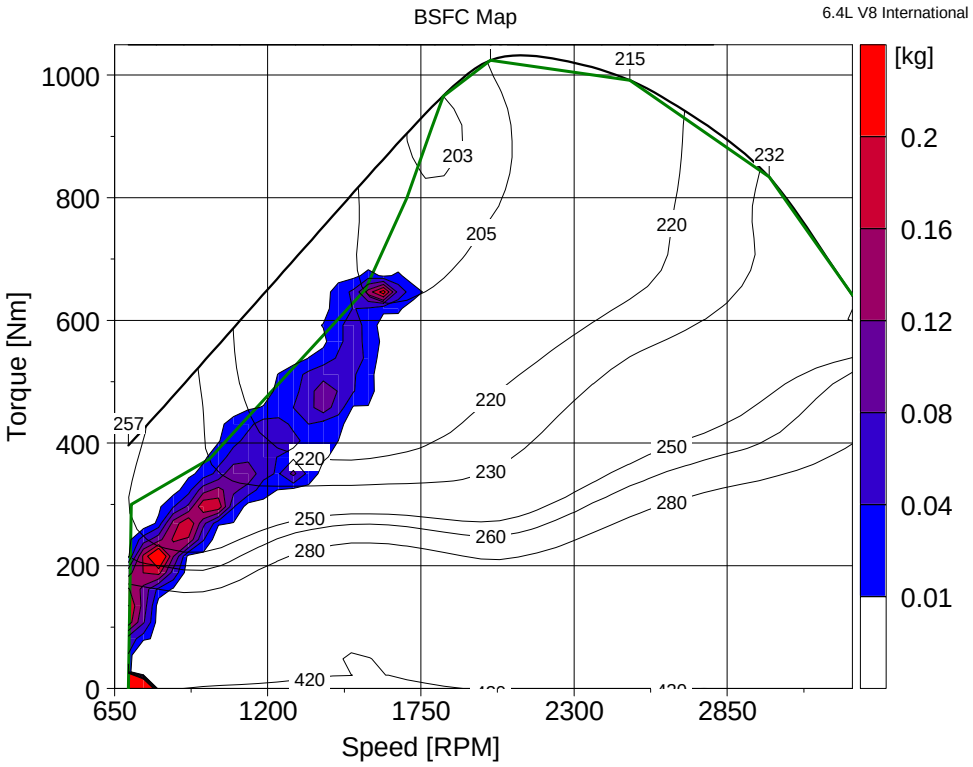


Figure 4.5: Engine visitation points on the BSFC map, with a color scale indicating the relative amount of fuel consumed in a given zone during simulation over FUDS for self-learning neural controller with 3 inputs in S-HHV.

Table 4.1: Fuel economy (mpg) comparison for a S-HHV with system centric SDP control and NDP control with 3 inputs, both designed with fuel economy objective over different driving schedules

	FUDS	**HWFET**	**LA92**
S-HHV with SDP controller	17.47	17.4	14.2
S-HHV with NDP controller	17.84	17.47	14.14

Table 4.1 gives the results of series hydraulic hybrid with SDP and NDP based controllers over different driving schedules. The engine operation and management of SOC by NDP based controller is very similar to SDP based controller and hence, the fuel economy benefits are very similar. This effectively validates the NDP approach to design

optimal power management controllers. This is a favorable outcome providing evidence of NDP's ability to learn and discover best modes of operation.

4.5 NDP with Transient Emission and Fuel Economy Objective

This section examines the infinite horizon power management problem with multiple objectives. The power management tries to minimize the weighted sum of transient emission and fuel consumption. This problem is motivated in section 3.2.1. The instantaneous cost is given by

$$\begin{aligned} g = w_{FC} \cdot FC(x_k, u_k) + w_{NOx} \cdot NO_x(x_k, u_k) + w_{PM} \cdot PM(x_k, u_k) \\ + \mu \cdot (SOC - SOC_{ref})^2 \cdot (SOC < SOC_{ref}) \end{aligned} \tag{4.35}$$

where FC is the normalized fuel consumption, NO_X is the normalized transient NO_X emission, PM is the normalized transient particulate matter emission. The w_{FC}, w_{NOx} and w_{PM} are the normalized weighting parameter and $\sum(w_{FC} + w_{NOx} + w_{PM}) = 1$.

The state vector x and control vector u is given by

$$x_k = \begin{bmatrix} SOC \\ \omega_{wh} \\ \omega_e \\ m_f \\ P_{im} \\ NOx^{k-1} \\ PM^{k-1} \\ P_{dem} \end{bmatrix} \tag{4.36}$$

$$u_k = \begin{bmatrix} \omega_e^{dem} \\ T_e^{dem} \end{bmatrix} \tag{4.37}$$

where k is the time index, SOC is the state of charge, ω_{wh} is the wheel speed, ω_e is the engine speed, m_f is the mass of fuel injected, P_{im} is the inlet manifold pressure, NO_x^{k-1} is the previous predicted NO_X and PM^{k-1} is the previous predicted particulate matter, P_{dem} is the driver power demand, ω_e^{dem} is the desired engine speed and T_e^{dem} is the desired engine torque. The additional augmented states compared to states required for section 4.4 is due to transient emission model (refer section 2.3.2.2).

The addition of extra states results in additional state constraints along with the ones given in section 3.2.3.

$$\begin{aligned} 0 &\le m_f^k(T_e^k, \omega_e^k) \le m_{f,\max}(T_{e,\max}, \omega_e^k) \\ 0 &\le NOx^k \\ 0 &\le PM^k \end{aligned} \tag{4.38}$$

The problem formulated above with each state and control input discretized with cardinality of 25 has approximately 10^{13} state-action pairs. A state-action space of this size is computationally intractable with conventional policy iteration algorithm. In addition, even if the problem could be solved, it will require vast amounts of memory to store every action value for every combination of states. NDP provides an approach to suboptimally solve the above problem by approximating the optimal cost-to-go function with a functional approximation.

The self-learning controller, much like the controller in section 4.4, has three neural networks, one critic network and two actor networks which are trained using TD(λ) approach, Figure 4.2. The training of the self-learning controller is similar to supervisory controller in section 4.4.

The critic and actor neural networks are multilayer feedforward perceptron network with one hidden layer. The input and hidden layers have *hyperbolic tan-sigmoid* activation function whereas output layer has linear activation function. The inputs to all the networks are the states of the system, equation (4.36) i.e. each network have 8 inputs. The networks are trained incrementally by backpropogating the temporal difference and the weights are updated using equation (4.26). The critic network estimates of the cost-to-go value whereas the outputs of the actor controllers are optimal demanded engine speed and demanded engine torque. This implementation is referred to as NDP based controller with 8 inputs in this dissertation.

Implementation of above controller in real world will require feedback of transient NO_X and soot emissions. To avoid virtual soot and NO_X emission sensors during implementation of the controller in either simulation or real vehicle, another approach is explored. Instead of training critic and actor neural networks with complete state vector, the three states, namely previous NO_X, previous PM and inlet manifold pressure are

excluded from the inputs to the critic and actor network. The networks are trained with only remaining 5 inputs. This can be viewed as contraction mapping and has an effect of averaging out the effect of these 3 states. This implementation requires only 5 inputs to the power management controller and does not require a feedback of transient soot and NO_X emission during real world implementation. This eliminates the need for concurrent running of transient soot and NO_X virtual sensors with the power management controller. This implementation is referred to as NDP based controller with 5 inputs in this dissertation. The NDP framework still uses a transient emission model for calculating the objective function during Monte Carlo simulations for training self-learning controller and the algorithm internally keeps track of 8 plant states. The algorithm ignores the aforementioned 3 states while training actor and critic neural networks.

Figure 4.6 shows the implementation of NDP controller with 5 inputs i.e. with no feedback from virtual emission sensor, and Figure 4.7 shows the implementation of NDP controller with 8 inputs i.e. the neural controller requires feedback from transient emission sensors.

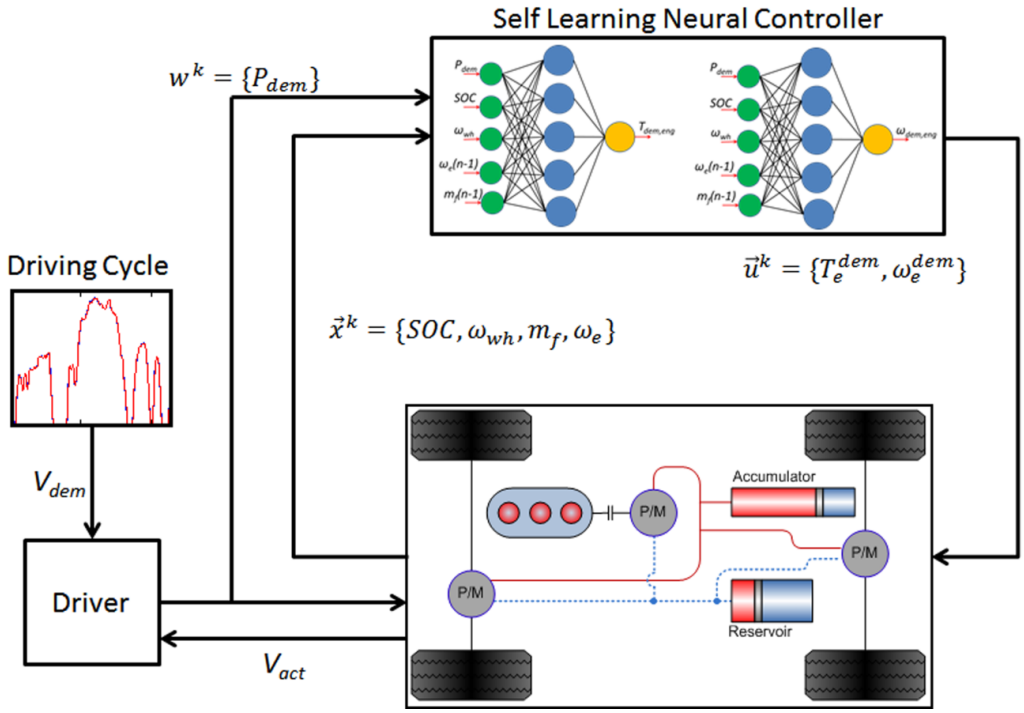


Figure 4.6: Schematic representation of self-learning neural controller with 5 inputs along with a S-HHV.

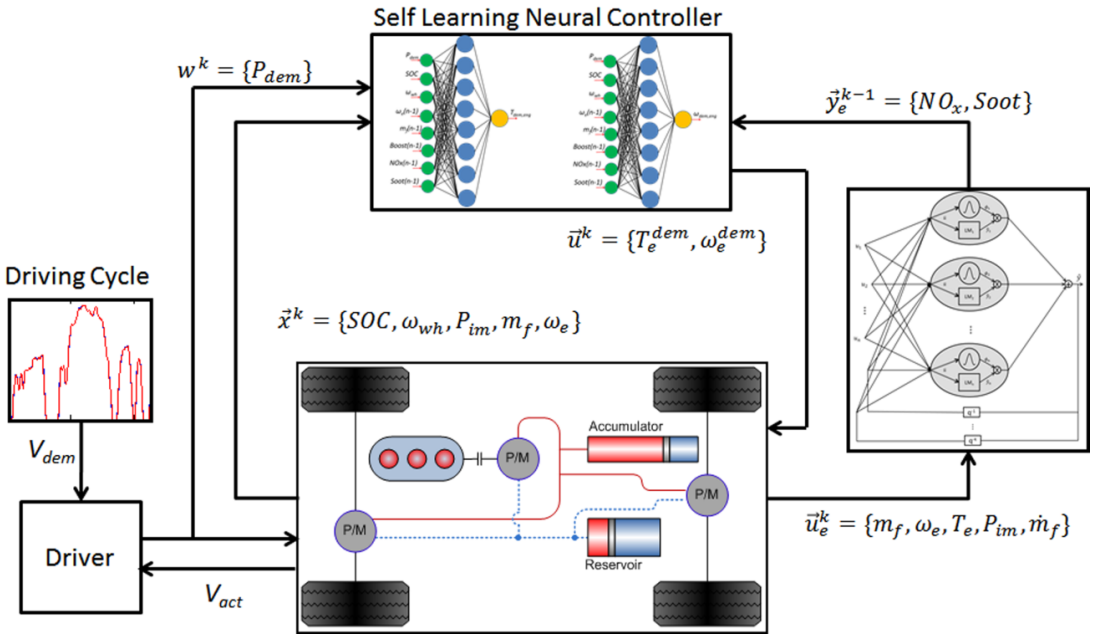


Figure 4.7: Schematic representation of self-learning neural controller with 8 inputs along with a S-HHV.

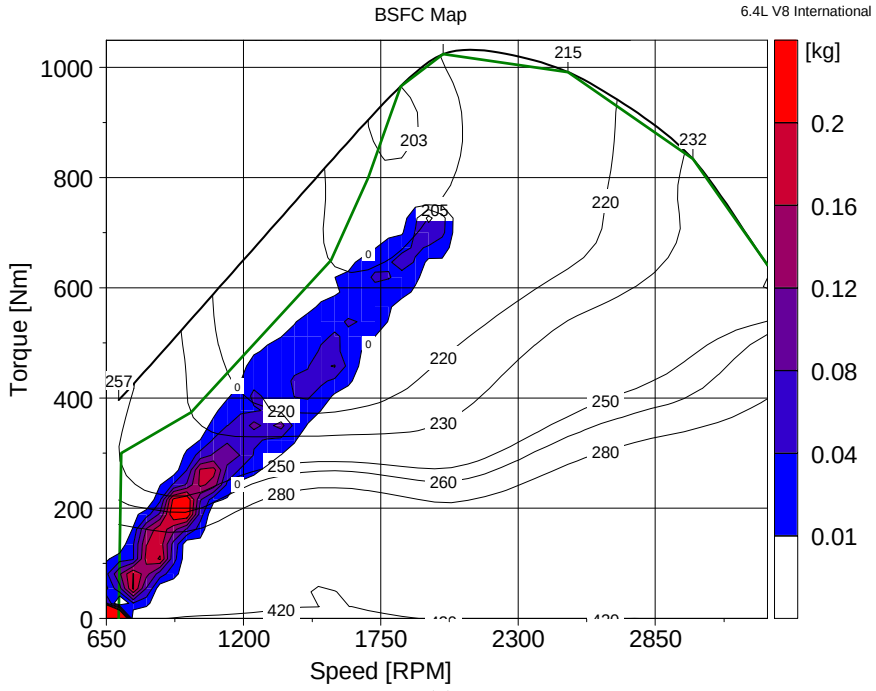
4.5.1 Simulation Results for NDP Controller with Multiple Objectives

The performance of self-learning controller and its true impact on transient engine emissions require validating it in the Engine-In-the-Loop (EIL) facility. EIL facility at the University of Michigan is described in section 2.4. The EIL facility allows for concurrent running of real engine with virtual drivetrain/vehicle models in real time. EIL facilitates in studying effects of different powertrain/power management strategies on engine operation. EIL facility is equipped with fast emission analyzers and allows for quantifying transient engine-out emissions. However, before the self-learning controller can be evaluated in EIL facility, its performance is evaluated using the high fidelity simulation platform. This step is required to perform sanity check on the controller and prevent any undesirable operation in the test cell. An ill-performing controller can damage the EIL facility. The engine-out transient emissions in simulation are calculated using transient particulate matter and NO_X models, which will be discussed in Chapter 5.

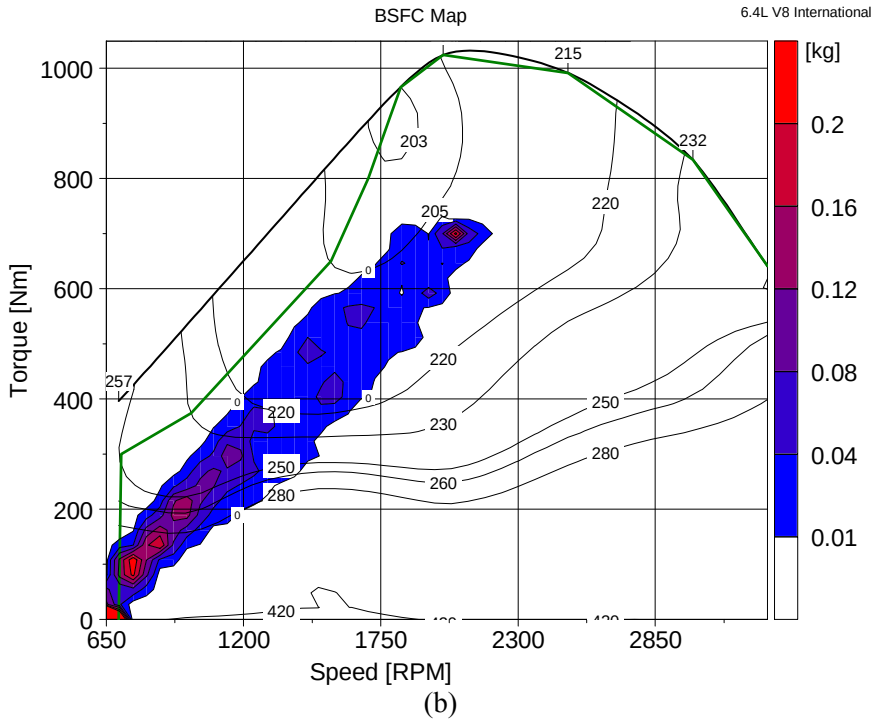
Three different self-learning neural network controllers are generated using NDP algorithm. The first controller is designed with fuel economy objective (refer section 4.4). The other two controllers have multiple objectives of fuel consumption and transient

engine emissions. These two controllers differ in the required number of inputs to the controller with one requiring 5 inputs and other requiring 8 inputs (refer section 4.5). The two NDP controllers with multiple objectives are designed with weights $w_{FC} = 0.7$, $w_{NOx} = 0.1$ and $w_{PM} = 0.2$. The selection of weights are random and are to show the ability of algorithm to not only learn to manage two power sources but to do by minimizing weighted sum of fuel consumption and transient emissions. The purpose of this dissertation is not to create pareto optimality front and show the tradeoff between fuel economy and emission with different choice of objective function weights. The algorithm presented in this dissertation is proposed to solve problems that are very large and computationally intractable. Though the proposed NDP algorithm implementation allows for numerically solving a problem with very large state space, it still is not fast enough for creation of pareto optimality front. The computational load to generate one single controller is around 5 days on a desktop computer. The biggest bottleneck is the poor implementation of neural network routines in Matlab and can be accelerated by using C. In addition, the algorithm can be designed to use cluster and parallel computing to further reduce computational time.

All three controllers are simulated over different EPA driving schedules and the fuel economy results are reported in Table 4.2. The engine operation over FUDS with the self-learning controller with emission and fuel economy objectives can be seen from Figure 4.8. Figure 4.8a and Figure 4.8b show the engine visitation points on the BSFC map with 5 input and 8 input self-learning controllers respectively. The color scale indicates the amount of fuel consumed by the engine at different operating points. The self-learning controllers with multi objective move the engine operation away from the best BSFC line in contrast to self-learning controller with only fuel economy objective, Figure 4.5. The engine operates at higher speeds and lower loads. This results in lower NO_X emissions.



(a)



(b)

Figure 4.8: Engine visitation points on the BSFC map, with a color scale indicating the relative amount of fuel consumed in a given zone during simulation over FUDS for self-learning neural controller with: (a) 5 inputs, and (b) 8 inputs in S-HHV.

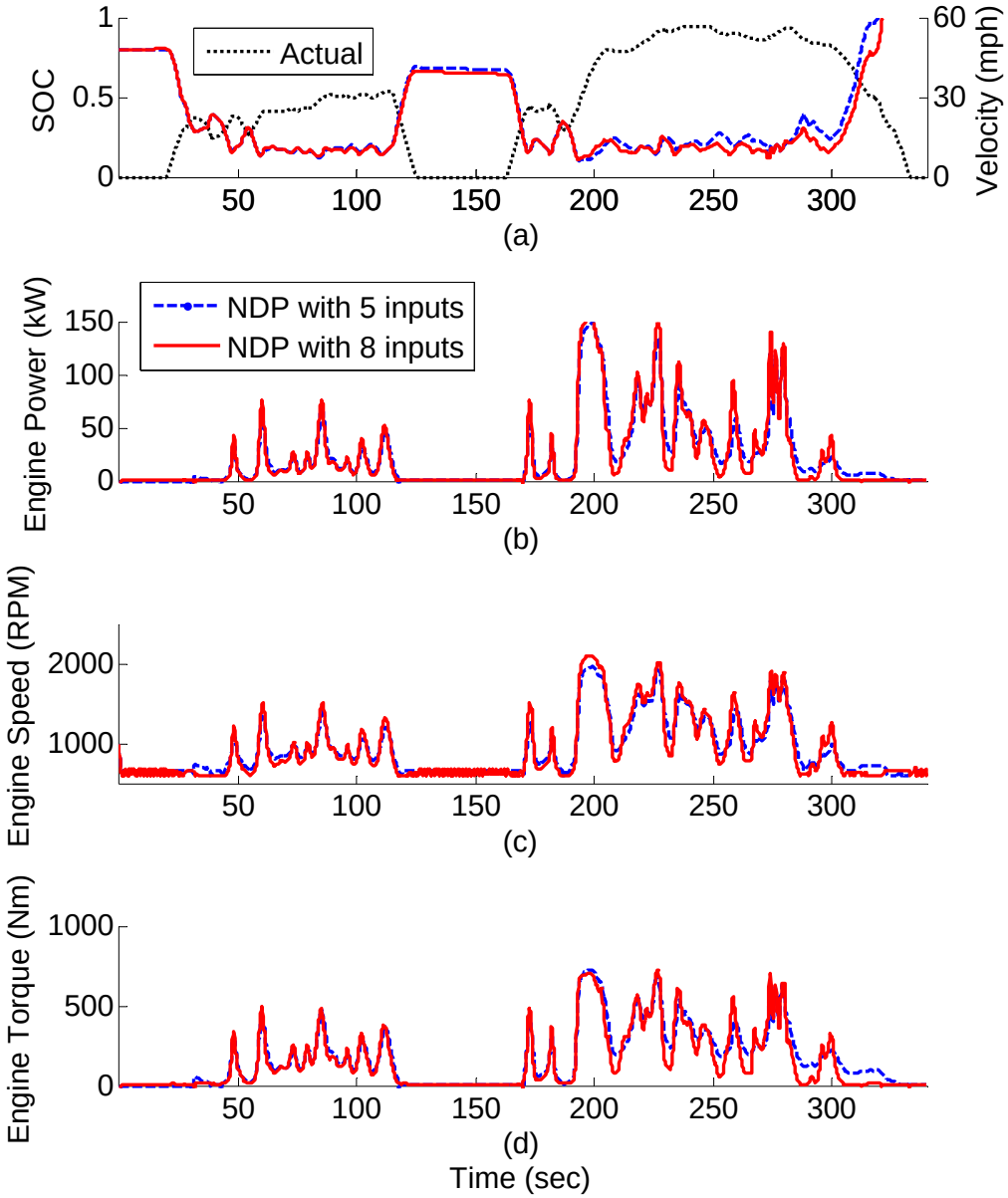


Figure 4.9: Comparison of simulated series hybrid powertrain behavior with two different NDP controllers, both designed with fuel economy and transient emission consideration: a) vehicle speed and SOC during first 350 sec of FUDS, b) engine power, c) engine speed, and d) engine torque.

The self-learning controller with 5 inputs operate engine slightly differently compared to self-learning controller with 8 inputs (Figure 4.9). To explain this, consider the engine and powertrain is operating at a given operating point and the next engine speed and torque demand is to be calculated by the two self-learning controllers with emission objectives. The self-learning controller with 5 inputs calculates these demands

based on vehicle speed, state of charge, engine speed, mass of fuel injected and power demand from driver. The controller estimates present engine operating point and calculates the next engine operating point to minimize engine transients to reduce emissions and fuel consumption. The self-learning controller with 8 inputs uses three additional inputs, namely engine inlet manifold pressure, previous NO_X and previous soot to calculate the estimate for present engine operating point. This estimate is more accurate compared to NDP controller with 5 inputs' estimate and hence the controller is able to calculate more accurately the desired engine speed and desired engine torque.

Figure 4.10 and Figure 4.11 shows the normalized cumulative engine-out NO_X and soot emissions for different controllers. By systematically evaluating the system behavior NDP controllers with transient emission objective manages significant reduction of NO_X emission with minimal fuel economy penalty. The simulation engine model with injection dynamics is capable of predicting fuel economy with reasonable degree of confidence. The engine simulation, however, does not accurately model the nonlinear dynamics of a real diesel engine. The soot formation is highly correlated to transient engine dynamics and nonlinear actuator response making it difficult to evaluate the soot emissions results using simulation. The next section simulates a virtual hybrid concurrently with a real engine in the EIL facility and provides a better estimate of engine transients and reduction in emissions.

Table 4.2: Fuel economy (mpg) comparison for a S-HHV with NDP control with 3, 5 and 8 inputs over different driving schedules. The former is designed with fuel economy objective and latter two are designed with fuel economy and transient emission objective

	FUDS	**HWFET**	**LA92**
NDP controller with 3 inputs	17.84	17.47	14.14
NDP controller with 5 inputs	16.86	16.92	13.56
NDP controller with 8 inputs	17.01	17.02	13.60

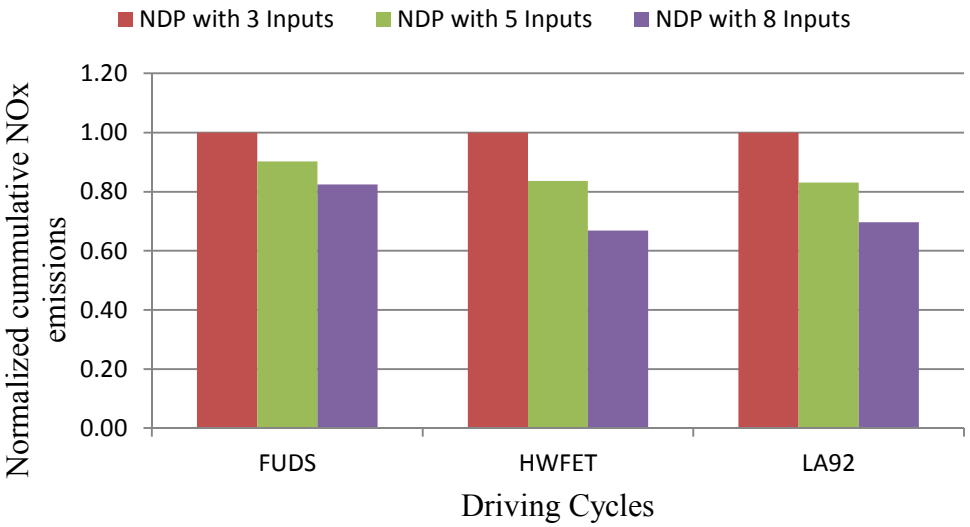


Figure 4.10: Comparison of cumulative normalized NO_X emissions from a S-HHV over different driving schedules with three different NDP based power management strategies.

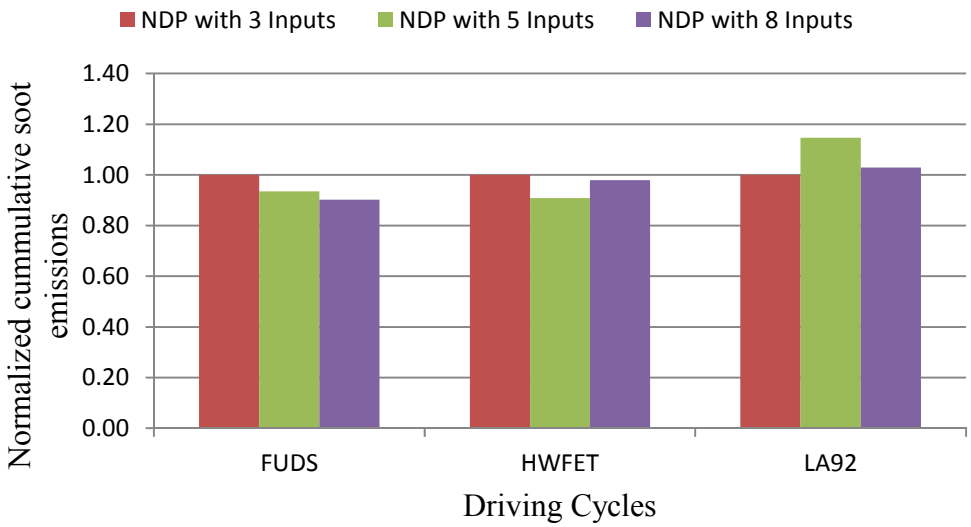


Figure 4.11: Comparison of cumulative normalized particulate matter emissions from a S-HHV over different driving schedules with three different NDP based power management strategies.

4.6 Engine-In-the-Loop Results

To assess the performance of the NDP based controllers in real world application, the NDP based controllers are evaluated using EIL over FUDS. The previous work done

by Liu et al. [42] showed that EIL facility is a valuable tool in evaluating the real-life performance of the supervisory controller and demonstrated that a supervisory controller which performed well in simulation failed to perform adequately with physical engine. The NDP based controllers not only needs to successfully orchestrate and manage the two energy sources but need to do by minimizing weighted sum of fuel consumption and transient emissions. The controllers are compared against the baseline SDP based controller and judged on the basis of fuel consumption as well as transient particulate matter and NO_X emissions. The SDP based controller is system centric controller (refer section 3.3) with sole objective to improve fuel economy.

Work done by other researchers using simulation platforms often rely on assumptions like simplified system response and noise free signal for feedback. While these assumptions make problem easier and are generally appropriate for simulation, the real world system performance can deviate significantly due to nonlinear and unmodeled system dynamics. The measured signals from sensors are often corrupted by noise. This severely undermines the applicability of these works in real world and robustness of controller in presence of noise. The application of EIL and evaluation of the proposed controller strategies in the experimental facility sets this work apart. The effectiveness of the proposed self-learning neural controller is demonstrated by their performance with actual engine in a real world environment. The operation of controllers in noisy environment demonstrates their robustness.

Two different self-learning neural network controller are generated using NDP algorithm with 5 and 8 inputs respectively with weights, $w_{FC} = 0.7$, $w_{NOx} = 0.1$ and $w_{PM} = 0.2$. The selection of weights is random and is to show the ability of algorithm to minimize a multi-objective problem.

4.6.1 NDP Controller with 5 Inputs

Figure 4.12 shows the EIL implementation of NDP controller with 5 inputs. The controller resides on the dSPACE real-time platform and interacts with the engine through AVL PUMA. The virtual vehicle along with NDP controller is simulated over FUDS.

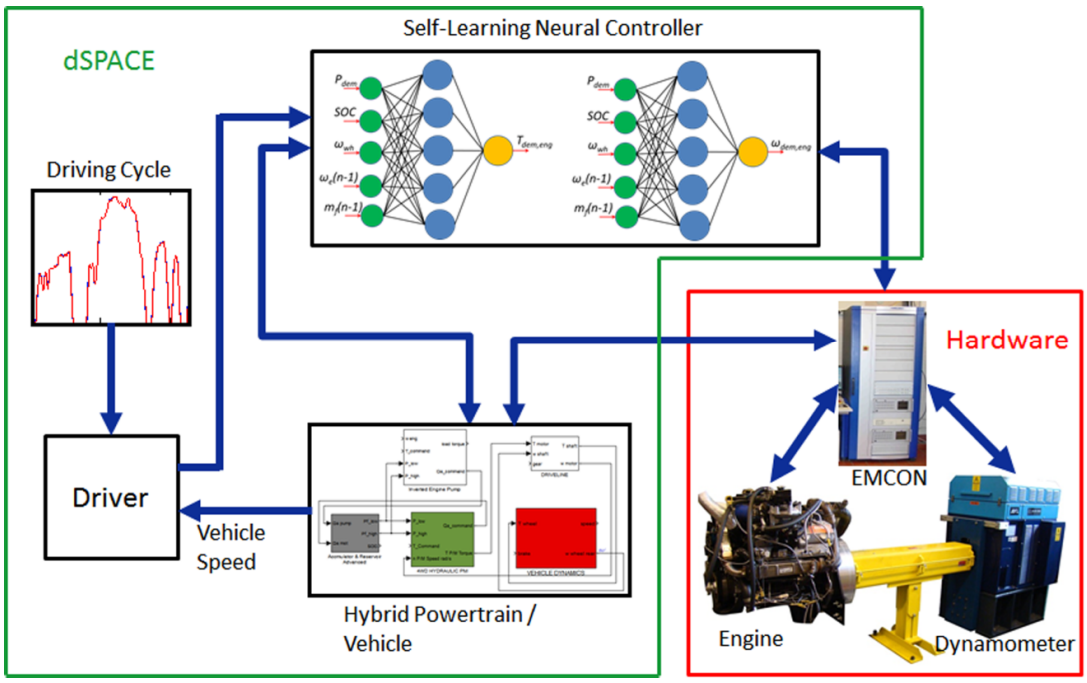
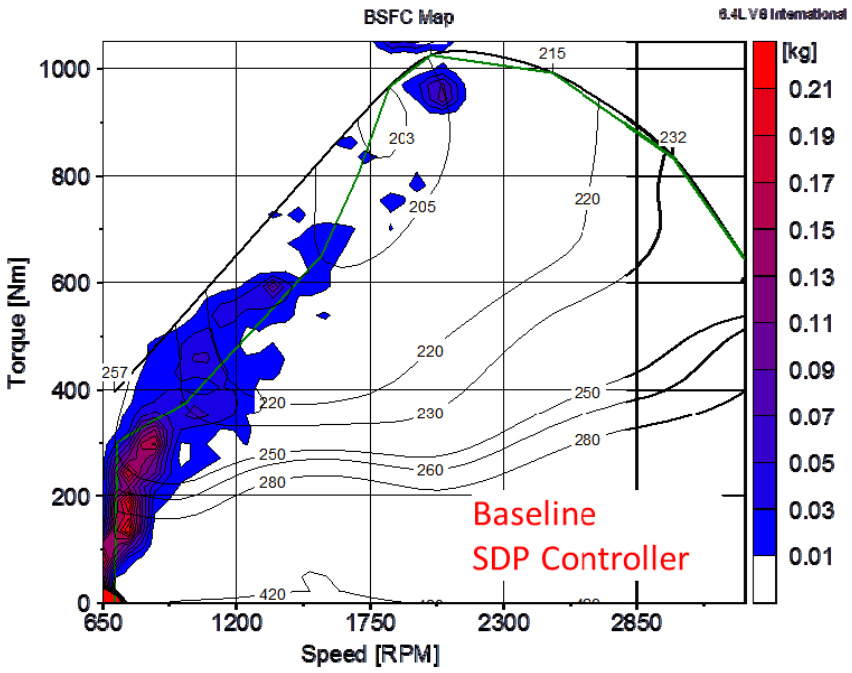
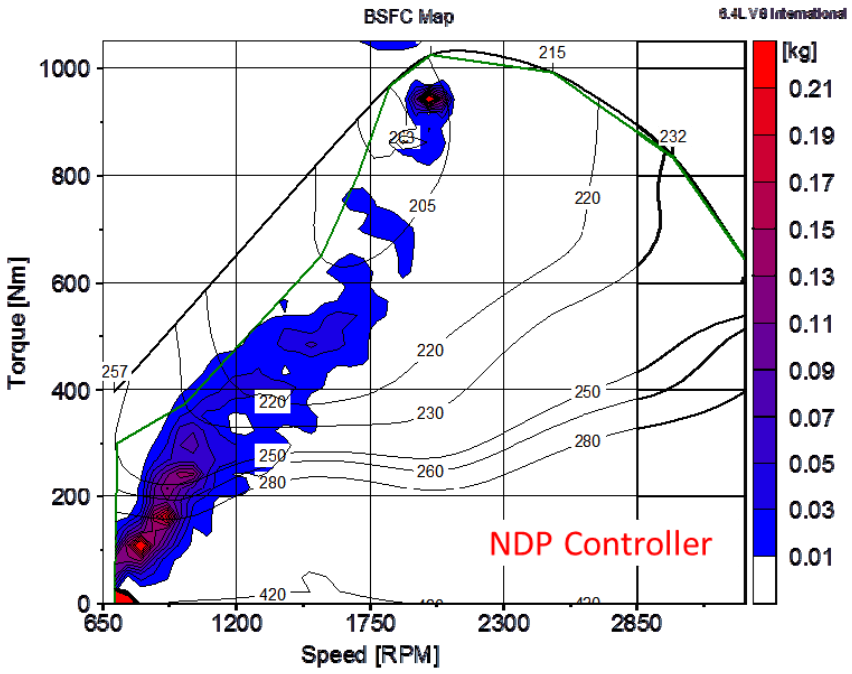


Figure 4.12: Schematic representation of EIL setup for the series hydraulic hybrid powertrain with self-learning neural controller with 5 inputs.

The NDP controller successfully learns to drive the vehicle by coordinating the physical engine and virtual hybrid powertrain. The hydraulics is used intelligently to minimize the harsh transient acceleration of the engine. To quantify the differences in engine operation with NDP and SDP based controllers, Figure 4.13 shows the engine operating points over BSFC map. The color scale indicates the amount of fuel consumed in each region during FUDS. The plot also shows the best BSFC line. It can be seen that the engine operation for NDP based controller deviates from the best BSFC line to reduce NO_X. The engine would have operated near the best BSFC line if the engine efficiency is the sole objective, as is the case with SDP based controller. Figure 4.14 shows the comparison of NO_X from NDP and SDP based controllers for a section of FUDS. The NO_X peaks from NDP based controller are considerably smaller and this result in 29.3% improvement of NO_X over SDP based controller in EIL during FUDS.



(a)



(b)

Figure 4.13: Engine visitation points on the BSFC map, with a color scale indicating the relative amount of fuel consumed in a given zone during EIL test over FUDS: a) system centric SDP controller designed with fuel economy considerations, and b) NDP controller with 5 inputs designed with fuel economy and transient emission consideration.

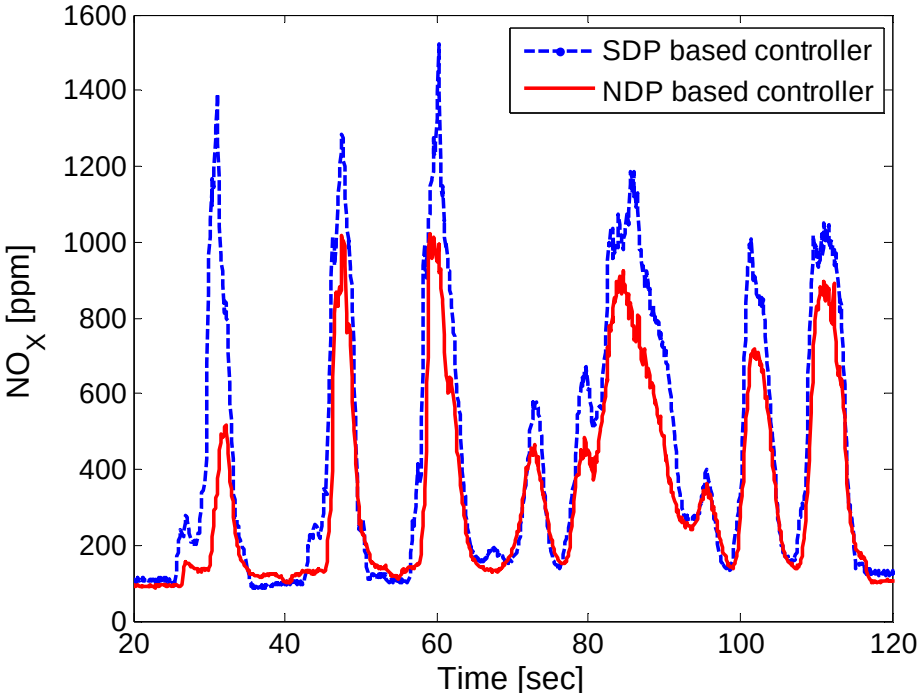


Figure 4.14: Instantaneous NO_X measurements during the EIL test of S-HHV over section of FUDS with SDP and NDP based controllers.

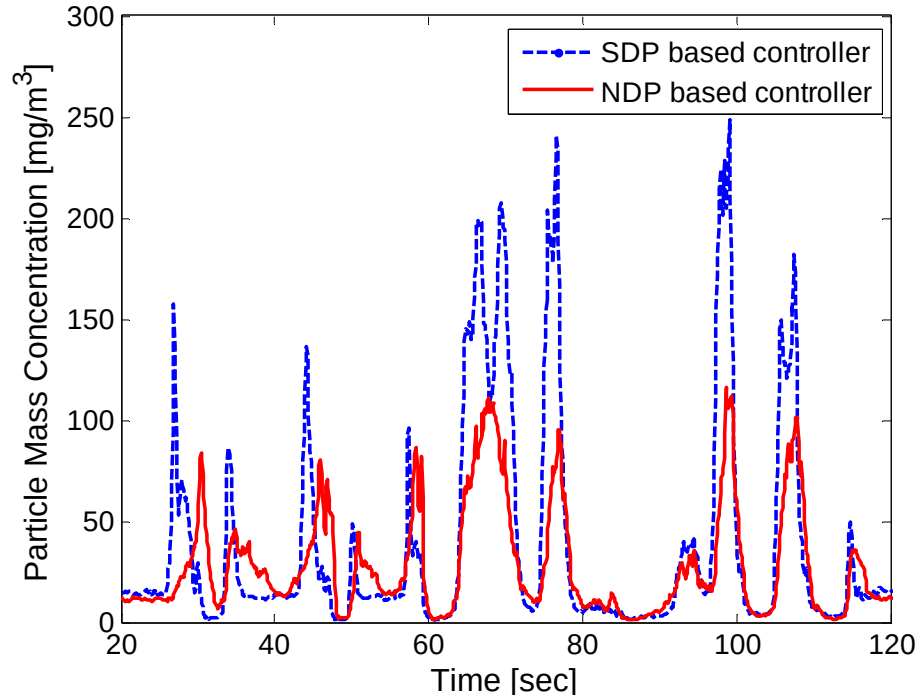


Figure 4.15: Instantaneous particulate concentration measurements during the EIL test of S-HHV over section of FUDS with SDP and NDP based controllers.

Figure 4.15 shows the time trace of particulate matter for SDP and NDP based controllers over a section of FUDS with the NDP based controller resulting in lower transient particulate matter spikes. Cumulatively for first two hills of FUDS, NDP based controller results in 10% reduction in particulate matter over SDP based controller. To explain the reasoning for reduction in transient soot spikes, one needs to consider the fuel injected per stroke for these two controllers, Figure 4.16. It can be seen that the fuel injection ramps up and down slowly for NDP based controller to reduce transient spikes of emission. This is in agreement with finding by Hagena et al. [11] that a step change in fueling results in large spike in transient emissions and can be significantly reduced if fueling change occurs gradually. The effect is particularly strong when a step change is initiated from idle [11]. The NDP strategy successfully avoids this, e.g. the engine is not brought down to idle at ~35 sec, and the ramp-up rate is milder.

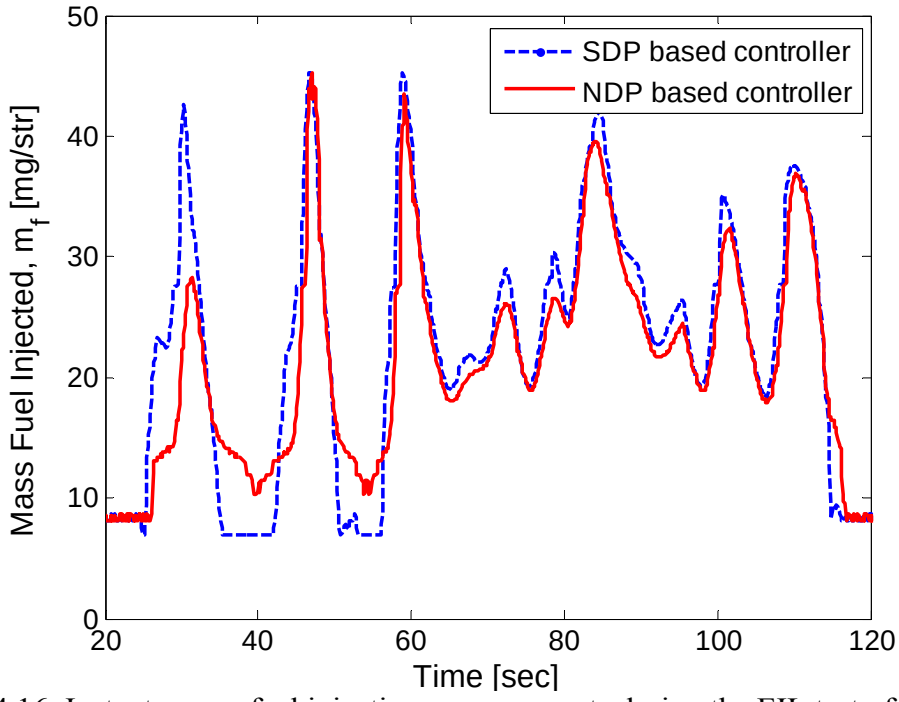


Figure 4.16: Instantaneous fuel injection measurements during the EIL test of S-HHV over section of FUDS with SDP and NDP based controllers.

By systematically evaluating the system behavior, the NDP controller manages such significant emission reduction with minimal fuel economy penalty. The S-HHV fuel economy with NDP based controller is 16.2 mpg, only 3% lower than the result obtained using SDP with fuel economy being sole objective. It should be noted that conventional

vehicle fuel economy is 10.2 mpg over FUDS, and both NDP and SDP based controllers give around 60% fuel economy improvement.

4.6.2 NDP Controller with 8 Inputs

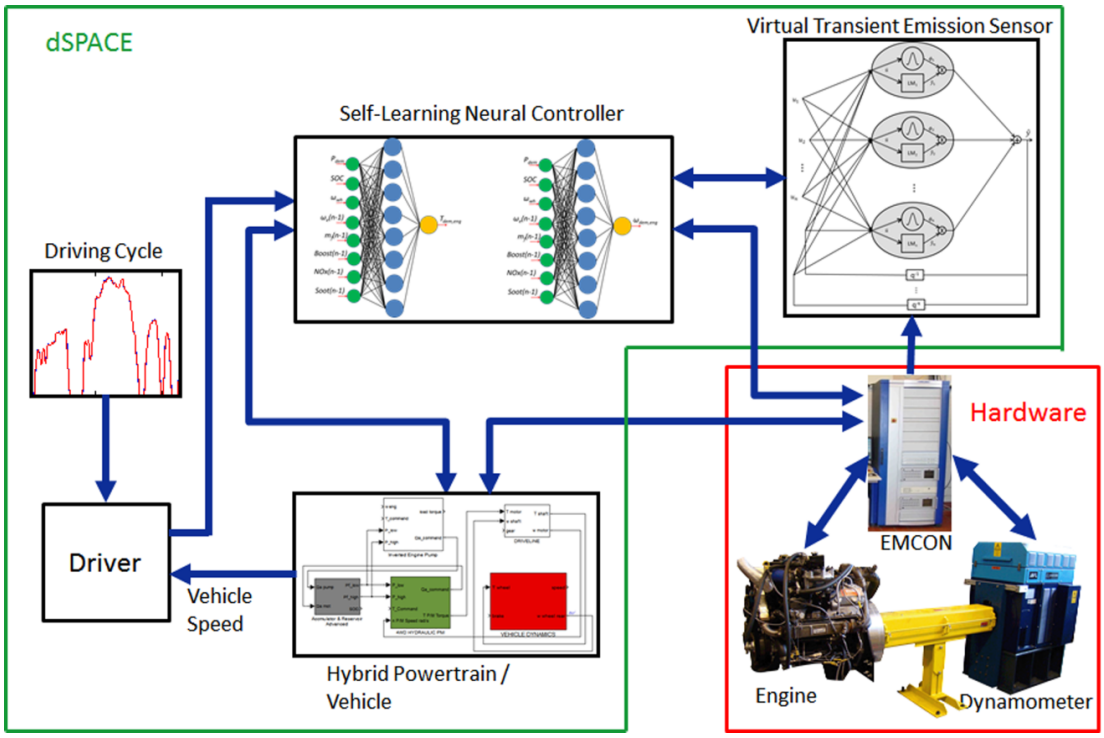


Figure 4.17: Schematic representation of EIL setup for the series hydraulic hybrid powertrain with self-learning neural controller with 8 inputs and virtual transient emission sensors.

Figure 4.17 shows the EIL implementation of NDP controller with 8 inputs. The implementation is similar to the NDP controller with 5 inputs. The self-learning controller resides on the dSPACE real-time platform along with the virtual soot and NO_X emission sensors. A neuro-fuzzy model tree with orthogonal least square based transient virtual emission sensors is used in this section. The details about the structure and their design are given in section 5.5.2.1. The controller and emission sensors interact with the engine through AVL PUMA. Figure 4.18 shows the behavior of NDP controller with 8 inputs along with NDP controller with 5 inputs over a section of FUDS. The engine operation with NDP controller with 8 inputs deviates very slightly from that of NDP controller with 5 inputs. This difference in engine operation is due to additional

information available to controller in terms of intake manifold pressure, previous estimated soot and NO_X from virtual sensors. Figure 4.19 shows the engine operating points over BSFC map with the color scale indicating the amount of fuel consumed in each region during FUDS. The modified engine operation results in further reduction in NO_X, Figure 4.20 and almost similar soot, Figure 4.21 with even better fuel economy.

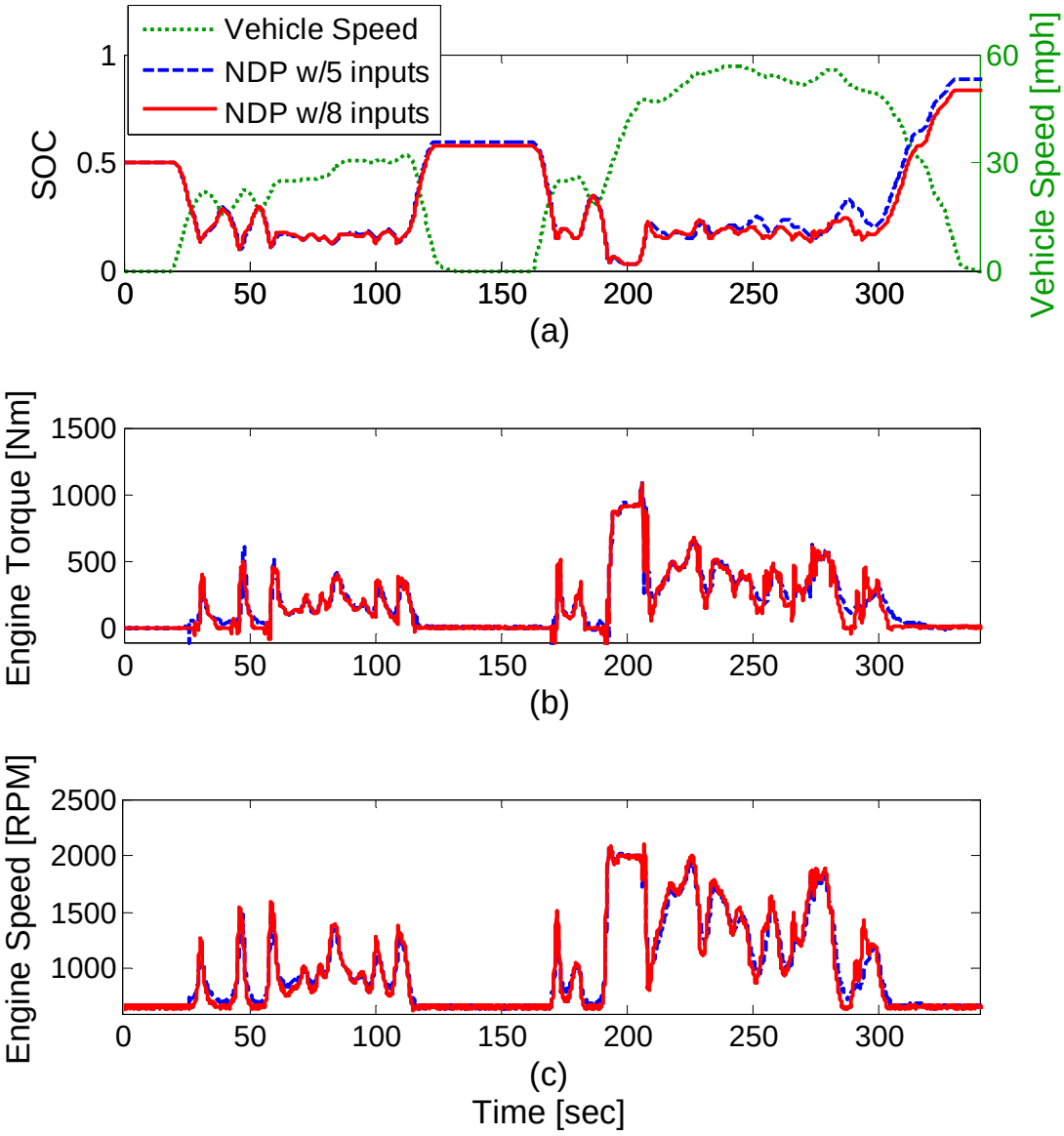


Figure 4.18: EIL results of series hybrid powertrain behavior with two different NDP controllers, both designed with fuel economy and transient emission consideration over section of FUDS: a) vehicle speed and SOC, b) engine speed, and c) engine torque.

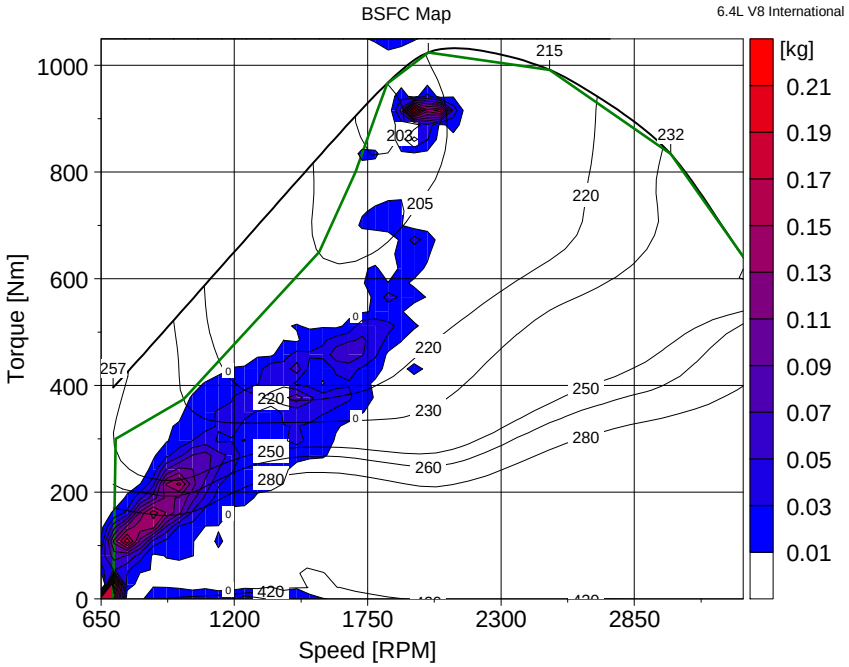


Figure 4.19: Engine visitation points on the BSFC map, with a color scale indicating the relative amount of fuel consumed in a given zone during EIL test over FUDS for self-learning neural controller with 8 inputs in S-HHV.

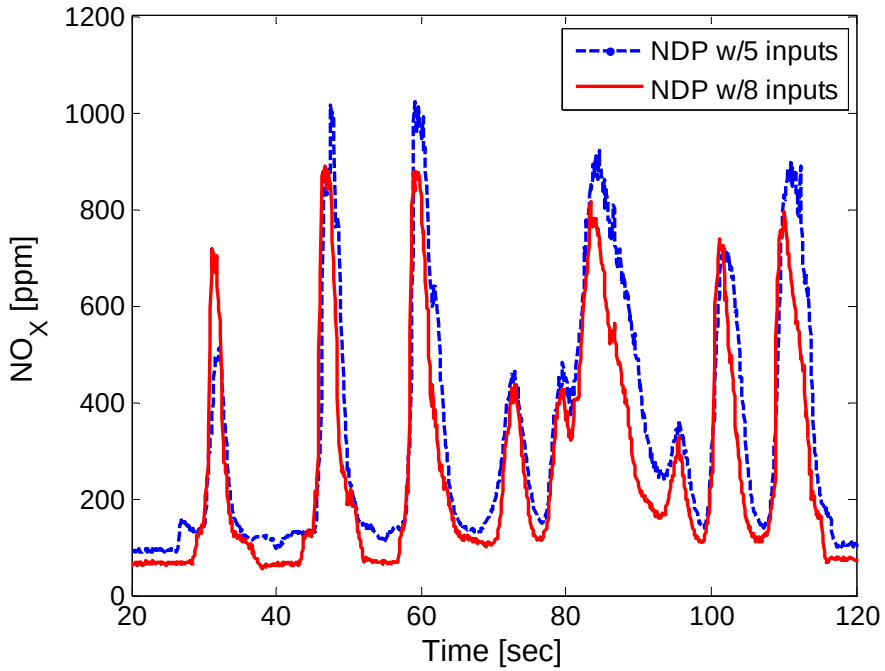


Figure 4.20: Instantaneous NO_X measurements during the EIL test of S-HHV over section of FUDS with two different NDP based controllers.

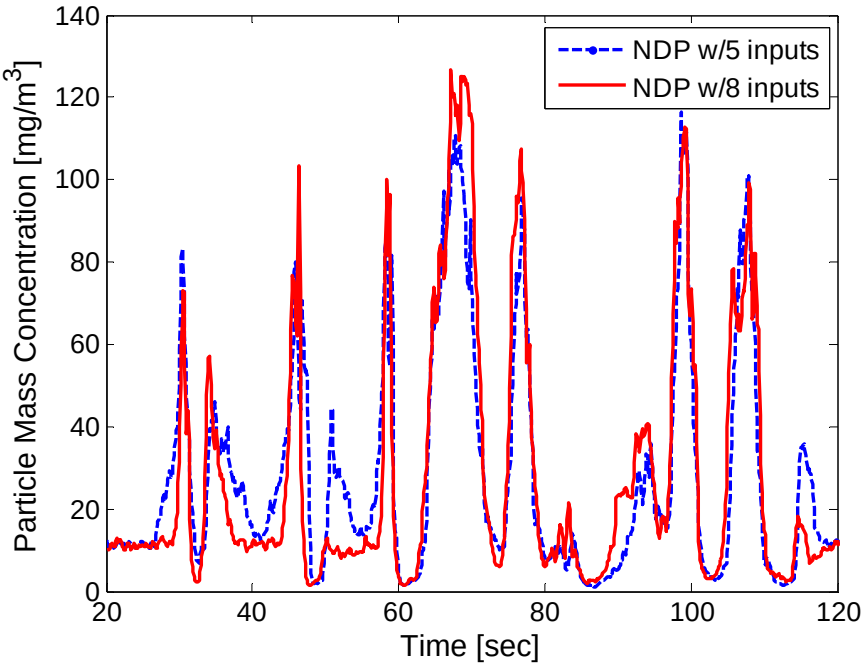


Figure 4.21: Instantaneous particulate concentration measurements during the EIL test of S-HHV over section of FUDS with two different NDP based controllers.

Overall, the NDP controller with 8 inputs manages to further reduce soot and NO_X emissions compared to SDP controller and almost recovers the fuel economy penalty associated with NDP controller with 5 inputs. Table 4.3 summarizes the percentage change in fuel economy along with cumulative emissions of particulate matter and NO_X with NDP based controllers over SDP based controller.

Table 4.3: Comparison of fuel economy and cumulative particulate matter and NO_X emissions for a S-HHV with two different NDP controllers and SDP controller during EIL test over FUDS

	Fuel Economy (mpg)	**Particulate Matter***	**NO_X**
SDP controller	16.7	-	-
NDP controller with 5 inputs	16.2 (- 3.0 %)	- 10 %	- 29.3 %
NDP controller with 8 inputs	16.6 (- 0.7 %)	- 16 %	- 38.5 %

* Only first two hills of FUDS

Chapter 5

MODELING TRANSIENT EMISSIONS

5.1 Introduction

The energy security and climate change concerns provide strong impetus for research of efficient engine concepts. In light of the stringent CAFE regulations, diesel engines are becoming an attractive option. However, diesel engines are facing challenges in meeting emission regulations owing to the nature of diesel combustion. Heavy-duty diesels are facing particularly hard challenges regarding emissions regulations. Transient diesel engine emissions of particulates and NO_X are very complex phenomena owing to the nature of diesel combustion. Desire to maximize the efficiency and power density, while addressing the emissions challenge too, leads to increased complexity with many actuators like variable geometry turbochargers (VGT), exhaust gas recirculation (EGR) and variable valve timing (VVT). Aftertreatment is necessary to bring the tailpipe emissions down to compliance levels. However, the burden is equally shared by the in-cylinder clean combustion strategies, advanced catalyst and diesel particulate filters since size and cost of aftertreatment is an issue.

Advanced approaches such as model based predictive control, closed loop combustion control and development of advanced supervisory strategies for hybrid propulsion systems will be essential for coping with new regulations and complex hardware. In all cases, model-based soot and NO_X virtual sensors can provide real-time predictions and enable strategies that require feedback of emissions under transient operating conditions.

Virtual sensors have gained recent importance in many fields of technology like aerospace and medical science [114]. Virtual sensors are useful in cases where direct measurement of signal is not possible and model based estimators are used for predicting

the state [115]. They have been used as backup sensors to provide failsafe mode operation. Finally, they have found applications where the original sensor is either too expensive or impractical to be actually deployed. Present day, soot and NO_X sensors fall into this category. Developing sophisticated real sensor for particulate matter and NO_X is challenging and is active an area of research.

Present day emission models generally fall into two categories, at the extreme end of the spectrum when it comes to complexity and computational speed: (i) lookup table based steady state models, and (ii) computational fluid dynamics (CFD) with chemical kinetics based models. CFD and chemical kinetics based models can capture transient effects but are very complex and computationally slow. This makes them impractical to be used with optimization routines and as virtual sensors with physical engine. Steady state map based models fail to capture the transient nature of emission when engine is operated transiently and underestimate soot production [10], [12]. The transient spike is higher and precedes the prediction by quasi-steady state. Therein lies the impetus for transient emission models, fast enough to be employed for control-oriented problems or as virtual real-time sensors, and yet detailed enough to capture system dynamics accurately.

The emission (particulate matter and NO_X) formation in diesel engine is highly nonlinear and displays complex dynamic behavior with change in operating condition. The problem becomes even more challenging if one considers stochastical nature of combustion. To design a unified model capable of capturing all the nonlinearities, particularly under highly dynamic operating conditions and valid for entire engine operating range will invariably have complex structure and very high order. In addition, training such model will pose numerical challenges. This is especially true for particulate matter emission and explains the reason for lack of progress in modeling transient particulate matter emission for diesel engines.

This chapter discusses the development of models for transient emissions of particulate matter and NO_X for medium duty diesel engine capable of predicting over a wide-range of operating conditions. The model is intended to run on a microprocessor in real-time and predict instantaneous engine-out particulate and NO_X emissions using signals from the ECU and low-cost physical sensors. An approach with multiple local

models is proposed. The models belong to hierarchical class of models, specifically neuro-fuzzy model tree. The underlying principle is a divide-and-conquer strategy, whereby, the engine operating space is subdivided into multiple smaller subspaces and individual submodels are used for identification. Hence, the complex problem is subdivided into multiple simpler problems, which are then identified using simpler models. The neuro-fuzzy model tree based emission sensor is capable of learning complex, nonlinear and multidimensional dimension association between inputs and outputs. The neuro-fuzzy model has parallel structure with respect to local models and thus can be efficiently implemented in hardware. Emission models developed in this study are driven by experimental data and are specific to a particular diesel engine but the methodology developed is universal and can be applied to any other engine.

The chapter begins with a brief background on combustion and emission formation in diesel engine. Next, the chapter discusses the methodology for design of experiment and importance of design of perturbation signal for system identification of nonlinear plants. A perturbation signal is designed specifically for characterizing the dynamic engine operation and resulting emissions. Next, the chapter describes the neuro-fuzzy modeling approach and the training algorithm. The training algorithm is augmented with input regressor selection techniques like orthogonal least squares and automatic relevance determination. This prevents local models to be trained to inconsequential inputs and improves the overall model performance. The neuro-fuzzy modeling technique is then used to develop two different classes of virtual sensors. One specific for real-time implementation with physical engine, and the other for dynamic programming framework with extremely fast computation, very small number of states and inputs available only from engine simulation model. Finally, the architecture of virtual sensors along with validation results is presented.

5.2 Diesel Combustion and Emission Background

Diesel combustion is a very complex process. Optical studies combined with analyses of engine cylinder pressure data have led to the widely accepted phenomenological understanding proposed by Dec [116]. After injection, the fuel

evaporates and mixes with air, and owing to very high temperatures, autoignites after a delay (*ignition delay*). The fuel/air mixture prepared during the ignition delay period burns rapidly and this is referred to as *premixed phase* of burning. Following the premixed combustion, the fuel injection continues through the *mixing-controlled burn phase.* The liquid core of injected fuel persists and the fuel droplets downstream of the liquid core are evaporated, facilitated by turbulent air entrainment. This results in formation of relatively uniform, high equivalence ratio (fuel/air of 2-4) zone, extending ahead and around the liquid fuel core. A standing premixed flame forms at the boundary of this gaseous fuel/air zone, and owing to excessive rich conditions, produces polycyclic aromatic hydrocarbons (PAH - soot precursor) and solid particles. The soot particles are initially small but grow in size and concentration as they move towards the head vortex. The particle accumulation process continues in the head-vortex zone surrounded by a thin diffusion flame. The outer edge of diffusion flame is surrounded by OH radicals and oxygen molecules, which oxidize particles that reach outer boundary. The high temperature and presence of oxygen is highly conducive for NO_X production. The NO_X production continues even after end of injection due to latter part of diffusion burning and more oxygen available due to further mixing.

Even though the fundamentals of the emission formation process remain same with transient operation of engine, it makes modeling them even more complex due to constantly changing combustion environment and engine subsystem interactions. Fluctuations in charge composition, stochastic nature of turbulence, mixing and combustion makes the interaction of species in engine cylinder inconsistent. Instantaneous composition in the cylinder and flow field goes through dramatic excursions from the steady state values after a rapid change of engine command. Thus, steady state emission models cannot capture these effects and are incapable of predicting transient emissions [117]. In summary, multiple aspects of the very complex phenomena in the combustion chamber have to be understood, and experimental insights are necessary to support virtual sensor development.

Next subsection briefly describes the NO_X and soot formation. The detailed chemical process behind the formation of these species is beyond the scope of this work.

5.2.1 NO_X Emissions Formation

NO and NO_2 are generally grouped together as NO_X emissions with *NO* being the more dominant species. The principal source of *NO* is the oxidation of atmospheric nitrogen with presence of nitrogen in fuel further adds to *NO* formation. The mechanism of *NO* formation is given by extended Zeldovich mechanism.

$$O + N_2 = NO + N \tag{5.1}$$

$$N + O_2 = NO + O \tag{5.2}$$

$$N + OH = NO + H \tag{5.3}$$

NO_2 can be 10 to 30% of the total exhaust oxides of nitrogen in diesel engines [118], [119]. Plausible explanation is the conversion of *NO* formed in the flame zone via

$$NO + HO_2 \rightarrow NO_2 + OH \tag{5.4}$$

Subsequently, conversion of this NO2 to NO occurs via

$$NO_2 + O \rightarrow NO + O_2 \tag{5.5}$$

unless NO_2 formed in the flame is quenched by mixing with cooler fluid. This explains the reason for highest NO_2/NO ratio occurring at light loads in diesel when cooler regions are more predominant. Due to the nature of diesel combustion, the local temperature variation throughout the cylinder leads to heterogeneous spatial and temporal NO_X formation. Studies have shown that NO_X is primarily formed in the diffusion flame surrounding the spray head vortex. Increase in oxygen concentration levels and thereby increase in adiabatic flame temperatures leads to increase in NO_X formation. Addition of dilutants like EGR and nitrogen reduces peak flame temperatures and helps in reducing NO_X emissions. The critical time for NO formation is when the burned gas temperatures are highest i.e. start of combustion when the burned gases are compressed further and shortly after the peak pressures are reached in the cylinder. The decreasing temperature due to expansion and mixing of high temperature gases with air or cooler burned gases eventually freezes NO chemistry. A chemical kinetics based model would still be impossible to use with system level studies due to the need for instant local temperature and pressure.

5.2.2 Particulate Matter and Soot Emissions Formation

Diesel particulates consist principally of combustion generated carbanaceous materials (soot) on which some organic compounds (soluble organic fraction, SOF) have been absorbed [118]. Most particulate formation is due to incomplete combustion of fuel and engine oil. The characteristics of diesel combustion - high gas temperatures and pressures, complex fuel composition, high turbulent mixing and three-dimensional geometry make it difficult to interpret soot formation. Haynes et al. [120] summarized the soot formation in two stages. Firstly, *particle formation*, the condensed phase material arises from the oxidation or pyrolysis. This phase is known as nucleation mode. The products mainly consist of unsaturated hydrocarbons and polycyclic aromatic hydrocarbons (PAH) formed during mixing-controlled burn. In second stage, *particle growth* occurs, which include surface growth where bulk of solid phase material is generated by attachment of gas-phase species to the surface of particles, coagulation and aggregation where particles collide and coalesce. This phase is known as agglomeration mode. Figure 5.1 shows the stages in formation of soot in diesel engine [121].

The particulate matter encompasses both soot and soluble organic fraction, the terms particulate matter and soot are interchangeably used in this work.

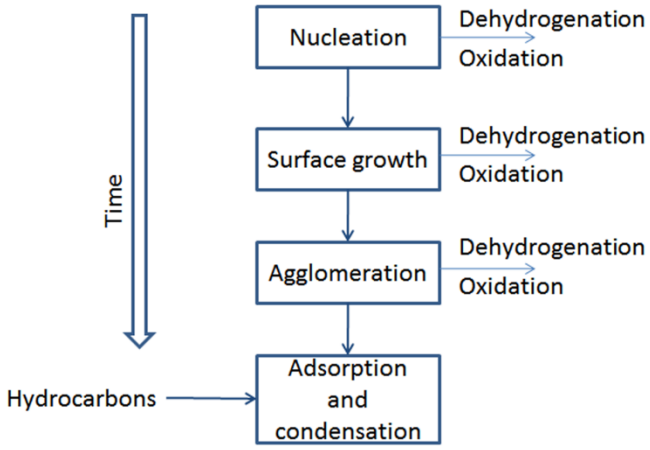


Figure 5.1: Schematic illustrating the process leading to soot formation in a diesel engine.

5.3 System Identification

Modeling and identification of nonlinear dynamic systems is a challenging task. Nonlinear models do not share many properties and require iterative approach to identify not only model parameters but also the type of model itself. Figure 5.2 shows a system identification flowchart followed in this chapter for design of transient emission models.

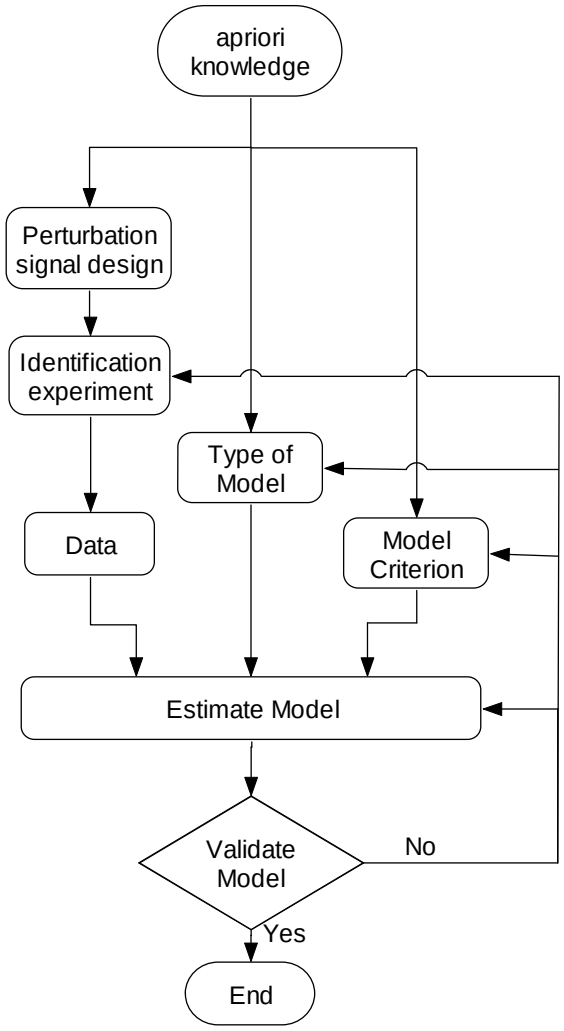


Figure 5.2: System identification flowchart for black-box modeling.

One of the most crucial tasks in system identification of black box model is the design of appropriate perturbation signals for excitation and gathering data. This step is even more important for nonlinear systems than linear systems owing to the inherent complexity of the model. Independently of the chosen model architecture and structure,

the quality of system identification signal determines an upper bound on the accuracy that can be best achieved by the given model.

Two aspects are very important in selecting test signals for process identification, signal waveform and its power spectrum or frequency content. In order to have data with high signal-to-noise ratio (SNR), the test signal should have high signal power. Due to constraints on signal amplitudes, a small amplitude is desirable for a given signal power. This property can be expressed in terms of crest factor, C_r. A good signal should have small crest factor. Binary signal have the smallest crest factor possible, 1. For a signal *u(t)* with 0 mean, its crest factor as defined by Ljung [122] is

$$C_r = \sqrt{\frac{\max_t u^2(t)}{\lim_{N\to\infty} \frac{1}{N}\sum_{t=1}^{N} u(t)}} \tag{5.6}$$

Ljung [122] showed that model error distribution in the frequency domain is affected by the spectrum of the test signal. Also, in order to guarantee that the estimation algorithms have unique solutions, the test signals need to be persistent. A discrete-time signal *u(t)* is said to persistently exciting of order *n* if the following limit exists

$$R_u(\tau) = \lim_{N\to\infty} \frac{1}{N}\sum_{t=1}^{N} u(t)u(t-\tau) \tag{5.7}$$

and the matrix

$$\begin{bmatrix} R_u(0) & R_u(1) & \dots & R_u(n-1) \\ R_u(-1) & R_u(0) & & \vdots \\ \vdots & & \ddots & \\ R_u(1-n) & & \dots & R_u(0) \end{bmatrix} \tag{5.8}$$

is nonsingular. Therefore, test signal needs to be designed carefully.

For a linear model, a binary random signal suffices, but it is not suitable for nonlinear system identification because it is not persistently exciting in amplitude. The selection of perturbation signal for nonlinear system identification requires more careful consideration. A multi-level Pseudo Random Signal (m-PRS) or Amplitude Modulated PRBS (APRBS) is applied in this work for nonlinear system identification. m-PRS signals are periodic, deterministic, persistently exciting and have autocorrelation function

similar to white noise. These characteristics make m-PRS signal well suited for this type of work.

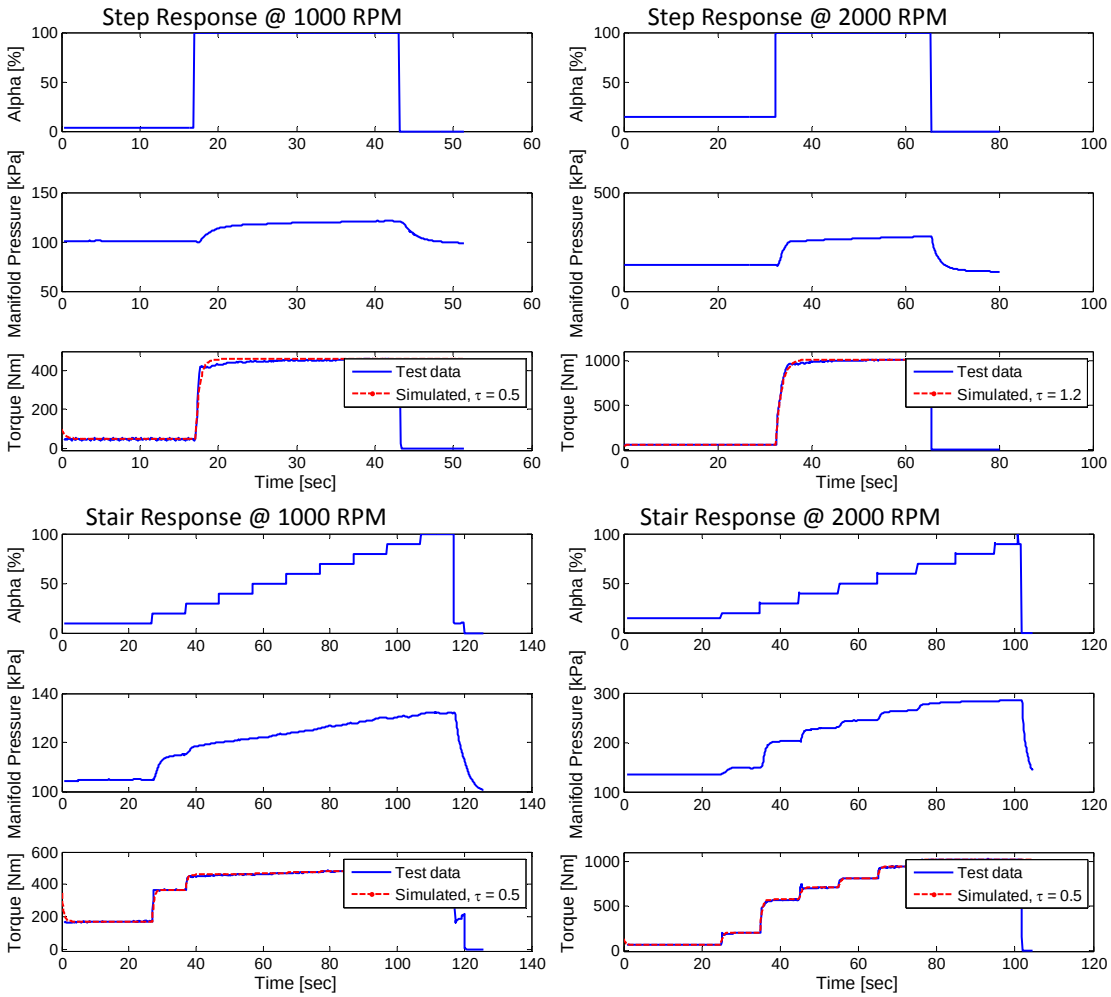


Figure 5.3: Preliminary step and staircase tests performed at different engine speeds to characterize the engine.

Modern diesel engine is highly complex system with nonlinear responses to action of multiple actuators. Furthermore, the diesel engine emission formation is a highly nonlinear process owing to complex in-cylinder mixing and chemistry. To create a transient diesel engine emission model valid over entire operating region, the test signal should excite the entire engine operating frequencies. This will ensure the training data are "rich". Using a priori information about engine and preliminary tests like step and stair case excitation, information about bandwidth of system dynamics, dominant settling time, etc. is obtained [122], [123], [123]. Figure 5.3 shows results from the step and

staircase test at different engine speeds. This information is used to create desired perturbation signal i.e. the signal with appropriate frequency range, switching time and amplitude level specific for this work. The switching time for signal is short enough to prevent capturing predominantly steady state data, but long enough to allow engine transients to fully develop before the next instance of signal is sent.

5.3.1 Multilevel Pseudo Random Signal

Theory behind generation of m-PRS signals is well developed [124], [125]. m-PRS are generated using *q*-level shift registers where *q* is the number of levels and is prime or power of prime, i.e. *q = 2, 3, 4, 5, 7, 8, 9, 11, 13, ...* (Godfrey [124]). The basis of generation of these m-PRS signals lies in finite field theory [124]. A Galois field, defined by *q*, guides the creation of pseudo-random sequence, x_r. The length of sequence $\{x_r\}$ is $q^n - 1$, where *n* is an integer. The sequence is generated by shift register, Figure 5.4, with feedback to the first stage consisting of the modulo *q* sum of the outputs of the other stages multiplied by coefficients $c_1, \ldots, c_n$ which are also integers *0, 1, ..., (q-1)*.

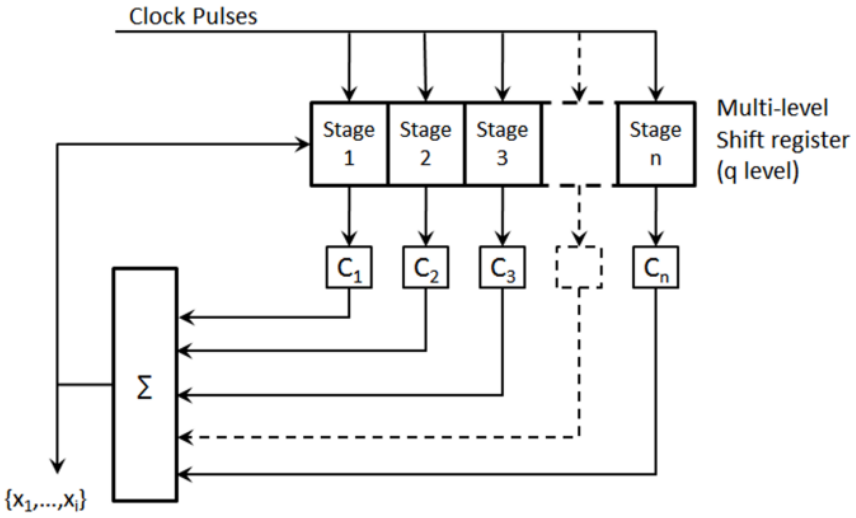


Figure 5.4: Schematic illustrating the q-level shift register configuration for generating m-sequence pseudo random signal.

The feedback configuration corresponds to a primitive polynomial, modulo *q*. Consider a primitive polynomial, *modulo q*,

$$a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0 \tag{5.9}$$

then the corresponding characteristic equation in delays, D, introduced by shift register

$$a_n D^n + a_{n-1} D^{n-1} + \ldots + a_1 D + a_0 = 0, \ modulo\, q \tag{5.10}$$

For Figure 5.4, the logic connections to the first stage is

$$a_0 X = -a_1 DX - \ldots - a_{n-1} D^{n-1} X - a_n D^n X, \ modulo\, q \tag{5.11}$$

where $a_0 = 1$ and the remaining coefficients $a_1, \ldots, a_n$ are integer values in range 0 to $q-1$.

Applying modulo q subtraction,

$$X = c_1 DX + \ldots + c_{n-1} D^{n-1} X + c_n D^n X, modulo\, q \tag{5.12}$$

where $c_r = (q - a_r), \ r = 1, 2, \ldots, n$

The sequence, $\{x_i\}$, generated from above equation is then mapped into pseudo-random sequence $\{u_i\}$ of real numbers according to equation

$$u_i = u(x_i) \quad \forall i \tag{5.13}$$

The pseudo-random sequence $\{u_i\}$ is converted by a zero-order-hold into the stepwise continuous pseudo-random signal $u(t)$, according to equation

$$u(t) = u_i \quad i\Delta t \le t \le (i+1)\Delta t \tag{5.14}$$

The clock interval Δt determines the temporal nature of the pseudo random signal $u(t)$.

Test signal chosen for transient engine emission identification models is an 11-level m-PRS signal. Advantage of using a deterministic signal is that it can be applied to the system multiple times and any corrupted data can be easily discarded. A Galois field (GF), defined by 11 elements and a primitive polynomial

$$f(x) = 1 + x^2 + 4x^3 \tag{5.15}$$

is used to create the 11-level sequence which is then mapped to 0-100 based on following mapping.

$$\begin{aligned}
&u(0) = 0, && u(g^0) = u(1) = 10, && u(g^1) = u(4) = 40 \\
&u(g^2) = u(5) = 50, && u(g^3) = u(9) = 90, && u(g^4) = u(3) = 30 \\
&u(g^5) = u(10) = 100, && u(g^6) = u(7) = 70, && u(g^7) = u(6) = 60 \\
&u(g^8) = u(2) = 20, && u(g^9) = u(8) = 80
\end{aligned} \tag{5.16}$$

where g is the primitive element, given by $g = c_n$ and modulo q arithmetic is applied.

Figure 5.5 shows the 100 sec period of input signal generated in Simulink, Figure 5.6. The signal has a mean of 49.85 and crest factor of 1.59.

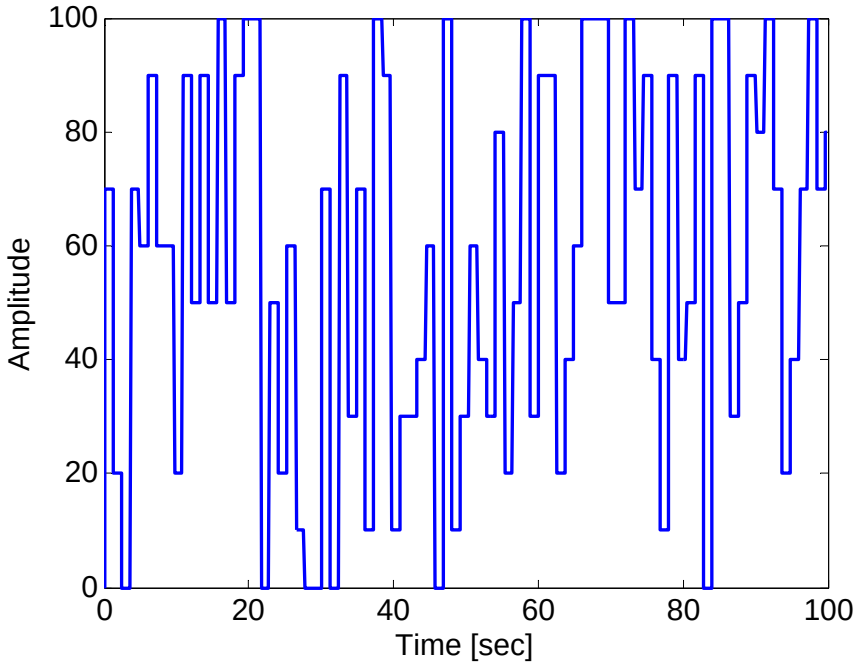


Figure 5.5: A section of m-level pseudo random signal.

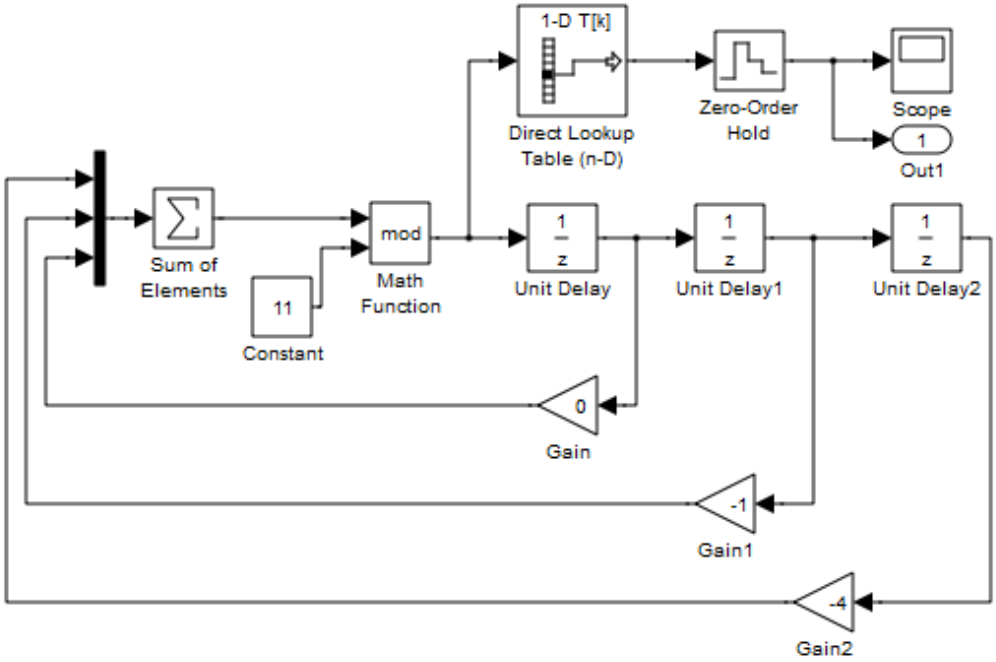


Figure 5.6: Illustration of Simulink implementation of m-PRS generator.

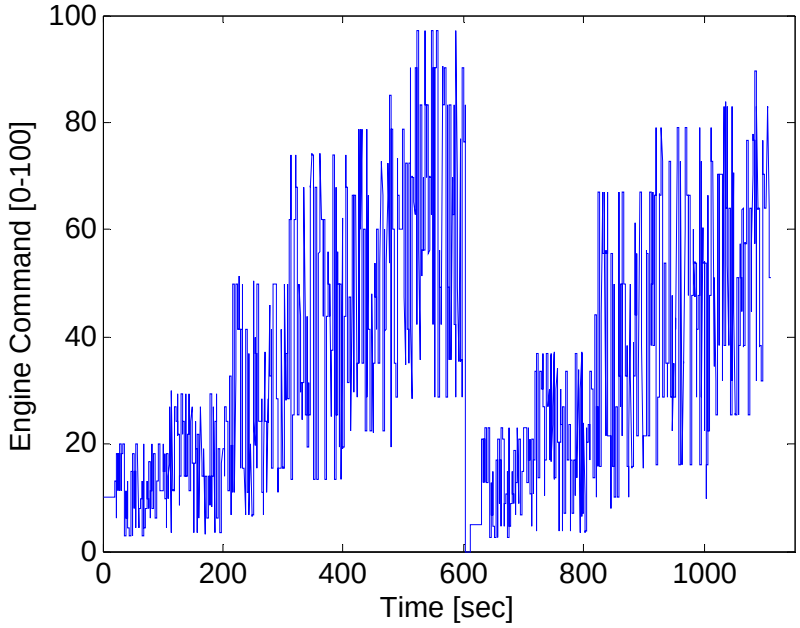


Figure 5.7: Throttle signal applied to engine for generating training data for system identification.

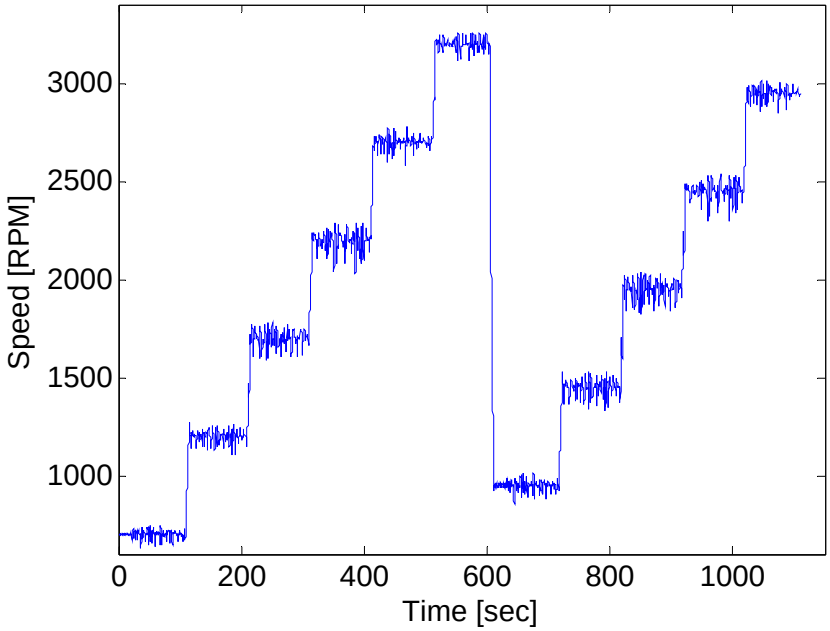


Figure 5.8: Speed signal applied to engine for generating training data for system identification.

The 11-step m-PRS signal created earlier is applied to engine as throttle (fueling) command. The signal is further scaled between maximum and minimum throttle command possible to the engine for a given speed. Figure 5.7 and Figure 5.8 shows the

engine throttle and engine speed signal applied to engine. Figure 5.9 shows the engine operating points on 2-D engine operating space spanned by engine speed and engine torque. Similar plots can be plotted with different axis like engine boost and EGR valve opening. It can be seen from Figure 5.9 that the whole engine operating space is completely covered and the training data obtained is "rich".

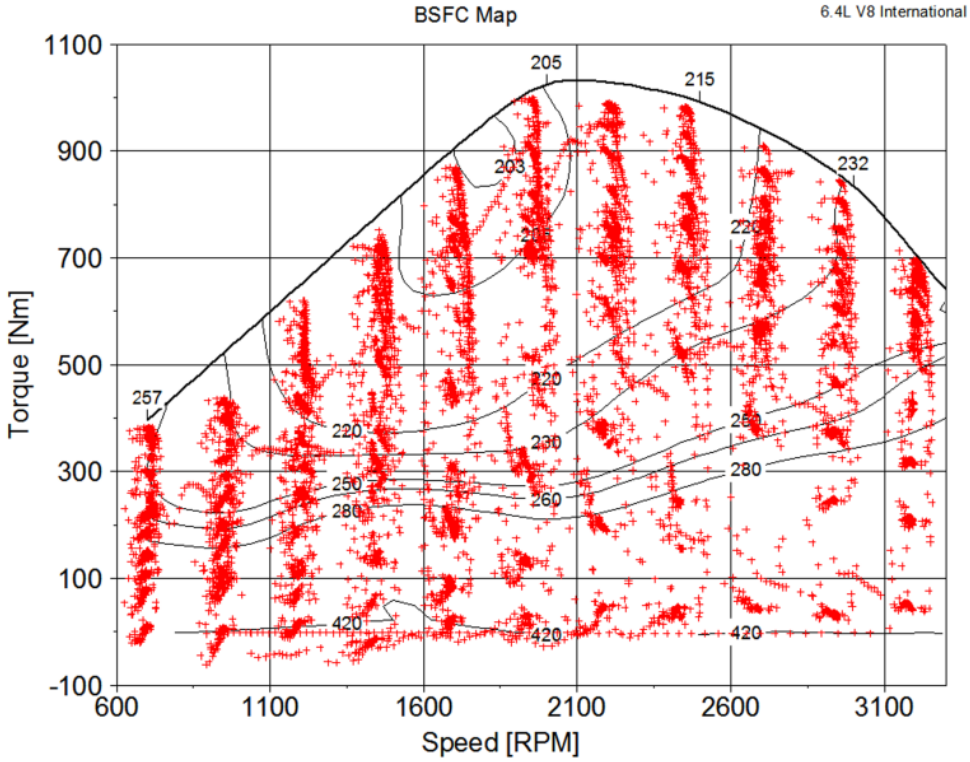


Figure 5.9: Engine visitation points on the speed vs. torque map during system identification test.

5.3.2 Selection of Input Regressor

A three-step process is employed in this dissertation for selection of input regressor set. First, a set of potential input signals is generated based on the knowledge of diesel engine combustion and emission formation like engine speed and fuel injection. Hagena's [14], [11] work gives an insight into potentially relevant signals for predicting transient emissions. Hagena et al. [11] showed that transient NO_X spikes are primarily dependent on lag between increased fueling and boost combined with EGR starvation while the

particulate transients are initiated by step change in fueling commands. The latter are highly dependent on rate of change in fueling command. Important inclusions based on Hagena's work are the rate of change of fuel injection and post injection.

The above set of input variables are further shortlisted based on ease of availability of signal. The ease of availability means the signal should be readily available either through onboard engine sensors or from ECU. Residual content in the fresh charge cannot be easily measured onboard engine and hence is not included in the set of input signals. However, EGR valve is actuated by ECU and the percentage opening of that valve is available. Hence, this signal is considered instead.

Finally, the available input signals are cross-correlated with soot and NO_X emissions. Cross-correlation is a sliding inner product of two signals and gives the measure of similarity of two signals with one signal time-shifted.

$$corr_{xy}(k) = \frac{1}{N} \sum_{k=1}^{N-|\kappa|} x(k) y(k-\kappa) \tag{5.17}$$

where *x(k)* is one of the selected input variable, *y(k)* is the emission with time instant *k* and κ is the time shift or "lag" between two signals.

The signals with high correlation are included in the input data set. Figure 5.10 gives few such example of cross-correlation of emission species with input regressors. Cross-correlation of soot and NO_X emission with various individual variables also shows that emission formation is not only a function of present information but depends on the time history as well. Therefore, time shifted input signals are also considered as separate inputs.

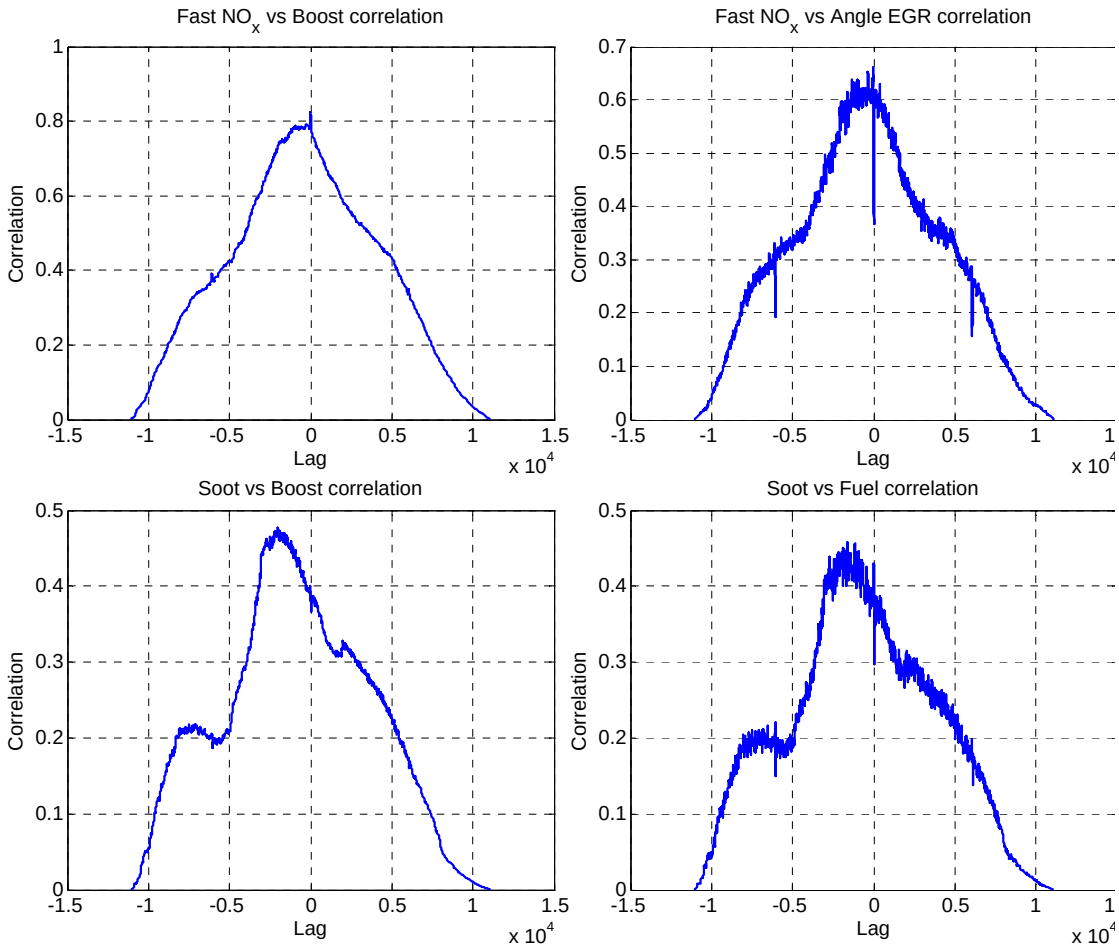


Figure 5.10: Cross-correlation of NOx and soot emissions with different potential input regressors.

5.3.3 Training Data

The specially designed perturbation signal (section 5.3.1) is applied to the engine and the engine response along with the engine-out emissions is sampled at 10 Hz. The training data set is then generated from the recorded data using the input regressor selection procedure described in section 5.3.2. Figure 5.11 shows a section of training data set with some of the input regressors, and engine-out soot and NO_X emissions.

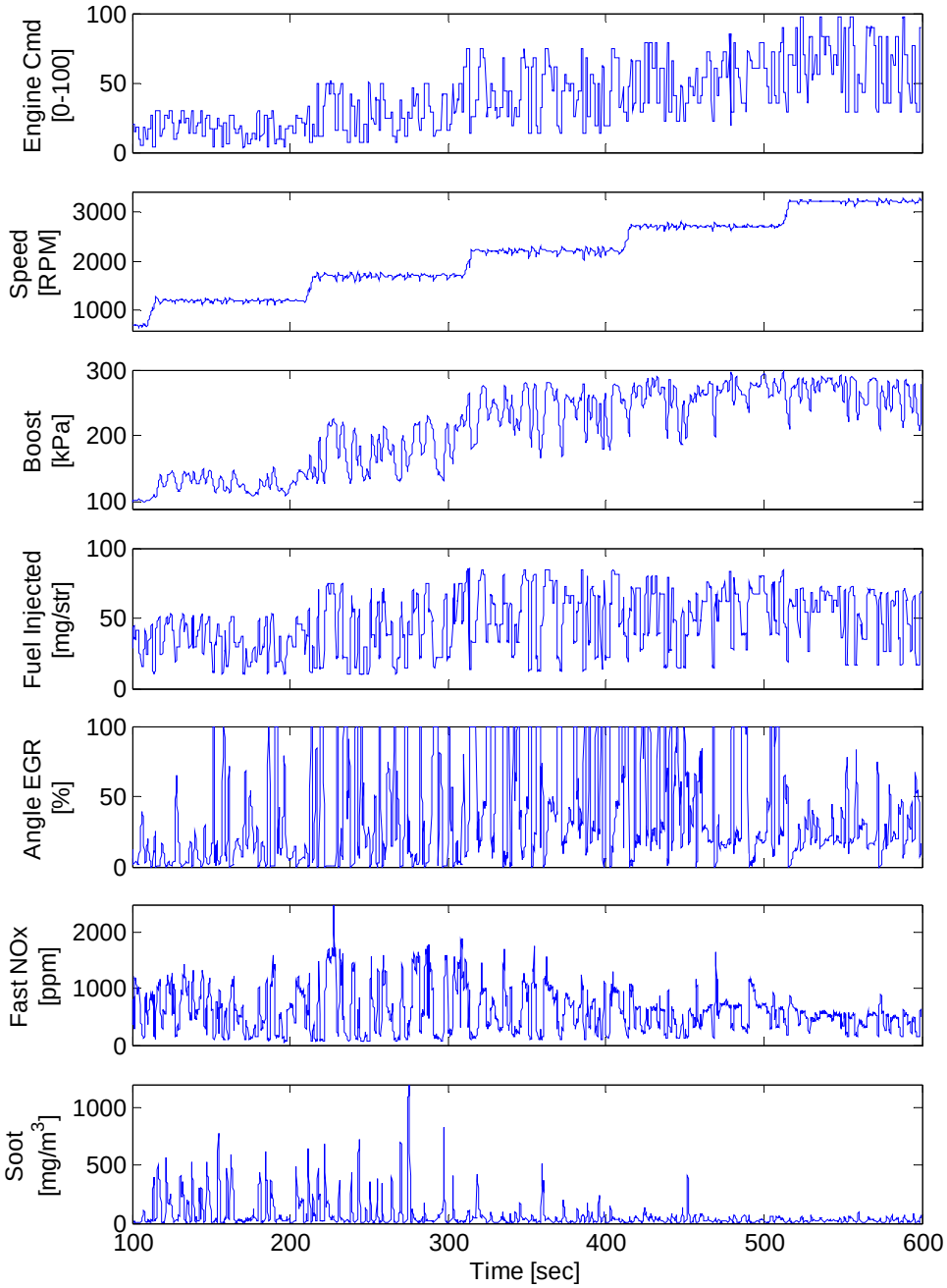


Figure 5.11: Section of training data set used for creating NO_X and soot emission virtual sensors.

5.3.4 Validation Data

The engine-out emissions are stochastic in nature and are affected by environmental condition like air humidity, fuel properties, and ambient air temperature. To assess the

robustness of the emission models for different ambient conditions, the model is validated against multiple sets of data. The validation data is recorded at different days and different operating conditions to capture wide range of variability seen by production engine. The engine is operated with virtual series hydraulic hybrid powertrain over different driving schedules and with different power management strategies in the EIL setup. The engine operation along with engine-out emissions is recorded and used for validation. Figure 5.12 shows one such validation data set recorded over section of FUDS. The emission models are not "aware" of these data sets during training.

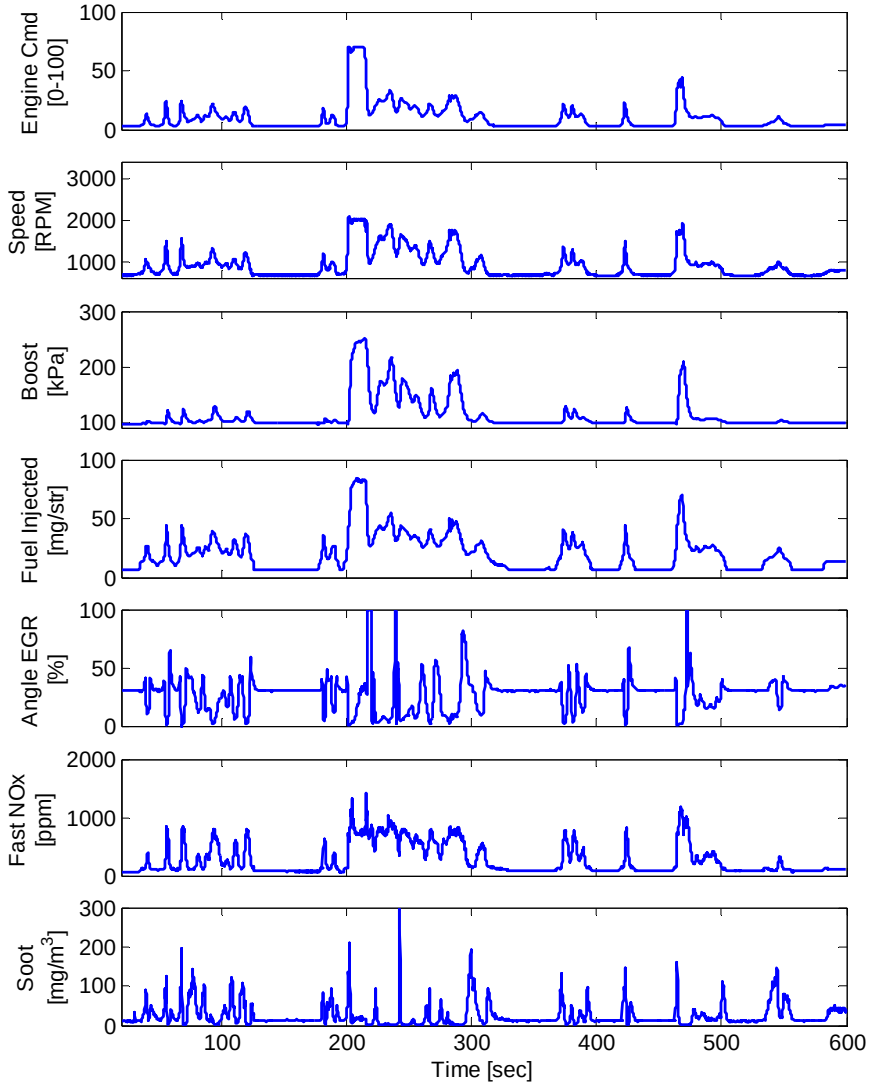


Figure 5.12: Section of one particular validation data set used for validating the NO_X and soot emission virtual sensors.

5.4 Neuro-Fuzzy Model Tree

The neuro-fuzzy models belong to hierarchical class of models and are also known as Takagi-Sugeno fuzzy models. The approach to modeling is based on divide-and-conquer strategy i.e. to divide a complex problem into multiple simpler subproblems, which then can be identified using simpler class of models like linear, polynomial or neural networks. Each local model is then identified independently. The challenge lies in the devising the correct division strategy for the original complex problem, selection of local model structure and training strategy for local models.

The concept of multiple models to represent physical system has been independently developed in many fields like artificial intelligence and statistics with different names like operating regime based models [126], [127], [128] and piecewise local models [129]. Different local modeling approaches can be categorized based on the manner in which they combine different models. The nested structure of the model combined with automatic selection of relevant regressor makes the model developed in this work different from previous approaches. Neuro-fuzzy models developed in this work have soft partitioning strategies but other strategies involving hard switching between local models based on finite state automata or probabilistic approach do exist.

Neural-fuzzy models can be considered as an extension of radial basis function networks (RBF neural networks). The output weights of RBF are replaced with a local function of the inputs. Hence, the validity functions are weighted with their corresponding local models [130].

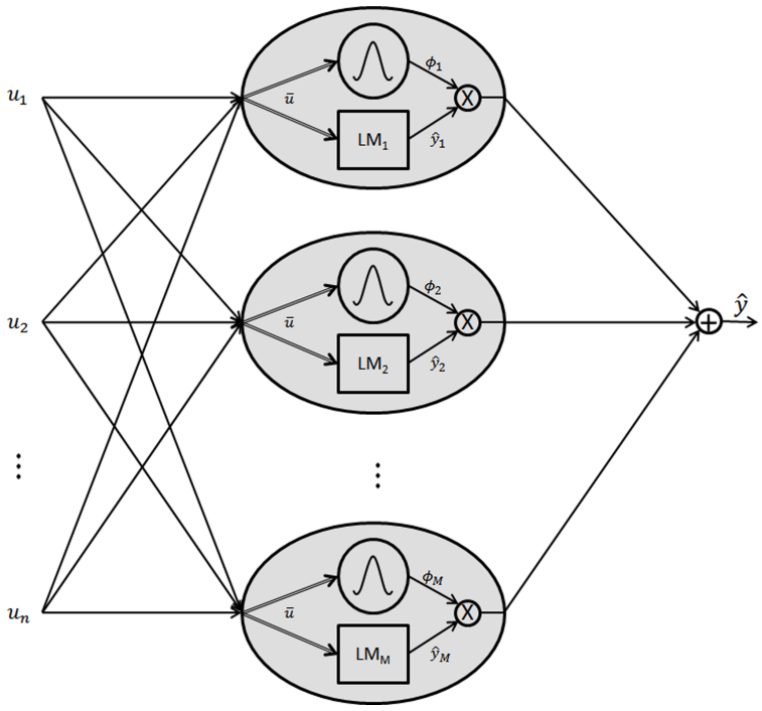


Figure 5.13: Network structure of a static neuro-fuzzy model with *M* local models and *n* inputs [130].

Figure 5.13 shows a representation of local neuro-fuzzy model with *M* local models and *n* inputs. Each submodel has its own local model with associated validity function, which determine the regions of input space where that particular model is active. Each local model is allowed to have its own unique structure different from other local models. The output of local neuro-fuzzy model is given by

$$\hat{y} = \sum_{i=1}^{M} f\left(\overline{w}, \overline{u}\right) \cdot \Phi_i(\overline{u}) \tag{5.18}$$

i.e. the output of model is the weighted sum of all local sub models. In above equation, *M* is the number of local models and □ is the validity function. Output of k^{th} submodel is given by

$$\hat{y}_k = f\left(\overline{w}, \overline{u}\right) \tag{5.19}$$

where $\overline{w}$ and $\overline{u}$ represents weight and input vector for k^{th} local model. The validity functions are normally normalized Gaussian functions

$$\Phi_i(\bar{u}) = \frac{\mu_i}{\sum_{j=1}^{M} \mu_j(\bar{u})} \tag{5.20}$$

where,

$$\mu_j(\bar{u}) = \exp\left(-\frac{1}{2} \cdot \left(\frac{(u_1 - c_{i1})^2}{\sigma_{i1}^2} \right) + \ldots + \left(\frac{(u_n - c_{in})^2}{\sigma_{in}^2} \right) \right) \tag{5.21}$$

with center coordinates c_{ij} and dimension individual standard deviation σ_{ij} [130]. The validity functions are normalized and add up to one. This is a required property to ensure the contribution of all the local models add up to 100%.

$$\sum_{i=1}^{M} \Phi_i(\bar{u}) = 1 \tag{5.22}$$

The neuro-fuzzy models can be used to model nonlinear transient processes by extending it with the concept of external dynamics model. External dynamics model consists of two parts [130]: a nonlinear static approximator and an external dynamic filter. Figure 5.14 shows a generic example of external dynamics model [130]. In principle any model architecture can be chosen for approximator *f(.)*. The approximator should be able to cope with relatively high dimensional of mapping. The dynamic filter is generally chosen as the simple time delay, q^{-1}. Neuro-fuzzy models serve as a nonlinear static approximate, while a time delay feedback of output behave as a dynamic filter, Figure 5.15. In other words, recurrent neuro-fuzzy model tree architecture is adopted to predict transient systems. The neuro-fuzzy model tree in Figure 5.15 has *M* local models, each with an associated validity function that determines the region of validity.

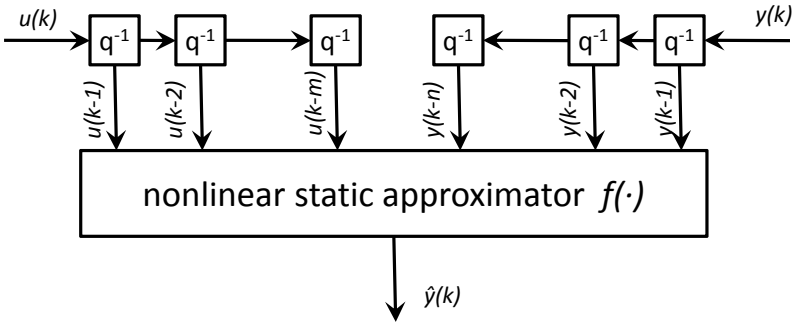


Figure 5.14: External dynamics approach: the model is separated into static approximator and an external filter bank.

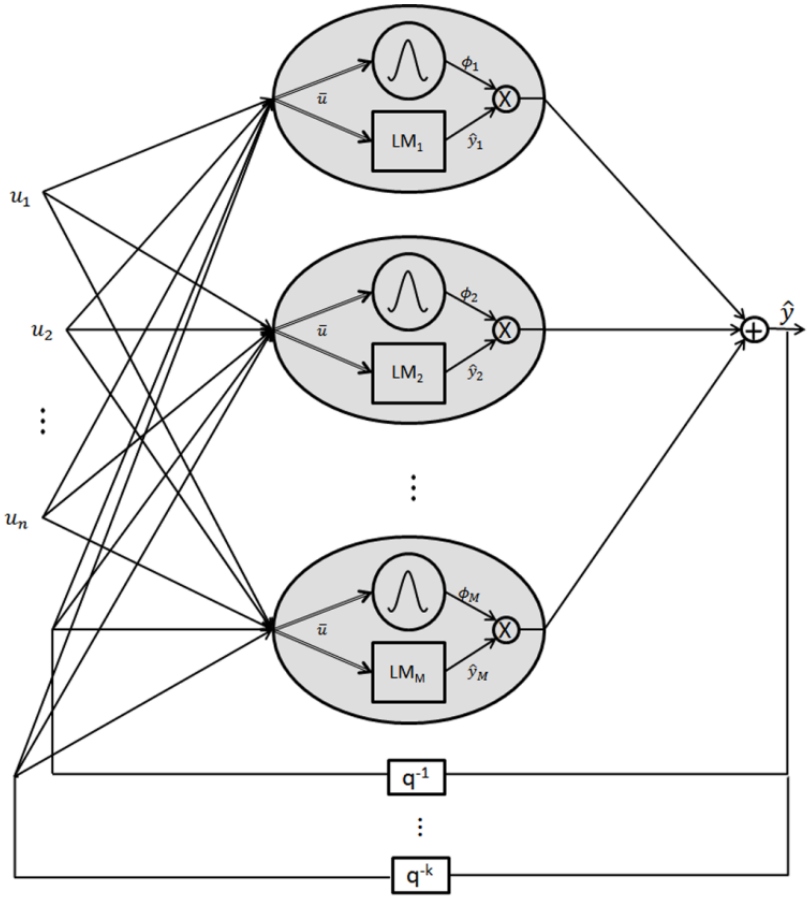


Figure 5.15: Network structure of a dynamic neuro-fuzzy model with *M* local models and *n* inputs with *k* tapped-delay output feedback.

5.4.1 Neuro-Fuzzy Training Algorithm

The algorithm for training local linear neuro-fuzzy model is based on algorithm proposed by Nelles [130] for Local Linear Model Tree (LOLIMOT). The algorithm consists of two loops. The outer loop optimizes the rule premise structure i.e. the partitioning of input space, which is defined by center and standard deviation of the validity function. The inner loop then finds the optimal structure for local model and estimates weight of the local model parameters.

The algorithm partitions the input space into hyperractangles and trains a local model for every hyperranctagle space. The center of each hyperractangle holds the validity function with standard deviation based on the boundary of this space. The size of hyperrectangle can vary thus allowing certain validity functions to have large standard

deviation i.e. large area of influence and some with very small standard deviation i.e. extremely local influence area. The local model tree grows with iteration where the worst performing local model space i.e. model with maximum local error, equation (5.23) is divided further into two smaller regions with their corresponding local models. The new cut dimension for hyperrectangle is chosen by evaluating cuts in all the possible directions and selecting the one that gives the maximum global improvement.

$$J_i = \sum_{j=1}^{N} \Phi_i\left(\bar{u}(j)\right)\left(y(j) - \hat{y}(j)\right)^2 \tag{5.23}$$

i.e. the local error, J_i is the sum squared error for all data, N weighted by the validity function.

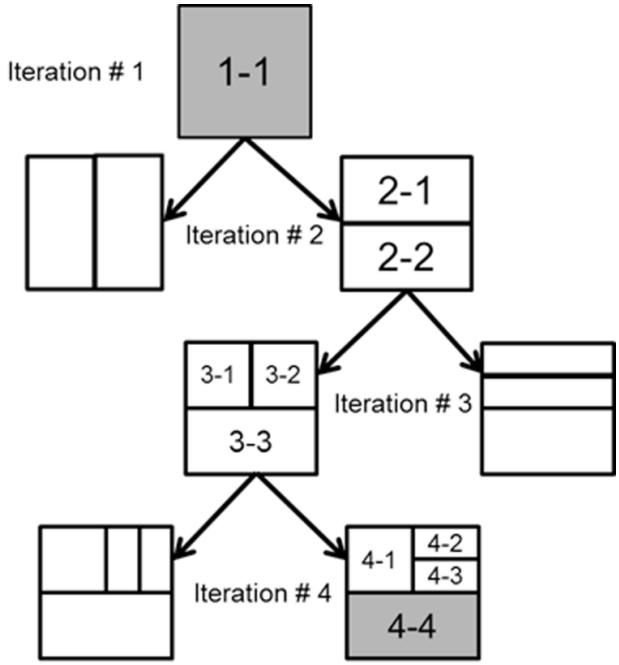


Figure 5.16: Operation of LOLIMOT structure search algorithm for a two-dimensional input space [130].

Figure 5.16 illustrates the algorithm in action. The inner loop estimates the optimal weights for the local model i.e. parameter optimization for rule consequents. The estimation can be carried out either globally or locally. Global estimation requires calculating weights for all the local models together and hence takes into consideration influence of overlapping validity function. It is more computationally intensive as the weights are refreshed for every model with addition of newer local model. In contrast,

local estimation optimizes the rule consequent parameters separately, neglecting the interaction of other local models. The local model estimation approaches global estimation with validity function standard deviation, $\sigma \rightarrow 0$, i.e. no interaction between local models. With σ increasing i.e. interaction between local models grow, the model error increases. Different algorithms can be employed to calculate the optimal weights of the local model. Choice of training algorithm is driven by the structure of local models like weighted least square for local polynomial models and backpropagation algorithm for optimizing the network weights in a multilayer perceptron network.

5.4.2 Local Models

The neuro-fuzzy framework allows for integration of different type of local models. The models can range from basic linear models to complex high order models. The choice of model depends on a priori system knowledge. However, in general, complex models do not have any advantage over simpler models as they are against the philosophy of the neuro-fuzzy modeling of dividing the complex problem into multiple simpler problems. In addition, neuro-fuzzy model training is computationally intensive and training algorithms with lower computational cost are preferred. In this dissertation, two different types of local models are used namely linear regression models and neural networks.

5.4.2.1 Linear regression models

A linear regression model can be represented mathematically as

$$y = c_0 + c_1\phi_1(x_1) + \cdots + c_n\phi_n(x_n) + \varepsilon \tag{5.24}$$

where y is the observed output , $u = \{x_1, \ldots, x_n\}$ are the input regressors, ε is independent Gaussian noise, $\{c_0, \ldots, c_n\}$ is the regression coefficients and $\varphi_1, \ldots, \varphi_n$ may be the nonlinear functions.

The model is "linear" in relation to how the regression coefficients $\{c_0, \ldots, c_n\}$ appear in the regression model and not with inputs. For a given training sets of inputs and outputs in the form $\{u_1,y_1\}$, $\{u_2,y_2\}$ ... $\{u_n,y_n\}$ where u is the inputs and y is the target , the objective is to minimize the sum of squared errors

$$E = \min \sum_{i=1}^{n} \left(y_i - \hat{y}_i \right)^2 = \min \left\| Y - Xc \right\|^2 \tag{5.25}$$

where $\hat{y}_i$ is the linear regression model response, Y is the vector of sampled outputs, X is the matrix of input regressors and c is the vector of coefficients to be determined.

The numerical solution to the above least square minimization problem is given by

$$c = (X^T X)^{-1} X^T Y \tag{5.26}$$

The above equation requires numerically calculating inverse, which is computationally intensive. In this dissertation, built-in Matlab routines are used to calculate coefficient matrix efficiently.

5.4.2.2 Neural networks

A neural network is a universal function approximator and has been shown capable of fitting any nonlinear function $\hat{y} = f(u)$ to an arbitrary degree of precision [112]. A recurrent neural network can be represented mathematically as

$$f(u) = \sum_{i=0}^{M} w_i \sigma \left(\sum_{j=0}^{p} w_{ij} u_j + \theta_j \right) \tag{5.27}$$

where $\sigma(x)$ is any continuous and monotonically increasing function with $\sigma(x) \rightarrow 1$ as $x \rightarrow \infty$ and $\sigma(x) \rightarrow 0$ as $x \rightarrow -\infty$. A suitable activation function is *log-sigmoid* or *tan-sigmoid*. Equation (5.27) is the describing equation for a feedforward artificial neural network (ANN) with p inputs, a hidden layer comprised of M nodes with sigmoidal transfer functions, and a linear output node (implementing a summation of its inputs) for each k. Nodes in successive layers are exhaustively interconnected, and there are no intralayer links. The θ are nodal biases, treated as adjustable parameters like the weights.

Multilayer perceptron network is trained using supervised training with training sets of inputs and outputs in the form $\{u_1,y_1\}$, $\{u_2,y_2\}$ ... $\{u_n,y_n\}$ where u is the inputs and y is the target and

$$y_i = f(u_i) + \varepsilon_i \tag{5.28}$$

where ε is independent Gaussian noise.

The objective is generally to minimize the sum of squared errors

$$E_D = \sum_{i=1}^{n} \left(y_i - \hat{y}_i \right)^2 \tag{5.29}$$

where $\hat{y}_i$ is the neural network response.

The goals of the neural network are to produce small error for the training set, avoid overfitting and to respond properly to new unseen inputs, i.e. generalize well. A common approach is to add a regularizing term based on the weights of the neural network. The new objective function could then be written as

$$F(w) = \beta E_D + \alpha E_W \tag{5.30}$$

where α and β are hyperparamters and

$$E_W = \frac{1}{2} \sum_i w_i^2 \tag{5.31}$$

with $\boldsymbol{w}$ is the weight vector.

The main problem with implementing regularization is setting correct values for hyperparameters α and β. MacKay [131], [132] applied Baye's formulation to neural network training and network regularization. This provides a systematic way of finding optimal hyperparameters. In a Bayesian neural network learning framework, the weights of the network are considered random variables and are characterized by joint probability distribution. Using Baye's rule

$$p\left(w \mid D, \alpha, \beta, H \right) = \frac{p\left(D \mid w, \beta, H \right) \cdot p\left(w \mid \alpha, H \right)}{p\left(D \mid \alpha, \beta, H \right)} \tag{5.32}$$

$$posterior \propto likelihood * prior$$

where D is the observed data, $\boldsymbol{w}$ is the weights of neural network model H.

Assuming a Gaussian prior, it can be seen that maximizing only likelihood estimate, $p(D|\boldsymbol{w},\beta,H)$ is equivalent to minimizing least square error problem, E_D and minimizing $F(w)$ is equivalent of maximizing posterior probability, $p(\boldsymbol{w}|D,\alpha,\beta,H)$. Baye's rule can be applied to optimize the hyperparameters α and β in the objective function.

5.4.3 Regressor and Structure Selection

An important aspect of black box model training is the selection of input regressors. The regressors should capture the process characteristics and nonlinearities. Considering

irrelevant input regressors can hurt the model performance, as any conventional training algorithm will not set the coefficients of these irrelevant inputs to zero. However, selecting the best subset of inputs from a given set of *n* inputs for a given model is a non-trivial problem. The problem is equivalent of evaluating all combinatorial possibilities i.e. 2^n-1 input subsets, [133]. With large *n*, this quickly becomes infeasible. This dissertation presents two approaches for selecting the most relevant input regressors, namely Orthogonal Least Squares and Automatic Relevance Determination.

5.4.3.1 Orthogonal least squares

Orthogonal Least Squares (OLS) is a linear subset selection technique. Chen et al. [134] gave a detailed overview of algorithm and its application for nonlinear system identification. The OLS method involves the transformation of the set of input regressors x_i into set of orthogonal basis vectors and then calculating the individual contribution to the desired output variance from each basis vector.

$$\bar{y} = \bar{X}\bar{\theta} + e \tag{5.33}$$

where $\bar{X}$ is the regression matrix, $\bar{\theta}$is the parameter vector and e is the error. The regression matrix is decomposed into $\bar{X} = \bar{V}\bar{W}$ with $\bar{W}$ being the triangular matrix and $\bar{V}$ being the matrix with orthogonal columns. The space spanned by set of orthogonal basis vectors v_i is the same as space spanned by the set of x_i. The model then can be rewritten as

$$\bar{y} = \bar{V}g + e \tag{5.34}$$

with $g = \bar{W}\bar{\theta}$ which can be derived by any orthogonalization method like Gram-Schmidt or Householder transformations. The output variance is then be given by

$$\bar{y}^T\bar{y} = \sum_{i=1}^{n} g_i^2 v_i^T v_i + e^T e \tag{5.35}$$

The output variance due to regressor v_i is given by term $g_i^2 v_i^T v_i$ in the equation (5.35). A regressor is important if this term is large or in other words, the error reduction ratio provides a criterion for subset selection.

$$[err]_i = \frac{g_i^2 v_i^T v_i}{\overline{y}^T \overline{y}} \tag{5.36}$$

Utilizing the conventional Gram-Schmidt orthogonalization, the OLS subset selection procedure can be summarized as follows.

In the first step, for $i = 1, \ldots, n$ compute

$$v_1^i = x_i \tag{5.37}$$

$$\theta_1^i = \frac{(v_1^i)^T y}{(v_1^i)^T v_1^i} \tag{5.38}$$

$$err_1^i = \frac{(\theta_1^i)^2 (v_1^i)^T v_1^i}{y^T y} \tag{5.39}$$

The regressor associated with largest error reduction ratio is selected

$$err_1^{i_1} = \max_i \left(err_1^i \right) \qquad i = 1, \ldots, n \tag{5.40}$$

$$v_1 = v_1^{i_1} = x_{i_1} \tag{5.41}$$

Define projection operator

$$proj_u(v) = \frac{\langle v, u \rangle}{\langle u, u \rangle} u \tag{5.42}$$

At the k^{th} step ($k = 2, \ldots, n_r$), for $i = 1, \ldots, n$ and $i \neq i_1, \ldots, i_{k-1}$ compute

$$v_k^i = x_i - \sum_{j=1}^{k-1} proj_{v_j}(x_i) \tag{5.43}$$

$$\theta_k^i = \frac{(v_k^i)^T y}{(v_k^i)^T v_k^i} \tag{5.44}$$

$$err_k^i = \frac{(\theta_k^i)^2 (v_k^i)^T v_k^i}{y^T y} \tag{5.45}$$

The regressor associated with largest error reduction ratio is selected

$$err_k^{i_k} = \max_i \left(err_k^i \right) \qquad i = 1, \ldots, n \text{ and } i \neq i_1, \ldots, i_{k-1} \tag{5.46}$$

$$v_k = v_k^{i_k} = x_{i_k} - \sum_{j=1}^{k-1} proj_{v_j}(x_{i_k}) \tag{5.47}$$

The algorithm proceeds with transformation of original regressors into orthogonalized basis vectors. Then, the regressor with maximum error reduction ratio is selected. The remaining regressors are transformed into new vectors by subtracting the

linear combination of all previously selected regressors and re-orthogonalization. The idea is for every new piece of data, the actual information content or innovation is calculated, where *innovation = new data – prediction of the new data based on the existing data.* The innovation is 0 if the new incoming data can be regenerated based on present information. The innovations are mutually uncorrelated (orthogonal in case of vectors). The whole algorithm is repeated until desired number of regressors, n_r have been obtained or the unexplained error fall below a given threshold.

The standard OLS algorithm computational demand grows with increase in potential regressors and size of data set. Chen et al. [135] noted that the standard OLS algorithm spends a lot of computation in calculating vector inner products and substantial saving can be achieved by updating the scalar inner product. The OLS algorithm implemented for input structure selection in this dissertation is derived from Chen et al. [135] to greatly reduce computational effort.

The OLS routines for local subset selection are modified in this dissertation, to include the weighting of data in local neuro-fuzzy models.

$$\tilde{\bar{X}} = \sqrt{Q}\bar{X} \tag{5.48}$$

$$\tilde{\bar{y}} = \sqrt{Q}\bar{y} \tag{5.49}$$

where Q is a diagonal weighting matrix. The weighting matrix is given by equation (5.68).

5.4.3.2 Automatic relevance determination

Automatic Relevance Determination (ARD) provides a Bayesian framework for ranking the inputs and thereby provides a systematic procedure for the selection of relevant inputs from all the input variables. The ARD model introduces a weight decay hyperparameter for each input [136], [133]. The idea is that the inputs then can eventually be ranked based on their hyperparameters. Inputs with higher priority will be used for training model and all irrelevant inputs will be ignored.

For a multilayer perceptron network, separate hyperparameters α_k are defined for each input. Along with input hyperparameters, two additional hyperparamaters are considered, one for hidden and output layer weights and another one for bias in all the

layers (input, hidden and output), Figure 5.17. All weights of subclass are assumed to be distributed with Gaussian prior with 0 mean and 1/ α_k variance. The evidence framework is then applied to optimize all the hyperparameters, α_k. The new training objective with input relevance is then given by

$$F(w) = \beta E_D + \sum_k \alpha_k E_{W(k)} \tag{5.50}$$

where $E_{W(k)} = \frac{1}{2}\sum_i^{m_k} w_i^2$ and m_k is the number of weights in weight class k.

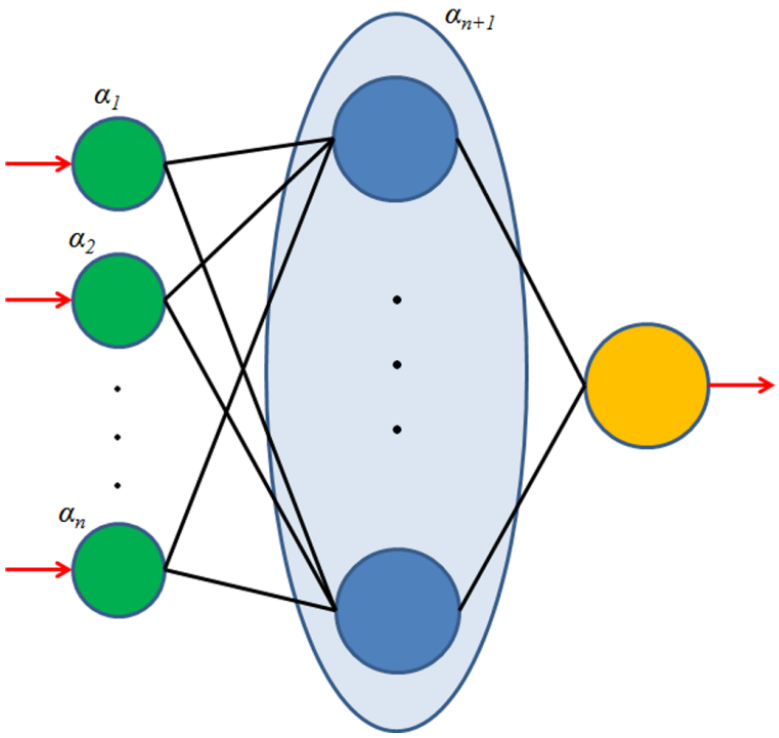


Figure 5.17: Overview of hyperparameters introduced by automatic relevance determination technique for input regressor selection.

Let $p(\{w_i\}|\{\alpha_k\},H)$ be the *prior* probability distribution of a given neural network model, H. This refers to the information about the weight vector, $\boldsymbol{w}$, prior to any observation made on the network and typically, is a flat distribution i.e. all the weight values are priori equiprobable. When the data D is observed, the prior distribution of weight vector, $\boldsymbol{w}$ is adjusted to a posterior distribution according to Baye's rule

$$p\left(\{w_i\} \mid D, \{\alpha_k\}, \beta, H\right) = \frac{p\left(D \mid \{w_i\}, \beta, H\right) \cdot p\left(\{w_i\} \mid \{\alpha_k\}, H\right)}{p\left(D \mid \{\alpha_k\}, \beta, H\right)} \tag{5.51}$$

where $p(D|\{w_i\},\beta,H)$ is the *likelihood* function, which is the probability of data occurring given network, H and weight vector, $\boldsymbol{w}$. $p(D|\{\alpha_k\},\beta,H)$ is called *evidence* and plays a role of normalizing constant. Hence, the posterior density of the weight vector, $\boldsymbol{w}$ parameterizing the network, H is proportional to the product of likelihood function and prior, i.e.

$$posterior \propto likelihood * prior \tag{5.52}$$

Assuming prior distribution of weights to be Gaussian with 0 mean and $1/\alpha_k$ variance [132]

$$\begin{aligned} p\left(\{w_i\} \mid \{\alpha_k\}, H\right) &= \prod_k \left[\prod_i^{m_k} \left(\frac{1}{\sqrt{2\pi / \alpha_k}} \exp\left(-\frac{\alpha_k}{2} w_i^2 \right) \right) \right] \\ &= \prod_k \left[\frac{1}{\left(\sqrt{2\pi / \alpha_k} \right)^{m_k}} \prod_i^{m_k} \left(\exp\left(-\frac{\alpha_k}{2} w_i^2 \right) \right) \right] \\ &= \frac{1}{\prod_k \left(\sqrt{2\pi / \alpha_k} \right)^{m_k}} \exp\left(-\sum_k \left[\frac{\alpha_k}{2} \sum_i^{m_k} w_i^2 \right] \right) \\ &= \frac{1}{\prod_k Z_W(\alpha_k)} \exp\left(-\sum_k \alpha_k E_{W(k)} \right) \end{aligned} \tag{5.53}$$

and noise in the training data set also to be Gaussian with 0 mean and $1/\beta$ variance

$$\begin{aligned} p\left(D \mid \{w_i\}, \beta, H\right) &= \prod_i^n \left(\frac{1}{\sqrt{2\pi / \beta}} \exp\left(-\frac{\beta}{2} (y_i - \hat{y}_i)^2 \right) \right) \\ &= \frac{1}{\left(\sqrt{2\pi / \beta} \right)^n} \prod_i^n \left(\exp\left(-\frac{\beta}{2} (y_i - \hat{y}_i)^2 \right) \right) \\ &= \frac{1}{\left(\sqrt{2\pi / \beta} \right)^n} \exp\left(-\frac{\beta}{2} \sum_i^n (y_i - \hat{y}_i)^2 \right) \\ &= \frac{1}{Z_D(\beta)} \exp\left(-\beta E_D \right) \end{aligned} \tag{5.54}$$

By substituting these probabilities in equation (5.51)

$$p\left(\{w_i\} \mid D, \{\alpha_k\}, \beta, H\right) = \frac{1}{Z_F(\{\alpha_k\}, \beta)} \exp\left(-\beta E_D - \sum_k \alpha_k E_{W(k)}\right) = \frac{1}{Z_F(\{\alpha_k\}, \beta)} \exp\left(-F(w)\right) \tag{5.55}$$

It can be seen that maximizing only likelihood estimate, $p(D|\{w_i\},\beta,H)$ is equivalent to minimizing least square error problem, E_D and minimizing $F(\mathbf{w})$ is equivalent of maximizing posterior probability, $p(\{w_i\}|D,\{\alpha_k\},\beta,H)$. However, solving for *maximum a posteriori estimate* (MAP) requires knowledge of hyperparameters $\{\alpha_k\}$ and β. Baye's rule is applied to optimize the hyperparameters $\{\alpha_k\}$ and β in the objective function.

$$p\left(\{\alpha_k\}, \beta \mid D, H\right) = \frac{p\left(D \mid \{\alpha_k\}, \beta, H\right) p\left(\{\alpha_k\}, \beta \mid H\right)}{p\left(D \mid H\right)} \tag{5.56}$$

Assuming uniform prior density $p(\{\alpha_k\},\beta|H)$ for the hyperparameters $\{\alpha_k\}$ and β, the posterior is maximized by maximizing the likelihood function, $p(D|\ \{\alpha_k\},\beta,H)$. Note the likelihood function is same as the normalizing constant in equation (5.51). Therefore solving for $p(D|\ \{\alpha_k\},\beta,H)$

$$\begin{aligned} p\left(D \mid \{\alpha_k\}, \beta, H\right) &= \frac{p\left(D \mid \{w_i\}, \beta, H\right) p\left(\{w_i\} \mid \{\alpha_k\}, H\right)}{p\left(\{w_i\} \mid D, \{\alpha_k\}, \beta, H\right)} \\ &= \frac{1}{\frac{1}{Z_F(\{\alpha_k\}, \beta)} \exp\left(-F(w)\right)} \times \frac{1}{\prod_k Z_W(\alpha_k)} \exp\left(-\sum_k \alpha_k E_{W(k)}\right) \\ &\quad \times \frac{1}{Z_D(\beta)} \exp\left(-\beta E_W\right) \\ &= \frac{Z_F(\{\alpha_k\}, \beta)}{\prod_k Z_W(\alpha_k) \cdot Z_D(\beta)} \times \frac{\exp\left(-\sum_k \alpha_k E_{W(k)}\right) \exp\left(-\beta E_W\right)}{\exp\left(-F(w)\right)} \\ &= \frac{Z_F(\{\alpha_k\}, \beta)}{\prod_k Z_W(\alpha_k) \cdot Z_D(\beta)} \end{aligned} \tag{5.57}$$

The denominator is known from previous equations and the only unknown is $Z_F(\{\alpha_k\},\beta)$. The objective function $F(w)$ has quadratic shape in a small area surrounding minimum point. Applying Taylor series expansion around the minimum point of

posterior density $\boldsymbol{w}_{MP}$ and equating the gradient of posterior density function, $p(\{w_i\}|D,\{\alpha_k\},\beta,H)$ at $\boldsymbol{w}_{MP}$ to zero [137], Z_F is given by

$$Z_F = (2\pi)^{N/2} \left(\det\left(\left(H_k^{MP} \right)^{-1} \right) \right)^{1/2} \exp\left(-F(w_{MP}) \right) \tag{5.58}$$

where $H_k^{MP} = \beta \nabla^2 E_D + \sum_k \alpha_k \nabla^2 E_{W(k)}$ is the hessian matrix of objective function, $F(\boldsymbol{w})$.

The optimal values of hyperparameters $\{\alpha_k\}$ and β is calculated by taking derivative of *log* of $p(D|\{\alpha_k\},\beta,H)$ w.r.t. $\{\alpha_k\}$ and β and setting it to zero [137].

$$\alpha_k^{MP} = \frac{\gamma_k}{2E_{W(k)}} \quad and, \quad \beta^{MP} = \frac{n - \sum_k \gamma_k}{2E_D(w_{MP})} \tag{5.59}$$

where γ_k is the number of well determined parameters for weight class k with $\gamma_k = N_k - 2\alpha_k^{MP} tr(H_k^{MP})^{-1}$. N_k is the number of weights in weight class k and trace is taken over only those parameters.

The most relevant inputs have small $\{\alpha_k\}$, since $\{\alpha_k\}$ is inversely proportional to variance of corresponding Gaussian prior. Hence, large number of potential inputs can be included while training and only the most relevant will be considered automatically without the degrading effects [133].

5.5 Neuro-Fuzzy Tree based Emission Models

Figure 5.18 shows the universal model with one model spanning the whole input space. As mentioned before, the drawback with this approach is that one unified model has to capture all the process nonlinearities. This leads to challenges in picking the appropriate model structure and training algorithm. Figure 5.19 summarizes the neuro-fuzzy model with regressor selection strategy. The neuro-fuzzy model tree algorithm divides the input space based on rule premise, $\boldsymbol{z}$. The algorithm is able to extract variables in rule premise vector $\boldsymbol{z}$, which have a significant nonlinear influence on the process output and create partition rules. It subsequently trains the local model valid for a specific input region. The regressor selection algorithm helps in structure selection of local models by pruning the input variables. Each local model can have different set of input

variables. The premise vector, z need not be similar to consequent vector, $\boldsymbol{x}$. The validity function (premises) depends on z and local models (consequents) depend on $\boldsymbol{x}$.

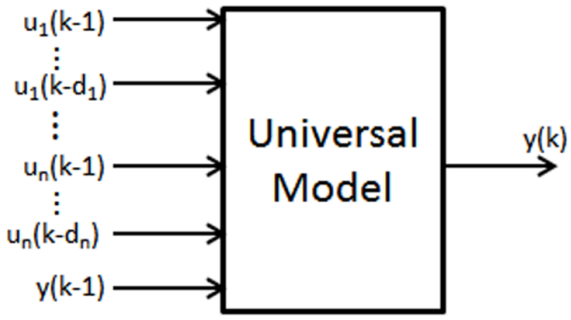


Figure 5.18: A global model spanning the whole input space.

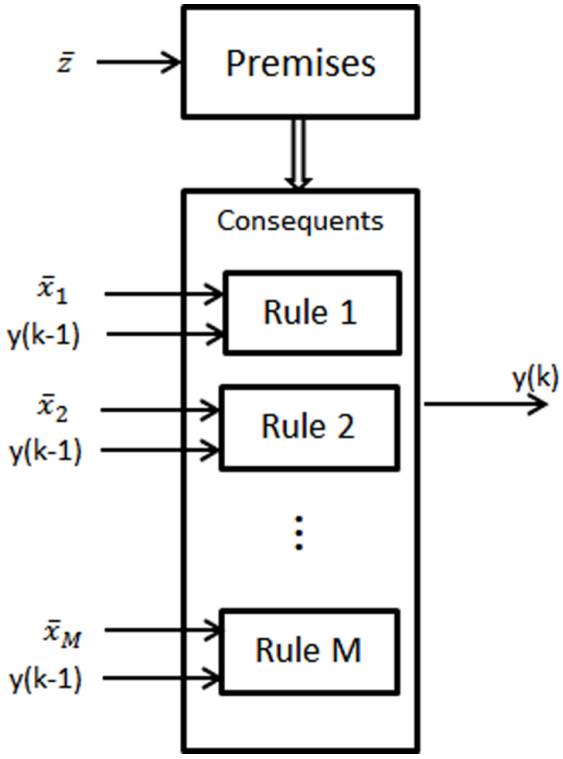


Figure 5.19: A neuro-fuzzy model with different rule premise and consequent inputs.

In this dissertation, two different types of transient emission models are developed using the concepts of neuro-fuzzy model tree and regressor selection techniques presented in previous sections.

1. Neuro-fuzzy model tree for dynamic programming framework
2. Neuro-fuzzy model tree based real-time virtual sensors

5.5.1 Neuro-Fuzzy Model Tree for Dynamic Programming Framework

The transient emission models developed in this section are designed specifically to be used with dynamic programming framework for supervisory controller design for hybrid vehicles. The transient emission model is designed with three considerations. Firstly, the transient emission models for particulate matter and NO_X are to be employed in DP framework for solving power management problem. The DP suffers from curse of dimensionality i.e. exponential increase in memory and computational cost with increase in state space spanned by problem. This effectively puts a constraint on the number of states of the problem being solved with DP. Therefore, the transient emission models needs to have a minimum number of states. Secondly, the transient emission model needs to be computationally efficient to prevent any significant increase in computational burden. Finally, the input regressors to the transient emission models are limited to the signals available during DP calculations. A simplified engine model with no actuator dynamics is used in the DP framework and only a very small set of engine signals are available as input to emission model.

A separate model for predicting transient particulate matter and transient NO_X is developed. The transient NO_X and soot emission models can be mathematically represented as

$$\begin{aligned} \widehat{PM}^{k} &= \sum_{i=1}^{M} f_{NN\_PM}\left(\bar{w}_{PM}, \bar{u}_{PM}\right) \cdot \Phi_i(\omega_e^k) \\ \bar{u}_{PM} &= \left\{\omega_e^k, \tau_e^k, P_{im}, m_f, \dot{m}_f, \widehat{PM}^{k-1}\right\} \end{aligned} \tag{5.60}$$

$$\begin{aligned} \widehat{NOx}^{k} &= \sum_{i=1}^{M} f_{NN\_NOx}\left(\bar{w}_{NOx}, \bar{u}_{NOx}\right) \cdot \Phi_i(\omega_e^k) \\ \bar{u}_{NOx} &= \left\{\omega_e^k, \tau_e^k, P_{im}, m_f, NOx_{ss}, \widehat{NOx}^{k-1}\right\} \end{aligned} \tag{5.61}$$

where k is the time index, $\bar{w}$ is the set of neuron weights, $\bar{u}$ is the set of input regressors,ω_e^k is current engine speed, τ_e^k is current engine torque, $P_{im} = f(\omega_e^k, \tau_e^k)$ is steady state inlet manifold pressure, m_f is current mass of fuel injected, $\dot{m}_f$ is the rate of change of fuel injection, $NOx_{ss} = f(\omega_e^k, \tau_e^k)$ is the steady state NO_X, $\widehat{PM}^{k-1}$ is the previous predicted particulate matter and $\widehat{NOx}^{k-1}$is the previous predicted NO_X. The

inputs are based on the signals available during DP simulation from the engine model. Different combinatorial sets of inputs are tried and the final selection is based on input set with best performance with least number of regressors.

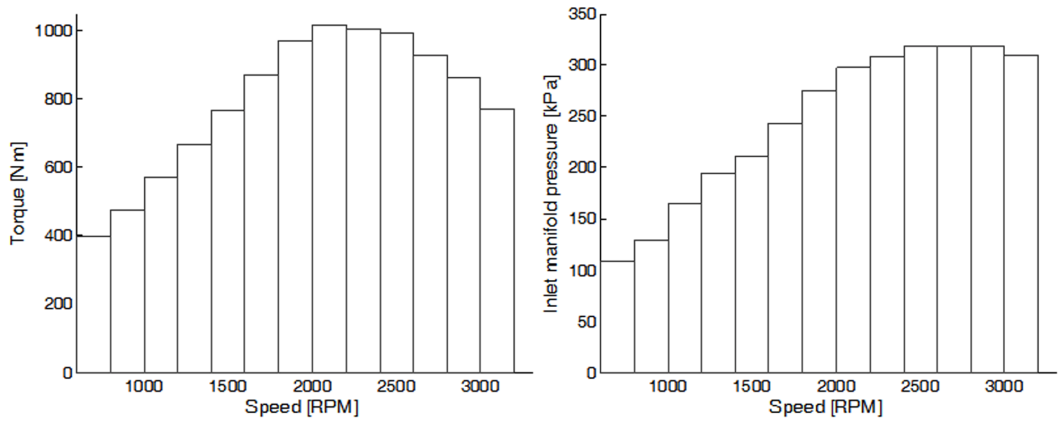


Figure 5.20: Input space partitioning in the neuro-fuzzy model for dynamic programming framework. Input space partitioning along speed dimension is shown with respect to engine torque and inlet manifold pressure.

To reduce the computational effort, the operating regimes of local model and the input space partitioning of the model is not carried out using neuro-fuzzy model tree training algorithm described in section 5.4.1. Instead, the input space is divided based on engine operating speed into 15 regions and a separate recurrent neural network is trained for each operating region. This choice of input space division is based on the experiment carried out to characterize the engine described in section 5.3. Figure 5.20 shows the partitioning along the engine speed dimension for the models. Triangular membership function, Figure 5.21 are used as validity function. This different choice of membership function compared to typical Gaussian membership function is to reduce the computational cost of calculating the membership function values. Each color in Figure 5.21 represents a separate validity function and defines the validity of different submodels based on engine operating speed.

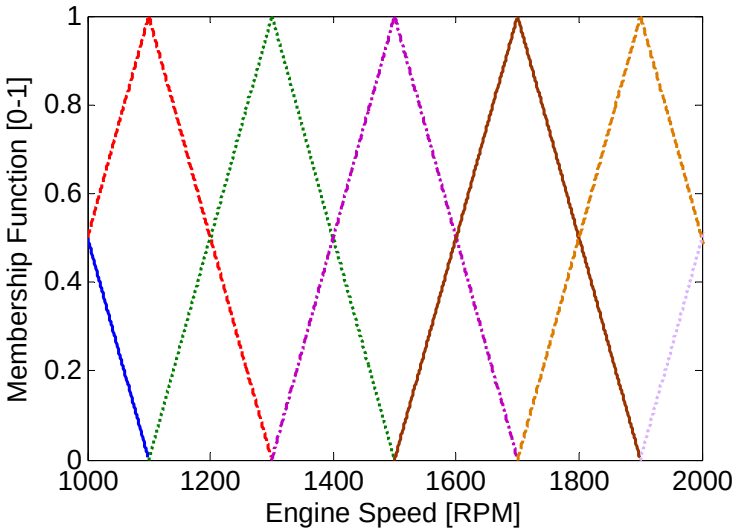


Figure 5.21: Triangular membership function in the neuro-fuzzy model tree for dynamic programming framework.

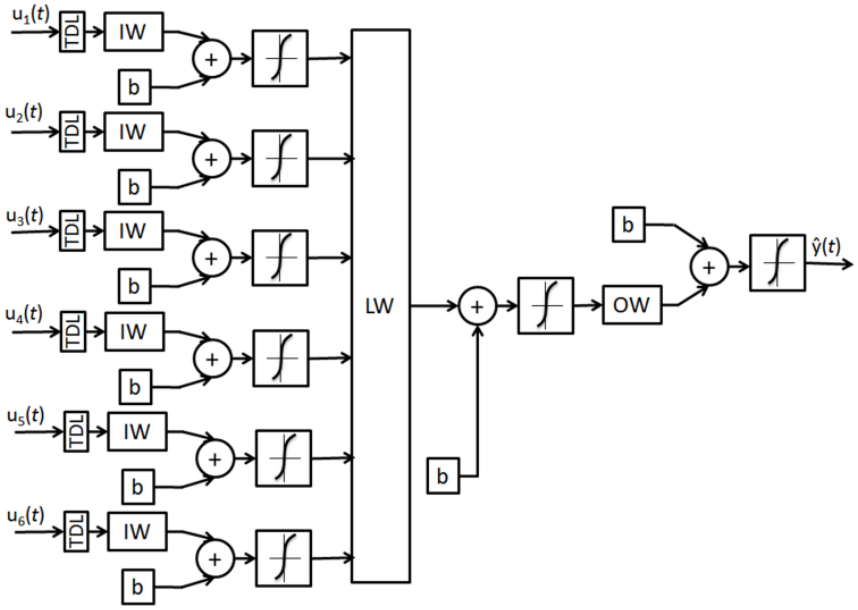


Figure 5.22: Local neural network model in the neuro-fuzzy model tree for dynamic programming framework.

Figure 5.22 shows the overview representation of a layer recurrent neural network used in this model. Each local neural network has 6 inputs i.e. all the inputs to neuro-fuzzy model are available to all the local neural networks. Each input is preprocessed and normalized between -1 to 1 and is fed to separate input layer with 5 neurons and *tan-*

sigmoid activation function. The feedback from output is also considered as different input. The first set of layers then feed to a hidden layer, which has 30 neurons and *tan-sigmoid* activation function. The output layer also uses a *tan-sigmoid* activation function rather than usual linear function to prevent extrapolation and outputs going out of bounds.

Bayesian regularization backpropogation is used for training the neural network [131], [138], [130], [139].

$$\begin{aligned} H &= J^T J \\ g &= J^T e \\ x_{k+1} &= x_k - \left[H + \mu I\right]^{-1} g \end{aligned} \tag{5.62}$$

where H is the approximate hessian and J is the Jacobian matrix of first derivatives of network error with respect to weight and bias, g is the gradient. The addition of μI to the approximation of hessian, H prevents inversion of poorly conditioned J^TJ. For small values of μ, the algorithm approaches Gauss Newton algorithm, while for large values of μ it approaches the steepest descent method. The mean squared error with regularization performance function is used as cost function.

$$msereg = \gamma \frac{1}{N}\sum_{i=1}^{N}(e_i)^2 + (1-\gamma)\frac{1}{n}\sum_{j=1}^{n} w_j^2 \tag{5.63}$$

where N is number of training data, n is the number of weights and γ is the performance ratio. The performance ratio is optimally calculated using Bayesian approach and prevents the neural network from overfitting [131], [140].

Training a neural network with output feedback requires dynamic backpropagation algorithm, which is very slow. Since the true output is available during the training of the network, a series-parallel architecture [139] is created. The true output is used during training instead of feeding back the estimated output, as shown in Figure 5.23. This has two advantages [139]. First, the input to the feedforward network is more accurate and second, the resulting network has a purely feedforward architecture and static backpropagation can be used for training, which is faster.

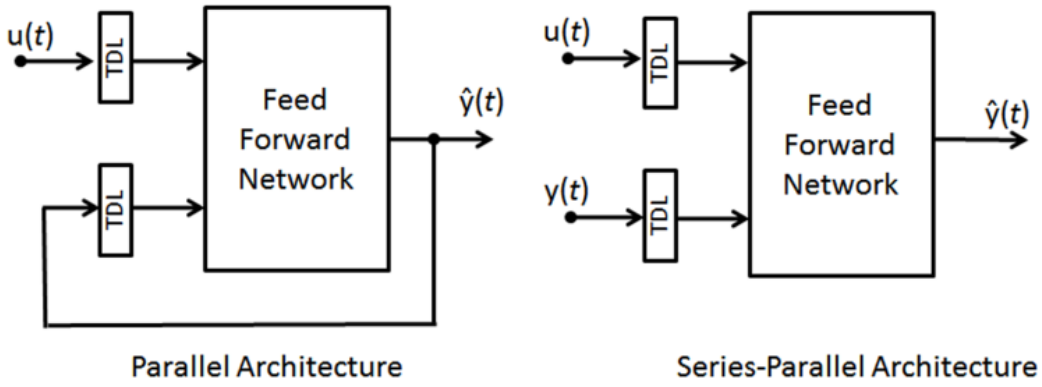


Figure 5.23: Parallel vs Series-Parallel architecture for training NARX models.

Figure 5.24 and Figure 5.25 show the model prediction along with steady state model predictions and the test cell measurement of particulate matter and NO_X respectively using fast emission analyzers. It can be seen that neuro-fuzzy models predict particulate matter transients more accurately compared to the steady state models with little overprediction. Figure 5.26 gives the relative error in predicting transient particulate matter emission using neuro-fuzzy model and steady state model for different validation data sets. The neuro-fuzzy model is within 20% of cumulative soot recorded by fast analyzers compared to 50% error in steady state model. The neuro-fuzzy model for NO_X has smaller cumulative error of about 10% compared to 20% error in steady state model prediction, Figure 5.27.

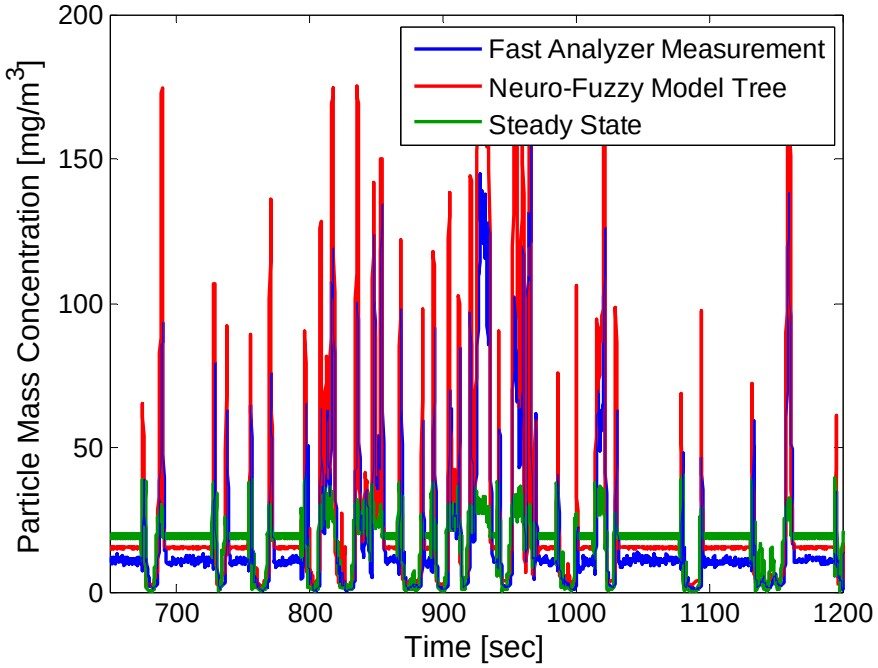


Figure 5.24: Neuro-fuzzy based model predictions along with measured data using fast analyzers and quasi-steady state model predictions for transient particulate matter for a particular validation data set.

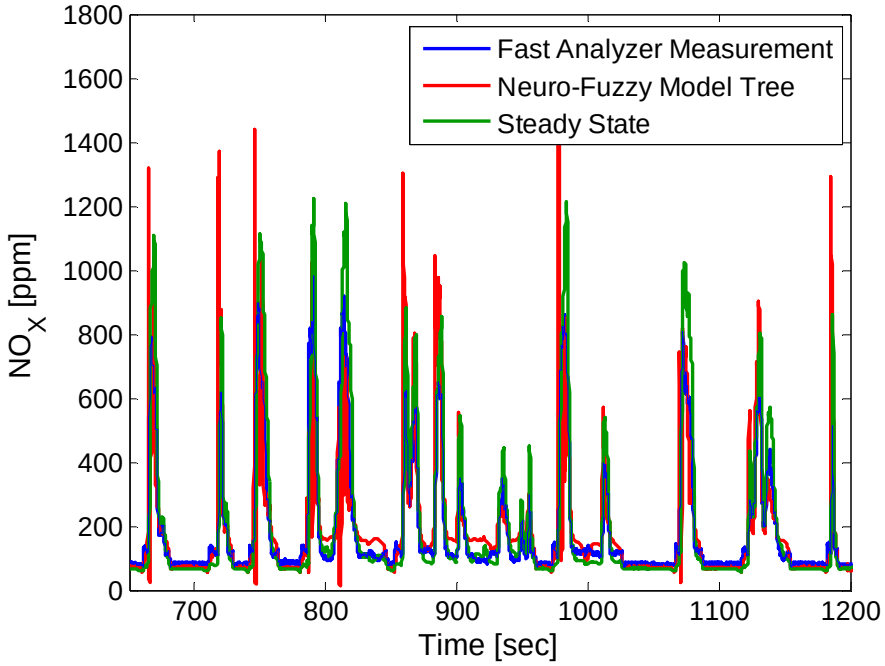


Figure 5.25: Neuro-fuzzy based model predictions along with measured data using fast analyzers and quasi-steady state model predictions for transient NO_X for a particular validation data set.

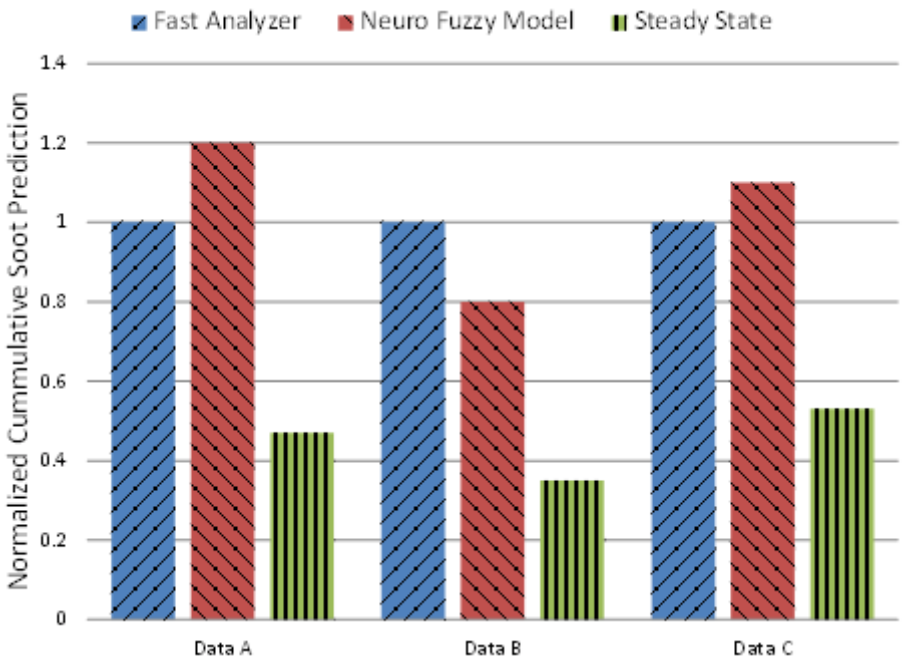


Figure 5.26: Comparison of cumulative particulate matter prediction by neuro-fuzzy tree based transient emission models for dynamic programming framework and steady state models for different validation data set.

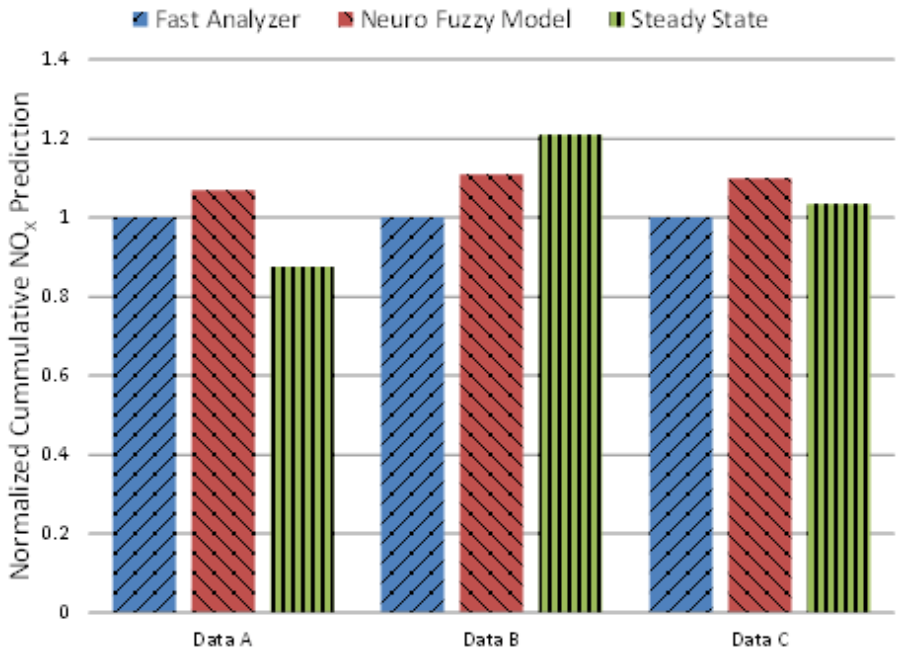


Figure 5.27: Comparison of cumulative NO_X prediction by neuro-fuzzy tree based transient emission models for dynamic programming framework and steady state models for different validation data set

5.5.2 Neuro-Fuzzy Model Tree based Real-Time Virtual Sensors

The neuro-fuzzy model for transient emission developed in previous section is designed specifically to be employed in dynamic programming framework. The model is constrained to have minimum number of input regressors and choice of regressors is limited. In this section, two different transient emission models are designed to be used as virtual sensors along with physical engine in real world operation. The neuro-fuzzy model tree algorithm and the experimental data generated above are used for creating these models. These models, unlike previous model are not limited to number of input regressors and have a larger set of input regressors available. They have access to all the signals available from sensors onboard engine and ECU. However, the models should be capable of running in real time on an embedded controller along with real engine. These two models employ a more generalized concept of neuro-fuzzy tree. Although the neuro-fuzzy model tree is a hierarchical structure, the generated models are "flat" in the sense that they are non-hierarchical. A natural extension of the neuro-fuzzy model tree is a two level hierarchical structure. The local models for the top-level neuro-fuzzy model are also a neuro-fuzzy model tree. The second tier neuro-fuzzy model trees in turn have "flat" submodels. Figure 5.28 depicts the high-level representation of such a hierarchical model.

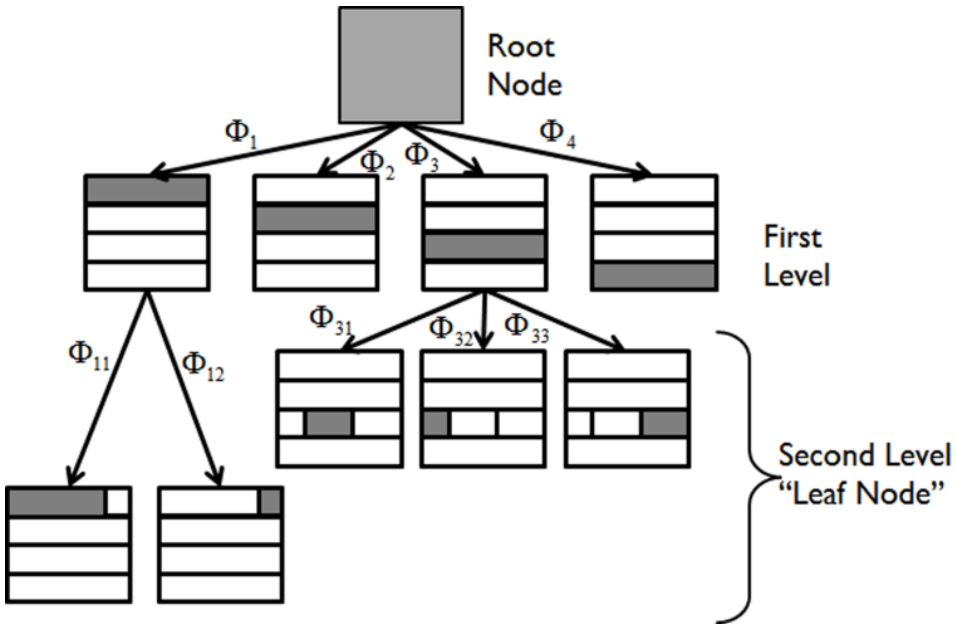


Figure 5.28: Hierarchical model structure of neuro-fuzzy model tree based virtual emission sensors.

The overall model output is calculated by summing the contributions of every local model at leaf nodes, weighted with their validity function values. The validity functions pass their contribution to the next higher node (parent).

$$\begin{aligned}\hat{y} &= \Phi_1\left(\sum_i \Phi_{1i} f_{1i}(\overline{w},\overline{u})\right) + \ldots + \Phi_m\left(\sum_i \Phi_{mi} f_{mi}(\overline{w},\overline{u})\right) \\ &= \sum_j^m \Phi_j\left(\sum_i \Phi_{ji} f_{ji}(\overline{w},\overline{u})\right)\end{aligned} \tag{5.64}$$

where, Φ_{ji} is the validity function of leaf node local models and $y_{ji} = f_{ji}(\overline{w},\overline{u})$ are the individual outputs of local submodels.

The problem of developing transient emission models based on the experimental data generated in section 5.3.3 lends itself to the above hierarchical structure. The root node denotes the whole input space. The root node is partitioned by engine speed into 15 first level models. The choice of this partition is logical based on the training signal used to generate training data shown in Figure 5.8. The Figure 5.29 show the soft partition of first level models based on Gaussian validity functions. The first level models are further partitioned into multiple local models (leaf nodes) using the neuro-fuzzy algorithm.

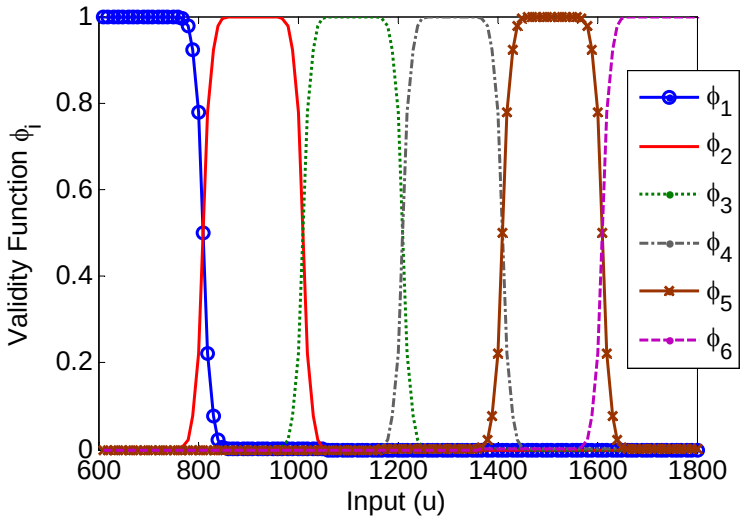


Figure 5.29: Soft model partition of first level models based on engine speed using Gaussian validity functions.

The leaf nodes are partitioned in a 4-D hyperspace. The rule premise space is spanned by mass of fuel injected, boost, EGR valve angle and post fuel injection,

whereas the rule consequent space includes engine speed, fuel injected per cylinder, boost pressure, EGR valve angle, rate of change of fuel injection, post fuel injection, their previous time histories and previous emission output. Table 5.1 and Table 5.2 give the list of included time histories for each signal. The delay for time history is calculated using cross correlation of input variable with emission output (section 5.3.2).

Table 5.1: Time delay in input signals considered for soot model

Input Signal	**Time History (sec)**
Speed (RPM)	0
Fuel Injected (mg/str)	0, 0.1, 0.2, 0.3, 0.4, 0.5
Boost (bar)	0
Angle of EGR Valve (θ)	0, 0.1, 0.2
Post Fuel Injection (mg/str)	0, 0.4, 0.5
Rate of Fuel Injection	0, 0.1, 0.2, 0.3, 0.4, 0.5

Table 5.2: Time delay in input signals considered for NO_X model

Input Signal	**Time History (sec)**
Speed (RPM)	0
Fuel Injected (mg/str)	0, 0.1, 0.2, 0.3, 0.4
Boost (bar)	0
Angle of EGR Valve (θ)	0, 0.1
Post Fuel Injection (mg/str)	0
Rate of Fuel Injection	0

Each input is preprocessed and normalized between 0 and 1 before training. The normalization has the benefit of making the learning more numerically stable. The output is then anti-normalized to recover values in original range.

A multitude of different emission models can be developed based on the above framework with different local models and different regressor selection techniques. Two such different models are developed in next section.

5.5.2.1 Neuro-fuzzy model tree with orthogonal least squares

The local submodel at leaf nodes is polynomial in structure. The local submodel can be represented as

$$\hat{y}(k) = w_0 + w_1\phi_1(u_1) + \cdots + w_n\phi_n(u_n) + w_{n+1} \cdot \hat{y}(k-1) \tag{5.65}$$

where $u = [u_1, \ldots, u_n]$ is the set of input regressors. In order to introduce nonlinearity in input space, the second order multiplication of input data set is also used. Though the formulation now includes nonlinearities, the structure of submodel is still linear.

The leaf nodes and input space partitioning is carried out using neuro-fuzzy model tree training algorithm described in section 5.4.1. Each first level model is trained separately. The training is stopped if the number of local submodels reaches 20 or the sum square error falls below 0.01. Figure 5.30 shows a representative input space partitioning at leaf nodes with respect to two different division directions for soot model.

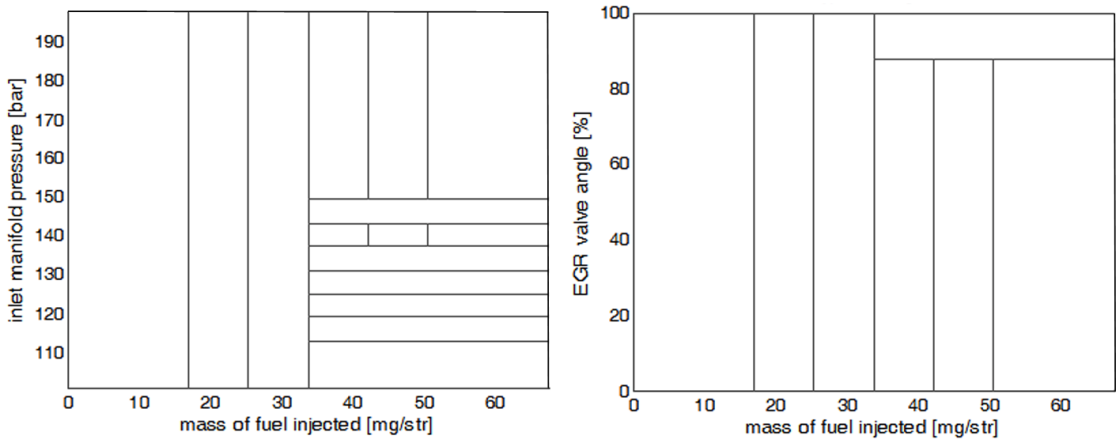


Figure 5.30: Input space partitioning for a representative first level neuro-fuzzy model for soot. Hyperplane divisions along two different input regressors are shown.

The local model is calculated to minimize the weighted least square error between the input and output samples. The solution is given by

$$\bar{w}_i = \left(X_{ji}^T Q_{ji} X_{ji} \right)^{-1} X_{ji}^T Q_{ji} Y \tag{5.66}$$

where $\bar{w}_i$ is the weight vector of local model i and this estimate has to be computed successively for all local models $i = 1, \ldots, M$. The X_{ji} is the input regression matrix and Q_{ji} is the diagonal weighting matrix given by

$$X_{ji} = \begin{bmatrix} 1 & u_1(1) & \cdots & u_{ni}(1) & y(0) \\ 1 & u_1(2) & \cdots & u_{ni}(2) & y(1) \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ 1 & u_1(N) & \cdots & u_{ni}(N) & y(N-1) \end{bmatrix} = \begin{bmatrix} \bar{u}(1) \\ \bar{u}(2) \\ \vdots \\ \bar{u}(N) \end{bmatrix} \tag{5.67}$$

$$Q_{ji} = \Phi_j \begin{bmatrix} \Phi_{ji}(\bar{u}(1)) & 0 & \cdots & 0 \\ 0 & \Phi_{ji}(\bar{u}(2)) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \Phi_{ji}(\bar{u}(N)) \end{bmatrix} \tag{5.68}$$

where N is the number of data samples, $\bar{u}(i)$ is the input regressor set with cardinality n chosen by OLS, Φ_j is the validity function of first level model j, Φ_{ji} is the validity function of leaf node local model i.

The training algorithm for neuro-fuzzy model tree is augmented with the OLS algorithm for automatic selection of input structure for leaf node local models from the available input vector. The number of regressors selected by OLS is tunable and is 20 for NO_X and 40 for soot model. The OLS had another termination criterion based on unexplained error with 10^{-2} as the limit. To keep computational cost low, both the weight estimation for models and input selection using OLS are done locally. The OLS algorithm is nested in the inner loop of neuro-fuzzy model tree algorithm and executed before optimization of local model weights. The relevant time lag for some input signals change with engine speed due to transport delay of emission specimen from engine exhaust manifold to fast emission analyzers. OLS is capable of figuring out the lag time automatically and includes only the relevant time delayed signals. This helps in keeping the number of input regressors small, making the model more efficient and robust.

The NO_X and soot emission model are trained using the data generated in section 5.3.3. The virtual sensors are then coded in C++ and implemented in Simulink as S-function. The Simulink based emission model is then cross-compiled and cross-linked for dSpace real-time platform to run in real-time with engine as virtual sensor. The virtual sensor interfaces with PUMA Open and receives engine speed, engine boost, in-cylinder injected fuel, EGR valve angle and post fuel injection at 10Hz and predicts instantaneous NO_X and soot emission. The transient emission models are then validated against set of validation data (section 5.3.4). Figure 5.31 and Figure 5.32 shows the instantaneous predicted and measured particulate matter and NO_X emissions respectively along with steady state model predictions for one validation data set. The predicted emissions show a good match with soot and NO_X measurements from fast analyzers. The model prediction over transient engine operation is far superior to steady state model predictions for soot

emissions, as seen in Figure 5.31. The ability of the virtual sensor to capture frequent transient spikes of soot emission is particularly relevant since steady-state model significantly underpredicts concentrations during dynamic engine operation.

The error in predicting cumulative particulate matter and NO_X by the neuro-fuzzy models for different data set is shown in Figure 5.33 and Figure 5.34 respectively along with steady state model errors. Looking at the cumulative emissions, the steady state model underpredicts soot consistently by around 50%. The proposed emission virtual sensor provides an order of magnitude improvement, since the soot predictions are only at max 10% higher compared to measurements.

To explain the deficiencies of steady state models to predict transients correctly, we need to look closely into engine operation during transients. Consider the time interval around 35 sec. The engine command changes nearly instantaneously and the fuel injected follows the demand. The intake manifold pressure lags due to turbocharger inertia and the delay in boost pressure results in lower in-cylinder air-to-fuel ratio. The EGR command also changes to zero, however, residual gas dynamics have slower time scales and it takes time to purge the intake manifold. In addition, step change of load results in increased exhaust backpressure to inlet manifold pressure thereby increasing the internal residual [10], [11]. The presence of residual helps in reduction of NO_X but results in higher soot production. The combined effect of high instantaneous values of Fuel/Air ratio at the onset of the load transient [10], [11] and increased residual lead to sharp spikes of particulate concentration. The steady state emission model is only function of engine load and hence, cannot capture the transient effects. The transient model, on the other hand, can capture the effect of turbocharger inertia and EGR valve actuator dynamics on in-cylinder constituents, thereby giving superior predictions of resulting emissions.

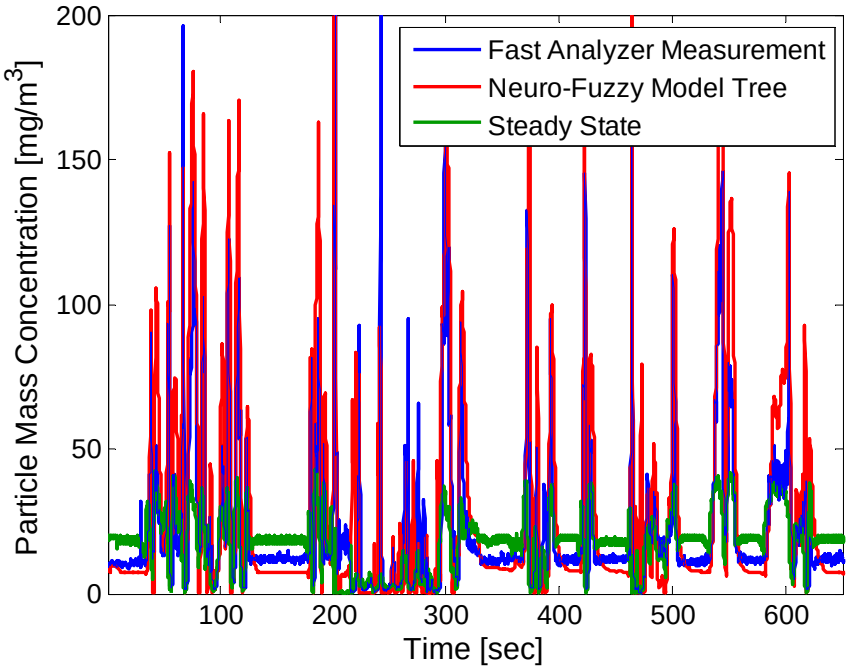


Figure 5.31: Neuro-fuzzy based model with OLS predictions along with measured data using fast analyzers and quasi-steady state model predictions for transient particulate matter emission for a particular validation data set.

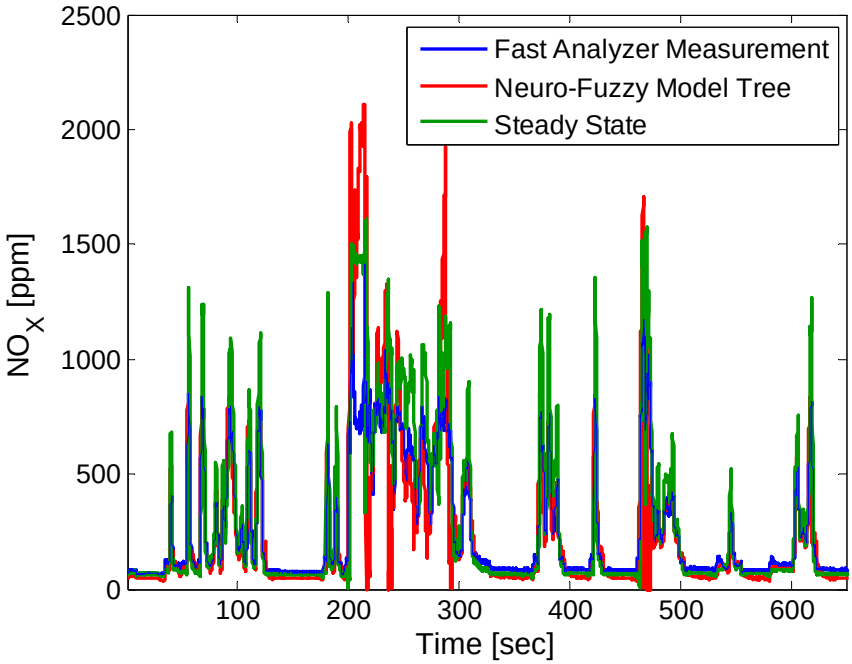


Figure 5.32: Neuro-fuzzy based model with OLS predictions along with measured data using fast analyzers and quasi-steady state model predictions for transient NO_X emission for a particluar validation data set.

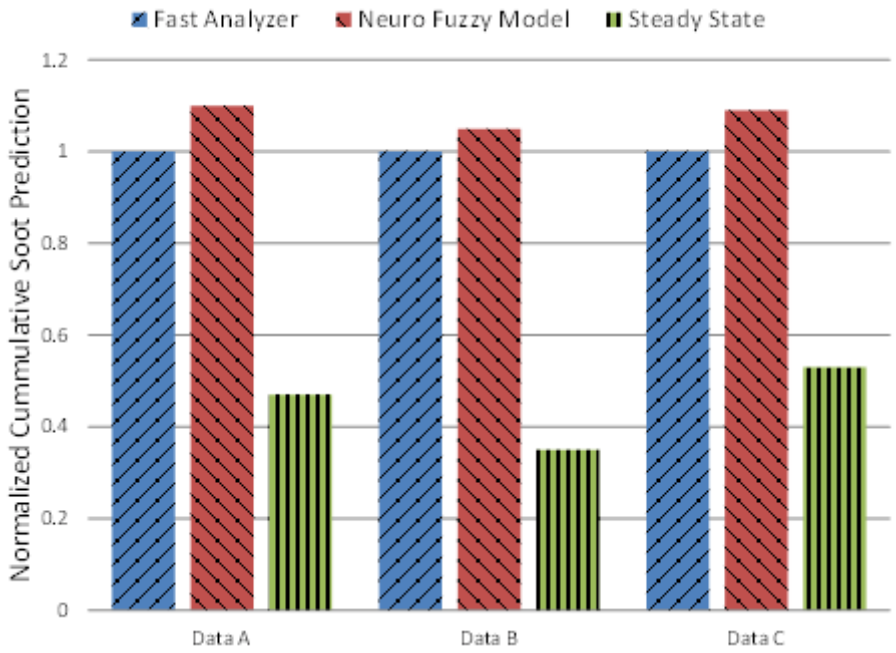


Figure 5.33: Comparison of cumulative particulate matter prediction by neuro-fuzzy tree based transient emission model with OLS and steady state models for different validation data set.

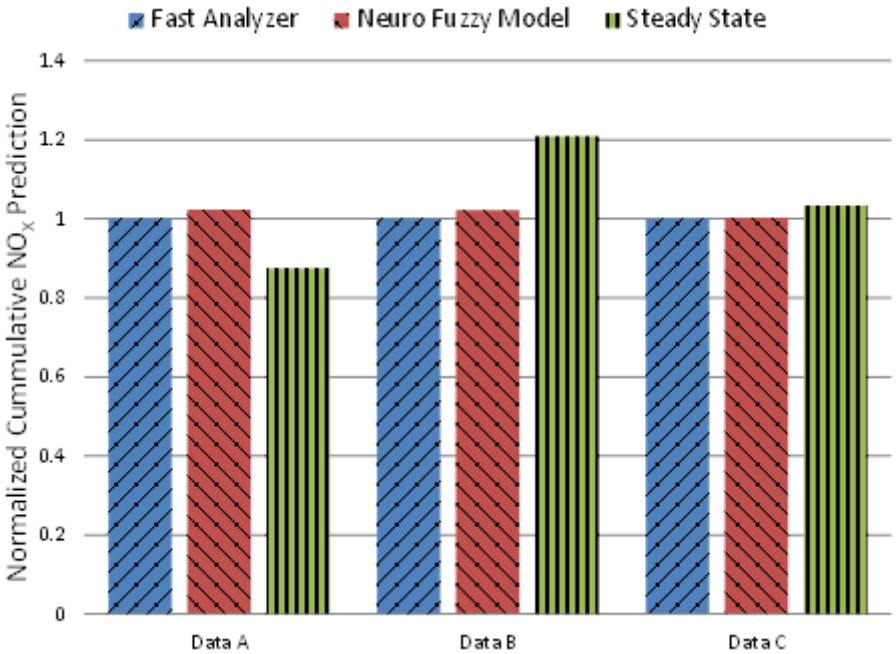


Figure 5.34: Comparison of cumulative NO_X prediction by neuro-fuzzy tree based transient emission models with OLS and steady state models for different validation data set.

5.5.2.2 Neuro-fuzzy model tree with automatic relevance determination

The leaf node submodels are multi-layer perceptron neural networks and can be represented as

$$\hat{y}(k) = f(w, u_1(k-1), \ldots, u_1(k-d_1), \ldots, u_n(k-1), \ldots, u_n(k-d_n), \hat{y}(k-1)) \qquad (5.69)$$

where w is the weight parameter, $u = [u_1, \ldots, u_n]$ is the inputs with maximum delay for each input obtained from cross correlation with output, $d = [d_1, \ldots, d_n]$ respectively.

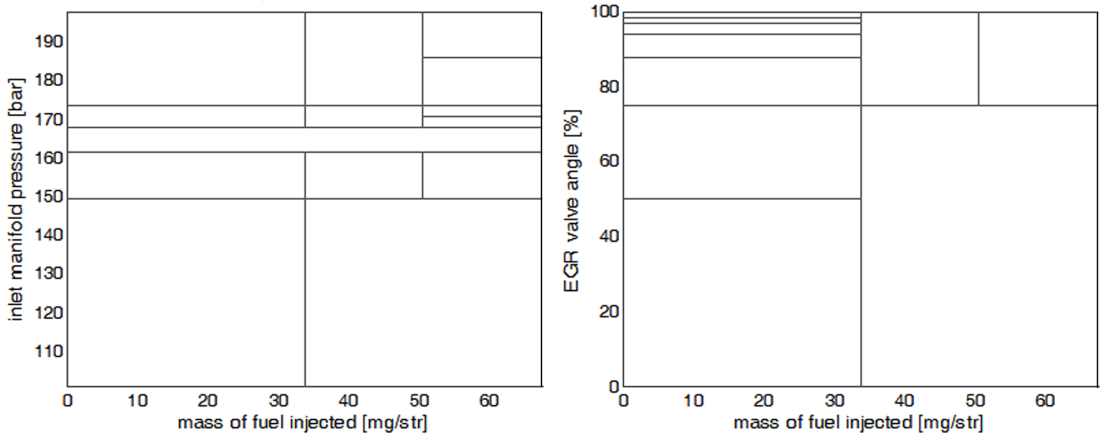


Figure 5.35: Input space partitioning for a representative first level neuro-fuzzy model for NO_X. Hyperplane divisions along two different input regressors are shown.

Each first level neuro-fuzzy model is further divided into several submodels using the local neuro-fuzzy training algorithm described in section 5.4.1. Each first level model is trained separately. The training is stopped if the number of local models reaches 20 or the sum square error falls below 0.01. Figure 5.35 shows a representative input space partitioning at leaf nodes with respect to 2 different division directions for NO_X model. Each hyperplane in Figure 5.35 represents input space modeled by a different local neural network.

The local neural network weights are trained using backpropogation with the objective function given by equation (5.50) and the update rule is given by

$$\overline{w}_{k+1} = \overline{w}_k - \gamma (J^T J + \mu I)^{-1} J^T e \qquad (5.70)$$

where γ is the step size, $\overline{w}$ is the weight vector, J is the Jacobian of error with respect to weights, e is the error and μ is the term added to deal with singularity of $J^T J$.

The input regressor selection for local models is based on three step procedure described section 5.3.2. A serious drawback with this approach i.e. relying solely on correlation for regressor selection is that correlating finite random signal will always show random correlation between input and output. This can lead to inclusion of irrelevant inputs for training of local neural network. A conventional neural network will not set the coefficients of these junk inputs to zero, thereby hurting the performance of the local model. However, models trained with ARD algorithm are capable of rejecting the junk inputs. The ARD is performed for optimization of every local neural network weights. The ARD ensures considering only relevant inputs and prevents the local models from overfitting.

The training is carried using the training data (section 5.3.3) and subsequently validated using validation data (section 5.3.4). Figure 5.36 and Figure 5.37 shows the validation results of the neuro-fuzzy based virtual sensors for particulate matter and NO_X respectively for one validation data. It can be seen from the results the neuro-fuzzy based models do a good prediction of transient particulate matter and NO_X emissions compared to the steady state models. This improvement in predictiveness is particularly strong in the case of transient soot emissions as shown in the Figure 5.36. The steady state (static) model is incapable of predicting spikes of every particulate concentration occurring at the initiation of rapid increase of load. The neuro-fuzzy model tree accurately predicts the timing and duration of spikes and slightly over predicts the magnitude of the peaks. These features make it very well suited for the system level analysis and control development. Finally, Figure 5.38 and Figure 5.39 give the error in cumulative prediction of particulate and NO_X for the proposed neuro-fuzzy model with ARD along with steady state models for 3 different data sets.

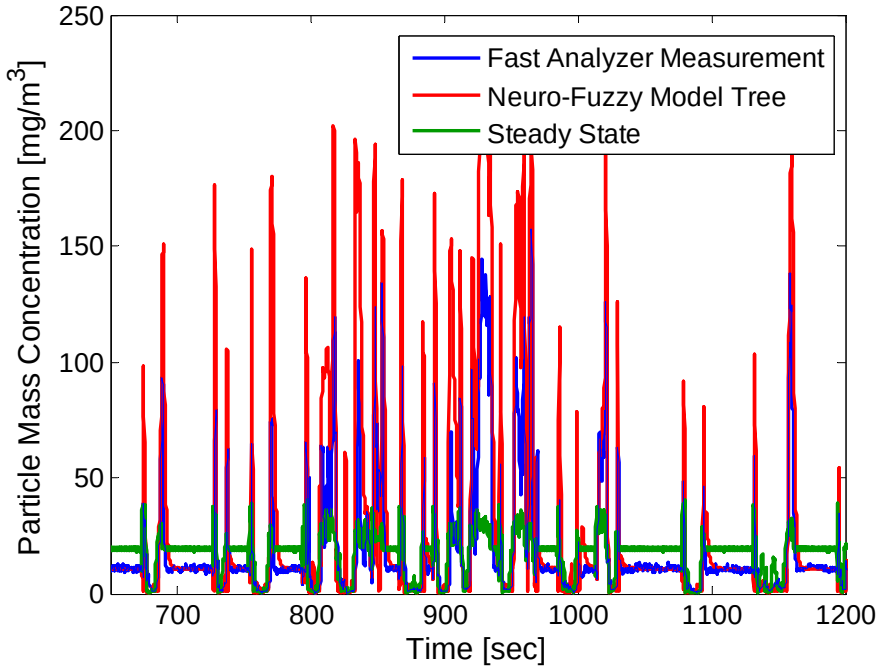


Figure 5.36: Neuro-fuzzy based model with ARD predictions along with measured data using fast analyzers and quasi-steady state model predictions for transient particulate matter emission for a particular validation data set.

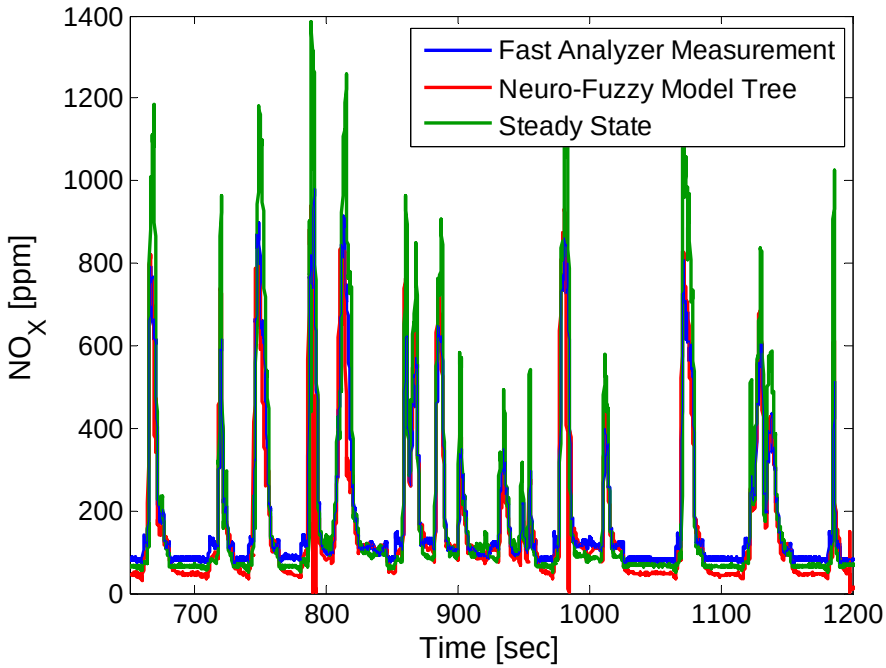


Figure 5.37: Neuro-fuzzy based model with ARD predictions along with measured data using fast analyzers and quasi-steady state model predictions for transient NO_X emission for a particular validation data set.

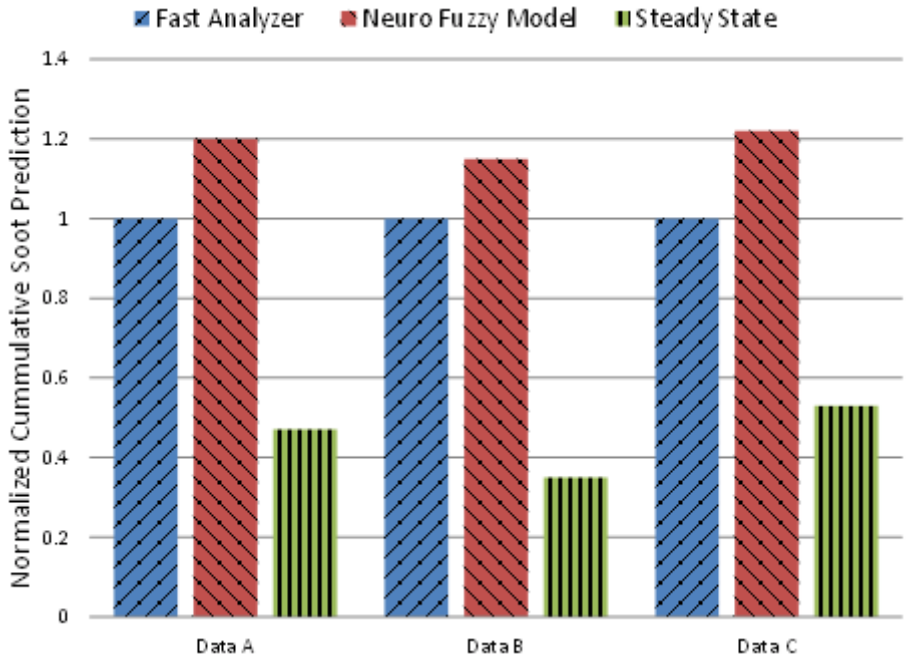


Figure 5.38: Comparison of cumulative particulate matter prediction by neuro-fuzzy tree based transient emission models with ARD and steady state models for different validation data set.

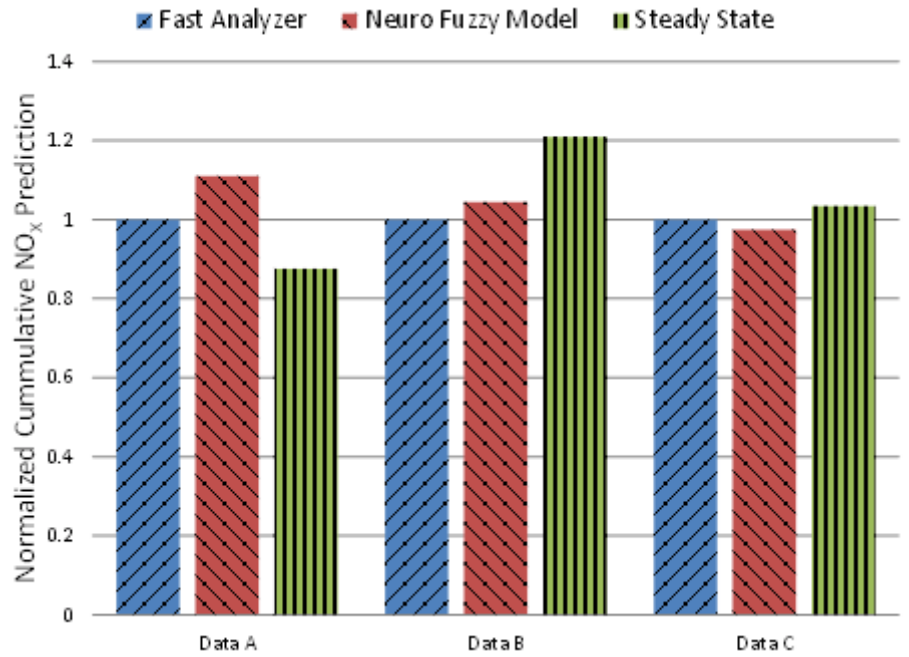


Figure 5.39: Comparison of cumulative NO_X prediction by neuro-fuzzy tree based transient emission models with ARD and steady state models for different validation data set.

Chapter 6

CONCLUSIONS

6.1 Summary

This dissertation provides a systematic framework for designing optimal power management controller for hybrid vehicles with multiple objectives. In particular, we apply the technique to minimize transient emissions along with fuel consumption in series hydraulic hybrid vehicle. The two key fundamental contributions of this work lies in the field of newer algorithms based on dynamic programming framework for problems with large state space, and advanced transient diesel emission models for soot and NO_X. The contributions of this work spans the full range of development process from theoretical analysis to model based controller development to implementation on hardware and real world testing.

The Chapter 2 introduced the mathematical models of vehicle and powertrain for development and validation of optimal controllers for hybrid vehicles. Two different categories of the models are developed. First, a high fidelity physics based models are developed for vehicle simulation and evaluation of different power management strategies. Second, control-oriented models are developed to be employed within dynamic programming framework. The control-oriented models ignore system dynamics faster than 1 Hz and are computationally very fast. The chapter also introduces the Engine-In-the-Loop (EIL) facility at the University of Michigan. The EIL allows for concurrent running of the real diesel engine connected with virtual powertrain/vehicle and measure real-time transient emissions using fast analyzers. Finally, a baseline thermostatic controller is developed. The controller is rule-based design with engine power demand based of present SOC. The controller thresholds are tuned using

parametric study similar to previous work done by Filipi et al. [23]. A 40% fuel economy is reported over FUDS compared to conventional vehicle.

The Chapter 3 investigates optimal power management for a series hydraulic hybrid vehicle. A Markov chain model for driver demand dynamics is developed with the transition probability matrix based on statistical sampling of naturalistic driving schedules. The power management problem with fuel economy objective is formulated as an infinite horizon discounted future problem and solved via stochastic dynamic programming techniques. The analysis includes additional degree of freedom compared to the traditional approach, as the engine power demand is split into two variables, namely engine torque and speed. The algorithm moved engine torque/speed points away from the best BSFC line to maximize the combined engine-pump power generation subsystem efficiency. This is a valuable lesson, indicating what is preferred from the system standpoint. The engine operation resembles load-following mode, except the speeds are much lower than in the case of a mechanical transmission, hence pushing the loads up into the high-efficiency region. The performance of supervisory controller is compared with baseline thermostatic controller and an improvement of 17% in fuel economy is reported. The system centric SDP controller also showed an additional 3-4% improvement over engine centric SDP controller.

The Chapter 4 introduces the concept of neuro-dynamic programming (NDP). NDP is an extension of dynamic programming framework designed to alleviate the curse of dimensionality. The idea centers on evaluation and approximation of optimal cost-to-go function. Two key aspects of the NDP approach are:

1. Cost-to-go approximation with functional representation: The cost-to-go is approximated using neural networks. This reduces the memory requirement as only the parameter vector needs to be stored in comparison to complete cost-to-go vector.
2. Incremental learning of cost-to-go: The learning of the cost-to-go function is performed in an incremental fashion using temporal difference algorithm.

The significance of NDP approach lies in the fact that, in contrast to deterministic dynamic programming or stochastic dynamic programming, the computational effort and

memory resources in case of NDP increases linearly with parameters in functional representation rather than exponentially with state space. This makes the approach scalable to more complex problems with larger state-action space.

A self-learning neural network based power management controller is designed for series hydraulic hybrid. The vehicle power management problem is formulated as a Markov decision process and solved using NDP techniques. To validate the NDP approach, first a single objective problem with only fuel consumption consideration is solved. The controller not only learned to manage the two power sources, namely engine and hydraulic energy stored in accumulator but learned to do so efficiently. The NDP based controller is compared against a baseline SDP controller. The fuel economy and system operation by both the controllers i.e. NDP and SDP are very similar and this effectively validates the NDP based approach for design of supervisory controllers. Next, the power management problem is augmented with transient emission model and newer power management problem is formulated with multiple objectives i.e. minimization of fuel consumption, and transient NO_X and soot emission. The NDP controller successfully solved the problem with a state-action space of 10^{13}, which is computationally intractable with classical SDP. The NDP controller is simulated over different EPA driving schedules and results were reported.

To assess the real world potential of NDP controllers, the performance of different power management controllers over FUDS is evaluated using the EIL facility. The controller and virtual hybrid powertrain is simulated on dSPACE real-time platform. EIL results showed that self-learning neural controller is able to successfully orchestrate the power in a series hydraulic hybrid to meet performance objectives while significantly reducing particulate matter and NO_X emissions and preserving most of the fuel economy gain attainable with optimized SDP policy. EIL results also demonstrate the robustness of the self-learning neural controllers' performance in real world environment and ability to perform exceedingly well in the presence of sensor noise or virtual sensors.

The Chapter 5 proposed modeling neuro-fuzzy based transient emission models for particulate matter and NO_X in diesel engine. They accurately capture the transient dynamics of soot and NO_X emissions unlike steady state models. In particular, the new model is able to capture extreme spikes of soot emissions occurring at the onset of rapid

load increases. The model is intended to run on a microprocessor in real-time and predict engine-out soot and NO_X emissions using signals from the ECU and low-cost physical sensors. The key aspects of this modeling work are:

1. Modeling relies on dividing the input space into smaller subspaces and fitting local models.
2. Recurrent architecture of the model allows for capturing transient characteristics.
3. Selection technique, orthogonal least squares is applied for selecting the structure of local model inputs.
4. Multi-level pseudo random perturbation signal is designed specifically for characterizing the diesel engine transients.
5. Virtual sensors are fast and capable of running in real-time along with the real engine.

In particular, three different transient emission models were proposed. The first model is designed to be employed with optimal control design framework. The model traded complexity and accuracy for very fast computation. The model have heuristic division of design space and pre-selection of local model structure. Each local model is a recurrent neural network and connected by triangular membership function. The second transient emission model has two-level hierarchical structure with polynomial local submodels and employs orthogonal least squares selection technique for selecting the structure of local submodel inputs. The model employs neuro-fuzzy model tree training algorithm for automatic division of input space and selection of local submodel. Gaussian validity functions are chosen for determining influence region of each local submodel. The model combines orthogonal least square selection technique for selecting the relevant input regressors for each local model. Finally, the third model also has two-level hierarchical model with local multilayer perceptron network and also employs neuro-fuzzy model tree training algorithm for input space partitioning. A selection technique based on Baye's framework for automatic relevance determination is applied for shortlisting the relevant inputs. All the three type of models are coded in C for performance. The training and validation data is obtained from the experimental setup at the University of Michigan for transient testing of a medium duty diesel engine-in-the-

loop with virtual vehicle. Comparison of the predictions with transient measurements demonstrates good agreement. The models accurately capture the transient dynamics of soot and NO_X emissions, and predict the extreme spikes of soot with change in load unlike steady state models.

In summary, the dissertation advances the knowledge in the field of optimal control design for developing advanced power management controllers with very large state space. The controller effectively manages multiple objectives of fuel consumption and transient NO_X and soot emissions, another first in the area of power management. The dissertation also advances the knowledge in the development of transient emission virtual sensors for control studies and onboard vehicular application.

6.2 General Comment

Throughout this dissertation, the design of optimal controller with multiple objectives has been emphasized. The work is novel and different from previous work done in the field of power management controller for hybrids in sense that it incorporates transient emission objectives. Nevertheless, the main contribution of this work lies in the innovative neuro-dynamic programming algorithm for stochastic optimization and its viable applicability to alleviate the curse of dimensionality associated with dynamic programming based algorithms.

Dynamic programming is an effective tool for solving Markov decision problems i.e. problems of sequential decision making under uncertainty (stochastic control). Dynamic programming can be applied to find optimal solutions using constrained non-linear model based systems. However, as mentioned before, dynamic programming based algorithms have to deal with curse of dimensionality. A detailed modeling of multiple phenomena invariably involves increase in system/plant states and the space spanned by the states grows exponentially. This makes the application of dynamic programming to practical real-world problems somewhat limited. Neuro-dynamic programming combines the strengths of classical dynamic programming with innovative techniques like reinforcement learning, temporal difference learning and cost-to-go functional approximation. This allows more complex real-world problems to be optimally solved.

6.3 Summary of Contributions

The contributions of this dissertation are summarized below:

1. A self-learning power management controller is introduced which learns by interacting with the environment. The power management controller successfully learns to minimize combined weighted objective of fuel consumption and transient engine emissions.
2. The neuro-dynamic programming is successfully applied for the first time in context of designing power management controller for hybrids with multiple objectives. The algorithm is the first demonstration of application of policy optimization based power management controller with 8 states and state-action space cardinality of 10^{13}. Mathematical techniques like functional approximation of cost-to-go function and incremental training using temporal difference learning are introduced for the first time in the context of designing hybrid power management controller.
3. Transient emission models for soot and NO_X are developed. These transient models are implemented in real time and are validated by running concurrently with physical engine. The models are based on neuro-fuzzy model tree with the modeling technique motivated by the idea of divide and conquer of the input-output space. The models are capable of predicting transient soot and NO_X emissions over complete range of engine operation with input parameters available from standard ECU and sensors.
4. The proposed power management controller is simulated along with virtual hybrid powertrain concurrently with real engine in the EIL facility for realistic evaluation in real-life conditions. A systematic approach for transitioning from simulation to embedded controller for real world application is presented.
5. Two different powertrain models, namely control oriented and vehicle simulation models are developed for series hydraulic hybrid architecture.
6. A power management controller with fuel economy objective is solved using SDP. The problem is formulated to produce state feedback of desired set point for both engine speed and torque i.e. the engine is not restricted to operate along the

best BSFC line unlike previous works where the sole controller output is engine power demand i.e. the engine is operated along best BSFC line.

7. A multi-Pseudo Random Signal (m-PRS) perturbation signal is designed specifically for characterizing the diesel engine. The test signal is designed to excite all the engine operating frequencies to ensure the training data obtained captures all the operating regimes of the engine
8. Multiple numerical techniques are introduced to reduce the computational time of policy optimization techniques.
9. Fast transient emission analyzers are used to develop deep insight into engine system behavior and quantifying the emission reduction with new proposed controller.

6.4 Perspective on Future Work

The dissertation proposes a new approach to design supervisory controller for hybrid vehicles. The proposed algorithm scales well with increase in state-action space and framework allows for integrating multiple objectives. Nonetheless, there exists several opportunities to advance the work presented here. From the application standpoint, the algorithms can be applied to different hybrid architecture or with different cost objectives. From theoretical perspective, the algorithms can be augmented to solve other problems. Some of these future works is presented next.

6.4.1 Application to Other Optimal Power Management Problems

The proposed neuro-dynamic programming technique can be applied for designing optimal control for hybrid vehicles with other objectives like battery health [141], [142], drivability [143], [144] and thermal management [145]. Previous work has relied on model order reduction techniques to allow integration of these models within dynamic programming framework. Another common approach is to discretize continuous state variables coarsely to reduce space spanned by states. Since the NDP algorithm scales well with increase in state space, advanced models can be included without exponential increase in computational resources. For example, the HEV problem formulated by

Moura et al. [142] can be expanded to include full electrochemistry model or the hybrid model can be augmented with thermal models to include thermal effects in power management controller design.

6.4.2 Adaptive Markov Model for Driver

The NDP algorithm presented in this dissertation relies on an underlying Markov chain model for drive cycle. The transition probabilities are calculated a priori by statistically analyzing the naturalistic driving schedules. The underlying Markov driver model can be adapted to learn driver behavior based on real-time observations similar to work done by Bichi et al. [146]. The NDP algorithm is incremental in nature and can be modified to learn the optimal power management strategy on-board vehicle with modified Markov model. This will result in adaptive self-learning power management strategies that adapts to vehicle driving schedule and driver driving behavior.

6.4.3 Powertrain-In-the-Loop

The proposed power management strategies are demonstrated using the EIL setup. EIL facility provides a systematic evaluation of engine transients and engine-out emissions. However, the effects of system level interaction of powertrain components with engine are limited to dynamical effects modeled in virtual real-time models. The setup is presently being upgraded to Powertrain-in-the-Loop (PIL) facility with physical pump connected to engine and hydraulic motor connected to dynamometer [80]. The PIL facility will allow in-depth analysis of the effect of component level dynamics on engine performance i.e. fuel consumption and transient emissions. The findings from PIL can then be used to improve and re-calibrate the simulation models to better capture transient behavior and increase fidelity of optimal power management techniques.

Figure 6.1 shows a picture of the hardware included in the H-PIL setup at the University of Michigan. The power generation subsystem comprises the V8 diesel engine and the hydraulic pump, shown in the forefront. The propulsion subsystem includes a hydraulic traction motor coupled with a dynamometer, which simulates the vehicle inertia. There is no physical connection between the power generation subsystem and propulsion subsystem except for the hydraulic fluid.

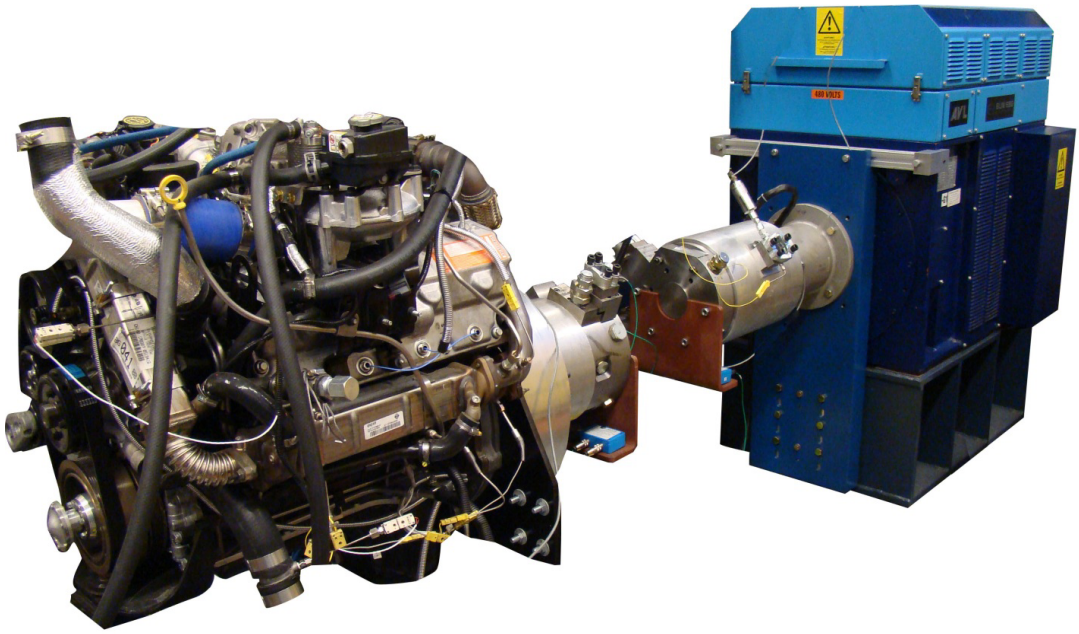


Figure 6.1: Hydraulic hybrid powertrain-in-the-loop setup.

6.4.4 Different Approximate Dynamic Programming Algorithms

The dissertation proposes temporal difference learning based algorithm for incremental upgrade of cost-to-go functional. Other machine learning techniques by Sutton [108] like SARSA and Q-learning can also be explored.

6.4.5 Different Approximation Architectures

The cost functional in this dissertation is approximated using neural networks. Though neural networks are universal approximators and lend them beautifully to the problem, training them requires large computational resources and sophisticated numerical techniques. Specifically, when neural network is trained incrementally and the training is feedback in nature i.e. the output of neural network is also used to train the network. Simpler functional approximations [109], [108] like polynomials, splines, wavelets, coarse coding, tile coding and Kanerva coding can be used instead for functional approximation. These approximating functions are particularly useful where the cost-to-go function does not have very complex shape or prior information about cost-to-go function is available.

6.4.6 Adaptive Grid

The state variables in this dissertation are discretized with uniform spacing. In [147] the concept of adaptive grid generation is introduced. The idea is to use a coarse grid to solve the problem. The solution of this controller is used as starting point for next problem with finer grid. This process is repeated until the solution for the desired grid resolution is obtained. This idea with one-dimensional grid was demonstrated in [147] with problem solved using linear programming. This idea can be extended to neuro-dynamic programming algorithm outlined in this dissertation.

6.4.7 Emission Models

The emission models developed in this dissertation are capable of being employed either in control optimization studies or running along with physical engine as a virtual sensor. The transient model-based NO_X and soot virtual sensors can provide real-time predictions and can be used in engine-oriented strategies that require feedback of emissions under transient operating conditions.

6.4.8 Pareto Optimality Sets

It is straight forward with the proposed NDP approach to generate pareto optimality sets by sweeping through different weighting matrix in the objective function. The goal of the dissertation is to develop new power management techniques to alleviate the curse of dimensionality in dynamic programming framework. The code is not suitable for generating optimal front, as the time required to solve each NDP problem is large with present algorithm. The algorithm uses Matlab based implementation of neural network, which is suboptimal. If goal is to generate pareto optimality set, the design process can be significantly improved by using more efficient C implementation of neural networks.

APPENDICES

APPENDIX A

TEST CELL SPECIFICATION

A.1 Engine

A 6.4 L V-8 direct-injection diesel engine manufactured by the Navistar Ltd. is used for this work. Engine specifications are given in Table A.1. The engine is intended for a variety of medium duty truck applications covering the range between Classes IIB and VII. The engine is a modern, state-of-art and incorporates advanced technologies to provide high power density while meeting 2007 emissions standards. The engine incorporates common rail direct injection (CRDI) system, which permits precise control of fuel injection timing, pressure, quantity, and number of injections. In order to meet NO_X regulations, engine uses exhaust gas recirculation (EGR) circuit to introduce cooled exhaust gases into the intake manifold. EGR flow rate is controlled through modulation of the EGR valve. The dual stage variable geometry turbocharger (VGT) is used to enhance engine performance, as it reduces boost lag and allows control of the intake manifold pressure. Figure A.1 gives the engine brake specific fuel consumption map used in this dissertation. The BSFC map was generated at the University of Michigan.

Table A.1: Diesel engine specifications

Engine Type	DI 4-Stroke Diesel Engine
Configuration	V-8, Cam-in-Crankcase, 90°
Bore x Stroke	98 mm x 105 mm
Displacement	6.4L
Rated Power	261 kW @ 3000 RPM
Rated Torque	881 Nm @ 2000 RPM
Compression Ratio	16.7 : 1
Valve Lifters	Push Rod-Activated Rocker Arm
Aspiration	Variable Geometry Dual stage Turbocharger / Intercooler
Fuel Delivery System	Common Rail Direct Injection (CRDI)

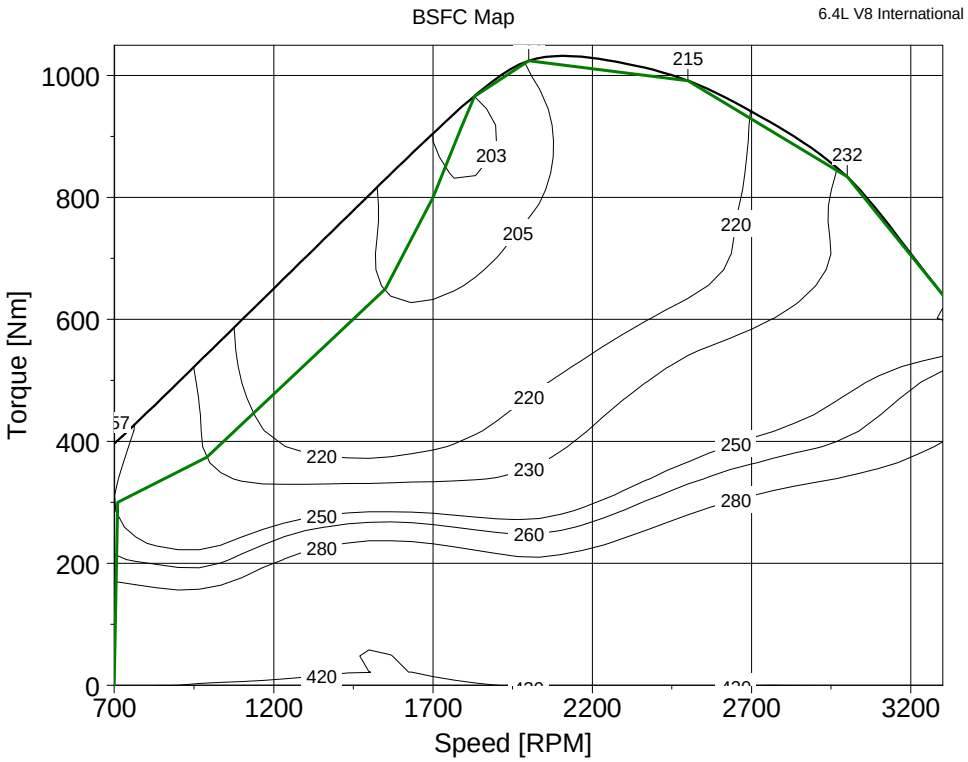


Figure A.1: International 6.4L V8 BSFC map with best BSFC line shown in green.

A.2 Dynamometer

The engine is coupled to a 330 kW AVL ELIN series 100 APA Asynchronous Dynamometer. This dynamometer is especially suited to perform transient testing with a 5 ms torque response time and a -100% to +100% torque reversal time of 10 ms. AVL PUMA Open system orchestrates engine operation in the test cell, and provides monitoring and control of test cell functions. The AVL PUMA interfaces and communicates with dSPACE real-time system. This facilitates concurrent running of engine with virtual driveline and vehicle systems. The engine is fully instrumented with both 10 Hz time-based measurements of pressures, temperatures and flow rates at various locations in the system, and 0.1 crank-angle resolved in-cylinder pressure traces.

ETAS INCA software is used for interfacing with the engine's Powertrain Control Module (PCM). INCA allows full control over injection parameters, EGR valve and VGT vane settings. The setup allows synchronized data recording between INCA and PUMA Open system.

A.3 dSPACE

The dSPACE real-time platform is a modular system with flexible processor and I/O board arrangement. The dSPACE system simulates virtual components in real-time, communicates with the engine and the dynamometer via a test cell controller, and finally supervises and coordinates operation of the hybrid powertrain. The DS1006 simulates the virtual models in real-time, and coordinate the communications between the I/O boards. The custom AVL board interfaces the dSPACE control box with the PUMA Open system and allows two-way communication between dSPACE and AVL PUMA. The dSPACE electronics specifications are given in Table A.2.

Table A.2: dSPACE electronic system specifications

Item	Description
PX10	Power supply, motherboard, optical network card and electronics casing.
DS1006	2.2 GHz processor board, with 256 MB DDR SDRAM, 128 MB SDR SDRAM and 2 MB flash memory.
AVL interface	2 link interfaces, 10 frequency inputs and 10 frequency outputs or up to 20 digital inputs and 20 digital outputs.
DS2202	I/O hardware with 16 differential A/D channels, 20 D/A channels, 38 digital inputs, 24 PWM measurement inputs, 16 digital outputs, 9 PWM outputs, 2 CAN channels, 2 serial interface and 2 CAN interface.
DS4302	4 independent CAN channels, 50 MHz processor, 128K 32-bit SRAM, 512K 8-bit flash memory

A.4 Emissions measurement

A.4.1 Fast NO_X Analyzer

A CLD 500 Fast NO_X analyzer is used for accurate temporal measurement of NO_X. It consists of a chemiluminescent detector with a 90%→10% response time of less than 3 ms for NO, and less than 10 ms for NO_X. The detectors in remote sample heads are positioned very close to the sample point in the engine and use vacuum to convey the sample gas to the detectors through narrow heated capillaries.

The Fast NO_X analyzer provides NO_X concentration in parts per million (ppm). This can be subsequently converted to mass flow of NO_X with the equation

$$\dot{m}_{NOx} = \frac{ppm_{NOx}}{10000} \cdot \frac{MW_{NOx}}{MW_{exhaust}} \cdot \left(\dot{m}_{air} + \dot{m}_{fuel}\right) \tag{A.1}$$

where MW_{NOx} is molecular weight of NO_X, $MW_{exhaust}$ is molecular weight of exhaust, and $(\dot{m}_{air} + \dot{m}_{fuel})$ is the total mass flow rate of exhaust. The total mass flow rate is measured using AVL Combustion Emissions Bench (CEB-II).

A.4.2 Fast Particulate Differential Mobility Spectrometer

Temporally resolved particulate concentrations are obtained using a differential mobility spectrometer (DMS) 500. This instrument measures the number of particles and their spectral weighting in the 5 nm to 1000 nm size range with a time response of 200 ms. The DMS provides aerosol size spectral data by using a corona discharge to place a prescribed charge on each particle. The charged particles are then carried along a classifier column by a sheath of clean air, as shown in Figure A.2. Within the column, particles are subjected to a radial electric field from a central electrode, which repels them towards the periphery. Particles with lower aerodynamic drag-to-charge ratio will deflect more quickly and are attracted towards electrode rings closer to the beginning of the classifier column, and vice versa.

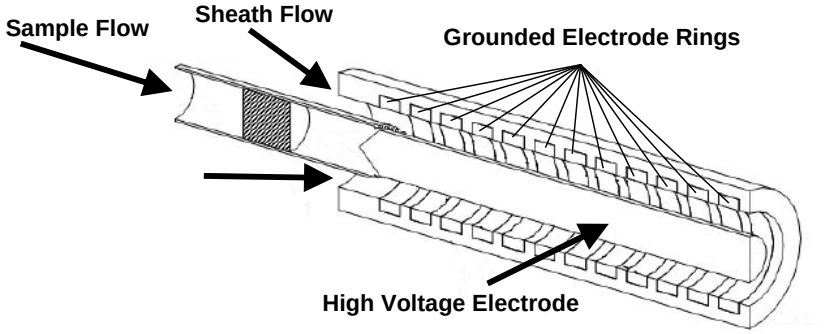


Figure A.2: DMS 500 Classifier Column – from [151].

As the particles land on the grounded rings, they give up their charge and these outputs from the electrometers are processed in real time to provide spectral data and other desired parameters. Typical spectral data from the DMS 500 are shown in Figure

A.3 where the x-axis is particle diameter (D_P) in nanometers and the y-axis is the spectral density with a unit of dN/dlog D_P/cc. Thus, the area under the curve represents the number of particles per cubic centimeter of sample.

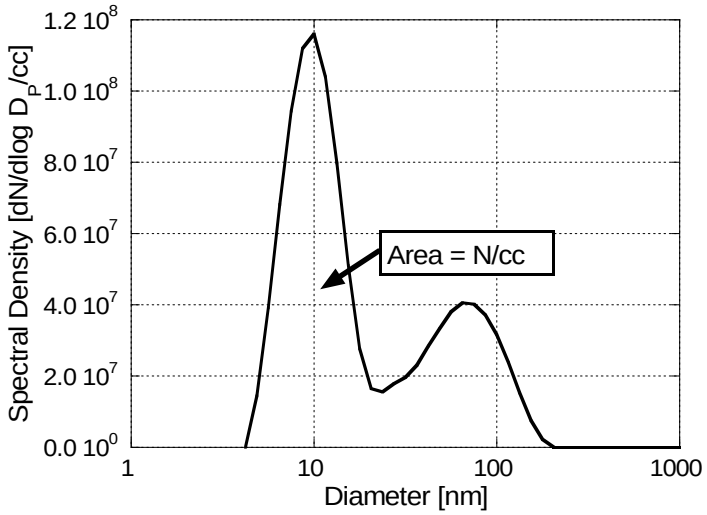


Figure A.3: Sample particulate spectral density curve.

To obtain total particle mass, the data is grouped into particle diameter bins. The density of the particles within each bin is assumed to be constant and then the mass of particles in each bin is determined by [148]:

$$\text{Particle Mass} = 6.95\times10^{-3}\cdot D_P^{2.34}\cdot \text{Number of Particles} \qquad \text{(A.2)}$$

In this equation, the non-spherical nature of particle shapes is accounted for by the diameter (D_p) exponent smaller than three. The leading coefficient acts as a "pseudo-density" for the particles in that bin; its magnitude is affected by particle constituents and also the unit changes that occur with the non-integer particle diameter exponent. After the particle mass is calculated for each bin, the total mass is found by summing the masses in each bin and dividing it by the number of bins per decade.

$$\text{Total Mass} = \frac{\left[\sum_{Bins} Mass\right]}{Bins/Decade}\ [\text{kg/m}^3] \qquad \text{(A.3)}$$

This summation is used as an approximation that accounts for integration over a logarithmic scale.

APPENDIX B

TRAINING ALGORITHMS

The training of neural network is an optimization problem where the goal is to find the optimum set of parameters of an approximating architecture that provide best fit between input/output pairs. The estimation problem can be stated as optimizing the expectation of a function

$$\min_{r \in R} \mathrm{E}\left\{F\left(r\right)\right\} \tag{B.1}$$

where typically F is the mean square error (MSE).

The challenges arise when approximating a DP cost-to-go function using sample pair $\left(i,\hat{J}\right)$ of states i and corresponding noisy cost-to-go estimate $\hat{J}$ generated using Monte Carlo simulation. In the context of cost-to-go function, the optimal parameter vector r can be obtained using following equation.

$$r = \arg\min_{r \in R} \mathrm{E}\frac{1}{2}\left(\hat{J} - \tilde{J}(r)\right)^2 = \min_{r \in R} \frac{1}{2}\sum_{i}^{m}\left(\hat{J}_i - \tilde{J}(r_i)\right)^2 \tag{B.2}$$

where $\hat{J}$ is the estimate of J^* at present instance and $\tilde{J}(r)$ is the present representation of cost-to-go functional.

The Monte Carlo based algorithm described in Chapter 4 generates the sample pair $\left(i,\hat{J}\right)$ incrementally, one at a time. Batch training algorithms, owing to the nature of the problem, i.e. iterative generation of sample pair and extremely large size of state space, cannot be employed for optimizing the parameter vector. The optimization problem needs to be solved using iterative algorithms. Next section gives a brief overview of different iterative algorithms adapted for training cost-to-go function in this work.

B.1 Steepest Descent

The stochastic gradient descent method is given by

$$r_{n+1} = r_n - \gamma_n \nabla F(r) \tag{B.3}$$

where $\gamma > 0$ is the step size. Substituting for F

$$r_{n+1} = r_n - \gamma_n \nabla \tilde{J}(r)\left(\hat{J} - \tilde{J}(r)\right) \tag{B.4}$$

B.2 Gauss-Newton Method

Given r_n, the Gauss-Newton iteration is based on linearizing F around r_n to obtain the function

$$\tilde{F}(r, r_n) = F(r_n) + \nabla F(r_n)^T (r - r_n) \tag{B.5}$$

And then minimizing the norm of the linearized function $\tilde{F}$

$$\begin{aligned} r_{n+1} &= \arg\min \frac{1}{2}\left\|\tilde{F}(r, r_n)\right\|^2 \\ &= \arg\min \frac{1}{2}\left(\left\|F(r_n)\right\|^2 + 2(r - r_n)^T \nabla F(r_n) F(r_n) + (r - r_n)^T \nabla F(r_n) \nabla F(r_n)^T (r - r_n)\right) \end{aligned} \tag{B.6}$$

The above quadratic minimization leads to

$$r_{n+1} = r_n + \gamma_n \left(-\left(\nabla F(r_n) \nabla F(r_n)^T\right)^{-1} \nabla F(r_n) \cdot F(r_n)\right) \tag{B.7}$$

where γ is the step-size. Substituting for F

$$r_{n+1} = r_n - \gamma_n \left(\nabla \tilde{J}(r) \cdot \nabla \tilde{J}(r)^T\right)^{-1} \nabla \tilde{J}(r)\left(\hat{J} - \tilde{J}(r)\right) \tag{B.8}$$

B.3 Levenberg-Marquardt Method

The Gauss-Newton method can be modified to ensure steady descent, and to deal with singular matrix $(\nabla \tilde{J}(r) \cdot \nabla \tilde{J}(r)^T)$ and to enhance convergence when matrix is near singular). The Gauss-Newton method is modified with inclusion of μI where μ is a small positive multiplier and I is the identity matrix.

$$r_{n+1} = r_n - \gamma_n \left(\nabla \tilde{J}(r) \cdot \nabla \tilde{J}(r)^T + \mu I\right)^{-1} \nabla \tilde{J}(r)\left(\hat{J} - \tilde{J}(r)\right) \tag{B.9}$$

B.4 Extended Kalman Filter (EKF)

The extended Kalman filter can be employed for parameter estimation by augmenting the system states with unknown parameters. If Kalman filter is to be employed only for estimation, the system can be considered stationary with no system dynamics. The augmented states and system equation is then given by

$$\hat{x} = \begin{pmatrix} x \\ r \end{pmatrix} \tag{B.10}$$

$$\begin{aligned} \hat{x}_n &= f(\hat{x}_{n-1}, u_n) \\ y_{n-1} &= h(\hat{x}_{n-1}) + v_n \\ \hat{x}_0 &= \begin{bmatrix} 0 \\ r_0 \end{bmatrix} \end{aligned} \tag{B.11}$$

where y_n is the output of the system at observation time n and, nonlinear functions $f(\cdot)$ and $h(\cdot)$ are defined below. The v_n is zero mean white Gaussian noise with covariance R.

$$\begin{aligned} f(\hat{x}_n, u_n) &= \begin{bmatrix} x_n \\ r_n \end{bmatrix} \\ h(\hat{x}_n) &= F(r_n) \end{aligned} \tag{B.12}$$

Using preceding model and observations, the estimate of the system states can be updated as follows

$$\begin{aligned} \tilde{x}_n &= f(\tilde{x}_{n-1}, u_{n-1}) + K_n \left(y_{n-1} - h(\tilde{x}_{n-1}) \right) \\ K_n &= P_{n-1} H_{n-1}^T \left[H_{n-1} P_{n-1} H_{n-1}^T + R \right]^{-1} \\ P_n &= P_{n-1} - K_n H_{n-1} P_{n-1} \end{aligned} \tag{B.13}$$

where K_n is the Kalman gain matrix, y_{n-1} is desired output, H_n is the gradient matrix obtained by linearizing the $h(\cdot)$, and P_n is the matrix representing the uncertainty in the estimates of the states of the system.

$$H_n = \frac{\partial}{\partial \hat{x}} h(\hat{x}) \bigg|_{\hat{x}=\tilde{x}_n} = \nabla F(r_n) \tag{B.14}$$

Substituting for F, the EKF update can be written as

$$r_{n+1} = r_n + \gamma_n \left(K_n \cdot \nabla F(r_n) \cdot F(r_n) \right) \tag{B.15}$$

and,

$$K_{n+1} = K_n - \frac{\left(K_n \nabla \tilde{J}(r_{n+1})\right) \cdot \left(K_n \nabla \tilde{J}(r_{n+1})\right)^T}{1 + \nabla \tilde{J}(r_{n+1})^T K_n \nabla \tilde{J}(r_{n+1})} \tag{B.16}$$

APPENDIX C

STEP SIZE RECIPES

The step size plays an important role in convergence of stochastic iterative algorithm. The parameter vector r in stochastic environment cannot converge to limit vector r^*. In the best circumstances, r will reach neighborhood of a solution r^* and will move randomly in that neighborhood. The size of neighborhood can be controlled and can be made smaller by using smaller γ but convergence to r^* cannot be obtained until γ is positive constant. If on the other hand, γ is allowed to decrease to 0, the possibility of parameter vector converging to r^* exists. However, if the γ decreases too quickly, the algorithm will never succeed in converging to r^*. One of the challenges in Monte Carlo methods discussed in this dissertation is finding the appropriate step size γ. A standard technique in deterministic problems is to find the value of γ so that r gives the smallest objective function value (among all possible values of γ). This is not difficult for a deterministic problem. For a stochastic problem, it means calculating the objective function, which involves computing an expectation. For the Monte Carlo based methods discussed in this dissertation, this is computationally intractable making it impossible to find optimal step size. This section provides a brief literature survey of different step size recipes, followed by the step size recipes used in this work (Chapter 4). A more detailed survey of step sizes can be found in [107].

C.1 Deterministic Step Size Rules

Deterministic step size rules depend only on the iteration number. The step sizes are based on simple update rule and are generally heuristic in nature.

C.1.1 Constant

A constant step size rule (Figure C.1) is given by

$$\gamma_n = \begin{cases} 1 & \text{If } n = 1 \\ \bar{\gamma} & \text{otherwise} \end{cases} \tag{C.1}$$

where $\bar{\gamma}$ is a constant less than 1. Constant step sizes are easy to implement and do not require any knowledgeable guess about the rate of convergence of optimization algorithm. However, a constant step size will not result in convergence to r^* under stochastic environment.

C.1.2 Annealed

The step size is proportional to the iteration number, Figure C.1.

$$\gamma_n = \frac{1}{n} \tag{C.2}$$

Annealed step size rule is arguably the most popular rule but a major drawback with the annealed step size is that it drops to 0 very quickly. This results in convergence issues and can result in apparent convergence even when solution is far from optimal.

C.1.3 Harmonic

A generalization of the annealed step size rule is the generalized harmonic sequence, Figure C.1, given by

$$\gamma_n = \frac{a}{a+n-1} \tag{C.3}$$

where a is the constant. For $a > 1$, the harmonic rule provides larger step sizes than annealed rule. Increasing a slows down the rate at which the step size drops to 0.

C.1.4 Polynomial

An extension of harmonic step size rule (Figure C.1) is

$$\gamma_n = \frac{1}{n^{\beta}} \tag{C.4}$$

where $\beta \in (0.5, 1]$. A smaller value of β slows the rate of step size decline improving the response of algorithm in presence of initial transient conditions.

C.1.5 Search-then-Converge

The Search-then-converge step size rule (Figure C.1) is a variation of harmonic step size rule and is given by

$$\gamma_n = \gamma_0 \frac{\left(\frac{b}{a} + a\right)}{\left(\frac{b}{a} + a + n^{\beta}\right)} \tag{C.5}$$

where a, b and β are different tunable parameters and depending on the choice of these parameters, basic harmonic and polynomial step size rules can be generated. The algorithm provides a period of high step size for search and then step size decreases for convergence. The degree of delayed learning is controlled by parameter b. The exponent β has the effect of increasing step size in later iterations. By controlling β, a slow descent i.e. increased learning period can be obtained. Figure C.1 shows different step size rules compared to each other.

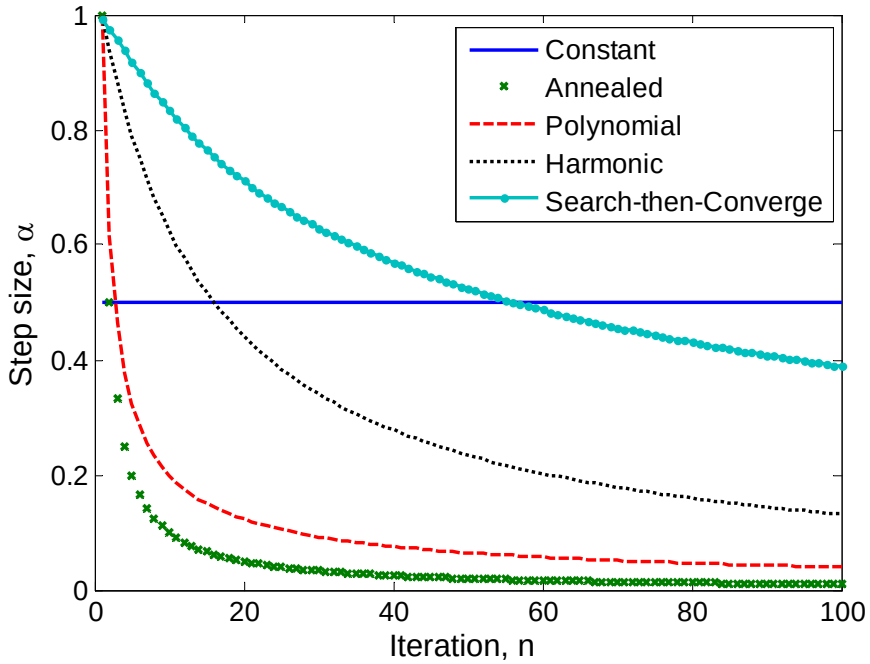


Figure C.1: Different step size rules.

C.2 Stochastic Step Size Rules

Stochastic step size rules adapts with the data and change step size depending on the trajectory of the algorithm. Stochastic step size tries to keep the step size large when the parameter being estimated is changing quickly. The stochastic rules offer freedom to have different step size for different parameters based on the individual convergence rate.

C.2.1 Sompolinsky-Barkai-Seung

The Sompolinsky-Barkai-Seung (SBS) algorithm for step size calculation [149] is given by

$$\gamma_n = \gamma_{n-1} + a\gamma_{n-1}\left\{b\left(f(\cdot, r_{n-1}) - F^*\right) - \gamma_{n-1}\right\} \tag{C.6}$$

where γ is the step size, a and b are positive constants, r is the parameter vector, $f(\cdot, r)$ is a differentiable loss function defined by $f(\cdot, r) = \sum\left(J^* - \tilde{J}(\cdot, r)\right)^2$, and $F^* = \min_r \mathbb{E}[f(\cdot, r)]$ is minimal loss function.

The idea is that when the error is large, the step size γ takes a large value, $\gamma_n \approx b(f(\cdot, r_n) - F^*)$. When the error is small, i.e. the estimator is close to optimal value, α approaches 0 automatically as $\gamma_n = \gamma_{n-1} - a\gamma_{n-1}^2$.

C.2.2 Bias Adjusted Kalman Filter

Powell [150], [107] derived an optimal step size rule, called Bias Adjusted Kalman Filter (BAKF), for estimating parameter from sequence of independent observations $\hat{\theta}^n$ with unknown mean θ^n and variance σ^2. The optimal step size is a solution to

$$\min_{0 \le \gamma_{n-1} \le 1} \mathrm{E}\left[\left(\bar{\theta}^n(\gamma_{n-1}) - \theta^n\right)^2\right] \tag{C.7}$$

i.e. γ_{n-1} minimizes the unconditional expectation of the error between $\bar{\theta}^n$ with true meanθ^n. The solution is given explicitly by formula

$$\gamma_{n-1} = 1 - \frac{\sigma^2}{(1+\lambda^{n-1})\sigma^2 + (\beta^{n-1})^2} \tag{C.8}$$

where λ is computed recursively using

$$\lambda^n = \begin{cases} (\gamma_{n-1})^2 & n = 1 \\ (1-\gamma_{n-1})^2 \lambda^{n-1} + (\gamma_{n-1})^2 & n > 1 \end{cases} \tag{C.9}$$

and $\beta^{n-1} = \theta^n - \mathbb{E}[\bar{\theta}^{n-1}]$, i.e. bias in smoothed estimate from previous iteration. The bias itself is computed recursively as it is also unknown.

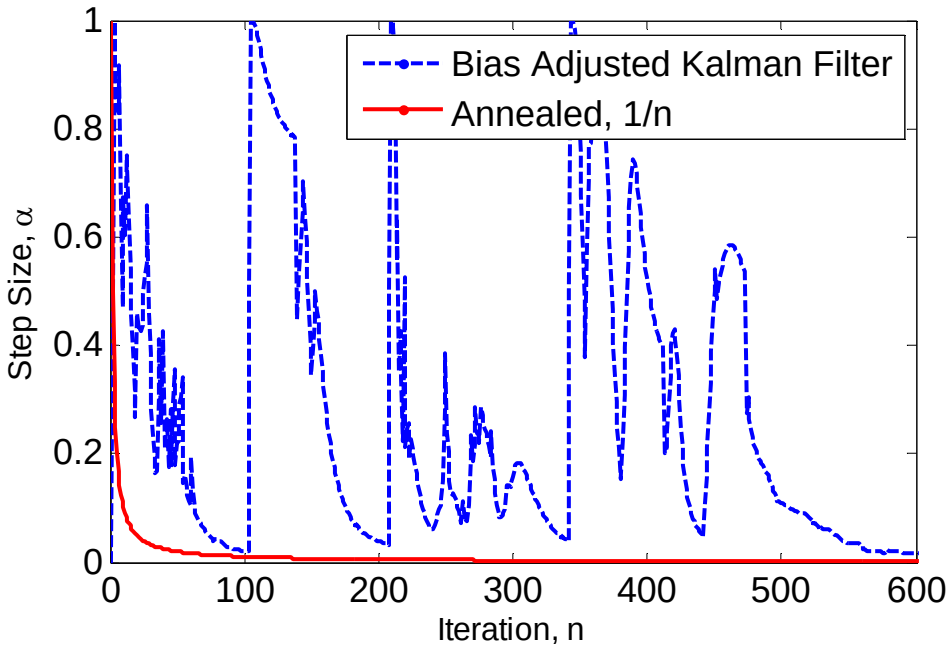


Figure C.2: Comparison between Bias Adjusted Kalman filter and annealed step size rules.

Figure C.2 shows the comparison between step size using BAKF and annealed algorithm. It can be see that with annealing the step size falls to a very small value within small number of iterations and hence will not respond to the non-stationary nature of cost-to-go data. On the other hand, BAKF tries to regulate the step size based on the cost-to-go data.

REFERENCES

REFERENCES

[1] EIA, , "International Energy Outlook 2010," :MRN:DOE/EIA-0484(2010), U.S. Energy Information Administration, , 2010.

[2] EIA, , "Annual Energy Review 2009," :MRN:DOE/EIA-0384(2009), U.S. Energy Information Administration, , 2010.

[3] EIA, , "Annual Energy Outlook 2011 with Projections to 2035," :MRN:DOE/EIA-0383(2011), U.S. Energy Information Administration, , 2010.

[4] EPA, , "EPA and NHTSA to Propose Greenhouse Gas and Fuel Efficiency Standards for Heavy-Duty Trucks; Begin Process for Further Light-Duty Standards: Regulatory Announcement," Regulatory Announcement :MRN:EPA-420-F-10-038, Environmental Protection Agency, , 2010.

[5] Santini, D. , "Comparison of Cost Effectiveness of (Some of the Many Possible) Hybrid Configurations," The 20th International Electric Vehicle Symposium and Exposition, 2003.

[6] Wu, B. , Lin, C. C., Filipi, Z. , Peng, H. , and Assanis, D. , "Optimal power management for a hydraulic hybrid delivery truck," *Vehicle System Dynamics*, **42**(1-2), pp. 23-40, 2004.

[7] Filipi, Z. , Louca, L. , Daran, B. , Lin, C.-C. , Yildir, U. , Wu, B. , Kokkolaras, M. , Assanis, D. , Peng, H. , Papalambros, P. , Stein, J. , Szkubiel, D. , and Chapp, R. , "Combined optimisation of design and power management of the hydraulic hybrid propulsion system for the 6 X 6 medium truck," *International Journal of Heavy Vehicle Systems*, **11**(3-4), pp. 372-402, 2004.

[8] Kim, Y. , "Integrated modeling and hardware-in-the-loop study for systematic evaluation of hydraulic hybrid propulsion options," PhD thesis Mechanical Engineering, University of Michigan, , 2008.

[9] Samulski, Michael J. and Jackson, Cleophas C., "Effects of Steady-State and Transient Operation on Exhaust Emissions from Nonroad and Highway Diesel Engines," SAE Technical Paper 982044, 1998.

[10] Filipi, Z. , Hagena, J. , and Fathy, H. , "Investigating the impact of in-vehicle transients on diesel soot emissions," *Thermal Science*, **12**(1), pp. 53-72, 2008.

[11] Hagena, J. R., Filipi, Z. S., and Assanis, D. N., "Transient Diesel Emissions: Analysis of Engine Operation During a Tip-In," SAE Technical Paper 2006-01-

1151, 2006.

[12] Kirchen, P. , Obrecht, P. , and Boulouchos, K. , "Soot Emission Measurements and Validation of a Mean Value Soot Model for Common-Rail Diesel Engines during Transient Operation," *SAE Int. J. Engines*, **2**(2009-01-1904), pp. 1663-1678, 2009.

[13] Rakopoulos, C.D. , Dimaratos, A.M. , Giakoumis, E.G. , and Rakopoulos, D.C. , "Evaluation of the effect of engine, load and turbocharger parameters on transient emissions of diesel engine," *Energy Conversion \& Management*, **50**(9), pp. 2381-2393, 2009.

[14] Hagena, J. , "An experimental technique for determining cycle-resolved pre-combustion in in-cylinder composition and its application towards the understanding of diesel engine emissions during transient operation," PhD thesis Mechanical Engineering, University of Michigan, , 2008.

[15] Filipi, Z. , Louca, L. , Stefanopoulou, A. , Pukrushpan, J. , Kittirungsi, B. , and Peng, H. , "Fuel cell APU for silent watch and mild electrification of a medium tactical truck," *Journal of Commercial Vehicles*, **113**(2004-01-1477), pp. 1029-1039, 2004.

[16] Jalil, N. , Kheir, N.A. , and Salman, M. , "A rule-based energy management strategy for a series hybrid vehicle," Proceedings of the American Control Conference, **1**, pp. 689-93, 1997.

[17] Caratozzolo, P. , Serra, M. , and Riera, J. , "Energy management strategies for hybrid electric vehicles," IEEE Electric Machines and Drives Conference, **1**, pp. 241-248, 2003.

[18] Liang, C. , Weihua, W. , and Qingnian, W. , "Energy Management Strategy and Parametric Design for Hybrid Electric Military Vehicle," SAE Technical Paper 2003-01-0086, 2003.

[19] Barsali, S. , Miulli, C. , and Possenti, A. , "A control strategy to minimize fuel consumption of series hybrid electric vehicles," *IEEE Transactions on Energy Conversion*, **19**(1), pp. 187-95, 2004.

[20] Barsali, S. , Ceraolo, M. , and Possenti, A. , "Techniques to control the electricity generation in a series hybrid electrical vehicle," *IEEE Transactions on Energy Conversion*, **17**(2), pp. 260-266, 2002.

[21] Wu, P. , Luo, N. , Fronczak, F. J., and Beachley, N. H., "Fuel economy and operating characteristics of a hydropneumatic energy storage automobile," *SAE International*, **94**(851678), 1985.

[22] Kapellen, D. R, Jamzadeh, F. , Frank, A. , and Wang, S. , "Analysis of Energy Storage Concepts for Refuse Collection Trucks," SAE Technical Paper 840056, 1984.

[23] Filipi, Z and Kim, Y J, "Hydraulic Hybrid Propulsion for Heavy Vehicles:

Combining the Simulation and Engine-In-the-Loop Techniques to Maximize the Fuel Economy and Emission Benefits," *Oil \& Gas Science and Technology - Revue de l'Institut Francais du Petrole*, **65**(1), pp. 155-178, 2010.

[24] Tavares, F. , Johri, R. , and Filipi, Z. , "Simulation Study of Advanced Variable Displacement Engine Coupled to Power-Split Hydraulic Hybrid Powertrain," ASME 2009 Internal Combustion Engine Division Spring Technical Conference, **2009**, pp. 463-476, 2009.

[25] Baumann, B. M., Washington, G. , Glenn, B. C., and Rizzoni, G. , "Mechatronic design and control of hybrid electric vehicles," *IEEE/ASME Transactions on Mechatronics*, **5**(1), pp. 58-72, 2000.

[26] Won, J. and Langari, R. , "Fuzzy torque distribution control for a parallel hybrid vehicle," *Expert Systems*, **19**(1), pp. 4-10, 2002.

[27] Phillips, A.M. , Jankovic, M. , and Bailey, K.E. , "Vehicle system controller design for a hybrid electric vehicle," IEEE International Conference on Control Applications, pp. 297-302, 2000.

[28] Pisu, P. and Rizzoni, G. , "A supervisory control strategy for series hybrid electric vehicles with two energy storage systems," 2005 IEEE Vehicle Power and Propulsion Conference, p. 8, 2006.

[29] Paganelli, G. , Tateno, M. , Brahma, A. , Rizzoni, G. , and Guezennec, Y. , "Control development for a hybrid-electric sport-utility vehicle: Strategy, implementation and field test results," Proceedings of the American Control Conference, **6**, pp. 5064-5069, 2001.

[30] Paganelli, G. , Delprat, S. , Guerra, T.M. , Rimaux, J. , and Santin, J.J. , "Equivalent consumption minimization strategy for parallel hybrid powertrains," IEEE 55th Vehicular Technology Conference, **4**, pp. 2076-81, 2002.

[31] Sciarretta, A. , Back, M. , and Guzzella, L. , "Optimal control of parallel hybrid electric vehicles," *IEEE Transactions on Control Systems Technology*, **12**(3), pp. 352-63, 2004.

[32] Kim, N. , Cha, S. , and Peng, H. , "Optimal Control of Hybrid Electric Vehicles Based on Pontryagin's Minimum Principle," *IEEE Transactions on Control Systems Technology*, **PP**(99), pp. 1-9, 2011.

[33] Serrao, L. , Onori, S. , and Rizzoni, G. , "ECMS as a realization of Pontryagin's minimum principle for HEV control," American Control Conference, pp. 3964-3969, 2009.

[34] Borhan, H. A., Vahidi, A. , Phillips, A. M., Kuang, M. L., and Kolmanovsky, I. V., "Predictive energy management of a power-split hybrid electric vehicle," Proc. of American Control Conference, pp. 3970-3976, 2009.

[35] Borhan, H.A. , Zhang, Chen , Vahidi, A. , Phillips, A.M. , Kuang, M.L. , and

Cairano, S. Di, "Nonlinear Model Predictive Control for power-split Hybrid Electric Vehicles," 49th IEEE Conference on Decision and Control, pp. 4890-4895, 2010.

[36] Brahma, A. , Guezennec, Y. , and Rizzoni, G. , "Optimal energy management in series hybrid electric vehicles," Proceedings of the American Control Conference, **1**, pp. 60-64, 2000.

[37] Sciarretta, A. , Guzzella, L. , and Onder, C.H. , "On the power split control of parallel hybrid vehicles: from global optimization towards real-time control," *Automatisierungstechnik*, **51**(5), pp. 195-203, 2003.

[38] Lin, C. C., Peng, H. , Grizzle, J.W. , and Kang, J. , "Power management strategy for a parallel hybrid electric truck," *IEEE Transactions on Control Systems Technology*, **11**(6), pp. 839-49, 2003.

[39] Liu, J. and Peng, H. , "Control optimization for a power-split hybrid vehicle," Proceedings of the 2006 American Control Conference, 2006.

[40] Lin, C. C., Peng, H. , and Grizzle, J.W. , "A stochastic control strategy for hybrid electric vehicles," Proc. of American Control Conference, **5**, pp. 4710-4715, 2004.

[41] Tate, E. D., Grizzle, J. W., and Peng, H. , "Shortest path stochastic control for hybrid electric vehicles," *International Journal of Robust and Nonlinear Control*, **18**(14), pp. 1409-1429, 2008.

[42] Liu, J. , Hagena, J. , Peng, H. , and Filipi, Z.S. , "Engine-in-the-loop study of the stochastic dynamic programming optimal control design for a hybrid electric HMMWV," *International Journal of Heavy Vehicle Systems*, **15**(2-4), pp. 309-26, 2008.

[43] Johri, R. and Filipi, Z. , "Low-Cost Pathway to Ultra Efficient City Car: Series Hydraulic Hybrid System with Optimized Supervisory Control," *SAE International Journal of Engines*, **2**(2), pp. 505-520, 2010.

[44] Kepner, R. P., "Hydraulic power assist - A demonstration of hydraulic hybrid vehicle regenerative braking in a road vehicle application," SAE Technical Paper 2002-01-3128, 2002.

[45] Gray, D. L. and Hentea, T. I., "Engine emissions modeling for a hybrid electric vehicle," Proceedings of the Intersociety Energy Conversion Engineering Conference, pp. 745-750, 2002.

[46] Kleimaier, A. and Schroder, D. , "Optimal control and emission behaviour of a hybrid powertrain," *VDI Berichte*, **1709**(1709), pp. 603-621, 2002.

[47] Lindenkamp, N. , Stöber-Schmidt, C. , and Eilts, P. , "Strategies for Reducing NOX- and Particulate Matter Emissions in Diesel Hybrid Electric Vehicles," SAE Technical Paper 2009-01-1305, 2009.

[48] Tate, E.D. , Grizzle, J.W. , and Peng, H. , "SP-SDP for Fuel Consumption and Tailpipe Emissions Minimization in an EVT Hybrid," *IEEE Transactions on Control Systems Technology*, **18**(3), pp. 673-87, 2010.

[49] Kum, D. , "Modeling and optimal control of parallel HEVs and plug-in HEVs for multiple objectives," PhD thesis Mechanical Engineering, University of Michigan, , 2010.

[50] Hiroyuki, H. , Toshikazu, K. , and Masataka, A. , "Development and Use of a Spray Combustion Modeling to Predict Diesel Engine Efficiency and Pollutant Emissions : Part 1 Combustion Modeling," *Bulletin of JSME*, **26**(214), pp. 569-575, 1983.

[51] Bayer, J. and Foster, D. E., "Zero-Dimensional Soot Modeling," SAE Technical Paper 2003-01-1070, 2003.

[52] Khan, I. M., Greeves, G. , and T., C. H., "Factors Affecting Smoke and Gaseous Emissions from Direct Injection Engines and a Method of Calculation," SAE Technical Paper 730169, 1973.

[53] Ericson, C. , Westerberg, B. , Andersson, M. , and Egnell, R. , "Modelling Diesel Engine Combustion and NOx Formation for Model Based Control and Simulation of Engine and Exhaust Aftertreatment Systems," SAE Technical Paper 2006-01-0687, 2006.

[54] Andersson, M. , Johansson, B. , Hultqvist, A. , and Nohre, C. , "A Real Time NOx Model for Conventional and Partially Premixed Diesel Combustion," SAE Technical Paper 2006-01-0195, 2006.

[55] Lyons, C. M. and Timoney, D. J., "Sensitivity of a DI Diesel NOx Model to Changes in Common Rail Pressure, Injection Patterns and EGR," SAE Technical Paper 2005-01-2118, 2005.

[56] Seykens, X. , Baert, R. , Somers, B. , and Willems, F. , "Experimental Validation of Extended NO and Soot Model for Advanced HD Diesel Engine Combustion," SAE Technical Paper 2009-01-0683, 2009.

[57] Pitsch, H. , Barths, H. , and Peters, N. , "Three-dimensional modeling of NOx and soot formation in DI-diesel engines using detailed chemistry based on the interactive flamelet approach," SAE Technical Paper 962057, 1996.

[58] Bazari, Z. , "Diesel exhaust emissions prediction under transient operating conditions," SAE Technical Paper 940666, 1994.

[59] Jung, D. and Assanis, D. N., "Multi-Zone DI Diesel Spray Combustion Model for Cycle Simulation Studies of Engine Performance and Emissions," SAE Technical Paper 2001-01-1246, 2001.

[60] Patterson, M. A., Kong, S.-C. , Hampson, G. J., and Reitz, R. D., "Modeling the Effects of Fuel Injection Characteristics on Diesel Engine Soot and NOx

Emissions," SAE Technical Paper 940523, 1994.

[61] Liu, Y. , Tao, F. , Foster, D. E., and Reitz, R. D., "Application of A Multiple-Step Phenomenological Soot Model to HSDI Diesel Multiple Injection Modeling," SAE Technical Paper 2005-01-0924, 2005.

[62] Tao, F. , Liu, Y. , RempelEwert, B. H., Foster, D. E., Reitz, R. D., Choi, D. , and Miles, P. C., "Modeling the Effects of EGR and Injection Pressure on Soot Formation in a High-Speed Direct-Injection (HSDI) Diesel Engine Using a Multi-Step Phenomenological Soot Model," SAE Technical Paper 2005-01-0121, 2005.

[63] Eastwood, P.G. , Tufail, K. , Winstanley, T. , Darlington, A. , Karagiorgis, S. , Hardalupas, Y. , and Taylor, A.M.K.P. , "Estimation of deviations in no and soot emissions between steady-state and eudc transient operation of a common-rail diesel engine," *SAE International Journal of Engines*, **2**(2), pp. 648-659, 2010.

[64] Giakoumis, E. G. and Alafouzos, A. I., "Comparative study of turbocharged diesel engine emissions during three different Transient Cycles," *International Journal of Energy Research*, **34**(11), pp. 1002-1015, 2010.

[65] Schilling, A. , Amstutz, A. , Onder, C.H. , and Guzzella, L. , "A real-time model for the prediction of the NOx emissions in DI diesel engines," Proceedings of the 2006 IEEE International Conference on Control Applications, p. 6, 2006.

[66] Kirchen, P. , "Steady-State and Transient Diesel Soot Emissions: Development of a Mean Value Soot Model and Exhaust-Stream and In-Cylinder Measurements," PhD thesis ETH Zurich, , 2008.

[67] Brahma, A. , Upadhyay, D. , Serrani, A. , and Rizzoni, G. , "Modeling, identification and state estimation of diesel engine torque and NOx dynamics in response to fuel quantity and timing excitations," Proceeding of the 2004 American Control Conference, **vol.3**, pp. 2166-71, 2004.

[68] Warth, M. , Obrecht, P. , Bertola, A. , and Boulouchos, K. , "Predictive Phenomenological C.I. Combustion Modeling Optimization on the Basis of Bio-Inspired Algorithms," SAE Technical Paper 2005-01-1119, 2005.

[69] Brahma, I. , Rutland, C. J., Foster, D. E., and He, Y. , "New Approach to System Level Soot Modeling," SAE Technical Paper 2005-01-1122, 2005.

[70] Ouladsine, M. , Bloch, G. , and Dovifaaz, X. , "Neural modelling and control of a Diesel engine with pollution constraints," *Journal of Intelligent \& Robotic Systems*, **41**, pp. 157-171, 2005.

[71] Krijnsen, H. C., J., W. E., A., H. P., Verbeek, R. P., and v., C. M., "Prediction of NOx Emissions from a Transiently Operating Diesel Engine Using an Artificial Neural Network," *Chemical Engineering \& Technology*, **22**(7), pp. 601-607, 1999.

[72] Tóth-Nagy, C. , Conley, J. J., Jarrett, R. P., and Clark, N. N., "Further validation of artificial neural network-based emissions simulation models for conventional and

hybrid electric vehicles," *Journal of the Air and Waste Management Association*, **56**(7), pp. 898-910, 2006.

[73] Wang, Y.-Y. , He, Y. , and Rajagopalan, S. , "Design of Engine-Out Virtual NOx Sensor Using Neural Networks and Dynamic System Identification," SAE Technical Paper 2011-01-0694, 2011.

[74] Alberer, D. , Re, L. del, Winkler, S. , and Langthaler, P. , "Virtual Sensor Design of Particulate and Nitric Oxide Emissions in a DI Diesel Engine," SAE Technical Paper 2005-24-063, 2005.

[75] Re, L. del, Langthaler, P. , Furtmueller, C. , Winkler, S. , and Affenzeller, M. , "NOx Virtual Sensor Based on Structure Identification and Global Optimization," SAE Technical Paper 2005-01-0050, 2005.

[76] Johri, R. and Filipi, Z. , "Self-Learning Neural controller for Hybrid Power Management using Neuro-Dynamic Programming," SAE Technical Paper 2011-24-0081, 2011.

[77] Johri, R. , Salvi, A. , and Filipi, Z. , "Optimal Energy Management for a Hybrid Vehicle Using Neuro-Dynamic Programming to Consider Transient Engine Operation," Proc. of 4th Annual Dynamic Systems and Control Conference, 2011.

[78] Johri, R. , Salvi, A. , and Filipi, Z. , "Real-Time Transient Soot and NOx Virtual Sensors for Diesel Engine using Neuro-Fuzzy Model Tree and Orthogonal Least Squares," Proceedings of the ASME 2011 Internal Combustion Engine Division Fall Technical Conference, 2011.

[79] Johri, R. , Baseley, S. , and Filipi, Z , "Simultaneous Optimization Of Supervisory Control And Gear Shift Logic For A Parallel Hydraulic Hybrid Refuse Truck Using Stochastic Dynamic Programming," Proc. of 4th Annual Dynamic Systems and Control Conference, 2011.

[80] Tavares, F. , Johri, R. , Salvi, A. , Baseley, S. , and Filipi, Z. S., "Hydraulic Hybrid Powertrain-In-the-Loop Integration for Analyzing Real-World Fuel Economy and Emissions Improvements," SAE Technical Paper 2011-01-2275, 2011.

[81] Tavares, F. , Johri, R. , and Filipi, Z. , "Simulation Study of Advanced Variable Displacement Engine Coupled to Power-Split Hydraulic Hybrid Powertrain," *accepted at ASME Journal of Engineering for Gas Turbines and Power*, 2011.

[82] Matheson, P. and Stecki, J. , "Modeling and Simulation of a Fuzzy Logic Controller for a Hydraulic-Hybrid Powertrain for Use in Heavy Commercial Vehicles," SAE Technical Paper 2003-01-3275, 2003.

[83] EPA, , "World's First Full-Size Hydraulic Hybrid SUV," :MRN:EPA420-F-04-019, Environmental Protection Agency, , 2004.

[84] Williamson, S. S., Emadi, A. , and Rajashekara, K. , "Comprehensive efficiency modeling of electric traction motor drives for hybrid electric vehicle propulsion

applications," *IEEE Transactions on Vehicular Technology*, **56**(4 I), pp. 1561-1572, 2007.

[85] Backe, W. , "Present and future of fluid power," *Proceedings of the Institution of Mechanical Engineers. Part I, Journal of systems and control engineering*, **207**(4), pp. 193-212, 1993.

[86] Kim, Y. and Filipi, Z. , "Simulation Study of a Series Hydraulic Hybrid Propulsion System for a Light Truck," *SAE Transactions, Journal of Commercial Vehicles*, **116**(2007-01-4151), pp. 147-161, 2007.

[87] Assanis, D. , Filipi, Z. , Gravante, S. , Grohnke, D. , Gui, X. , Louca, L. , Rideout, G. , Stein, J. , and Wang, Y. , "Validation and Use of SIMULINK Integrated, High Fidelity, Engine-In-Vehicle Simulation of the International Class VI Truck," *SAE Transactions: Journal of Engines*, **109**(2000-01-0288), 2000.

[88] Fathy, H.K. , Filipi, Z.S. , Hagena, J. , and Stein, J.L. , "Review of hardware-in-the-loop simulation and its prospects in the automotive area," *Proceedings of the SPIE - The International Society for Optical Engineering*, **6228**(1), pp. 117-136, 2006.

[89] Pourmovahed, A. , Beachley, N.H. , and Fronczak, F.J. , "Modeling of a hydraulic energy regeneration system. Part I. Analytical treatment," *Journal of Dynamic Systems, Measurement and Control, Transactions of the ASME*, **114**(1), pp. 155-159, 1992.

[90] Alson, J. , Barba, D. , Bryson, J. , Doorag, M. , Haugen, D. , Kargul, J. , McDonald, J. , Newman, K. , Platte, L. , and Wolcott, M. , "Progress Report On Clean And Efficient Automotive Technologies Under Development At Epa," :MRN:EPA420-R-04-002, Environmental Protection Agency, , 2004.

[91] Pourmovahed, A. and Otis, D.R. , "Experimental thermal time-constant correlation for hydraulic accumulators," *Journal of Dynamic Systems, Measurement and Control, Transactions of the ASME*, **112**(1), pp. 116-121, 1990.

[92] Pourmovahed, A. , Beachley, N.H. , and Fronczak, F.J. , "Modeling of a hydraulic energy regeneration system. Part II. Experimental program," *Journal of Dynamic Systems, Measurement and Control, Transactions of the ASME*, **114**(1), pp. 160-165, 1992.

[93] Pourmovahed, A. , Baum, S.A. , Fronczak, F.J. , and Beachley, N.H. , "Experimental Evaluation Of Hydraulic Accumulator Efficiency With And Without Elastomeric Foam.," *Journal of Propulsion and Power*, **4**(2), pp. 185-192, 1988.

[94] Tartt, C. Jason and Moskwa, J. J., "A hardware-in-the-loop transient diesel engine test system for control and diagnostic development," ASME Dynamic Systems and Control Division, **70**, pp. 255-261, 2002.

[95] Nabi, S. , Balike, M. , Allen, J. , and Rzemien, K. , "An overview of hardware-in-the-loop testing systems at Visteon," SAE Technical Paper 2004-01-1240, 2004.

[96] Pasquier, M. and Monnet, G. , "Diesel hybridization and emissions.," Technical Report :MRN:ANL/ES/RP-113184, Argonne National Lab., Argonne, IL (US), , 2004.

[97] Filipi, Z. , Fathy, H. , Hagena, J. , Knafl, A. , Ahlawat, R. , Liu, J. , Jung, D. , Assanis, D. , Peng, H. , and Stein, J. , "Engine-in-the-Loop Testing for Evaluating Hybrid Propulsion Concepts and Transient Emissions – HMMWV Case Study," *SAE Transactions, Journal of Commercial Vehicles*, **115**(2006-01-0443), 2006.

[98] Simon, M. , Compere, M. , Connolly, T. , Lors, C. , Smith, W. , and Brudnak, M. , "Hybrid Electric Power and Energy Laboratory Hardware-in-the-Loop and Vehicle Model Implementation," SAE Technical Paper 2006-01-1162, 2006.

[99] Moura, S. J., Callaway, D. S., Fathy, H. K., and Stein, J. L., "Tradeoffs between battery energy capacity and stochastic optimal power management in plug-in hybrid electric vehicles," *Journal of Power Sources*, **195**(9), pp. 2979-2988, 2010.

[100] Kim, Y. and Filipi, Z. , "Series Hydraulic Hybrid Propulsion for a Light Truck – Optimizing the Thermostatic Power Management," *SAE Transactions, Journal of Engines*, **116-3**(2007-24-0080), pp. 1597-1609, 2007.

[101] Rahman, Z. , Butler, K. L., and Ehsani, M. , "A Comparison Study Between Two Parallel Hybrid Control Concepts," SAE Technical Paper 2000-01-0994, 2000.

[102] Wipke, K. , Markel, T. , and Nelson, D. , "Optimizing Energy Management Strategy and Degree of Hybridization for a Hydrogen Fuel Cell SUV," EVS 18 Berlin, 2001.

[103] Farrall, S.D. and Jones, R.P. , "Energy management in an automotive electric/heat engine hybrid powertrain using fuzzy decision making," Proceedings of the 1993 IEEE International Symposium on Intelligent Control, pp. 463-468, 1993.

[104] Sciarretta, A. and Guzzella, L. , "Control of hybrid electric vehicles," *IEEE Control Systems Magazine*, **27**(2), pp. 60-70, 2007.

[105] Lee, T.-K. and Filipi, Z. S., "Synthesis and validation of representative real-world driving cycles for Plug-In Hybrid vehicles," Vehicle Power and Propulsion Conference, pp. 1-6, 2010.

[106] Adornato, B. , Patil, R. , Filipi, Z. , Baraket, Z. , and Gordon, T. , "Characterizing Naturalistic Driving Patterns For Plug-in Hybrid Electric Vehicle Analysis," Proc. IEEE Vehicle Power and Propulsion Conference VPPC '09, pp. 655-660, 2009.

[107] Powell, W. B., Approximate dynamic programming: solving the curses of dimensionality, Wiley-Interscience., 2007.

[108] Sutton, R. S. and Barto, A. G., Reinforcement learning an introduction, MIT Press., 1998.

[109] Bertsekas, D. P. and Tsitsiklis, J. N., Neuro-dynamic programming, Athena

Scientific., 1996.

[110] Tsitsiklis, J.N. and Roy, B. van, "Feature-based methods for large scale dynamic programming," *Machine Learning*, **22**(1-3), pp. 59-94, 1996.

[111] Bertsekas, D. P., Homer, M. L., Logan, D. A., Patek, S. D., and Sandell, N. R., "Missile defense and interceptor allocation by neuro-dynamic programming," *IEEE Transactions on Systems, Man, and Cybernetics Part A:Systems and Humans.*, **30**(1), pp. 42-51, 2000.

[112] Cybenko, G. , "Approximation by superpositions of a sigmoidal function," *Mathematics of Control, Signals, and Systems*, **2**(4), pp. 303-14, 1989.

[113] Tesauro, G. , "Practical issues in temporal difference learning," *Machine Learning*, **8**(3-4), pp. 257-77, 1992.

[114] Goodwin, G.C. , "Evaluating the performance of virtual sensors," Proc. of IEEE Information, Decision and Control, pp. 5-12, 1999.

[115] Sevilla, J. and Pulido, C. , "Virtual industrial sensors trough neural networks. Demonstration examples in nuclear power plants," IEEE Instrumentation and Measurement Technology Conference, **1**, pp. 293-297, 1998.

[116] Dec, J. E., "A Conceptual Model of DI Diesel Combustion Based on Laser-Sheet Imaging*," *SAE Transactions, Journal of Engines*, **106**(970873), 1997.

[117] Galindo, J. , Bermudez, V. , Serrano, J.R. , and Lopez, J. J., "Cycle-to-cycle diesel combustion characterization during engine transient operation," *SAE Transactions, Journal of Engines*, **110**(2001-01-3262), 2001.

[118] Heywood, J. B., Internal combustion engine fundamentals, McGraw-Hill., 1988.

[119] Hilliard, J. C. and Wheeler, R. W., "Nitrogen dioxide in engine exhaust," SAE Technical Paper 790691, 1979.

[120] Haynes, B.S. and Wagner, H. Gg., "Soot formation," *Progress in Energy and Combustion Science*, **7**(4), pp. 229-273, 1981.

[121] Amann, C. A. and Siegla, D. C., "Diesel Particulates - What They Are And Why.," *Aerosol Science and Technology*, **1**(1), pp. 73-101, 1982.

[122] Ljung, L. , System identification : theory for the user, Prentice-Hall., 1987.

[123] Zhu, Y. , Multivariable system identification for process control, Pergamon., 2001.

[124] Godfrey, K. , Perturbation signals for system identification, Prentice Hall., 1993.

[125] Braun, M.W. , Rivera, D.E. , Stenman, A. , Foslien, W. , and Hrenya, C. , "Multi-level pseudo-random signal design and model-on-demand estimation applied to nonlinear identification of a RTP wafer reactor," Proceedings of the American Control Conference, **3**, pp. 1573-7, 1999.

[126] Johansen, T. A. and Foss, B. A., "Identification of non-linear system structure and parameters using regime decomposition," *Automatica*, **31**(2), pp. 321-326, 1995.

[127] Murray-Smith, R. and Johansen, T.Arne , "Local learning in Local Model Networks," IEE Conference on Artificial Neural Networks, pp. 40-46, 1995.

[128] Johansen, T. A. and Foss, B. A., "Operating regime based process modeling and identification," *Computers \& Chemical Engineering*, **21**(2), pp. 159-176, 1997.

[129] Ragot, J. , Mourot, G. , and Maquin, D. , "Parameter estimation of switching piecewise linear system," Proceedings of 42nd IEEE Conference Decision and Control, **6**, pp. 5783-5788, 2003.

[130] Nelles, O. , Nonlinear system identification: from classical approaches to neural networks and fuzzy models, Springer., 2001.

[131] MacKay, D.J.C. , "Bayesian methods for adaptive models," PhD thesis California Institute of Technology, , 1991.

[132] MacKay, D.J.C. , "Bayesian interpolation," *Neural Computation*, **4**(3), pp. 415-47, 1992.

[133] Baesens, B. , Viaene, S. , den, D. Van, Vanthienen, J. , and Dedene, G. , "Bayesian neural network learning for repeat purchase modelling in direct marketing," *European Journal of Operational Research*, **138**(1), pp. 191-211, 2002.

[134] Chen, S. , Billings, S.A. , and Luo, W. , "Orthogonal least squares methods and their application to non-linear system identification," *International Journal of Control*, **50**(5), pp. 1873-1896, 1989.

[135] Chen, S. and Wigger, J. , "Fast orthogonal least squares algorithm for efficient subset model selection," *IEEE Transactions on Signal Processing*, **43**(7), pp. 1713-1715, 1995.

[136] MacKay, D.J.C. , "Bayesian nonlinear modeling for the prediction competition," *Ashrae Transactions*, **100**(2), pp. 1053-1062, 1994.

[137] Foresee, F. Dan and Hagan, M.T. , "Gauss-Newton approximation to Bayesian learning," International Conference on Neural Networks,1997., **3**, pp. 1930--1935 vol.3, 1997.

[138] Papalambros, P. Y. and Wilde, D. J., Principles of optimal design: modeling and computation, Cambridge University Press., 2000.

[139] H., M. Hagan, "Neural Network Toolbox™ 6," , 2008.

[140] MacKay, D.J.C. , "A practical Bayesian framework for backpropagation networks," *Neural Computation*, **4**(3), pp. 448-72, 1992.

[141] Bashash, S. , Moura, S. J., Forman, J. C., and Fathy, H. K., "Plug-in hybrid electric vehicle charge pattern optimization for energy cost and battery longevity," *Journal*

of Power Sources, **196**(1), pp. 541-549, 2011.

[142] Moura, S. J., "Techniques for Battery Health Conscious Power Management via Electrochemical Modeling and Optimal Control," PhD thesis Univeristy of Michigan, , 2011.

[143] Opila, D.F. , Aswani, D. , McGee, R. , Cook, J.A. , and Grizzle, J.W. , "Incorporating drivability metrics into optimal energy management strategies for Hybrid Vehicles," 47th IEEE Conference on Decision and Control, pp. 4382-4389, 2008.

[144] Opila, D. , "Incorporating Drivability Metrics into Optimal Energy Management Strategies for Hybrid Vehicles," PhD thesis Univeristy of Michigan, , 2010.

[145] Park, S. and Jung, D. , "Design of Vehicle Cooling System Architecture for a Heavy Duty Series-Hybrid Electric Vehicle Using Numerical System Simulations," *Journal of Engineering for Gas Turbines and Power*, **132**(9), p. 092802, 2010.

[146] Bichi, M. , Ripaccioli, G. , Cairano, S. Di, Bernardini, D. , Bemporad, A. , and Kolmanovsky, I.V. , "Stochastic model predictive control with driver behavior learning for improved powertrain control," 49th IEEE Conference on Decision and Control (CDC), pp. 6077-6082, 2010.

[147] Trick, M. A. and Zin, S. E., "A Linear Programming Approach to Solving Stochastic Dynamic Programming," Tepper School of Business. Paper 517., 1993.

[148] Symonds, J. P.R., Reavell, K. St.J., Olfert, J. S., Campbell, B. W., and Swift, S. J., "Diesel soot mass calculation in real-time with a differential mobility spectrometer," *Journal of Aerosol Science*, **38**(1), pp. 52-68, 2007.

[149] Saad, D. , On-line learning in neural networks, Cambridge University Press., 1998.

[150] George, A. and Powell, W. , "Adaptive stepsizes for recursive estimation with applications in approximate dynamic programming," *Machine Learning*, **65**, pp. 167-198, 2006.

[151] Reavell, K. , Hands, T. , and Collings, N. , "A Fast Response Particulate Spectrometer for Combustion Aerosols," SAE Technical Paper 2002-01-2714, 2002.