Optimal Energy Management in Electric Vehicles

Convex Optimization for Model Predictive Control

A thesis submitted for the degree of

Doctor of Philosophy

Sebastian East

St. John's College, University of Oxford

Trinity Term 2020

This thesis is dedicated to my parents,
Janet & Bernd.

ACKNOWLEDGEMENTS

Firstly, I would like to extend my deepest gratitude to my supervisor, Professor Mark Cannon, whose guidance and support has been absolutely phenomenal. Mark's dedication to both research and teaching is inspiring, and I am extremely fortunate to have had the opportunity to work with him for four years. I am particularly grateful for the weekly meetings, the helpful and unbelievably detailed feedback on my work, the insightful advice and suggestions, and the rapid completion of countless forms and references. I can't possibly thank him enough for all of the time and effort he has invested in me.

More broadly, I would like to thank the Control Group for providing a friendly and positive research environment throughout my doctoral studies; a particular highlight was an unforgettable couple of months spent developing a ventilator at the start of the coronavirus outbreak. I would also like to thank the engineering fellows at St. John's College for my teaching opportunities, Marco Gallieri and the team at NNAISENSE for an incredible summer in Lugano, and Professor Luigi Del Re and Dr. Philipp Polterauer from Johannes Kepler University Linz for generously sharing the driver behaviour data used throughout this thesis.

My time in Oxford wouldn't have been the same without the MCR community at St. John's College; thank you to everyone who has been a part of it for making Oxford feel like home. I would particularly like to thank the following for their contributions to the successful completion of my doctoral studies: Ed, Angelica, Kelli, and Cameron for several years of helpful (and entertaining) discussions over lunch, and Tunrayo for keeping me motivated whilst writing this thesis during lockdown. I am also very grateful to Abby for all of the weekends in and out of Oxford, some amazing holidays, and just being great at taking my mind off of work.

Finally, I am forever grateful to my parents, Janet and Bernd, for the years of loving support and dedication to my academic development, without whom I would never have made it to Oxford.

ABSTRACT

Increased electrification of road vehicles has been identified as an important method of mitigating air pollution and climate change, and the past ten years has seen a paradigm shift in the automotive market towards hybrid, plug-in hybrid, and all-electric vehicles. Electric vehicle battery technology currently suffers from several limitations relating to cost, lifespan, and energy density, so a common approach to electric powertrain design is a hybrid system in which an additional power or energy source is included to reduce the battery's shortcomings. The performance of a hybrid powertrain is strongly dependent on the control system used to determine the power delivered from each source. Therefore the design of control systems that can balance the power demand in a way that is optimal (in the sense of minimizing fuel consumption, for example) is of great interest to both academia and industry. Model predictive control (MPC) is a strong candidate for this problem as it explicitly considers predictions of future driver and vehicle behaviour together with hard constraints on system performance. However MPC requires that a potentially challenging constrained optimization problem is solved repeatedly throughout a given journey. This introduces a trade off between the closed loop performance of the controller and the tractability of the associated MPC optimization problem.

This thesis presents an investigation into the use of convex optimization for model predictive energy management in electric vehicles. The aim is to generate convex formulations that permit more general descriptions of system dynamics than linear quadratic MPC, whilst still ensuring that the resulting optimization problem can be readily solved online using limited hardware. A particular contribution relative to previous work in the area is the development of optimization algorithms that are specifically tailored to the structure of the resulting convex optimization problem. The major part of the thesis is focussed on energy management in plug-in hybrid electric vehicles (where the objective is to minimize fuel consumption over a given journey), although the methods are extended to all-electric vehicles with hybrid energy storage systems consisting of both conventional batteries and supercapacitors (in this case the objective is to minimize energy consumption and battery degradation). The MPC optimization problem for plug-in hybrid electric vehicles is first formulated considering only the terminal constraint on the battery state of charge, then progressively extended to consider the general state-of-charge constraints, optimal engine switching, and driver behaviour uncertainty. In each case, the performance of the MPC formulation and optimization algorithm is demonstrated in numerical studies, and it is shown the the controllers obtain a highly accurate approximation of the globally optimal solution, whilst dramatically reducing the computational requirement relative to the current state-of-the-art.

NOMENCLATURE

Definitions, Equality, & Inequality

$A := B$	A is defined as equal to B
$A :\leftarrow C$	A is redefined as equal to C
$A =: B$	B is defined as equal to A
$A \in B$	A is an element of B
$A \approx B$	A is approximately equal to B
$A > (\geq) B$	A is element-wise greater than (or equal to) B
$A < (\leq) B$	A is element-wise less than (or equal to) B
$A \,(\succeq) \succ B$	$A - B$ is positive (semi)definite
$f : A \mapsto B$	f is a function that maps A onto B

Number Sets

$\mathbb{Z}$	the integers	
$\mathbb{Z}_{++}$	the positive integers	
$\{a, \ldots, b\}$	the interval of integers between a and b	$\{x \in \mathbb{Z} : a \leq x \leq b\}$
$\mathbb{R}$	the real numbers	
$\bar{\mathbb{R}}$	the extended real numbers	
$\mathbb{R}_+$ ($\mathbb{R}_{++}$)	the non-negative (positive) real numbers	
$\mathbb{R}_-$ ($\mathbb{R}_{--}$)	the non-positive (negative) real numbers	
(a, b)	the open interval between a and b	$\{x \in \bar{\mathbb{R}} : a < x < b\}$
$[a, b]$	the closed interval between a and b	$\{x \in \bar{\mathbb{R}} : a \leq x \leq b\}$
$\mathbb{R}^n$	the set of n dimensional real vectors	

$\mathbb{R}^{m\times n}$	the set of m-by-n dimensional real matrices

Set operations

$A \cap B$	the intersection of A and B
$A \cup B$	the union of A and B
$A \setminus B$	the difference set of A and B
$A \oplus B$	the Minkowski sum of A and B

Vectors & Matrices

$\mathbf{0}$	a vector of zeros
$\mathbf{1}$	a vector of ones
O	a matrix of zeros
I	the identity matrix
Ψ	a lower triangular matrix of ones

Throughout this thesis, the dimensions of the vectors and matrices defined above can be inferred from the context in which they are used.

Vector Operations

x_i	the ith element of vector x	
$x^\top$	the transpose of x	
$\|x\|$	a vector norm	
$\|x\|_2$	the vector 2-norm	
$\|x\|_Q$	the vector Q-norm $(Q \in \mathbb{R}^{n\times n})$	$\|x\|_Q := \sqrt{x^\top Q x}$
$\text{diag}(x)$	a matrix formed with the elements of x on the diagonal	
$\mathcal{I}_A(\cdot)$	the indicator function of set A	
$\Pi_A(x)$	the Euclidean projection of x onto A	$\Pi_A(x) := \text{argmin}_{\hat{x}\in A}\|x - \hat{x}\|_2$

Matrix Operations

$A_{i,j}$	the element on the ith row and jth column of A
$A_{i,:}$	the ith row of A
$A_{:,j}$	the jth column of A
$A \otimes B$	the Kronecker product of A and B
$\text{vec}\,(A)$	the columns of A stacked into a vector

Calculus

$f'(\cdot)$	the derivative of f
$\nabla f(\cdot)$	the gradient of f
$\nabla_i f(\cdot)$	the ith element of the gradient of f
$\nabla_A f(\cdot)$	the $i \in A$ elements of the gradient of f
$\nabla^2 f(\cdot)$	the Hessian of f
$\partial f(\cdot)$	the subdifferential of f

Logic

$A \Rightarrow B$	A implies B
$A \Leftrightarrow B$	A is equivalent to B
$A \wedge B$	A and B
$A \vee B$	A or B

Algorithms

$\sim$	empty placeholder
$A \leftarrow B$	the value of B is assigned to A

NOMENCLATURE

Common Acronyms

ADMM	Alternating Direction Method of Multipliers
BSEV	Battery/Supercapacitor Electric Vehicle
CDCS	Charge-Depleting/Charge-Sustaining
MPC	Model Predictive Control
PHEV	Plug-in Hybrid Electric Vehicle
PMP	Pontryagin's Minimum Principle
SOC	State-of-Charge

Commonly Used Variables (PHEV Model)

e	engine output power
m	motor output power
d	driver's power demand
b	mechanical braking power
p	powertrain output power
σ	engine on/off state
x	battery state-of-charge
ω_w	wheel rotational speed
ω_e	engine rotational speed
ω_m	motor rotational speed
$f(\cdot)$	engine output-input power mapping
$h(\cdot)$	motor output-input power mapping
$g(\cdot)$	battery output-input power mapping

CONTENTS

CHAPTER 1

INTRODUCTION

INTERNAL combustion engines (ICEs) have been a key technology in the development of modern society and the growth of the global economy for over a century. Innovations in design and manufacturing have led to lightweight, powerful, and cheap engines, that are very popular and found in a huge variety of applications and industries. This success has not been without its drawbacks, however, as ICEs are fuelled almost exclusively with oil derivatives, and can be directly linked to important environmental and societal issues including air pollution and climate change [1, §1].

Increased electrification of vehicles has been identified as a short-term solution to these problems, and improvements in practical and economic viability of electric powertrains have ensured that both hybrid and fully electric vehicles have surged in popularity in the past decade. It is expected that electric vehicles will reach a passenger vehicle market share of 10% by 2024 [2, p. 3]. Whilst all-electric vehicles provide regeneration capabilities, zero tailpipe emissions, and greater energy efficiency than internal combustion vehicles, a significant limitation is the driving range available from a full battery charge, due to the low energy density of the current state-of-the-art in battery technology (lithium-ion, $\sim$360kJ/kg [3, Fig.1]) compared to petrol ($\sim$45,000kJ/kg [4, Appendix. D]). This issue is exacerbated by the additional time required to charge a battery compared to filling a tank with fuel. A further limitation of all-electric vehicles is the cost of the battery-pack, which can currently reach 50% of the total cost of the vehicle [5, §3]. A factor contributing to this is the low cycle-life and power density of lithium-ion batteries ($\sim$2000 cycles and $\leq$2000 W/kg [6, Table B1]), which can cause the battery to be sized above its total energy requirement. That is, a larger battery will reduce the frequency of high and low charge states that increase the rate of battery ageing [7], and a larger battery may be needed to meet the power requirement of the vehicle.

1.1 HYBRID TECHNOLOGY

A method for reducing the limitations of all-electric vehicles is to hybridize the battery and motor with an additional power source and/or energy store. Many alternative energy/power sources have been proposed, including hydrogen fuel-cells, flywheels, and supercapacitors [8], and in a range of configurations. Consequently, a large vocabulary of terms has been generated for specific hybrid architectures that is not always used consistently. Therefore, before proceeding, the terminology used throughout this thesis is defined.

> **Electric vehicle:** A vehicle in which the powertrain includes an electric motor supplied at least in part by a battery, but possibly hybridised with other power sources and/or energy stores. Thus, all of the terms that follow are a subset of electric vehicles.
>
> **All-electric vehicle:** An electric vehicle with *only* electric elements in its powertrain, powered by a conventional battery *only*.
>
> **Plug-in hybrid electric vehicle (PHEV):** An electric vehicle for which the battery can be charged from an external source, and is supplemented with a secondary power source as a range extender. In this thesis, the second power source is always assumed to be an internal combustion engine.
>
> **Battery/supercapacitor electric vehicle (BSEV):** An electric vehicle with an electric motor supplied by a hybrid energy storage system, consisting of a battery and supercapacitor.
>
> **Hybrid electric vehicle:** An electric vehicle for which the battery *cannot* be charged from an external source, and is primarily powered by a non-electric power source (also assumed to be an internal combustion engine).

Each category can be further subdivided, but the above terms are sufficient for a concrete understanding of the following.

The focus of this thesis is on powertrain control for plug-in hybrid electric vehicles (PHEVs) and battery/supercapacitor electric vehicles (BSEVs). The principle of a PHEV is to address the range limitation of the battery by combining an electric system with an internal combustion engine as a range extender. Although this still relies on an internal combustion engine, analysis of daily driving statistics has revealed that 50% of daily driving distance is less that 30 miles [9], so a vehicle with an all-electric range of just 30 miles can displace at least 50% of internal combustion miles (assuming that the battery is charged from an external source each day). The principle of a BSEV is to address the low

cycle life and power density of the battery using a hybrid energy storage system consisting of both a lithium-ion battery pack and a supercapacitor. The idea is that the supercapacitor provides the high frequency, high magnitude content of the driver's power demand, whilst the battery delivers power at a reduced, more constant level. Supercapacitors are ideal for this application as they have a comparatively high power density (up to 23500 kW/kg, compared with 2000 W/kg for lithium-ion [6, Table B1]) and effectively infinite cycle life, but cannot be used for the storage system alone as they they have an energy density even lower than lithium-ion cells (up to 50 Wh/kg, compared with 350 Wh/kg for lithium-ion [6, Table B1]). A BSEV can therefore be interpreted as a compromise between the two technologies.

1.2 Energy Management

In both of the cases described above, the inclusion of an additional power source presents a problem: at each instant during a journey, the driver communicates a single power demand to the vehicle through the position of the throttle/accelerator pedal, but how much of this power should the powertrain deliver from each source?

A simple heuristic for the PHEV problem is to run the car in an all-electric mode until the battery is completely depleted, and then as a hybrid electric vehicle until the end of the journey (known as a charge-depleting/charge-sustaining (CDCS) strategy). It has, however, been demonstrated that by providing power from both simultaneously, and modulating the fraction of power delivered by each throughout the journey (a strategy known as 'blended-mode'), the fuel consumption can be significantly reduced without affecting total battery use [10]. Similarly, a simple method for controlling a BSEV is to use a low-pass filter to separate the frequencies of the demand power, then deliver the low frequency content from the battery and the high frequency content from the supercapacitor. It has also been demonstrated that this approach is sub-optimal (in the sense of minimizing battery use or maximizing efficiency), and that predictive methods that account for future driver behaviour can deliver significant performance improvements [11, §4].

Both problems can therefore be considered as optimal control problems, where the objective is to minimise fuel consumption in the former and minimize battery use in the latter, and both are achieved by controlling the relative fraction of a total power demand delivered by two power sources. This class of optimization problems are known as 'energy management problems'.

1.3 Model Predictive Control & Convex Optimization

It is possible to construct an optimal open-loop controller for a given journey *a priori* by constructing a mathematical model of the vehicle and solving a corresponding optimization problem based on a prediction of the driver's behaviour. A limitation of this approach is that there will always be a discrepancy between the modelled and physical dynamics of the vehicles, and between the predicted and ensuing driver behaviour. If a power split that is calculated this way is implemented during an actual journey, not only will the resulting performance be sub-optimal compared with that predicted by the optimization, but there is no guarantee that the vehicle will operate within hardware limits (e.g the upper and lower limits on state of charge of the battery), which can be dangerous.

A method for addressing these limitations is model predictive control (MPC), where instead of solving the optimization problem once it is solved repeatedly throughout the journey and the power split is updated at each instant using the most recently computed optimal control signal. This creates a feedback mechanism that provides a degree of robustness to modelling and prediction errors, as discrepancies between the model and the actual vehicle dynamics and errors in the predictions of driver behaviour and traffic conditions can be compensated for in future computations. A drawback of this approach is that it introduces challenging computational constraints: the energy management optimization problem is initially challenging to solve, and it must now be solved faster than the control variable update frequency using the vehicle's embedded powertrain control unit.

If the predictive powertrain model used for MPC is linear (or linearised), then the optimization problem reduces to a quadratic program that can be solved quickly and reliably using widely available solvers (e.g. [12]). In reality, the losses in the powertrain are highly nonlinear, and this approach therefore renders the applied control inputs suboptimal. A more general class of problems that are considered 'tractable' are convex optimization problems: those in which the objective function and constraint set are both convex. For these cases a local minimum is also a global minimum, and fast optimization algorithms exist for broad classes of canonical problem structures [13]. In the context of energy management, this approach permits more accurate models of powertrain dynamics than allowed by linear-quadratic MPC, and previous studies have demonstrated that the energy management problem can be formulated as a convex optimization problem with minor model approximation [14]. These studies, however, have typically used general purpose convex optimization software to model and solve the problem (e.g. CVX [15, 16]), which cannot obtain an accurate solution in real time using desktop hardware in simulation (e.g. up to 21.4s were required in [17]), and also cannot generally be used with embedded hard-

ware. To date, there has not been a systematic investigation into optimization algorithms that are specifically tailored to the structure of convex energy management problems.

1.4 Contributions & Outline

This thesis presents an investigation into convex formulations and optimization algorithms for predictive energy management in electric vehicles. The aim is to develop MPC frameworks that can consider detailed powertrain models and long, descriptive predictions, whilst also being readily solvable using an embedded powertrain controller. Several formulations are presented, in which varied levels of control authority and hardware constraints are considered, and the computational performance of each is demonstrated in numerical simulations.

The outline of this thesis is as follows: Chapters 2-6 are concerned with energy management in PHEVs, and propose a set of MPC frameworks in which the complexity of the model and constraints are incrementally increased. Chapter 7 is a standalone chapter concerned with energy management in BSEVs. The individual contributions of each chapter are described in more detail below.

Chapter 2: The Energy Management Problem

An illustrative formulation of the PHEV energy management problem is introduced, and common methods for optimization-based powertrain control are then formulated mathematically with a discussion of the relevant literature.

Chapter 3: Terminal State Constraint

The PHEV powertrain model used for Chapters 3-6 is presented, and a simple MPC controller is proposed where the battery is only subject to a terminal state-of-charge constraint. It is demonstrated that if the MPC problem is reformulated in terms of battery power it is convex; a key result used throughout the rest of the PHEV energy management chapters. A projected Newton method is proposed for the solution of the MPC optimization problem, and convergence results are presented. The MPC algorithm is simulated in closed-loop where it is demonstrated that the lack of general state-of-charge constraint is a fundamental limitation when the road is not flat.

A similar version of the proof of convexity presented in this chapter was published in [18], and the projected Newton method was previously published in [19].

CHAPTER 4: GENERAL STATE CONSTRAINT

The MPC controller proposed in Chapter 3 is extended to consider upper and lower bounds on the battery's state of charge throughout the journey. Two optimization algorithms are proposed for its solution: a first-order alternating direction method of multipliers (ADMM) algorithm and a second-order projected interior-point algorithm. The performance of the two algorithms is investigated in a set of numerical studies where it is demonstrated that the ADMM algorithm can obtain a sufficiently accurate solution to the MPC optimization problem in a fraction of a second, even with horizons of up to 1000 samples. The MPC controller is then simulated in closed loop where it is shown to satisfy the state of charge constraints at all times, and provide a close approximation of the globally optimal controls obtained using DP. It is also, however, demonstrated that if the engine switching controller is sufficiently suboptimal, then the optimization based controller may perform worse than a CDCS strategy.

The ADMM algorithm presented in this chapter was originally published in [18], and both the projected interior point algorithm and a subset of the numerical experiments were published in [20].

CHAPTER 5: ENGINE SWITCHING

The MPC controller proposed in Chapter 4 is extended to consider engine switching to improve its performance relative to the CDCS strategy. An ADMM algorithm is proposed as a fast heuristic for the MPC optimization problem, which is necessarily nonconvex due to the integer decision variable, and is shown to obtain a close approximation of the globally optimal solution in practice. The MPC controller is then simulated in closed-loop, where it is shown to provide significant fuel savings relative to CDCS control. A comprehensive analysis of the convergence properties of the ADMM algorithm is included in the appendix.

The formulation of the MPC controller and numerical experiments were published in [21], and a similar proof of convergence for nonconvex problems was published in [22].

CHAPTER 6: DRIVER BEHAVIOUR UNCERTAINTY

The MPC controller proposed in Chapter 4 is extended to consider uncertainty in future driver behaviour. A scenario-based MPC formulation is proposed where the controller optimizes over multiple predictions of future driver behaviour, and it is shown that the ADMM algorithm proposed in Chapter 4 readily extends to the new case. Results from

scenario optimization are used to derive a lower bound on the number of scenarios required for a given confidence of one-step-ahead feasibility. A data-based prediction method is proposed where predictions of future driver behaviour are generated from a database of previous examples of a given route being driven, and it is demonstrated in simulation that this approach provides an extremely close approximation of the results obtained with a nominal MPC controller that has full preview of future driver behaviour.

CHAPTER 7: BATTERY/SUPERCAPACITOR ELECTRIC VEHICLES

The approach used in the previous chapters for the PHEV energy management problem is extended to the case of an all-electric vehicle with a hybrid energy storage system consisting of batteries and supercapacitors. A general framework is proposed, considering limits on battery, supercapacitor, and motor power, and battery and supercapacity energy, and is shown to be convex under minor approximation. An ADMM algorithm is proposed for the solution of the convex optimal control problem. A set of numerical studies is presented, where the convex formulation is shown to provide significant reductions in battery degradation relative to a low-pass filtering controller.

This chapter was published in [23].

1.4.1 PUBLISHED WORKS

A significant fraction of the work presented in this thesis has been presented in the following peer-reviewed publications:

[18] S. East and M. Cannon, "An ADMM Algorithm for MPC-based Energy Management in Hybrid Electric Vehicles with Nonlinear Losses", *57th IEEE Conference on Decision and Control*, Miami, USA, pp. 2641–2646, 2018.

doi:10.1109/CDC.2018.8619731

[19] J. Buerger, S. East, and M. Cannon, "Fast Dual-Loop Nonlinear Receding Horizon Control for Energy Management in Hybrid Electric Vehicles", *IEEE Transactions on Control Systems Technology*, Vol. 27, no. 3, pp. 1060–1070, 2019.

doi:10.1109/TCST.2018.2797058

[20] S. East and M. Cannon, "Energy Management in Plug-in Hybrid Electric Vehicles: Convex Optimization Algorithms for Model Predictive Control", *IEEE Transactions on Control Systems Technology (Early Access)*, Vol. -, pp. 1–13, 2019.

doi:10.1109/TCST.2019.2933793

[21] S. East and M. Cannon, "Fast Optimal Energy Management with Engine On/Off Decisions for Plug-in Hybrid Electric Vehicles", *IEEE Control Systems Letters and CDC 2019*, Vol. 3, no. 4, pp. 1074–1079, 2019.

doi:10.1109/LCSYS.2019.2920164

[22] S. East and M. Cannon, "ADMM for MPC with State and Input Constraints, and Input Nonlinearity", *American Control Conference*, Milwaukee, USA, pp. 4514–4519, 2018.

doi:10.23919/ACC.2018.8431655

[23] S. East and M. Cannon, "Optimal Power Allocation in Battery/Supercapacitor Electric Vehicles using Convex Optimization", *IEEE Transactions on Vehicular Technology*, Vol. 69, no.11, pp. 12751-12762, 2020.

doi: 10.1109/TVT.2020.3023186

The following peer-reviewed publications were also completed during my doctoral studies, but the results are not included in this thesis.

- Z. Qureshi, S. East, and M. Cannon, "Parallel ADMM for Robust Quadratic Optimal Resource Allocation Problems", *American Control Conference*, Philadelphia, USA, pp. 3402–3407, 2019.

  doi:10.23919/ACC.2019.8814898

- S. East, M. Gallieri, J. Masci, J. Koutnik, and M. Cannon, "Infinite-Horizon Differentiable Model Predictive Control", *International Conference on Learning Representations (ICLR)*, Addis Ababa, Ethiopia, April 2020.

  arXiv:2001.02244

COMPUTATIONAL HARDWARE

All of the numerical experiments presented in this thesis were programmed in Matlab and run on a 2.60GHz Intel i7-9750H CPU.

CHAPTER 2

THE ENERGY MANAGEMENT PROBLEM

THIS chapter provides an introduction to the PHEV energy management problem, and an overview of methods for its solution. Firstly, a simplified version of the energy management problem is formalized mathematically, then a review of common rule-based and optimization-based methods for computing exact or approximate solutions is presented.

2.1 PROBLEM FORMULATION

Figure 2.1 shows a simplified overview of the powerflows in a PHEV powertrain, which is the architecture considered for the majority of this thesis. The specific architecture considered here is a *parallel, pre-transmission* PHEV[1], where the output of the engine is mechanically coupled with the motor to a common drivetrain. In a *series* PHEV the engine would be connected to a separate generator that would supply electric power to the motor and battery, resulting in a mathematical structure that fits within the hybrid energy storage formulation addressed in Chapter 7. A *post-transmission* PHEV has the motor(s) connected directly to the wheels, and can be controlled using the framework proposed in this chapter with minor modifications.

The high-level power transfers and kinematics considered for PHEV energy management are as follows: fuel mass flow is delivered from a fuel tank to the engine, where it is converted to mechanical power that is delivered through a clutch. Simultaneously, the battery's internal charge is converted to electrical power at its terminals, which is then converted to mechanical power by the motor (and this power-flow can be reversed to charge the battery). When the clutch is closed, the power from the motor and engine is combined additively to power the drivetrain, otherwise it is driven entirely by the motor. The drive-

[1] From this point, the preceding terms are dropped, and it can be assumed that every use of PHEV (without qualifying the type) refers to a parallel pre-transmission PHEV.

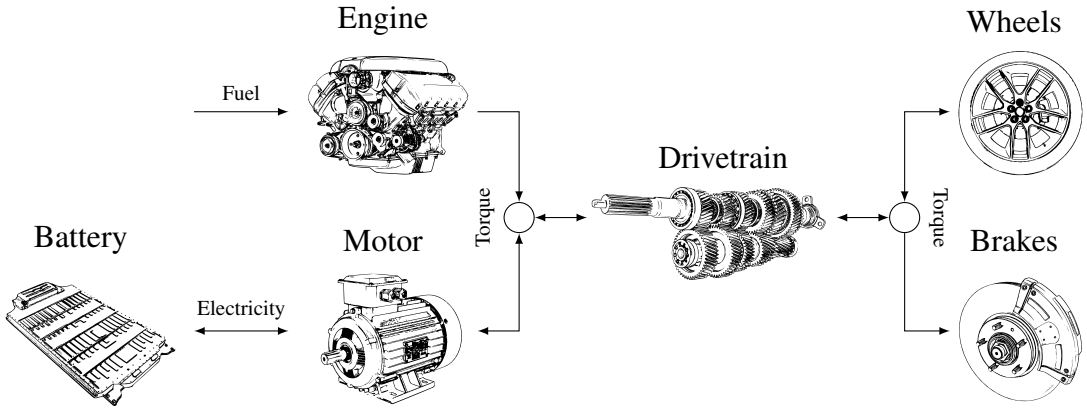


Figure 2.1: Simplified diagram of components and energy transfers that are relevant to the energy management problem.

train includes a gearbox to modulate the rotational speed of the engine and motor relative to the wheels, and mechanical brakes are available to remove energy at the wheels.

The principles described above are now formulated mathematically to aid with the presentation of the control methods that follow. Figure 2.2 shows a diagram of the power flows and rotational speeds that are relevant to this formalization, where $t \in [0, T]$ is the time and $T \in \mathbb{R}_{++}$ is the duration of a given journey. Energy management is an inverse problem: the behaviour of the vehicle is entirely dictated by the driver, and the control problem is to determine the signals allocated to various components to meet the demand whilst minimizing a given objective function. An inverse dynamic model is therefore typically used, where the relevant driver behaviour is defined by the rotational speed of the wheels, $\omega_w : [0, T] \mapsto \mathbb{R}$, and demand power, $d : [0, T] \mapsto \mathbb{R}\ \forall t \in [0, T]$. The rotational speed of the motor armature, $\omega_m : [0, T] \mapsto \mathbb{R}$, is subsequently defined by $\omega_m(t) := \omega_w(t)/r(t)$, where $r : [0, T] \mapsto \mathcal{R}$ is the effective gear ratio of the transmission such that $\mathcal{R}$ is the set of available ratios (only forwards driving is considered in this thesis, but the approach can be readily applied to reverse driving if $\mathcal{R}$ contains a negative element). The clutch control is defined by $\sigma : [0, T] \mapsto \{0, 1\}$, where $\sigma(t) = 1 \implies$ clutch closed, and $\sigma(t) = 0 \implies$ clutch open. It can be assumed that when the clutch is closed, the rotational speed of the engine, $\omega_e : [0, T] \mapsto \mathbb{R}$, is equal to the rotational speed of the motor armature, and that when it is open, the rotational speed of the engine is zero (i.e. the engine is off):

$$\omega_e(t) := \begin{cases} \omega_m(t) & \sigma(t) = 1 \\ 0 & \sigma(t) = 0 \end{cases}, \quad \forall t \in [0, T].$$

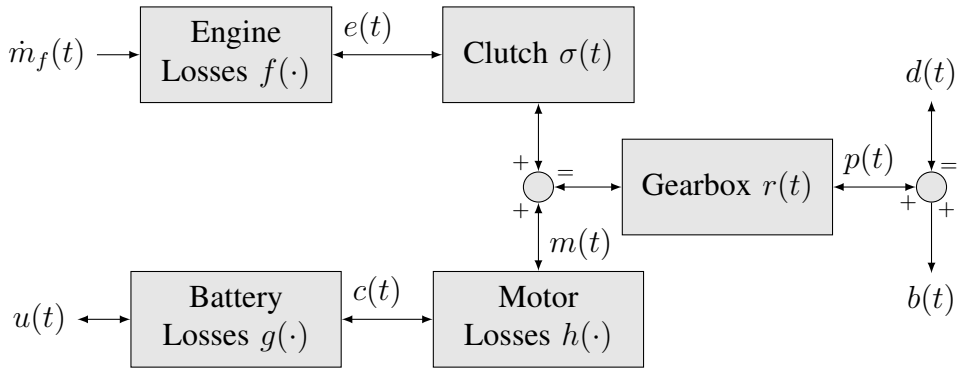


Figure 2.2: Diagram of the loss functions and variables that are used in the mathematical formulation of the energy management problem. The directions of the arrows represents the direction of power transfer (note that the arrow between the engine and clutch is bidirectional to allow the possibility of engine braking). Symbols are also included to indicate how the powerflows are combined at the summing junctions.

If it is also assumed that the gearbox and torque coupler are 100% efficient, then when the clutch is open the power delivered by the powertrain, $p : [0, T] \mapsto \mathbb{R}$, is equal to the mechanical power delivered by the motor, $m : [0, T] \mapsto \mathbb{R}$ (where $m(t) \in \mathbb{R}_{--}$ corresponds to regenerative braking). When the clutch is closed, the power delivered by the powertrain is equal to the sum of the power delivered by the motor and the power delivered by the engine, $e : [0, T] \mapsto \mathbb{R}$. Additionally, the driver demand power is equal to the power delivered by the powertrain plus the mechanical braking power (which must be non-positive), given by $b : [0, T] \mapsto \mathbb{R}_-$, so

$$d(t) =: \begin{cases} e(t) + m(t) + b(t) & \sigma(t) = 1 \\ m(t) + b(t) & \sigma(t) = 0 \end{cases}, \quad \forall t \in [0, T]. \tag{2.1}$$

The rate of fuel consumption, $\dot{m}_f : [0, T] \mapsto \mathbb{R}_+$, can be defined via a map, $f : \mathbb{R}^2 \mapsto \mathbb{R}_+$, of engine output power and rotational speed, so that $\dot{m}_f(t) := f(e(t), \omega_e(t))$. Similarly, the electrical power delivered to the motor, $c : [0, T] \mapsto \mathbb{R}$, can be defined via a map, $h : \mathbb{R}^2 \mapsto \mathbb{R}$, of motor output power and rotational speed, so that $c(t) := h(m(t), \omega_m(t))$. The rate of depletion of the internal energy of the battery, $u : [0, T] \mapsto \mathbb{R}$, is then given by a time varying function, $g : \mathbb{R} \times [0, T] \mapsto \mathbb{R}$, of the battery's electrical output power, so that $u(t) := g(c(t), t)\ \forall t \in [0, T]$. Therefore, the internal energy of the battery, $x : [0, T] \mapsto \mathbb{R}$,

can be given by

$$x(t) := x(0) - \int_0^t u(\tau)\,\mathrm{d}\tau, \quad \forall t \in (0, T],$$

where $x(0)$ is the battery's initial internal energy (for the purposes of this presentation it is assumed that u is Riemann integrable).

An important aspect of the energy management problem is that the system components are subject to hard constraints. The engine has upper and lower limits on torque, $\overline{T}_e : \mathbb{R} \mapsto \mathbb{R}$ and $\underline{T}_e : \mathbb{R} \mapsto \mathbb{R}$, that may be functions of engine speed, so the limits on engine power can given by

$$\underline{T}_e(\omega_e(t))\omega_e(t) \leq e(t) \leq \overline{T}_e(\omega_e(t))\omega_e(t) \quad \forall t \in [0, T].$$

Similarly, limits on motor power can be given as

$$\underline{T}_m(\omega_m(t))\omega_m(t) \leq m(t) \leq \overline{T}_m(\omega_m(t))\omega_m(t) \quad \forall t \in [0, T],$$

where $\overline{T}_m : \mathbb{R} \mapsto \mathbb{R}$ and $\underline{T}_m : \mathbb{R} \mapsto \mathbb{R}$ are upper and lower limits on motor torque. The engine and motor also have static upper and lower limits on rotational speed given by $\overline{\omega}_e, \underline{\omega}_e, \overline{\omega}_m, \underline{\omega}_m \in \mathbb{R}$, so that

$$\underline{\omega}_e \leq \omega_e(t) \leq \overline{\omega}_e \quad \text{and} \quad \underline{\omega}_m \leq \omega_m(t) \leq \overline{\omega}_m, \quad \forall t \in [0, T],$$

and the battery has static limits on internal energy and rate of charge and discharge, $\overline{x}, \underline{x}, \overline{u}, \underline{u} \in \mathbb{R}$, so that

$$\underline{x} \leq x(t) \leq \overline{x} \quad \forall t \in (0, T], \quad \text{and} \quad \underline{u} \leq u(t) \leq \overline{u} \quad \forall t \in [0, T].$$

Therefore, the necessary and sufficient set of variables to be chosen so that the operation of the powertrain is well defined are the engine output power, $e(t)$, clutch/engine state, $\sigma(t)$, gear selection, $r(t)$, and braking power, $b(t)$ (note that the motor power is implicitly defined by (2.1) given the listed control variables). It is important to note that the problem of interest is to determine how the driver's power demand should be distributed between the powertrain's components, and *not* to control the behaviour of the vehicle in any autonomous sense. The acceleration and deceleration of the vehicle are dictated entirely by the driver, so $d(t)$ and $\omega_w(t)$ are set at all times and cannot be changed by the powertrain's controller.

2.2 Optimal Energy Management

A simple PHEV powertrain control method is to determine the engine switching, gear selection, and braking control inputs with heuristics, and control the relative fraction

of power delivered by the motor and engine with a charge-depleting/charge-sustaining (CDCS) strategy [24]. In this case, all of the driver's demand power is delivered from the motor until the battery is completely depleted, after which the state-of-charge of the battery is sustained by either not using the electric system or by controlling the powertrain as a hybrid electric vehicle (e.g. only using the battery to brake and accelerate from a standstill). It has, however, been demonstrated that by delivering power from the motor and engine simultaneously, and modulating the power fraction delivered from each throughout the journey in a 'blended mode', the total fuel consumption can be reduced without affecting total electrical energy consumption, as shown in Figure 2.3.

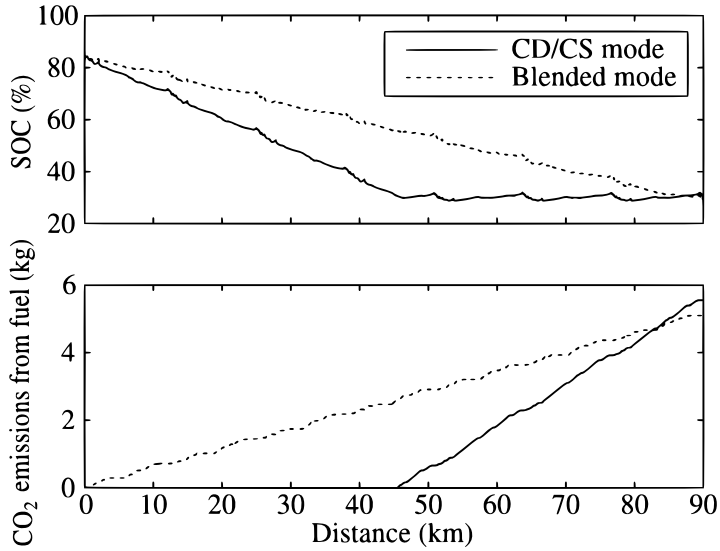


Figure 2.3: Comparison of blended mode power allocation and CDCS strategy (reproduced from [25]), illustrating reduction in fuel use (note that bottom figure represents emissions, but the same principle applies to fuel consumption).

There are two classes of control methods for reducing fuel consumption in this way: rule-based and optimization-based. Rule-based controllers are a further set of heuristic methods that decide the power-split based on a pre-determined set of rules, and whilst they can improve performance relative to CDCS, they are also inherently suboptimal in general [10, §2]. Conversely, optimization-based controllers attempt to explicitly address the question 'what are the control actions that result in the *lowest possible* fuel consumption?'. This chapter continues with an overview of five common methods for addressing this question: dynamic programming, Pontryagin's minimum principle, equivalent consumption minimization strategy, model predictive control, and convex optimization.

2.2.1 DYNAMIC PROGRAMMING

BACKGROUND

The concept of dynamic programming was introduced and developed by Richard E. Bellman during the 1950's, and is based on the *principle of optimality*:

> "An optimal policy has the property that whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision" [26]

Dynamic programming is a fundamental tool of control theory, and a common approach to energy management problems. The principles of dynamic programming are now developed mathematically to aid later discussion. Consider a system described by an ODE, $\dot{\mathbf{x}} = \mathbf{f}(t, \mathbf{x}(t), \mathbf{u}(t))$, where $\mathbf{x}(t) \in \mathbb{R}^n$ is the system state and $\mathbf{u}(t) \in \mathbb{R}^m$ is the control input. In order to measure the relative optimality of each candidate control input, consider a running cost function $\mathcal{L}(t, \mathbf{x}(t), \mathbf{u}(t))$, and terminal cost function $\mathcal{L}_T(\mathbf{x}(T))$, so that the cost functional for the optimal control problem is given by

$$J(t, \mathbf{x}(t), \mathbf{u}_{[t,T]}) := \int_t^T \mathcal{L}(\tau, \mathbf{x}(\tau), \mathbf{u}(\tau))\, \mathrm{d}\tau + \mathcal{L}_T(\mathbf{x}(T)) \quad \forall t \in [0, T].$$

The value function associated with this problem is the minimum (assuming that it exists) of the cost functional at a given time and state:

$$V(t, \mathbf{x}) := \min_{\mathbf{u}_{[t,T]}} J(t, \mathbf{x}(t), \mathbf{u}_{[t,T]}) \quad \forall t \in [0, T].$$

The principle of optimality states that the value function at $(t, \mathbf{x})$ is equal to the minimum of the running cost over an interval $[t, t+\Delta T]$, plus the value function at $(t+\Delta t, \mathbf{x}(t+\Delta t))$:

$$V(t, \mathbf{x}) = \min_{\mathbf{u}_{[t,t+\Delta t]}} \left\{ \int_t^{t+\Delta t} \mathcal{L}(\tau, \mathbf{x}(\tau), \mathbf{u}(\tau))\, \mathrm{d}\tau + V(t + \Delta t, \mathbf{x}(t + \Delta t)) \right\}.$$

By taking the limit as $\Delta t \to 0$, a partial differential equation known as the Hamilton-Jacobi-Bellman equation is obtained,

$$-\nabla_t V(t, \mathbf{x}) = \min_{\mathbf{u}} \Big\{ \mathcal{L}(t, \mathbf{x}, \mathbf{u}) + \langle \nabla_{\mathbf{x}} V(t, \mathbf{x}), \mathbf{f}(t, \mathbf{x}, \mathbf{u}) \rangle \Big\},$$

and the optimal control input, $\mathbf{u}^\star(t)$, can be obtained as the minimizing argument of the right-hand-side.

For a fuller exposition of dynamic programming and the Hamilton-Jacobi-Bellman equation in the context of optimal control, see [27, §5]. For Bellman's original paper on dynamic programming, see [28], and for his later textbook on the subject see [26]. For a historical overview of the development of dynamic programming, see [26].

DYNAMIC PROGRAMMING APPLIED TO PHEV ENERGY MANAGEMENT

If the PHEV management problem described in Section 2.1 were to be approached using Dynamic Programming as presented above, the state could be given by $\mathbf{x} = x$, the control input given by $\mathbf{u} = (u, \sigma, r, b)$, and dynamics given by $\mathbf{f}(\mathbf{x}) = -u(t)$. The hard constraints on system operation could be represented by a time-varying indicator function, $\mathcal{I}(\mathbf{u}(t), t)$, that maps to infinity if any of the constraints are violated and to zero otherwise, and a terminal constraint on the battery's state-of-charge could be included with $\mathcal{L}_T(\mathbf{x}(t))$ (which could also be an indicator function enforcing a terminal constraint set). The cost functional would then be

$$J(\mathbf{u}) := \int_0^T \Big[\dot{m}_f(t) + \mathcal{I}(\mathbf{u}(t), t)\Big] \, \mathrm{d}t + \mathcal{L}_T(\mathbf{x}(T)). \tag{2.2}$$

For the most general powertrain models and constraints, obtaining the optimal controller for (2.2) using dynamic programming is intractable due to the presence of multiple state and control constraints, and discontinuous and integer decision variables. The standard approach in the literature is to instead discretise the time, state, and control spaces, so that $\mathbf{x}_t \in \mathcal{X} \subset \mathbb{R}^n$ and $\mathbf{u}_t \in \mathcal{U} \subset \mathbb{R}^m$ for all $t \in \mathcal{T}$, where $\mathcal{T} := \{0, \Delta t, \ldots, T\}$, $\mathcal{X} := \mathcal{X}_1 \times \cdots \times \mathcal{X}_n$, and $\mathcal{U} := \mathcal{U}_1 \times \cdots \times \mathcal{U}_m$ (such that $|\mathcal{X}_i| \in \mathbb{Z}_{++}$ and $|\mathcal{U}_j| \in \mathbb{Z}_{++}$ for all $i \in \{1, \ldots, n\}$ and $j \in \{1, \ldots, m\}$). The value function can then be approximated by a finite-dimensional array, $\mathbf{V} : \mathcal{T} \times \mathcal{X} \mapsto \mathbb{R}$, which can be obtained by working backwards in time from the terminal cost function and constraint set. For a hard terminal constraint $\mathcal{X}_T = \mathbf{x}(T)$ (where $\mathbf{x}(T) \in \mathcal{X}$), the value function array is initialised at time T as

$$\mathbf{V}[T, \mathbf{x}_t] := \begin{cases} 0 & \mathbf{x}_t = \mathbf{x}(T) \\ \infty & \mathbf{x}_t \neq \mathbf{x}(T) \end{cases},$$

and then updated sequentially backwards in time using

$$\mathbf{V}[t, \mathbf{x}_t] := \min_{\mathbf{u}_t \in \mathcal{U}} \{\mathcal{L}(t, \mathbf{x}_t, \mathbf{u}_t) + \mathbf{V}[t + \Delta t, \mathbf{f}(t, \mathbf{x}_t, \mathbf{u}_t)]\} \quad \forall t \in \{T - \Delta t, \ldots, 0\}, \tag{2.3}$$

where $\mathbf{f} : \mathcal{T} \times \mathcal{X} \times \mathcal{U} \mapsto \mathcal{X}$ is is a function representing the discrete system dynamics and $\mathcal{L} : \mathcal{T} \times \mathcal{X} \times \mathcal{U} \mapsto \mathbb{R}$ is the stage cost. Simultaneously, the optimal control law, $\mathbf{U} : \mathcal{T} \times \mathcal{X} \mapsto \mathcal{U}$, is generated from

$$\mathbf{U}[t, \mathbf{x}_t] := \operatorname*{argmin}_{\mathbf{u}_t \in \mathcal{U}} \{\mathcal{L}(t, \mathbf{x}_t, \mathbf{u}_t) + \mathbf{V}[t + \Delta t, \mathbf{f}(t, \mathbf{x}_t, \mathbf{u}_t)]\} \quad \forall t \in \{T - \Delta t, \ldots, 0\}.$$

The optimal control actions are then determined for a given initial state by setting $\mathbf{x}_0 = x(0)$, then sequentially obtaining $\mathbf{u}_t = \mathbf{U}[t, \mathbf{x}_t]$ and updating the state for the next timestep as $\mathbf{x}_{t+\Delta t} = \mathbf{f}_t(\mathbf{x}_t, \mathbf{u}_t)$. This approximation converges to the continuous time optimal control law as $|\mathcal{X}_i|, |\mathcal{U}_j| \to \infty$ for all $i \in \{1, \ldots, n\}$ and $j \in \{1, \ldots, m\}$, and as $\Delta t \to 0$.

EXAMPLES IN THE LITERATURE

One of the limitations of dynamic programming is that it requires significant computational and memory resources, as the requirement for both increases exponentially with the dimension of the state and control spaces. Dynamic programming is therefore not suitable for online PHEV energy management control, and is typically only used offline to tune real-time controllers or as a performance benchmark. In an early paper (2003) [29], dynamic programming was used to optimize the gear shift strategy and power split of a parallel hybrid truck offline in order to understand the optimal behaviour, which was then used to tune a real-time rule-based controller. The same dynamic programming approach was then later used to determine the effect of varying levels of trip preview information [30]. In [31], dynamic programming was used as an optimal baseline to investigate the effect of road gradient information on other optimization-based control methods. The deterministic dynamic approach described in this section was extended to the stochastic case using policy iteration in [32], generating a control law that minimized the expected cost across a range of drive cycles. Power-split, gear selection, and engine switching control were considered in [33], where dynamic programming was used as a baseline for other optimization-based methods and shown to take 4799s to optimize a $\sim$1200s journey. Similarly, dynamic programming was also used as a globally optimal baseline in [34, 35, 36].

Steps have also been taken to reduce the computational and memory requirements of dynamic programming. For example, in [37] the minimization on the RHS of (2.3) was approximated analytically, reducing the computational time from 490s to 5.1s with only a 0.07% reduction in optimality (although this was only tested on a single drive-cycle). A vehicle-to-grid system was proposed in [38] where $\mathbf{U}$ is computed offline and transmitted to the vehicle from a centralised server. Neither approach, however, reduces the (potentially significant) memory requirements. Conversely, both the computational and memory requirements can be addressed by approximating the optimal control law obtained with dynamic programming using a neural network as in [39], although the approximation necessarily introduces a measure of sub-optimality, and it results in a black-box system for which it is impossible to interpret control decisions.

2.2.2 PONTRYAGIN'S MINIMUM PRINCIPLE

BACKGROUND

During approximately the same time period that the ideas behind dynamic programming were being developed, an alternative approach to optimal control based on the calculus of variations was formulated by Soviet Mathematician Lev Pontryagin, known as the minimum principle. The optimality conditions obtained using Pontryagin's minimum principle

(PMP) are now presented for the *free-time, fixed-endpoint* optimal control problem to aid the following discussion.

Consider a system described by a time invariant ODE, $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u})$, with a time invariant running cost, $\mathcal{L}(\mathbf{x}, \mathbf{u})$, and no terminal cost. *Fixed-endpoint* implies a terminal target set so that $\mathbf{x}(t_f) \in \{x_f\}$, whilst *free-time* implies that the finite horizon, t_f, is an optimization variable. For this problem, the *neccesary* optimality conditions are given by:

Theorem 2.2.1 (The Minimum Principle for the free-time fixed-endpoint control problem)**.** *Let $\mathbf{u}^\star : [0, t_f] \mapsto \mathcal{U}$ be an optimal control, and let $\mathbf{x}^\star : [0, t_f] \mapsto \mathbb{R}^n$ be the corresponding optimal state trajectory. Then there exists a function $\mathbf{p}^\star : [0, t_f] \mapsto \mathbb{R}^n$ satisfying $\mathbf{p}^\star(t) \neq 0 \; \forall t \in [0, t_f]$ and having the following properties:*

1. *$\mathbf{x}^\star$ and $\mathbf{p}^\star$ satisfy*

$$\dot{\mathbf{x}}^\star = \nabla_{\mathbf{p}} H(\mathbf{x}^\star, \mathbf{u}^\star, \mathbf{p}^\star)$$
$$\dot{\mathbf{p}}^\star = -\nabla_{\mathbf{x}} H(\mathbf{x}^\star, \mathbf{u}^\star, \mathbf{p}^\star)$$

   with the boundary conditions $\mathbf{x}^\star(t_0) = x_0$ and $\mathbf{x}^\star(t_f) = x_1$, where the Hamiltonian $H : \mathbb{R}^n \times \mathcal{U} \times \mathbb{R} \mapsto \mathbb{R}$ is defined as

$$H(\mathbf{x}, \mathbf{u}, \mathbf{p}) := \mathbf{p}^\top \mathbf{f}(\mathbf{x}, \mathbf{u}) - \mathcal{L}(\mathbf{x}, \mathbf{u})$$

2. *For each fixed t, the function $\mathbf{u} \mapsto H(\mathbf{x}^\star(t), \mathbf{u}, \mathbf{p}^\star(t))$ has a global minimum at $\mathbf{u} = \mathbf{u}^\star(t)$, i.e., the inequality*

$$H(\mathbf{x}^\star(t), \mathbf{u}^\star(t), \mathbf{p}^\star(t)) \leq H(\mathbf{x}^\star(t), \mathbf{u}, \mathbf{p}^\star(t))$$

   holds for all $t \in [0, t_f]$ and all $\mathbf{u} \in \mathcal{U}$.

3. *$H(\mathbf{x}^\star(t), \mathbf{u}^\star(t), \mathbf{p}^\star(t)) = 0$ for all $t \in [0, t_f]$.*

Proof. The proof is presented in [27, §4]. □

For an English translation of Pontryagin's results see [40]. For a historical overview of the development of Pontryagin's minimum principle, see [41] or [42].

PONTRYAGIN'S MINIMUM PRINCIPLE APPLIED TO PHEV ENERGY MANAGEMENT

It is not always trivial to translate the conditions for optimality obtained using PMP into an optimal control solution, and the general problem described in Section 2.1 is still intractable in general. For illustrative purposes, a reduced problem where only the power-split between the engine and motor is controlled is considered here. In this case, it

can be assumed that the gear selection, $r(t)$, braking power, $b(t)$, and clutch engagement, $\sigma(t)$, are controlled using an external method, and that the engine is always on ($\sigma(t) = 1\ \forall t \in [0, T]$). The free-time problem 2.2.1 can be converted to a fixed-time problem for energy management by augmenting the state vector so that $\mathbf{x}(t) = (x(t), t)$, and using the terminal constraint $\mathbf{x}(t_f) \in \{(x_f, T)\}$. Using this definition of the state, the system is defined by

$$\mathbf{f}(\mathbf{x}, \mathbf{u}) := \begin{bmatrix} g(h(d(t) - e(t))) \\ 1 \end{bmatrix}, \quad \mathbf{p}(t) := \begin{bmatrix} p_1(t) \\ p_2(t) \end{bmatrix}, \quad \mathbf{u} := e(t),$$

$$\text{and} \quad H(\mathbf{x}, \mathbf{u}, \mathbf{p}) := p_1(t) \cdot g_t(h_t(d(t) - e(t))) + p_2(t) - f_t(e(t)).$$

The necessary condition for optimality given by Theorem 2.2.1 Property 1 states that

$$\dot{\mathbf{p}} = -\nabla_{\mathbf{x}} H = (0, \nabla_t H),$$

which implies that the first element of the costate vector, $p_1(t)$, is constant[2], i.e. $p_1^\star(t) = p_1^\star(0)\ \forall t \in [0, T]$. The optimal control input is then defined implicitly by

$$\nabla_{\mathbf{u}} H = 0 \quad \Longrightarrow \quad -p_1^\star(t) \cdot g_t'(h_t(d(t) - e(t))) \cdot h_t'(d(t) - e(t)) - f_t'(e(t)) = 0.$$

Therefore, the PHEV optimal control problem in this case reduces to an initial value problem to determine $p_1^\star(0)$. This can be solved using a shooting method, where for each estimate of $p_1(0)$, the corresponding control input is obtained from

$$e(t) \in \{e(t) : -p_1(0) \cdot g_t'(h_t(d(t) - e(t))) \cdot h_t'(d(t) - e(t)) - f_t'(e(t)) = 0\}, \tag{2.4}$$

and the estimate of $p_1(0)$ is updated until $\mathbf{x}(t_f) = (x_f, T)$.

EXAMPLES IN THE LITERATURE

The benefit of the minimum principle relative to dynamic programming is that it uses significantly less computation and memory, although it only provides a necessary optimality condition, and is not sufficient (in general) to demonstrate global optimality. Furthermore, whilst a solution is simple to demonstrate for the case where the terminal state constraint is considered for power-split control, it is more challenging to address more general constraints and other control variables.

[2]The physical condition for this result to hold is that the electrical losses are independent of the state-of-charge of the battery, and is a common assumption in PMP approaches to the energy management problem.

The minimum principle was compared with dynamic programming in [34] for optimizing both the power split and gear selection variables, and it was found that it resulted in similar gear-shifting decisions and fuel consumption with significantly less computation. A hybrid approach was proposed in [33], where an algorithm alternated between optimizing the gear selection for a fixed power-split using dynamic programming, then optimizing the power-split for fixed gear selection using the minimum principle. This reduced the computational time from ~ 4000 s using dynamic programming to ~ 12 s using the hybrid algorithm (the target time for real-time operation is typically < 1 s). The minimum principle was also used in [43] to optimize both the power split and gear selection, where it was found to improve fuel consumption by up to 10% relative to a rule-based baseline, and used in [44] to optimize the power-split only, where it was shown to reduce fuel consumption by $\sim 20\%$ relative to a CDCS baseline. Engine switching was considered in [45] using both the minimum principle and simulated annealing. The minimum principle was also used in [46] the generate optimal behaviour used for training a rule-based algorithm.

The PMP formulation presented above has also been extended to consider other constraints and objectives. A constraint on the upper and lower state-of-charge of the battery was enforced at all times in [47] and [48] by introducing a term to the Lagrangian that penalises a linear cost on state violation. This approach can guarantee hard constraint satisfaction for a 'sufficiently' large constraint penalty, although the 'sufficient' threshold cannot be determined *a priori*, and an insufficient penalty will allow the constraint to be violated. An alternative method for enforcing the state-of-charge constraint was proposed in [49] using a quadratic cost on violations and was compared with dynamic programming in simulation, but was not tested in scenarios for which the constraint was active. The minimum principle has also been used to minimize CO_2 emissions [48] and battery degradation [50, 51] instead of fuel consumption.

2.2.3 EQUIVALENT CONSUMPTION MINIMISATION STRATEGY

BACKGROUND

The equivalent consumption minimisation strategy (ECMS) was originally developed for charge sustaining hybrid electric vehicles on the principle that the consumption of fuel energy and electrical energy can be equated using an equivalence factor [52]: the idea is that when electrical energy is sacrificed to reduce fuel consumption in the present, fuel consumption will need to be increased in the future to replenish the battery. Therefore, the equivalent fuel consumption, $\dot{m}_{f,equiv}$, can be given by

$$\dot{m}_{f,equiv}(e(t), \lambda(t), t) = \dot{m}_f(e(t), t) + \lambda(t) \cdot g_t(h_t(d(t) - e(t))),$$

where $\lambda(t)$ is the equivalence factor used to measure the relative cost of fuel and electrical energy consumption. The power delivered by the engine is then given by

$$e(t) = \underset{e(t)}{\operatorname{argmin}}\ \dot{m}_f(e(t), t) + \lambda(t) \cdot g_t(h_t(d(t) - e(t))) \quad \forall t \in [0, T]. \tag{2.5}$$

EXAMPLES IN THE LITERATURE

The goal when designing an ECMS-style algorithm for energy management is to determine $\lambda(t)\ \forall t \in [0, T]$ such that the hardware constraints are not violated. This echoes the issue of choosing the initial costate $\mathbf{p}(0)$ for PMP, and it has in fact been shown that ECMS and PMP are mathematically equivalent [53] (note that (2.4) is the first-order necessary condition for (2.5) if $\mathbf{p}(t) = \lambda(t)$ and $\dot{m}_{f,equiv}$ is differentiable). The emphasis for ECMS approaches in the literature, however, has been on developing adaptive methods for controlling the equivalence factor to generate a real-time optimization-based controller, known as adaptive-ECMS [54, 55], rather than on obtaining globally optimal solutions.

The results of a competition to develop a real-time energy management controller for PHEV fuel consumption minimization is detailed in [56], in which each of the candidate controllers had no predictive information about future driver behaviour. The submissions consisted of rule-based and ECMS optimization-based algorithms, and it was found that ECMS generally outperformed the rule-based controllers and was more robust to system variations. A common approach for introducing predictive information to adaptive ECMS is to generate a reference state-of-charge trajectory for a given journey, and then use the equivalence factor to penalize deviations in the actual state-of-charge trajectory. The authors of [57] considered hybrid-electric vehicles where a quadratic cost function penalizing deviations for a constant state-of-charge was added to the running cost to ensure that the controller was charge sustaining. In [58] a linearly decreasing state-of-charge reference was constructed that reached the terminal state-of-charge constraint at the end of the journey (which was assumed to be known), and a proportional-integral controller was then introduced to enforce the reference trajectory through the equivalence factor. A neural network was used in [59] to predict the energy demand of the vehicle, from which a state-of-charge trajectory was then generated, and in [60] a neural network was combined with vehicle-to-infrastructure and vehicle-to-vehicle technology to obtain more accurate predictions. Other predictive information has also been considered: the effect of road gradient preview was investigated in [31], and in [61] it was assumed that the controller was aware of both the future road gradient and distance to the next charging station when updating the equivalence factor. For a more comprehensive overview of ECMS and its parallels with PMP see [62].

2.2.4 MODEL PREDICTIVE CONTROL

BACKGROUND

Although the minimum principle has been shown to provide a comparable measure of optimality to dynamic programming with significantly reduced computational requirements, in general it is challenging to account for hard constraints in a systematic way. If constraints that are not considered in the optimal control formulation are active in operation, the performance of the controller may significantly deteriorate with respect to that predicted by the optimization. A method for extending the ideas from optimal control to explicitly consider hard constraints on states and control inputs is MPC. This a nonlinear feedback control method where the input applied to the system is defined implicitly by the solution of a constrained optimization problem. A general formulation of MPC for the discrete-time linear-time-invariant case is now presented to illustrate the fundamental ideas.

Consider a system with a state, $\mathbf{x}(t) \in \mathbb{R}^n$, and control input, $\mathbf{u}(t) \in \mathbb{R}^m$, where the state at time $t + \Delta t$ is defined by

$$\mathbf{x}(t + \Delta t) := A\mathbf{x}(t) + B\mathbf{u}(t) \quad \forall t \in \{0, \Delta t, \dots\},$$

such that $A \in \mathbb{R}^{n\times n}$ is the state transition matrix and $B \in \mathbb{R}^{n\times m}$ is the input matrix. The principle of MPC is that at each time sample the control input applied to the system is obtained from the solution of a finite-horizon optimal control problem, e.g.

$$\begin{aligned} \mathbf{u}^\star \in \underset{\mathbf{u}}{\operatorname{argmin}} \; & \sum_{k=0}^{N-1} \mathcal{L}_k(\mathbf{u}_k, \mathbf{x}_{k+1}), \\ \text{s.t. } & \mathbf{x}_0 = \mathbf{x}(t), \\ & \mathbf{x}_{k+1} = A\mathbf{x}_k + B\mathbf{u}_k, \\ & \mathbf{x}_{k+1} \in \mathcal{X}_{k+1}, \\ & \mathbf{u}_k \in \mathcal{U}_k, \\ & \forall k \in \{0, \dots, N-1\}, \end{aligned} \tag{2.6}$$

where $\mathbf{x} := (\mathbf{x}_1, \dots, \mathbf{x}_N)$ is the predicted state trajectory, $\mathbf{u} := (\mathbf{u}_0, \dots, \mathbf{u}_{N-1})$ is the corresponding control input trajectory, $N \in \mathbb{Z}_+$ is the prediction horizon, $\mathcal{L}_k$ is a running cost on control and state, $\mathcal{X}_{k+1}$ is a state constraint set, and $\mathcal{U}_k$ is a control constraint set. The control input applied to the system over each sampling interval is then defined as the first element of the optimized control vector, i.e. $\mathbf{u}(t) = \mathbf{u}_0^\star(\mathbf{x}(t)) \; \forall t \in \{0, \Delta t, \dots\}$.

One of the main technical challenges associated with MPC is obtaining the solution to the constrained optimization problem in real time. A solution can be obtained for arbitrary

cost and constraints using the discrete-time dynamic programming algorithm presented in Section 2.2.1[3], although this is too computationally demanding in general. One of the advantages of a discrete-time, finite-horizon MPC optimization problem is that it can instead be solved using standard finite dimensional numerical optimization techniques [64]. If the cost function is convex and quadratic and the constraint sets are polytopic, then (2.6) reduces to a quadratic program that can be reliably and efficiently solved on both desktop [12, 65, 66] and embedded hardware [67, 68]. Note that it is also possible in some cases to map the nonlinear feedback law explicitly (e.g. [69]), although the memory required to store the feedback law for each partition of the state-space may become prohibitive.

For a historical overview of the development of MPC since its inception in 1980 see [70]. For a review of techniques for ensuring closed-loop stability for MPC using a finite prediction horizon see [71]. For a textbook providing a more comprehensive treatment of MPC see [72].

STOCHASTIC MODEL PREDICTIVE CONTROL

The previous section considered the case where it is assumed that the dynamics of the system being controlled are known with complete precision, a.k.a 'nominal' MPC. In reality, the dynamics will not be known exactly and the system will be subject to random disturbances. The state transition equation can be given in this case by

$$\mathbf{x}(t+\Delta t) := A(\mathbf{w}(t))\mathbf{x}(t) + B(\mathbf{w}(t))\mathbf{u}(t) + \mathbf{d}(\mathbf{w}(t)) \quad \forall t \in \{0, \Delta t, \dots, \},$$

where $\mathbf{w}(t) \in \mathbb{R}^o$ is a vector parametrizing the uncertainty, $\mathbf{d} : \mathbb{R}^o \mapsto \mathbb{R}^n$ is the system disturbance, and A and B are now functions of the uncertainty vector. One approach to addressing this uncertainty is Robust MPC, where it is assumed that the uncertainty is restricted to a set, i.e.

$$\mathbf{w}(t) \in \mathcal{W}_t \quad \forall t \in \{0, \Delta t, \dots\}.$$

[3]This illustrates a separation between MPC and the previously described optimal control approaches. The principle of optimality and the minimum principle provide optimality *conditions* for optimal control problems, and in particular cases these conditions result in a state-feedback control law (e.g. the linear quadratic regulator [63]). Model predictive control, on the other hand, is fundamentally a nonlinear state-feedback control law $\mathbf{u}(t) = \phi(\mathbf{x}(t))$, where ϕ is defined implicitly by a state parametrized optimization problem.

The robust MPC problem associated with this uncertainty can then be given by

$$\begin{aligned}
\mathbf{u}^\star \in \underset{\mathbf{u}}{\operatorname{argmin}} \ \max_{\mathbf{w}} \ & \sum_{k=0}^{N-1} \mathcal{L}_k(\mathbf{u}_k, \mathbf{x}_{k+1}), \\
\text{s.t. } & \mathbf{x}_0 = \mathbf{x}(t), \\
& \mathbf{x}_{k+1} = A(\mathbf{w}_k)\mathbf{x}_k + B(\mathbf{w}_k)\mathbf{u}_k + \mathbf{d}(\mathbf{w}_k), \\
& \mathbf{x}_{k+1} \in \mathcal{X}_{k+1}, \\
& \mathbf{u}_k \in \mathcal{U}_k, \\
& \forall k \in \{0, \dots, N-1\}.
\end{aligned} \tag{2.7}$$

Note that the cost function now represents the worst case across the uncertainty set. A limitation of the robust approach is that it does not account for the probability of each possible realisation of the uncertainty set, so the controller will be conservative in practice. This limitation is addressed using stochastic MPC, where the uncertainty vectors, $\mathbf{w}_k$, are instead modelled as random variables that induce probability measure $\mathbb{P}$ on $\mathcal{W}_k$. The stochastic MPC problem associated with this uncertainty can instead be given by

$$\begin{aligned}
\mathbf{u}^\star \in \underset{\mathbf{u}}{\operatorname{argmin}} \ & \mathbb{E}\left[\sum_{k=0}^{N} \mathcal{L}_k(\mathbf{u}_k, \mathbf{x}_{k+1})\right], \\
\text{s.t. } & \mathbf{x}_0 = \mathbf{x}(t), \\
& \mathbf{x}_{k+1} = A(\mathbf{w}_k)\mathbf{x}_k + B(\mathbf{w}_k)\mathbf{u}_k + \mathbf{d}(\mathbf{w}_k), \\
& \mathbb{P}\left[\mathbf{x}_{k+1} \in \mathcal{X}_{k+1}\right] \geq \epsilon, \\
& \mathbf{u}_k \in \mathcal{U}_k, \\
& \mathbf{w}_k \sim \mathcal{W}_k, \\
& \forall k \in \{0, \dots, N-1\}.
\end{aligned} \tag{2.8}$$

Note that the objective now considers the expectation of the cost subject to the random uncertainty (although the worst case as presented in the robust version may also be used), and that the state is now constrained by a chance constraint. This is included because if the probability distribution $\mathcal{W}_t$ has infinite support, then it may be impossible to guarantee the state constraints with complete certainty. For a review of robust MPC see [73], and for a review of stochastic MPC see [74]. For a textbook addressing both see [75].

EXAMPLES IN THE LITERATURE

The complexity of the powertrain model and constraints that can be used for MPC are restricted by the specifications of the electronic control unit used for powertrain control[4]. As previously described, in the most general case the energy management problem can take several hours to solve over long prediction horizons using dynamic programming, so a degree of approximation and simplification is required.[5] One of the challenging aspects of using MPC for the energy management problem is that the prediction horizon can be 'long': a half hour journey sampled at 1Hz results in a prediction horizon of 1800 samples. Consequently, the state-of-charge trajectory is typically not explicitly optimized. Instead, a reference state-of-charge trajectory is generated so as to minimize fuel consumption (similarly to ECMS), and then 'short' horizon MPC (i.e. $10 - 20$ samples) is used to track the reference trajectory. For hybrid-electric vehicles, the reference trajectory can be set at a constant level so that the control algorithm is charge-sustaining, as demonstrated in [76], where a constant terminal state-of-charge constraint was used in the MPC optimization. Similarly, a quadratic cost on deviations from a constant terminal state-of-charge was used in [77], and a quadratic cost was used to penalise deviations from a constant state-of-charge reference across the entire prediction horizon in [78, 79, 80, 81]. For PHEVs, a charge-depleting state-of-charge trajectory is instead constructed. In [82, 83], a linearly decreasing trajectory was used (again, similarly to ECMS), although a reference trajectory that adapted to predictions of driver power demand was also investigated in [82]. A hierarchical framework was presented in [84], where a simplified powertrain model was optimized over a long horizon to generate the state-of-charge trajectory, then short-horizon MPC was used with a high fidelity model to track the optimized state-of-charge trajectory.

The real-time requirement of MPC influences the choice of optimization algorithm, as only short prediction horizons can be used if a computationally expensive method is selected. PMP was used in [76] as the MPC formulation only included a hard terminal constraint, whilst dynamic programming was used for optimization in [78, 36, 84, 83]. The use of dynamic programming places a significant restriction on the prediction horizon (e.g. a prediction horizon of 10 samples was used in [36], and a maximum of 20 samples was used in [83]), but this restriction can be reduced using the heirarchical approach of [84], where the long-horizon prediction is updated at a lower frequency than the high fidelity model. Another approach is to linearise the vehicle model about the reference state-of-charge trajectory so that the optimization reduces to a quadratic program [78, 85], although the linearization may introduce significant sub-optimality if the reference trajectory does

[4]The specifications of the powertrain controller are challenging to quantify as a modern car typically has a large range of available processing units.

[5]An energy management specific MPC formulation is not presented here as it was for dynamic programming and the minimum principle because several MPC formulations are presented in the following chapters.

not match the optimized trajectory. This approach was extended in [86], where the model was repeatedly re-linearized using sequential quadratic programming.

A further aspect of MPC that requires consideration is the methods used to predict future driver behaviour, which can be generally divided into artificial intelligence-based methods and methods based on Markov processes [87, §2.2-2.3]. In a Markov model it is assumed that future states only depend on the current state, and this approach has been used to model predictions of future velocity [83, 36], acceleration [83, 82], road gradient [36], speed limit [36], torque [80, 86] and power demand [79]. The minimum fuel consumption can be obtained for all possible realisations of a Markov model over the prediction horizon using stochastic dynamic programming [36], although this approach is even more computationally demanding than the deterministic dynamic programming discussed previously. Neural networks have also been used to model future driver behaviour [84, 82, 88], and have been shown to provide more accurate short-term predictions than Markov models [88]. The aforementioned methods are typically only used for short-term behaviour predictions, and long-term prediction is still an open problem [87].

2.2.5 CONVEX OPTIMIZATION

BACKGROUND

Convex optimization has its roots in linear programming, a class of optimization problems developed in the 1940s for solving military logistics problems [89]. The development of interior-point methods in the 1980s revealed that the principles of these methods for solving linear programs could be extended to nonlinear optimization problems [90]. In particular, interior point methods could be used to efficiently obtain a globally optimal solution to the convex optimization problem of minimizing a convex function over a convex set. To begin this section, some of the core concepts of convex optimization are presented:

Definition 2.2.1 (Convex Set)**.** A set, $\mathcal{C} \subseteq \mathbb{R}^n$, is convex if

$$\lambda x_1 + (1-\lambda)x_2 \in \mathcal{C}, \quad \forall x_1, x_2 \in \mathcal{C}, \quad \forall \lambda \in [0,1].$$

Definition 2.2.2 (Convex Function)**.** Let $\mathcal{C} \subseteq \mathbb{R}^n$ be a convex set. A function, $f : \mathcal{C} \mapsto (-\infty, \infty]$, is convex if

$$\lambda f(x_1) + (1-\lambda)f(x_2) \geq f(\lambda x_1 + (1-\lambda)x_2), \quad \forall x_1, x_2 \in \mathcal{C}, \quad \forall \lambda \in [0,1].$$

The function is *strictly convex* if

$$\lambda f(x_1) + (1-\lambda)f(x_2) > f(\lambda x_1 + (1-\lambda)x_2), \quad \forall x_1, x_2 \in \mathcal{C}, \quad x_1 \neq x_2, \quad \forall \lambda \in (0,1).$$

Definition 2.2.3 (Convex Optimization Problem)**.** An optimization problem

$$\begin{aligned} &\min_{x} f(x) \\ &\text{s.t. } x \in \mathcal{X} \end{aligned}$$

is convex if $f : \mathcal{C} \mapsto (-\infty, \infty]$ is a convex function and $\mathcal{X}$ is a convex set. A vector $x^\star$ is a minimum of f over $\mathcal{X}$ if

$$x^\star \in \mathcal{X} \cap \mathcal{C} \quad \text{and} \quad f(x^\star) = \inf_{x \in \mathcal{X}} f(x).$$

Proposition 2.2.1. *If $\mathcal{X} \subseteq \mathbb{R}^n$ is a convex set, and $f : \mathbb{R}^n \mapsto (-\infty, \infty]$ is a convex function, then a local minimum of f over $\mathcal{X}$ is also a global minimum. If in addition f is strictly convex, then there exists at most one global minimum of f over $\mathcal{X}$.*

Proof. The proof is presented in [91, p.118] □

Proposition 2.2.1 implies that if an optimization problem is convex, an algorithm that converges to a point satisfying the first-order necessary conditions (e.g. $\nabla f(x) = 0$, if $\mathcal{C} = \mathcal{X} = \mathbb{R}^n$ and f is differentiable) will also converge to the global optimum. Additionally, for many useful forms of convex optimization problem, including linear programs, quadratic programs, and semidefinite programs (subject to regularity conditions), algorithms exist that can approximate the solution to arbitrary accuracy in polynomial time [13]. These properties imply that convex optimization is, in a sense, 'easier' than nonconvex optimization. A further benefit of a convex formulation is that it can often be reliably solved using general purpose convex optimization software, so it may not be necessary to develop a dedicated optimization algorithm for a given problem. Consequently, convex optimization has been applied extensively to problems in optimal control [92, 93, 94, 95]. For a textbook covering the fundamental theory, applications, and algorithms of convex optimization, see [96].

SCENARIO OPTIMIZATION

Consider the chance constrained optimization problem

$$\begin{aligned} &\min_{x \in \mathbb{R}^n} c^\top x \\ &\quad \text{s.t. } x \in \mathcal{X} \\ &\qquad \mathbb{P}[x \in \mathcal{X}(\delta)] \geq \epsilon, \\ &\qquad \delta \in \Delta, \end{aligned} \tag{2.9}$$

where the constraint sets $\mathcal{X}$ and $\mathcal{X}(\delta)$ are closed and convex $\forall \delta \in \Delta$, and $\mathcal{X}(\delta)$ is a function of an uncertain parameter $\delta \in \Delta$ with an associated probability $\mathbb{P}$. Note that (2.9) can be used to represent an arbitrary convex cost function, f, if $c = (1, 0, \ldots, 0)$ and $\mathcal{X} \subseteq \{x : x_1 \geq f(x_2)\}$, and is a general form of both the stochastic MPC optimization problem (2.8) and the robust MPC optimization problem (2.7) (in the case of the latter $\epsilon = 1$).

In the general case, finding a solution to (2.9) with a high probability of constraint satisfaction is challenging (see [97] for a textbook addressing problems of the form of (2.9)). An alternative approach is to approximate (2.9) with $S \in \mathbb{Z}_{++}$ samples of the uncertainty as

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \; & c^\top x \\ \text{s.t. } & x \in \mathcal{X} \\ & x \in \bigcap_{i \in \{1,\ldots,S\}} \mathcal{X}(\delta_i) \end{aligned} \tag{2.10}$$

This is known as the scenario approach [98, 99], and results in a deterministic optimization problem that can now be approached using standard convex optimization algorithms. This significantly improves the tractability relative to the original stochastic program, and consequently, the scenario approach has been extensively investigated for designing optimal and model predictive controllers under stochastic uncertainty [100, 101, 102]. A limitation is that the solution of (2.10), $x^\star$, cannot now be guaranteed to satisfy the original chance constraint, $\mathbb{P}[x^\star \in \mathcal{X}(\delta)] \geq \epsilon$, as $x^\star$ is a random variable that depends on the chosen samples of the uncertainty set Δ. Theoretical treatments of the scenario approach therefore instead address issues such the confidence of constraint violation probability. For example, a seminal result in scenario optimization theory presented in [98] is that if the violation probability is defined as $V(x^\star) := \mathbb{P}\{\delta \in \Delta, x^\star \notin \mathcal{X}(\delta)\}$, then the confidence of a given violation probability is upper bounded by

$$\mathbb{P}\{V(x^\star) > \epsilon\} \leq \sum_{i=0}^{n-1} \binom{S}{i} \epsilon^i (1-\epsilon)^{S-i},$$

and that this bound is tight for *fully supported* problems (see [98] for further details). Results such as this provide a basis for selecting the number of scenarios, S, to provide an 'adequately' close approximation of the solution of (2.9).

EXAMPLES IN THE LITERATURE

Convex quadratic programs (in which the Hessian of the objective function is positive semi-definite) are a subclass of convex optimization problems, so all of the aforementioned

MPC approaches using quadratic programming are in fact using convex optimization. However, in order for the energy management problem to be formulated as a quadratic program, the model of the losses in the battery and motor needs to be linear (or linearized), rendering the obtained power-split suboptimal. The advantage of using more general convex optimization is that more descriptive models can be used for modelling the powertrain (for a more comprehensive overview of convex relaxations for hybrid electric powertrain modelling see [14] or [103]).

The design of a hybrid vehicle is dependent on the controller used for energy management, as the efficiency of the powertrain influences the required battery size. Consequently, a significant subset of the work on PHEV energy management using convex optimization has considered the problem of simultaneous component sizing and energy management. The first investigation using this approach was [17], where it was assumed that engine switching and gear selection were controlled using heuristics, and convex optimization was used to determine the optimal battery size and power-split between the battery and engine. This approach was extended in [104] and [105] to consider a PHEV with a battery/supercapacitor hybrid energy storage system. Engine switching was also optimized in [106] with an algorithm that alternated between using PMP to optimize the engine control sequence for a given power-split, and convex optimization to optimize the power split for a given engine control sequence. This approach was then extended to also optimize gear selection (using dynamic programming instead of PMP) in [107] and [108]. A continuously variable transmission was considered in [109], and an electric variable transmission was considered in [110]. Simultaneous sizing of the engine, motor, and battery was considered in [111], and simultaneous sizing, control, and charging was investigated in [112].

Other studies have approached the energy management problem alone using convex optimization. In [113], the power-split was controlled using convex optimization and compared against CDCS to investigate the effect of battery size on powertrain efficiency. Convex optimization was again combined with PMP to optimize both the power-split and engine switching in [114], and scenario optimization was used to consider multiple predictions of future driver behaviour in [80].

One of the limitations of the above approaches is that the convex optimization problem has typically been parsed with CVX and then solved using general purpose optimization software such as SDPT3 [115]. This limits the computational performance, and prevents a real-time solution: up to 13s were required in [114] for a horizon of ~ 1000 samples when optimizing the power-split alone. The only examples that could be found in the literature of optimization algorithms used to solve convex formulations of the energy management problem were the active set method presented in [116], and the interior point method presented in [117]. In both of these cases the losses in the battery were ignored and the

losses in the motor were modelled as linear functions, and as will be demonstrated in Chapter 4, these approximations necessarily render the obtained control inputs suboptimal.

2.3 Concluding Remarks

This chapter presents an illustrative formulation of the PHEV energy management problem, and a review of the common optimization-based methods for its solution and their limitations. It was demonstrated that dynamic programming can be used to solve the energy management problem for arbitrary models and control variables, but that it is too computationally demanding for a real-time online solution. This was contrasted with PMP, which is significantly computationally cheaper, but has limitations when enforcing more general constraints, and still requires an exact prediction of future driver behaviour. The ECMS was derived and shown to be a mathematically equivalent to PMP, and methods of adapting the costate/equivalence factor online using predictions of future driver behaviour were reviewed. Model-predictive control was introduced as an alternative method of including predictions of driver behaviour in a real-time, online framework, but was shown to be limited by the computation required to solve the associated optimization problem online. Finally, convex optimization was introduced as a method for forming computationally tractable energy management problems, and it was highlighted that there is a lack of studies that have investigated tailored algorithms for efficiently solving the resulting optimization problem.

CHAPTER 3

TERMINAL STATE CONSTRAINT

This chapter presents the first MPC formulation in which the state-of-charge of the battery is subject to a terminal constraint only. The chapter starts with a detailed description of the PHEV model used throughout this thesis, after which the MPC optimization is formulated using an approximate model. It is then demonstrated that the MPC optimization is convex under negligible further approximation when considered in terms of the battery's discharge rate. The MPC algorithm is then compared in simulation against PMP where it is shown to obtain near-optimal performance, but is also limited by the lack of general state constraints.

3.1 VEHICLE MODEL

This section presents the PHEV model used for simulation, for which the key parameters are summarized in Table 3.1. Figure 3.1 illustrates the transfers of power in the PHEV powertrain model used for simulation, which will now be formulated in discrete time. The velocity of the vehicle, $v(t) \in \mathbb{R}$, and road gradient, $\theta(t) \in \mathbb{R}$, for a given journey are sampled discretely for all $t \in \{0, \dots, T-1\}$, where $T \in \mathbb{Z}_{++}$ is the duration of the journey in seconds. This implies that the samples are taken at a uniform frequency of 1Hz, which is a common sampling rate used for energy management problems (e.g. [79, 82, 78, 84, 80]), although the approach presented here can be readily extended to arbitrary sampling intervals.

ROTATION

The rotational speed of the wheels, $\omega_w(t) \in \mathbb{R}$, is defined by

$$\omega_w(t) := \frac{v(t)}{r_w} \quad \forall t \in \{0, \dots, T-1\}, \tag{3.1}$$

where $r_w \in \mathbb{R}_{++}$ is the effective radius of the wheels. The first control variable considered is the gear ratio, $r(t)$, and the vehicle is modelled with a discretely variable transmission so

Table 3.1: Model Parameters for a Typical Passenger Vehicle

Parameter	Value
Vehicle mass (m_v)	1900 kg
Drag coefficient (C_D)	0.27
Rolling resistance coefficient (C_r)	0.015
Motor power rating	50 kW
Motor torque limits ($\overline{T}, \underline{T}$)	± 200 Nm
Engine power rating	100 kW
Engine torque limits ($\overline{T}, \underline{T}$)	(400, 0) Nm
Battery capacity	8 kWh
Battery current limits ($\bar{i}$, $\underline{i}$)	$\pm$200 A

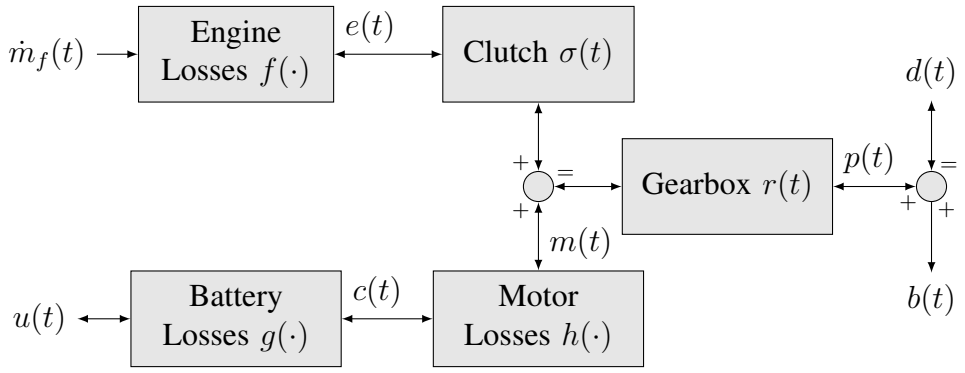


Figure 3.1: Diagram of the loss functions and variables that are used in the mathematical formulation of the energy management problem (duplicate of Figure 2.2 for convenience).

that the gear ratio is selected as $r(t) \in \{\hat{r}_1, \ldots, \hat{r}_g\}\ \forall t \in \{0, \ldots, T-1\}$, where $g \in \mathbb{Z}_{++}$ is the number of available gear ratios given by $\hat{r}_i \in \mathbb{R}_{++}\ \forall i \in \{1, \ldots, g\}$. The dynamics of gear selection are ignored as they are assumed to be significantly faster than the 1s sample period, so the rotational speed of the motor, $\omega_m(t) \in \mathbb{R}$, is defined by

$$\omega_m(t) := \frac{\omega_w(t)}{r(t)} \quad \forall t \in \{0, \ldots, T-1\}. \tag{3.2}$$

The second control variable considered is the clutch engagement, $\sigma(t) \in \{0, 1\}\ \forall t \in \{0, \ldots, T-1\}$, which is coupled to the engine state so that $\sigma(t) = 1$ implies that the clutch is engaged and the engine is on, and $\sigma(t) = 0$ implies that the clutch is disengaged and the engine is off. The engine switching and clutch engagement dynamics are also ignored as they are assumed to be too fast to be captured by the 1s sampling period. It is assumed (without loss of generality) that the torque coupling device does not have an effective gear ratio, so the rotational speed of the engine, $\omega_e(t) \in \mathbb{R}$, is defined by

$$\omega_e(t) := \begin{cases} \omega_m(t) & \sigma(t) = 1 \\ 0 & \sigma(t) = 0 \end{cases}, \quad \forall t \in \{0, \ldots, T-1\}. \tag{3.3}$$

POWER

The driver's power demand (which is externally imposed, and not a decision variable), $d(t) \in \mathbb{R}$, is defined using a longitudinal model (commonly used for energy management, e.g. [34, 32, 38, 37, 84]) as

$$d(t) := \Big[m_v \dot{v}(t) + \frac{1}{2}\rho_a C_D A_v v(t)^2 + C_r m_v g \cos\theta(t) + m_v g \sin\theta(t)\Big] v(t) \tag{3.4}$$

$\forall t \in \{0, \ldots, T-1\}$, where $m_v \in \mathbb{R}_+$ is the vehicle's mass (assumed constant), $\dot{v}(t) \in \mathbb{R}$ is the vehicle's acceleration (which can either be estimated from the velocity using numerical differentiation, or sampled itself), $\rho_a \in \mathbb{R}_+$ is the density of the atmosphere, $C_D \in \mathbb{R}_+$ is the vehicle's drag coefficient, $A_v \in \mathbb{R}_+$ is the vehicle's frontal area, $C_r \in \mathbb{R}_+$ is the vehicle's rolling resistance coefficient, and $g \in \mathbb{R}_{++}$ is the acceleration due to gravity. The third control variable considered in this framework is the mechanical braking power, $b(t) \in \mathbb{R}_-$, which is nonpositive and combined additively with the power delivered by the powertrain, $p(t) \in \mathbb{R}$, to meet the driver demand power:

$$d(t) := p(t) + b(t), \quad b(t) \leq 0, \quad \forall t \in \{0, \ldots, T-1\}. \tag{3.5}$$

The drivetrain components are modelled as 100% efficient[1], and the power delivered by the engine (through the clutch), $e(t) \in \mathbb{R}$, is combined additively with the power delivered by the motor, $m(t) \in \mathbb{R}$, and delivered to the drivetrain so that

$$p(t) := \begin{cases} e(t) + m(t) & \sigma(t) = 1 \\ m(t) & \sigma(t) = 0 \end{cases}, \quad \forall t \in \{0, \dots, T-1\}. \tag{3.6}$$

The fourth control variable is therefore the engine power, as the motor output power, $m(t)$, is then implicitly defined by (3.6).

The rate of fuel consumption, $\dot{m}_f(t) \in \mathbb{R}_+$, is modelled as a static map, $f : \mathbb{R} \times \mathbb{R} \mapsto \mathbb{R}_+$, of engine speed and power (as used in, for example, [82, 78, 80, 46, 49]) as

$$\dot{m}_f(t) := f(\omega_e(t), e(t)) \quad \forall t \in \{0, \dots, T-1\}. \tag{3.7}$$

The motor's input power, $c(t) \in \mathbb{R}$, is also modelled as a static map, $h : \mathbb{R} \times \mathbb{R} \mapsto \mathbb{R}$, of motor speed and power (as used in, for example, [82, 80, 46, 49]) as

$$c(t) := h(\omega_m(t), m(t)) \quad \forall t \in \{0, \dots, T-1\}. \tag{3.8}$$

The fuel mass flow and input power maps are illustrated in Figure 3.2, and obtained from experimental data for a 100 kW internal combustion engine and 50 kW electric motor.

An 8 kWh battery is modelled using an equivalent circuit with internal resistance only (as used in [37, 38, 32, 34, 51, 82, 78, 80]), where the open circuit voltage and resistance are functions, $V_{oc} : \mathbb{R} \mapsto \mathbb{R}$ and $R_{oc} : \mathbb{R} \mapsto \mathbb{R}$, of the battery's internal energy, $x(t) \in \mathbb{R}$, as illustrated in Figure 3.3. The power electronics connecting the battery to the motor are assumed to be 100% efficient, so the internal power of the battery, $u(t) \in \mathbb{R}$, is defined as a function, $g : \mathbb{R} \mapsto \mathbb{R}$, of motor output power and speed

$$u(t) = g(\omega_m(t), m(t)) := \frac{V_{oc}(x(t))^2}{2R_{oc}(x(t))} \left(1 - \sqrt{1 - \frac{4R_{oc}(x(t))h(\omega_m(t), m(t))}{V_{oc}(x(t))^2}}\right) \tag{3.9}$$

$\forall t \in \{0, \dots, T-1\}$. The power is modelled as constant between sampling intervals, so the internal energy of the battery, $x(t) \in \mathbb{R}$, is defined by

$$x(t) := x(0) - \sum_{i=0}^{t-1} u(0), \quad \forall t \in \{1, \dots, T\}, \tag{3.10}$$

where $x(0) \in \mathbb{R}$ is the battery's initial internal energy. Note that there is a one-to-one mapping between the battery's internal energy (e.g. in megajoules) and state-of-charge (as a percentage of its total capacity), and these terms are therefore used interchangeably.

[1]The drivetrain efficiency can be set arbitrarily using the proposed method, so long as a fixed fraction of negative power is provided by the mechanical brakes, as proposed in the following section.

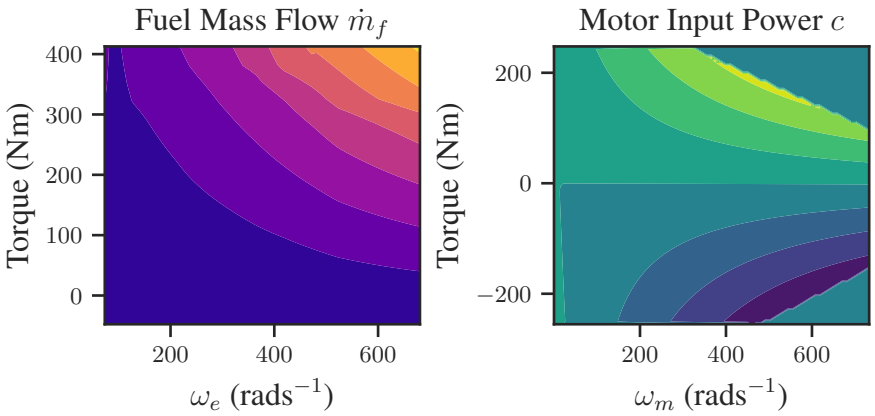
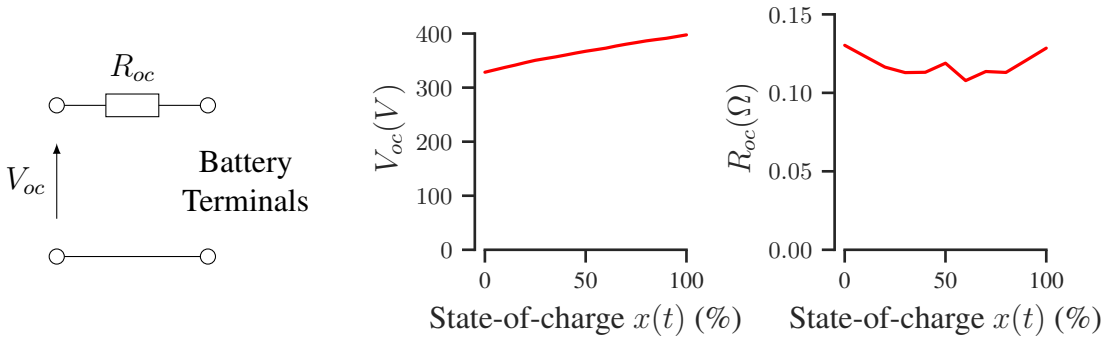


Figure 3.2: Experimental data used as the engine and motor maps for simulation (scale not provided as the data is proprietary).



(a) Equivalent circuit model.

(b) Equivalent circuit parameters, obtained from experimental data.

Figure 3.3: Battery equivalent circuit model

CONSTRAINTS

The engine is subject to upper and lower torque constraints, $\overline{T}_e : \mathbb{R} \mapsto \mathbb{R}$ and $\underline{T}_e : \mathbb{R} \mapsto \mathbb{R}$, that may be functions of rotational speed, so that

$$\underline{T}_e(\omega_e(t))\omega_e(t) \leq e(t) \leq \overline{T}_e(\omega_e(t))\omega_e(t) \quad \forall t \in \{0, \ldots, T-1\}. \tag{3.11}$$

Similarly, the motor is subject to upper and lower torque constraints, $\overline{T}_m : \mathbb{R} \mapsto \mathbb{R}$ and $\underline{T}_m : \mathbb{R} \mapsto \mathbb{R}$, that may be functions of rotational speed, so that

$$\underline{T}_m(\omega_m(t))\omega_m(t) \leq m(t) \leq \overline{T}_m(\omega_m(t))\omega_m(t) \quad \forall t \in \{0, \ldots, T-1\}. \tag{3.12}$$

The battery has upper and lower current limits, $\bar{i} \in \mathbb{R}$ and $\underline{i} \in \mathbb{R}$, so that

$$\underline{i}V_{oc}(x(t)) \leq u(t) \leq \bar{i}V_{oc}(x(t)) \quad \forall t \in \{0, \ldots, T-1\}. \tag{3.13}$$

The rotational speed of the engine and motor are constrained by lower and upper limits, $\underline{\omega}_m, \overline{\omega}_m, \in \mathbb{R}$ and $\underline{\omega}_e, \overline{\omega}_e(t) \in \mathbb{R}$,

$$\underline{\omega}_m \leq \omega_m(t) \leq \overline{\omega}_m, \quad \underline{\omega}_e \leq \omega_m(t) \leq \overline{\omega}_e \quad \forall t \in \{0, \ldots, T-1\} \tag{3.14}$$

Finally, the battery has energy limits, $\underline{x} \in \mathbb{R}$ and $\overline{x} \in \mathbb{R}$, such that

$$\underline{x} \leq x(t) \leq \overline{x} \quad \forall t \in \{1, \ldots, T\}. \tag{3.15}$$

The energy limits are generally not set equal to the battery's maximum and minimum capacity, as these charge states are unsafe.

ENERGY MANAGEMENT OPTIMIZATION

The objective is to minimize the total fuel consumed during the journey, so the energy management optimal control problem is

$$\min_{\substack{e(t),b(t),r(t),\sigma(t) \\ \forall t \in \{0,\ldots,T\}}} \sum_{t=0}^{T-1} \dot{m}_f(t)$$
$$\text{s.t. } (3.1) - (3.15).$$

As previously discussed, there are several limitations with approaching the energy management problem this way, including the fact that the solution at the start of the journey is dependent on a future prediction of driver behaviour, and that the optimization is challenging to solve in general, even if the future power demand is known exactly. A model predictive controller is now proposed where the control inputs, $e(t)$, $b(t)$, $r(t)$, and $\sigma(t)$, applied at each interval, t, are obtained from the solution of a simplified optimization problem.

3.2 MODEL PREDICTIVE CONTROLLER

PREDICTION

It is assumed that a prediction of the vehicle's velocity and the road gradient is available to the controller over a shrinking horizon of N timesteps (where $N = T - t$ and t is the time when the prediction is made) as $v := (v_0, \dots, v_{N-1}) \in \mathbb{R}^{N-1}$ and $\theta := (\theta_0, \dots, \theta_{N-1}) \in \mathbb{R}^{N-1}$. Additionally, it is assumed that the current power demand and wheel speed, d_0 and $\omega_{w,0}$, can be accurately measured using the vehicle's onboard sensors. Note that, for an example variable v, the notation $v(t)$ used to represent the simulated value of v at time t, whereas the subscript notation v_k is used to represent the value of $v(t + k)$ *predicted* at time t within the MPC framework. The future driver power demand, d_k, and rotational speed of the wheels, $\omega_{w,k}$, can then be determined using v and θ similarly to (3.1) and (3.4) for all $k \in \{0, \dots, N - 1\}$.

HEURISTICS

For the case considered in this chapter, only the relative fraction of power delivered by the engine and motor is considered by the MPC controller (i.e. the control vector is reduced to just u), and the other variables are completely determined by heuristics. The gear selection control, $r_k \in \mathbb{R}$, is defined by the heuristic

$$r_k := \begin{cases} \hat{r}_1 & \bar{v}_1 \leq v_t \leq \bar{v}_2 \\ \vdots & \vdots \\ \hat{r}_g & \bar{v}_g \leq v_t \leq \infty \end{cases}, \quad \forall k \in \{0, \dots, N-1\} \tag{3.16}$$

where $(\bar{v}_1, \dots, \bar{v}_g) \in \mathbb{R}^g$ is a vector of threshold velocities. With the gear selection controller chosen, the rotational speed of the motor is defined using (3.16) and a similar form of (3.2) $\forall k \in \{0, \dots, N - 1\}$. The intervals, k, within the prediction horizon are then separated into the sets

$$\mathcal{P} := \{k \in \{0, \dots, N-1\} : \omega_{m,k} \geq \underline{\omega}_e\}, \quad \mathcal{C} := \{0, \dots, N-1\} \setminus \mathcal{P},$$

i.e. $\mathcal{P}$ is the set of timesteps, k, where the motor speed is greater than or equal to the minimum engine speed, and $\mathcal{C}$ is the set of times where the motor speed is lower than the minimum engine speed. The clutch and engine control heuristic is then defined by

$$\sigma_k := \begin{cases} 1 & k \in \mathcal{P} \\ 0 & k \in \mathcal{C} \end{cases} \quad \forall k \in \{0, \dots, N-1\}. \tag{3.17}$$

The predicted rotational speed of the engine, $\omega_{e,k}$, is now well defined by (3.17) and a similar form of (3.3) for all $k \in \{0, \dots, N-1\}$. Finally, the braking power is set at a fixed fraction, $k_b \in \mathbb{R}_+$, of the negative driver demand power,

$$b_k := \begin{cases} 0 & d_k \geq 0 \\ k_b d_k & d_k < 0 \end{cases} \quad \forall k \in \{0, \dots, N-1\}, \tag{3.18}$$

so the power required from the drivetrain, p_k, can now be defined using (3.18) and a similar form of (3.5) for all $k \in \{0, \dots, N-1\}$.

It is likely that better heuristics exist, and the optimal gear selection, engine switching, and braking could be estimated using an external optimization routine (such as those used in [106, 107]), but the purpose of this chapter is to demonstrate that the resulting problem is convex, so the choice of control for these variables is arbitrary.

LOSSES

The engine and motor input-output maps are approximated by the functions

$$f(\omega_{e,k}, e_k) \approx \alpha_2(\omega_{e,k}) e_k^2 + \alpha_1(\omega_{e,k}) e_k + \alpha_0(\omega_{e,k})$$
$$h(\omega_{m,k}, m_k) \approx \beta_2(\omega_{m,k}) m_k^2 + \beta_1(\omega_{m,k}) m_k + \beta_0(\omega_{m,k})$$

where $\alpha_2(x) > 0 \; \forall x \in \mathrm{dom}(\alpha_2)$ and $\beta_2(x) > 0 \; \forall x \in \mathrm{dom}(\beta_2)$. This is a common approach to energy management using convex optimization (e.g. [112, 113, 17, 104, 111, 118, 105, 116, 107]), and typically provides a very close approximation of the experimental loss maps (see, for example, [116, §2], [17, §7]), [107, §2], or [112, §2]). The fuel consumption rate and input power to the motor, $\dot{m}_{f,k} \in \mathbb{R}$ and $c_k \in \mathbb{R}$, can then be defined as functions, $f_k : \mathbb{R} \mapsto \mathbb{R}$ and $h_k : \mathbb{R} \mapsto \mathbb{R}$, of engine power and motor power as

$$\dot{m}_{f,k} := f_k(e_k) := \begin{cases} \alpha_{2,k} e_k^2 + \alpha_{1,k} e_k + \alpha_{0,k} & k \in \mathcal{P} \\ 0 & k \in \mathcal{C} \end{cases} \tag{3.19a}$$

$$c_k := h_k(m_k) := \beta_{2,k} m_k^2 + \beta_{1,k} m_k + \beta_{0,k} \quad \forall k \in \{0, \dots, N-1\}, \tag{3.19b}$$

using the coefficients $\alpha_{i,k} := \alpha_i(\omega_{e,k}) \; \forall i \in \{0,1,2\} \; \forall k \in \mathcal{P}$ and $\beta_{i,k} := \beta_i(\omega_{m,k}) \; \forall i \in \{0,1,2\} \; \forall k \in \{0, \dots, N-1\}$.

It is assumed that the battery's open circuit voltage, $V_{oc,k} \in \mathbb{R}$, and resistance, $R_{oc,k} \in \mathbb{R}$, are constant across the prediction horizon at their current values, i.e.

$$V_k := V(x(t)) \quad R_k := R(x(t)) \quad \forall k \in \{0, \dots, N-1\}.$$

This is also an assumption that is commonly made for convex optimization-based energy management [17, 114, 106, 104], and Figure 3.3 illustrates that it results in a negligible approximation for moderate changes in the battery's internal energy. The predicted rate of depletion of the battery's internal energy, $u_k \in \mathbb{R}$, is then given by

$$u_k := g_k(m_k) := \frac{V_k^2}{2R_k}\left(1 - \sqrt{1 - \frac{4R_k h_k(m_k)}{V_k^2}}\right) \quad \forall t \in \{0, \dots, T-1\}. \tag{3.20}$$

Note that, under the condition that $\beta_{2,k} > 0$ for all k (which is true as $\beta_2(x) > 0$ by definition), the functions $g_k(m_k)$ are all convex (within a suitable domain). This can be shown using the fact that $-\sqrt{x}$ is strictly convex and strictly decreasing, and that every function $-h_k(m_k)$ is strictly concave and even (with respect to its maximizing argument). This result is not used within this thesis and is therefore not proven in detail.

CONSTRAINTS

The power required from the motor is defined by

$$m_k := \begin{cases} p_k - e_k & k \in \mathcal{P} \\ p_k & k \in \mathcal{C} \end{cases}, \quad \forall k \in \{0, \dots, N-1\}, \tag{3.21}$$

and subject to constraints $\underline{m}_k \leq m_k \leq \overline{m}_k$ defined by

$$\begin{aligned} \underline{m}_k &:= \begin{cases} \max\left\{\underline{T}_m(w_{m,k})w_{m,k}, \min \mathcal{Z} + \delta\right\} & k \in \mathcal{P} \\ p_k & k \in \mathcal{C} \end{cases} \\ \overline{m}_k &:= \begin{cases} \min\left\{\overline{T}_m(w_{m,k})w_{m,k}, \max \mathcal{Z} - \delta\right\} & k \in \mathcal{P} \\ p_k & k \in \mathcal{C} \end{cases}, \end{aligned} \tag{3.22}$$

$\forall k \in \{0, \dots, N-1\}$, where $\mathcal{Z} := \left\{m_k \in \mathbb{R} : 1 - \frac{4Rh_k(m_k)}{V^2} \geq 0\right\}$ is the interval in which a real solution of (3.20) exists, and $\delta \in \mathbb{R}_{++}$ is an arbitrarily small perturbation (the reason for including this perturbation is discussed later in Remark 3.2.2, and it is for now assumed to be equal to zero).

Remark 3.2.1 (Existence of $\min \mathcal{Z}$ and $\max \mathcal{Z}$)**.** The set $\mathcal{Z}$ is non-empty and closed if

$$1 \geq \frac{4R}{V^2}\left(\beta_{0,k} - \frac{\beta_{1,k}^2}{4\beta_{2,k}}\right). \tag{3.23}$$

For all operating regions of the motor data shown in Figure 3.2 it was found that $\beta_2 \approx 1 \times 10^{-5}$, $\beta_1 \approx 1$ and $\beta_0 \approx 0$, so (3.23) is always satisfied.

As the engine is off $\forall k \in \mathcal{C}$, and the constraints on engine and motor power can be consolidated into constraints on just engine power using (3.21) $\forall k \in \mathcal{P}$, the constraints $\underline{e}_k \leq e_k \leq \overline{e}_k$ are defined by

$$\underline{e}_k := \begin{cases} \max\left\{\underline{T}_e(w_{e,k})w_{e,k},\ p_k - \overline{m}_k\right\} & k \in \mathcal{P} \\ 0 & k \in \mathcal{C} \end{cases}$$
$$\overline{e}_k := \begin{cases} \min\left\{\overline{T}_e(w_{e,k})w_{e,k},\ p_k - \underline{m}_k\right\} & k \in \mathcal{P} \\ 0 & k \in \mathcal{C} \end{cases},$$

$\forall k \in \{0, \ldots, N-1\}$. Finally, the battery's internal power is constrained by

$$\underline{i}V_k \leq u_k \leq \overline{i}V_k \quad \forall k \in \{0, \ldots, N-1\}.$$

For all of the numerical experiments presented in this thesis, the same sampling frequency is used for both the simulation and MPC predictions (i.e. 1Hz), although this MPC framework can also be trivially extended to an arbitrary prediction sampling interval, which does not need to be the same as that used for the simulations. It is again assumed that the power is constant between sampling intervals, so the predicted internal energy of the battery, $x_k \in \mathbb{R}$, for the MPC optimization at time t is defined by

$$x_k := x(t) - \sum_{i=0}^{k-1} u_i \quad \forall k \in \{1, \ldots, N\}. \tag{3.24}$$

For the approach used in this chapter, the general state-of-charge constraint (3.15) is ignored, and only the constraint

$$x(t) - \sum_{i=0}^{N-1} u_i = X_T$$

is considered, where $X_T \in [\underline{x}, \overline{x}]$ is the target terminal state-of-charge. The terminal state of charge is useful for optimal energy management controllers for several reasons: if the optimal state-of-charge trajectory decreases (near) linearly throughout the journey, the optimal controller (in terms of minimizing fuel consumption) can be obtained with only the terminal state-of-charge constraint (it will later be shown that this condition does not hold in general). It will also be desirable when controlling a *HEV* powertrain for the state-of-charge to be the same at the start and end of the journey. More generally, a terminal state-of-charge constraint allows the relative efficiency of two candidate energy management strategies to be determined by simply comparing their total fuel consumption.

OPTIMIZATION

The objective is to minimize fuel consumption subject to the above constraints, and the model predictive controller is defined by $e(t) := e_0^\star$, where $e_0^\star$ is the first element of the vector $e := (e_0, \dots, e_{N-1})$ that minimizes

$$\begin{aligned} &\min_{e} \sum_{i=0}^{N-1} f_k(e_k), \\ &\text{s.t.} \left.\begin{array}{l} \underline{e}_k \le e_k \le \overline{e}_k \\ m_k = p_k - e_k \\ u_k = g_k(m_k) \\ \underline{i}V \le u_k \le \overline{i}V \end{array}\right\} \forall k \in \{0, \dots, N-1\} \\ &\qquad x(t) - \sum_{i=0}^{N-1} u_i = X_T. \end{aligned} \tag{MPC.1a}$$

Note that m_k and e_k are determined implicitly by any given value of u_k, and all other variables are fixed parameters. Problem (MPC.1a) is nonconvex in general due to the nonlinear equality constraint $u_k = g_k(m_k)$, defined in (3.20).

3.2.1 CONVEX FORMULATION

A convex reformulation of (MPC.1a) is now presented. This is not the first derivation of a convex energy management problem, and it has previously been demonstrated that if the constraint $u_k = g_k(m_k)$ is relaxed to the inequality constraint $u_k \ge g_k(m_k)$ the resulting problem is convex, and the constraint will be satisfied with equality at the solution [14]. The approach presented here is instead to reformulate the problem so that the rate of change of the battery's internal energy, u, is the decision variable. This method is computationally preferable when the MPC problem is extended to the general battery constraint in Chapter 4, as the previous approach then introduces N nonlinear inequality constraints whereas the proposed method introduces N affine inequality constraints.

MATHEMATICAL PRELIMINARIES

Definition 3.2.1 (Increasing Function)**.** A function $f : \mathcal{X} \mapsto \mathcal{Y}$ (where $\mathcal{X} \subseteq \mathbb{R}$ and $\mathcal{Y} \subseteq \mathbb{R}$) is increasing iff

$$x_1 > x_2 \implies f(x_1) \ge f(x_2) \quad \forall x_1, x_2 \in \mathcal{X}.$$

A function is *strictly increasing* iff

$$x_1 > x_2 \implies f(x_1) > f(x_2) \quad \forall x_1, x_2 \in \mathcal{X}.$$

Lemma 3.2.1. *The composition of a concave function, g, and a decreasing convex function, f, is convex.*

Proof. The property that g is concave implies that

$$g(\lambda x_1 + (1-\lambda)x_2) \geq \lambda g(x_1) + (1-\lambda)g(x_2) \quad \forall x_1, x_2 \in \mathrm{dom}(g),\ \forall \lambda \in [0,1].$$

The property that f is decreasing and convex can then be used to show that

$$\begin{aligned} f\big(g(\lambda x_1 + (1-\lambda)x_2)\big) &\leq f\big(\lambda g(x_1) + (1-\lambda)g(x_2)\big) && (g \text{ concave, } f \text{ decreasing}) \\ &\leq \lambda f\big(g(x_1)\big) + (1-\lambda) f\big(g(x_2)\big) && (f \text{ convex}) \end{aligned}$$

$\forall x_1, x_2 \in \mathrm{dom}(g),\ \forall \lambda \in [0,1]$, which concludes the proof. □

Lemma 3.2.2. *The composition of a strictly concave function, g, and a strictly decreasing, strictly convex function, f, is strictly convex.*

Proof. Can be shown with minor changes to the proof of Lemma 3.2.1. □

Lemma 3.2.3. *The composition of a strictly convex function, g, and a strictly increasing, strictly convex function, f, is strictly convex.*

Proof. Can be shown with minor changes to the proof of Lemma 3.2.1. □

CONVEX REFORMULATION

Proposition 3.2.1. *Redefine the lower limit on engine power as*

$$\underline{e}_k :\leftarrow \max\left\{\underline{e}_k, -\frac{\alpha_{1,k}}{2\alpha_{2,k}}\right\} \quad \forall k \in \mathcal{P}, \tag{3.25}$$

so that e_k is constrained to the domain of $f_k(e_k)$ in which it is an increasing function. Additionally, choose an arbitrarily small, but positive value of δ in (3.22), then redefine the limits on motor power as

$$\underline{m}_k :\leftarrow \max\left\{\underline{m}_k, -\frac{\beta_{1,k}}{2\beta_{2,k}}, p_k - \overline{e}_k\right\} \quad \overline{m}_k :\leftarrow \min\left\{\overline{m}_k, p_k - \underline{e}_k\right\} \quad \forall k \in \mathcal{P}. \tag{3.26}$$

These new constraints ensure that m_k is constrained to the domain of $h_k(m_k)$ in which it is an increasing function, and so that the limits on engine power are also satisfied when $e_k = p_k - m_k$. Under the above modifications, problem (MPC.1a) is equivalent to the convex optimization problem

$$\begin{aligned} \min_{u} \ & \sum_{i=0}^{N-1} F_k(u_k), \\ \textit{s.t. } & x(t) - \sum_{i=0}^{N-1} u_i = X_N, \\ & \underline{u}_k \leq u_k \leq \overline{u}_k \quad \forall k \in \{0, \dots, N-1\}, \end{aligned} \tag{MPC.1b}$$

where

$$F_k(u_k) := f_k(p_k - g_k^{-1}(u_k)),$$

and $\underline{u}_k \in \mathbb{R}$ and $\overline{u}_k \in \mathbb{R}$ are lower and upper limits on the battery power defined by

$$\underline{u}_k := \begin{cases} \max\{\underline{i}V, g_k(\underline{m}_k)\} & k \in \mathcal{P} \\ g_k(p_k) & k \in \mathcal{C} \end{cases} \quad \overline{u}_k := \begin{cases} \min\{\bar{i}V, g_k(\overline{m}_k)\} & k \in \mathcal{P} \\ g_k(p_k) & k \in \mathcal{C} \end{cases}$$

$\forall k \in \{0, \dots, N-1\}$.

The principle of Proposition 3.2.1 is that problem (MPC.1a) becomes convex under the change of variables $e_k = p_k - g_k^{-1}(u_k)$, subject to the specified constraint modifications. The proof of Proposition 3.2.1 is constructed in the three Lemmas and the Theorem that follow; the eventual result is that the functions $F_k(u_k)$ are convex, so (MPC.1b) is convex as it is only subject to affine equality and inequality constraints.

Lemma 3.2.4. *For every $k \in \mathcal{P}$, the function $f_k : [\underline{e}_k, \overline{e}_k] \mapsto \mathbb{R}$ defined by (3.19a) is strictly convex and strictly increasing (Definition 3.2.1).*

Proof. f_k is quadratic with positive second derivative ($\alpha_{2,k} > 0 \ \forall k \in \{0, \dots, N-1\}$) so is strictly convex and has a unique minimizing argument of $-\frac{\alpha_{1,k}}{2\alpha_{2,k}}$. Furthermore, $\underline{e}_k \geq -\frac{\alpha_{1,k}}{2\alpha_{2,k}}$, so f_k is strictly increasing on the interval $[\underline{e}_k, \overline{e}_k]$. □

Lemma 3.2.5. *For every $k \in \mathcal{P}$, the function $g_k : [\underline{m}, \overline{m}] \mapsto [g_k(\underline{m}_k), g_k(\overline{m}_k)]$ defined by (3.20) is strictly increasing, strictly convex, twice differentiable, and one-to-one.*

Proof. The function $\hat{h}_k(m_k) := 1 - \frac{4Rh_k(m_k)}{V^2}$ is quadratic with negative second derivative ($\beta_{2,k} > 0 \ \forall k \in \{0, \dots, N-1\}$) so is also concave and has a unique maximising argument

$-\frac{\beta_{1,k}}{2\beta_{2,k}}$. Note that $\underline{m} \geq -\frac{\beta_{1,k}}{2\beta_{2,k}} > \min \mathcal{Z}$ and $\overline{m} < \max \mathcal{Z}$, so $\hat{h}_k(m_t)$ is positive and strictly decreasing on the interval $[\underline{m}, \overline{m}]$. The function $-\sqrt{x}$ is strictly convex and strictly decreasing on the domain $\mathbb{R}_+$, so the composite function $-\sqrt{\hat{h}_k(m_k)}$ is strictly increasing (the composition of two strictly decreasing functions is necessarily strictly increasing) and strictly convex (Lemma 3.2.2). The additional constant and positive coefficient in the definition of g_k preserve this result. The derivatives of g_k are given by

$$g_k'(m_k) = -\frac{V\hat{h}'(x)}{4R\sqrt{\hat{h}(m_t)}}$$

and

$$g_k''(m_k) = \frac{V}{4R\sqrt{\hat{h}(m_t)}}\left(\hat{h}_k''(m_k) - \frac{\hat{h}_k'(m_k)}{2(\hat{h}_k(m_k))^2}\right)$$

which exist as $\hat{h}_k$ is positive on the interval $[\underline{m}, \overline{m}]$ and is twice differentiable, so g_k is twice differentiable.

The property that g_k is continuous (the composition of two continuous functions is necessarily continuous) implies that g_k maps $[\underline{m}, \overline{m}]$ to every element in $[g_k(\underline{m}_k), g_k(\overline{m}_k)]$, and the property that g_k is strictly increasing therefore implies that g_k maps $[\underline{m}, \overline{m}]$ uniquely to every element in $[g_k(\underline{m}_k), g_k(\overline{m}_k)]$. Therefore, g_k is one-to-one on the domain $[\underline{m}, \overline{m}]$ and range $[g_k(\underline{m}_k), g_k(\overline{m}_k)]$. □

Lemma 3.2.6. *The function* $g_k^{-1} : [g_k(\underline{m}_k), g_k(\overline{m}_k)] \mapsto [\underline{m}, \overline{m}]$*, defined by*

$$g_k^{-1}(u_k) := -\frac{\beta_{1,k}}{2\beta_{2,k}} + \sqrt{-\frac{Ru_k^2}{\beta_{2,k}V^2} + \frac{u_k - \beta_{0,k}}{\beta_{2,k}} + \frac{\beta_{1,k}^2}{4\beta_{2,k}^2}}$$

is strictly increasing, strictly concave, and twice differentiable.

Proof. The property that g_k is strictly convex implies that

$$\lambda g_k(x_1) + (1-\lambda)g_k(x_2) > g_k(\lambda x_1 + (1-\lambda)x_2) \quad \forall x_1, x_2 \in [\underline{m}, \overline{m}]\ \forall \lambda \in [0,1]. \tag{3.27}$$

The inverse map is strictly increasing (this can be trivially shown from the definition of a strictly increasing function), which implies that

$$y_1 > y_2 \implies g_k^{-1}(y_1) > g_k^{-1}(y_2) \quad \forall y_1, y_2 \in \mathrm{dom}(g_k^{-1}). \tag{3.28}$$

If $y = g_k(x)$, then

$$\begin{aligned} g_k^{-1}(\lambda y_1 + (1-\lambda)y_2) &= g_k^{-1}(\lambda g_k(x_1) + (1-\lambda)g_k(x_2)) \\ &> g_k^{-1}(g_k(\lambda x_1 + (1-\lambda)x_2)) \qquad \text{(using (3.27) and (3.28))} \\ &= \lambda g_k^{-1}(y_1) + (1-\lambda)g_k^{-1}(y_2), \end{aligned}$$

$\forall y_1, y_2 \in [g_k(\underline{m}_k), g_k(\overline{m}_k)]$ and $\lambda \in [0,1]$, which proves that g_k^{-1} is strictly concave. Finally,

$$\frac{\mathrm{d}x}{\mathrm{d}y} = \frac{1}{g_k'(x)}$$

and

$$\frac{\mathrm{d}^2x}{\mathrm{d}y^2} = \frac{-1}{(g_k'(x))^2}\frac{\mathrm{d}}{\mathrm{d}y}g_k'(x) = \frac{-1}{(g_k'(x))^2}g_k''(x)\frac{\mathrm{d}x}{\mathrm{d}y} = \frac{-g_k''(g_k^{-1}(y))}{(g_k'(g_k^{-1}(y)))^3},$$

which both exist on the interval $[g_k(\underline{m}_k), g_k(\overline{m}_k)]$, so g_k^{-1} is twice differentiable. □

Theorem 3.2.1. *For all $k \in \mathcal{P}$, the composite function $F_k : [g_k(\underline{m}_k), g_k(\overline{m}_k)] \mapsto \mathbb{R}$ defined by*

$$F_k(u_k) := f_k(p_k - g_k^{-1}(u_k))$$

is strictly decreasing, strictly convex, and twice differentiable.

Proof. Firstly, note that $\{e_k : e_k = p_k - m_k, m_k = g_k^{-1}(u_k), u_k \in [g_k(\underline{m}_k), g_k(\overline{m}_k)]\} \subseteq [\underline{e}_k, \overline{e}_k]$, so $f_k(\cdot)$ is strictly increasing and strictly convex (Lemma 3.2.4). Furthermore, $p_k - g_k^{-1}(\cdot)$ is strictly decreasing and strictly convex (immediately follows from Lemma 3.2.6 and the negative sign preceding g_k^{-1}). Therefore, F_k is strictly decreasing (the composition of a strictly decreasing function and a strictly increasing function is necessarily strictly decreasing) and strictly convex (Lemma 3.2.3). Twice differentiability can be demonstrated using the chain rule with the first and second derivatives of f_k and g_k^{-1}, which all exist on the specified domain. □

Theorem 3.2.1 therefore implies that (MPC.1b) is convex, as both the objective function and constraint set are convex with respect to $(u_0, \ldots, u_{N-1})$, which therefore completes the proof of Proposition 3.2.1. This section is now concluded with an analysis of the solutions of (MPC.1b), and a discussion of the suboptimality introduced by the updated constraints in Proposition 3.2.1.

Proposition 3.2.2. *If the constraint set of (MPC.1b) is nonempty, then the solution exists and is unique.*

Proof. The objective of (MPC.1b) twice differentiable, so is necessarily continuous, and the constraint set is closed and bounded, so is compact. On the assumption that the constraint set is nonempty[2], the solution of (MPC.1b) therefore exists (Weierstrass' Theorem for continuous functions [91, p.237]). Additionally, the objective of (MPC.1b) is strictly convex, so the solution is unique (Proposition 2.2.1). □

Remark 3.2.2 (On Conservatism of the Solution of (MPC.1b))**.** Two additional constraints are imposed on (MPC.1a) to obtain the convex formulation (MPC.1b): the functions f_k and g_k are constrained to the domain in which they are strictly increasing with the updated constraints (3.25) and (3.26), and the constraint perturbation δ is introduced in (3.22).

The first of these two constraints is reasonable from a physical interpretation of a loss function: it would be expected that an increase in engine or motor output power would require an increase in input power, and the domain in which f_k and h_k (and therefore g_k) are decreasing is typically a mathematical artefact of a quadratic function and does not represent important loss characteristics. It is technically possible for operating points to exist in which this is not the case (in particular, during low-speed high-power regenerative braking), but these operating points are highly inefficient and unlikely to be optimal with respect to (MPC.1b) anyway. Additionally, these constraints are also required in the alternative approach to convexity presented in [14].

The perturbation δ is included to ensure that $g_k(m_k)$ is twice differentiable $\forall m_k \in [\underline{m}, \overline{m}]$ in the proof of Lemma 3.2.5 (which is subsequently used to prove g_k^{-1} is twice differentiable in Lemma 3.2.6). The only points in $\mathcal{Z}$ at which the first and second derivative of g_k does not exist are $\min \mathcal{Z}$ and $\max \mathcal{Z}$, so δ can be made arbitrarily small, up to the finite precision of the processor used to solve (MPC.1b). Furthermore, $\min \mathcal{Z} \notin [\underline{m}_k, \overline{m}_k]$ as $\frac{-\beta_{1,k}}{2\beta_{2,k}} > \min \mathcal{Z}$, and in practice $\max \mathcal{Z}$ is typically an order of magnitude larger than the battery's power limit, so the conservatism introduced to the solution of (MPC.1b) using δ is negligible, if it exists at all.

Theorem 3.2.1 is therefore one of the central contributions of this chapter: the MPC optimization (MPC.1a) can be converted, under negligible further approximation, to a convex optimization problem by using the battery's internal power as the decision variable instead of the engine's output power.

[2]Feasibility is addressed in more detail in Chapter 4. For now, it is assumed that the predictions of velocity and road gradient always generate a feasible problem.

3.3 OPTIMIZATION

At each timestep, t, the model predictive control input is now defined by

$$u(t) := u_0^\star \quad \forall t \in \{0, \dots, T\},$$

where $u^\star$ is the solution of (MPC.1b). A projected Newton method [119] is now proposed to obtain this solution.

The Lagrangian equation for (MPC.1b) is

$$\mathcal{L}(u, \lambda, \nu, \mu) := \sum_{k=0}^{N-1} \left[F_k(u_k) + \lambda_k(u_k - \underline{u}_k) + \nu_k(\overline{u}_k - u_k) \right] + \mu \left(x(t) - \sum_{k=0}^{N-1} u_k - x_N \right) \tag{3.29}$$

where $u := (u_0, \dots, u_{N-1})$, and $\lambda := (\lambda_0, \dots, \lambda_{N-1}) \in \mathbb{R}^N$, $\nu := (\nu_0, \dots, \nu_{N-1}) \in \mathbb{R}^N$, and $\mu \in \mathbb{R}$ are Lagrange multipliers. Note that $\underline{u}_k = g_k(p_k) = \overline{u}_k \; \forall k \notin \mathcal{P}$, so the first order necessary and sufficient conditions defining the unique solution of (MPC.1b) are

$$F_k'(u_k) + \lambda_k - \nu_k - \mu = 0 \quad \forall k \in \mathcal{P}, \tag{3.30a}$$
$$u_k - \underline{u}_k \geq 0 \quad \forall k \in \mathcal{P}, \tag{3.30b}$$
$$\lambda_k \geq 0 \quad \forall k \in \mathcal{P}, \tag{3.30c}$$
$$\lambda_k(u_k - \underline{u}_k) = 0 \quad \forall k \in \mathcal{P}, \tag{3.30d}$$
$$\overline{u}_k - u_k \geq 0 \quad \forall k \in \mathcal{P}, \tag{3.30e}$$
$$\nu_k \geq 0 \quad \forall k \in \mathcal{P}, \tag{3.30f}$$
$$\nu_k(\overline{u}_k - u_k) = 0 \quad \forall k \in \mathcal{P}, \tag{3.30g}$$
$$\nu_k \lambda_k = 0 \quad \forall k \in \mathcal{P}, \tag{3.30h}$$
$$u_k - \underline{u}_k = 0 \quad \forall k \in \mathcal{C}, \tag{3.30i}$$
$$x(t) - \sum_{k=0}^{N-1} u_k - X_N = 0, \tag{3.30j}$$

(note that the values of λ_k and ν_k are arbitrary $\forall k \in \mathcal{C}$).

The elements k where the inequality constraints are binding at each iteration, $i \in \mathbb{Z}_+$, of the algorithm are estimated from

$$\mathcal{A}^{(i)} := \Big\{ k \in \mathcal{P} : \Big(F_k'(u_k^{(i)}) - \mu < 0 \; \wedge \; u_k^{(i)} - \underline{u}_k = 0 \Big) \\ \vee \; \Big(F_k'(u_k^{(i)}) - \mu > 0 \; \wedge \; \overline{u}_k - u_k^{(i)} = 0 \Big) \Big\}, \tag{3.31}$$

then the Newton step is defined by

$$\begin{bmatrix} \nabla^2_{\mathcal{D}^{(i)}} F(u^{(i)}) & -\mathbf{1} \\ -\mathbf{1}^\top & 0 \end{bmatrix} \begin{bmatrix} \Delta_{\mathcal{D}^{(i)}} u^{(i)} \\ \Delta\mu^{(i)} \end{bmatrix} := -\underbrace{\begin{bmatrix} \nabla_{\mathcal{D}^{(i)}} F(u^{(i)}) - \mu^{(i)}\mathbf{1} \\ x(t) - \sum_{k=0}^{N-1} u_k^{(i)} - X_N \end{bmatrix}}_{J^{(i)}}. \tag{3.32}$$

where, $\mathcal{D}^{(i)} := \mathcal{P} \setminus \mathcal{A}^{(i)}$, $F(u) := \sum_{k=0}^{N-1} F_k(u_k)$, $\nabla^2_{\mathcal{D}^{(i)}} F(u^{(i)})$ is a diagonal matrix of the $k \in \mathcal{D}^{(i)}$ diagonal elements of the Hessian of F, and $\nabla_{\mathcal{D}^{(i)}} F(u^{(i)})$ is the $k \in \mathcal{D}^{(i)}$ elements of the gradient of F. Note that the definition of (3.31) corresponds to the elements of $k \in \mathcal{P}$ where λ_k and ν_k are nonzero, and they are therefore not calculated explicitly.

Remark 3.3.1. Theorem 3.2.1 implies that (3.32) and its solution exist for all $u_k^{(i)} \in [\underline{u}_k, \overline{u}_k]$, for all k, and for all $i \in \{0, \dots, \infty\}$.

The solution to (3.32) is then equivalent to

$$\Delta\mu^{(i)} = \frac{-x(t) + \sum_{k=0}^{N-1} u_k^{(i)} + X_N - \sum_{k\in\mathcal{A}^{(i)}} \frac{F_k'(u_k^{(i)}) + \mu^{(i)}}{F_k''(u_k^{(i)})}}{\sum_{k\in\mathcal{A}^{(i)}} F_k''(u_k^{(i)})} \tag{3.33}$$

and

$$\Delta u_k^{(i)} = \begin{cases} \frac{-F_k'(u_k^{(i)}) - \mu^{(i)} - \Delta\mu^{(i)}}{F_k''(u_k^{(i)})} & k \in \mathcal{A}^{(i)} \\ 0 & k \notin \mathcal{A}^{(i)} \end{cases}. \tag{3.34}$$

Remark 3.3.2. The computation of $\Delta\mu^{(i)}$ and $(\Delta u_0^{(i)}, \dots, \Delta u_{N-1}^{(i)})$ scale linearly with N, with no additional memory requirement.

The projected Newton method iteration is then defined by

$$\begin{aligned} \mu^{(i+1)} &:= \mu^{(i)} + \Delta\mu^{(i)} \\ u_k^{(i+1)} &:= \min\{\overline{u}_k,\ \max\{\underline{u}_k,\ u_k^{(i)} + \Delta u_k^{(i)}\}\} \quad \forall k \in \{0, \dots, N-1\}, \end{aligned} \tag{3.35}$$

and the algorithm is terminated at the first iteration that the criterion

$$\|J^{(i)}\| \leq \epsilon$$

is satisfied, where $J^{(i)}$ is defined in (3.32) and $\epsilon \in \mathbb{R}_{++}$ is a pre-determined convergence threshold.

Proposition 3.3.1. *Let $u^\star$ and $\mu^\star$ denote the optimal argument of (MPC.1b) and optimal value of the costate in (3.29), and assume that $\mu^\star \neq 0$. Then $(x^{(i)}, \mu^{(i)})$ is a fixed point of the iteration (3.35) if and only if $(x^{(i)}, \mu^{(i)}) = (x^\star, \mu^\star)$. Furthermore, the iteration converges superlinearly in a neighbourhood of $(x^\star, \mu^\star)$ if no constraints are weakly active at the solution.*

Proof. The first-order optimality conditions (3.30) that define $(u^\star, \mu^\star)$ are equivalent to

$$F_k'(u_k^\star) - \mu^\star = 0 \quad \forall k \in \mathcal{P} \cap \{k : u_k^\star \in (\underline{u}_k, \overline{u}_k)\} \tag{3.36a}$$
$$F_k'(u_k^\star) - \mu^\star \leq 0 \quad \forall k \in \mathcal{P} \cap \{k : u_k^\star = \underline{u}_k\} \tag{3.36b}$$
$$F_k'(u_k^\star) - \mu^\star \geq 0 \quad \forall k \in \mathcal{P} \cap \{k : u_k^\star = \overline{u}_k\} \tag{3.36c}$$
$$u_k^\star - \underline{u}_k = 0 \quad \forall k \in \mathcal{C} \tag{3.36d}$$
$$x(t) - \sum_{k=0}^{N-1} u_k^\star - X_N = 0 \tag{3.36e}$$

Let $\mathcal{A}^\star$ denote the active set at the solution $(u^\star, p^\star)$, then from the optimality conditions (3.36) and the definition of the active set in (3.31) it follows that: 1) $\mathcal{P} \setminus \mathcal{A}^\star$ must be nonempty (since the constraints $\underline{u}_k \leq u_k \leq \overline{u}_k$ cannot be active $\forall k \in \mathcal{P}$ if (3.36e) holds because of the assumption that no constraint is weakly active at the solution), and 2) $F_k'(u_k^\star) - \mu^\star = 0 \; \forall k \in \mathcal{P} \setminus \mathcal{A}^\star$. Therefore, $\Delta\mu^\star = 0$ from (3.33), which therefore implies that $\Delta u_k^\star = 0 \; \forall k \in \mathcal{P}$ from (3.34), so $(u^\star, p^\star)$ is a stationary point of the iteration (3.35).

Conversely, if $(u^{(i)}, \mu^{(i)})$ is a fixed point of (3.35), then $\Delta u_k^{(i)} = 0 \; \forall k \in \mathcal{P}$ and $\Delta\mu^{(i)} = 0$. In this case, from (3.34) it follows that $-F_k'(u_k^{(i)}) - \mu^{(i)} = 0 \; \forall k \in \mathcal{A}^{(i)}$, so $x(t) + \sum_{k=0}^{N-1} x_k^{(i)} + x_N = 0$ from (3.33). Therefore, the first-order conditions are satisfied, so the point $u^{(i)}$ is optimal w.r.t (MPC.1b) as it is convex.

To complete the proof, note that if there are no weakly active constraints at the solution of (MPC.1b), then there must exist a neighborhood of $(u^\star, \mu^\star)$ within which the active set is constant, so the iteration (3.35) coincides with Newton's method applied to a subspace (namely, $\{u_k : k \in \mathcal{P}, u_k \neq \underline{u}_k \wedge u_k \neq \overline{u}_k\}$). A superlinear convergence rate can then be shown using the standard arguments for unconstrained optimization [64, §3.3]. □

Stronger convergence results may be obtained under less restrictive assumptions if $\Delta u^{(i)}$ and $\Delta\mu^{(i)}$ are scaled by a stepsize parameter in (3.35). It is demonstrated for a similar problem in [119] that the stepsize can be adapted at each iteration to ensure global convergence to the solution of (MPC.1b) from any initial feasible point. For the purposes of this chapter this modification is unneccessary as rapid convergence to the solution is observed for all of the following numerical simulations with the fixed iteration (3.35).

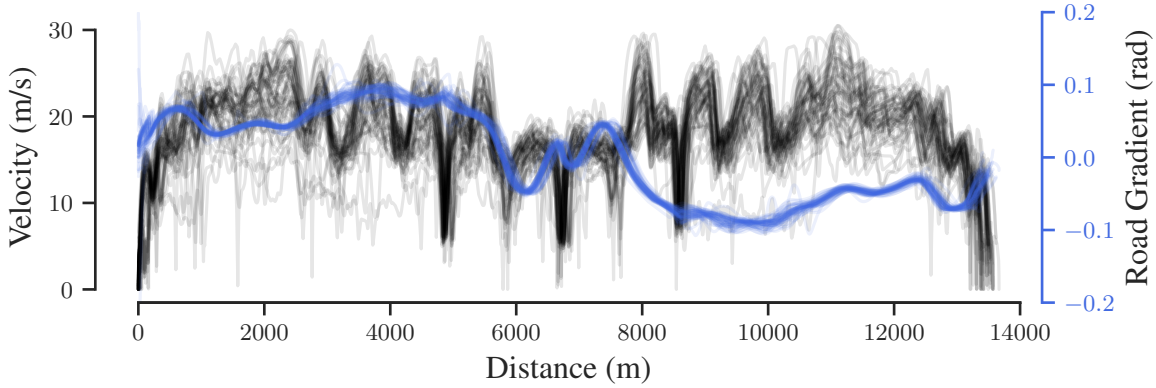


Figure 3.4: Velocity and road gradient against longitudinal distance for each of the 49 journeys.

3.4 Numerical Experiments

Driver Behaviour

It is common in the energy management literature to model the driver using standard government test cycles (e.g. [57, 39, 50, 57, 79, 114]), as these will typically be used to experimentally validate powertrain (and controller) performance. However, these cycles do not accurately represent real-world driver behaviour. Therefore, in this thesis the driver is modelled using real-world sensor data. Figure 3.4 shows the velocity and gradient trajectories used for the numerical simulations here and throughout this thesis. The data was obtained during a set of vehicle emissions tests conducted at Johannes Kepler University Linz, Austria, and represents 49 journeys driven by four different drivers. One of the key features of this dataset is that the same route is used for each journey, and therefore provides an accurate representation of the uncertainty in driver behaviour across driving conditions and drivers. This aspect is addressed further in Chapter 6.

Control

For each of the journeys the battery was initialised at 60% state of charge and the terminal state-of-charge constraint was set at 50%. Two control algorithms were used:

1. **PMP.** The PMP algorithm described in Section 2.2.2 was used with the engine, motor, and battery models that were used for simulation, and extended to include the instantaneous limits on battery power.

2. **MPC.** The MPC algorithm described in this Chapter was used where it was assumed that the controller had perfect predictions of the future velocity and road gradients

(i.e. the only uncertainty in predictions was due to model approximation error), and the predictions were implemented as a shrinking horizon (i.e the prediction horizon N reduced as the vehicle progressed through the journey).

EXPERIMENTS WITHOUT ROAD GRADIENT

Firstly, the simulations were conducted with the velocity profiles illustrated in Figure 3.4, but the road gradient was set to zero for every journey (in both the simulations and the MPC predictions). The state-of-charge trajectories and cumulative fuel consumption using both MPC and PMP are presented for all 49 journeys in Figure 3.5, with a single representative journey (chosen at random) highlighted. It can only be guaranteed that PMP obtains a stationary point of the energy management problem, and when the model used for simulation is considered in the optimization (as is the case here) then this may not be the global minimum as the problem is nonconvex in general. However, the results show that PMP provides marginally superior performance to MPC, reducing fuel consumption by 0.78 % on average, and is therefore likely to be a close approximation of the globally optimal controls in these examples. Additionally, the results also show that the state-of-charge and fuel consumption trajectories are extremely similar between MPC and PMP, which implies that MPC obtains a close approximation of the optimal controller, despite the model approximation used in the convex formulation. It can also be seen in this case that an assumption of a linearly decreasing state-of-charge trajectory is generally a good approximation of the true trajectory across all journeys. The average time taken to solve the MPC optimization was 0.76 ms (maximum 30 ms and minimum 17 μs), and therefore the algorithm is, computationally speaking, a suitable candidate for an online, embedded MPC controller.

EXPERIMENTS WITH ROAD GRADIENT

Figure 3.6 shows the experiments repeated with the real road gradient data presented in Figure 3.4. It is again demonstrated that MPC provides a very close approximation of the controls obtained with PMP, but two significant issues are now encountered. Firstly, it can now be clearly seen that that the state-of-charge trajectory is not linearly decreasing, and approaches that use short-horizon MPC to control the battery's state of charge to track a linearly decreasing reference trajectory (e.g. [82, 83]) will therefore introduce significant suboptimality to the controller. Secondly, it can be seen that the battery's state-of-charge dips significantly below its terminal constraint. This may not be an issue for the test cases presented here, but it is therefore possible that either control method could attempt to discharge the battery past its lower state-of-charge limit (or overcharge past its upper limit). In this case the controller will be overridden to prevent damage to the battery,

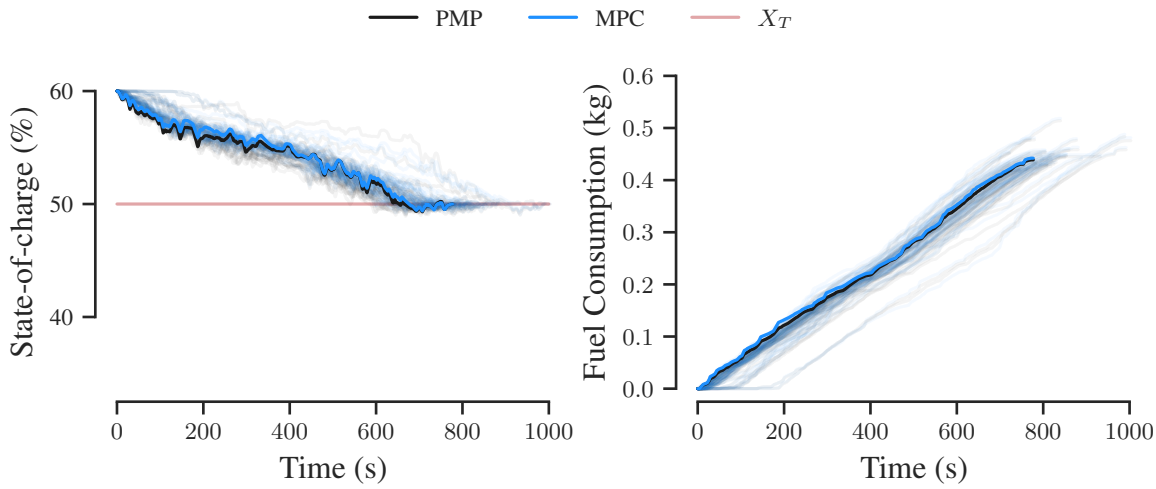


Figure 3.5: State-of-charge trajectories and cumulative fuel consumption for each of the 49 journeys where the road gradient is set to zero in both the MPC predictions and simulations. The results for a single journey are highlighted in both figures, and the terminal state-of-charge constraint is highlighted in red.

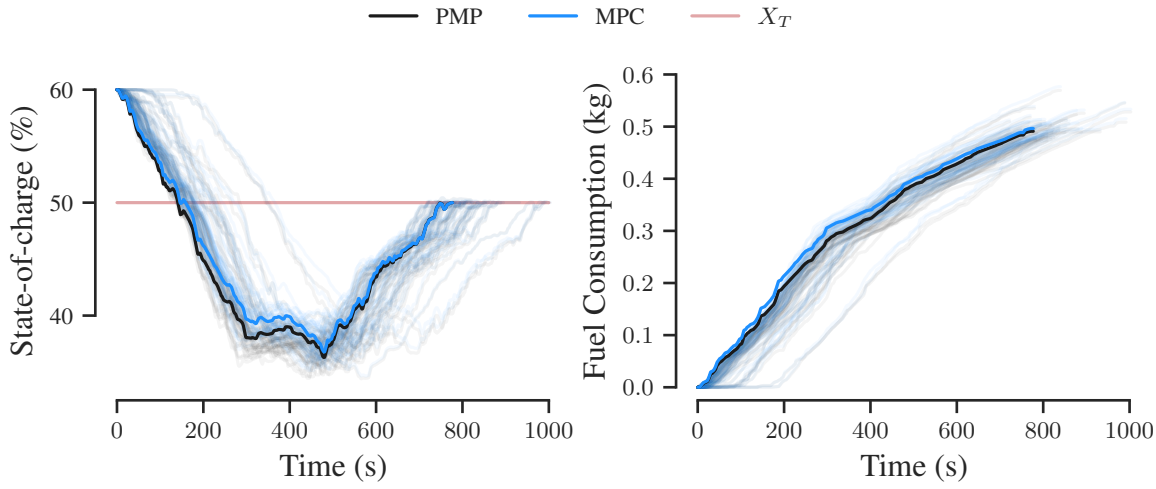


Figure 3.6: The results from the same experiments as depicted in Figure 3.5, except that the true road gradient is used for both simulation and MPC prediction.

which will render the control inputs suboptimal. It is therefore critical that the general state-of-charge constraints are explicitly considered by the MPC formulation.

3.5 Concluding Remarks

In this chapter a vehicle model was presented with the structure and fidelity similar to that typically used to investigate energy management problems, and a MPC framework was proposed where nonlinear models were used to obtain a high accuracy approximation of the powertrain losses used for simulation. It was subsequently shown that the MPC optimization problem is convex under negligible further approximation when stated in terms of the battery's discharge rate. The MPC controller was compared in simulation with PMP (for which the exact powertrain losses were considered) in simulation where it was shown to provide near-optimal control inputs, and the optimization algorithm was shown to be sufficiently fast for a real-time solution. However, it was also demonstrated that a linearly decreasing state-of-charge reference is inaccurate in general, and a terminal state-of-charge constraint is not sufficient to ensure optimal performance in a typical real-world driving scenario. In the following chapter, an MPC controller is proposed in which the state-of-charge constraints are explicitly enforced across the entire prediction horizon to address this limitation.

CHAPTER 4

GENERAL STATE CONSTRAINT

IN this chapter the convex model predictive energy management framework proposed in the previous is extended to consider upper and lower bounds on the battery's state-of-charge across the entire prediction horizon. Previous studies that have approached this problem using convex optimization have utilized general purpose convex optimization software (e.g. [17, 104, 106, 118]), and have generally not investigated algorithms that are tailored to the structure of the problem. Only two examples of convex optimization algorithms for solving the PHEV energy management problem could be found in the literature, [116] and [117], although both of these examples completely ignore the losses in both the battery and motor. The main contribution of this chapter is two algorithms for the solution of the resulting MPC optimization problem: an alternating direction method of multipliers (ADMM) algorithm, and a novel projected interior point algorithm. The computational efficiency of the two algorithms is compared on a set of randomly generated energy management problems, and then the extended MPC framework is implemented in closed loop in a simulation. It is shown that the ADMM can solve the MPC optimization in a fraction of a second, even over horizons of up to 1000 samples, and that the MPC controller provides a close approximation of the globally optimal controls obtained using dynamic programming. To date, these are the only examples of optimization algorithms that can solve the PHEV energy management problem in real time when long prediction horizons and nonlinear electric system dynamics are considered.

4.1 MODEL PREDICTIVE CONTROLLER

Problem (MPC.1b) can be trivially extended to enforce the bounds on state-of-charge

$$\underline{x} \leq x_k \leq \overline{x} \quad \forall k \in \{1, \ldots, N\}. \tag{4.1}$$

From (3.24) it can be shown that $x := (x_1, \ldots, x_N) = \mathbf{1}x(t) - \Psi u$, where Ψ is a lower triangular matrix of ones, and define $\underline{u} := (\underline{u}_0, \ldots, \underline{u}_{N-1})$ and $\overline{u} := (\overline{u}_0, \ldots, \overline{u}_{N-1})$. Ad-

ditionally, define $F : \mathbb{R}^n \mapsto \mathbb{R}$ as

$$F(u) := \sum_{i=0}^{N-1} F_k(u_k).$$

The MPC optimization including constraint (4.1) is then given by the convex optimization problem

$$\begin{aligned} &\min_u F(u), \\ &\text{s.t. } x = \mathbf{1}x(t) - \Psi u, \\ &\quad \mathbf{1}\underline{x} \leq x \leq \mathbf{1}\overline{x}, \\ &\quad \underline{u} \leq u \leq \overline{u}. \end{aligned} \tag{MPC.2}$$

Proposition 4.1.1. *If the constraint set of (MPC.2) is nonempty, then the solution exists and is unique.*

Proof. The proof can be shown with a minor extension of the proof of Proposition 3.2.2. □

4.1.1 FEASIBILITY

Proposition 4.1.2. *The constraint set in (MPC.2) is nonempty if and only if $\mathcal{F}_k \neq \emptyset \; \forall k \in \{1, \ldots, N\}$, where $\mathcal{F}_0 := \{x(t)\}$, $\mathcal{F}_{k+1} := \{\mathcal{F}_k \oplus -\mathcal{U}_k\} \cap \mathcal{X} \; \forall k \in \{0, \ldots, N-1\}$, $\mathcal{U}_k := [\underline{u}_k, \overline{u}_k]$, and $\mathcal{X} := [\underline{x}, \overline{x}]$.*

Proof. Before starting the proof, note that

$$\mathcal{F}_{k+1} = \{x_k - u_k \in \mathcal{X} : x_k \in \mathcal{F}_k, u_k \in \mathcal{U}_k\} \quad \forall k \in \{0, \ldots, N-1\} \tag{4.2}$$

from the definition of the Minkowski sum, and enumerate the constraints in (MPC.2) as

$$x = \mathbf{1}x(t) - \Psi u, \tag{4.3a}$$

$$\mathbf{1}\underline{x} \leq x \leq \mathbf{1}\overline{x}, \tag{4.3b}$$

$$\underline{u} \leq u \leq \overline{u}. \tag{4.3c}$$

Firstly, assume that $\mathcal{F}_{k+1} \neq \emptyset \; \forall k \in \{0, \ldots, N-1\}$. If it is also assumed that there exists $x_k \in \mathcal{F}_k$ such that x_k is feasible w.r.t (4.3b), then there exists a value of u_k that is feasible w.r.t (4.3c) such that x_{k+1} is feasible w.r.t (4.3a) and (4.3b). $x_0 := x(t)$ trivially satisfies (4.3b), so by induction there exists a pair (u_k, x_{k+1}) that satisfies (4.3a)-(4.3c) $\forall k \in \{0, \ldots, N-1\}$.

Conversely, assume that there exists u and x that satisfy (4.3a)-(4.3c), then $\mathcal{F}_{k+1} \supseteq \{x_{k+1}\} \; \forall k \in \{0, \ldots, N-1\}$, which completes the proof.

□

The property that $\mathcal{F}_0$, $\mathcal{X}$, and every $\mathcal{U}_k$ $\forall k \in \{0, \ldots, N-1\}$ are closed convex subsets of $\mathbb{R}$ implies that $\mathcal{F}_k$ is a closed convex subset of $\mathbb{R}$ $\forall k \in \{1, \ldots, N\}$. Consequently, this implies $\mathcal{F}_k$ can be parametrized by $\mathcal{F}_k = [\min \mathcal{F}_k, \max \mathcal{F}_k]$, where $\min \mathcal{F}_0 = \max \mathcal{F}_0 = x(t)$, $\min \mathcal{F}_{k+1} := \max\{\underline{x}, \min \mathcal{F}_k - \overline{u}_k\}$, and $\max \mathcal{F}_{k+1} := \min\{\overline{x}, \max \mathcal{F}_k - \underline{u}_k\}$ $\forall k \in \{0, \ldots, N-1\}$. Therefore, a feasibility certificate for (MPC.2) can be obtained using Algorithm 1 (note that $\max \mathcal{F}_k$ and $\min \mathcal{F}_k$ are treated as variables that are used to parameterize $\mathcal{F}_k$ in Algorithm 1).

Algorithm 1 Feasibility certificate for (MPC.2)

Input: $(x(t), \underline{u}, \overline{u}, \underline{x}, \overline{x})$
Output: ($\mathcal{F}$, Feasible)

$(\min \mathcal{F}_0, \max \mathcal{F}_0) \leftarrow (x(t), x(t))$
for $k = 0, \ldots, N-1$ **do**
  $\max \mathcal{F}_{k+1} \leftarrow \min\{\overline{x}, \max \mathcal{F}_k - \underline{u}_k\}$
  $\min \mathcal{F}_{k+1} \leftarrow \max\{\underline{x}, \min \mathcal{F}_k - \overline{u}_k\}$
  if $\max \mathcal{F}_{k+1} < \min \mathcal{F}_{k+1}$ **or** $\underline{u}_k < \overline{u}_k$ **then**
    return ($\sim$, FALSE).
  end if
end for
return ($\mathcal{F}$, TRUE).

Remark 4.1.1. The computational and memory requirements of Algorithm 1 scale linearly with the prediction horizon, N.

4.2 OPTIMIZATION

Two candidate convex optimization algorithms are proposed for the solution of (MPC.2): an interior point algorithm and an ADMM algorithm. These algorithms were chosen to investigate the relative performance benefits of first-order and second-order optimization algorithms. Additionally, the interior point algorithm was chosen for investigation as it is a common second-order method for solving MPC optimization problems [120, 121, 122, 123], and a novel approach is proposed that uses a *projected* Newton step to solve each successive barrier problem. The ADMM algorithm was chosen because it has recently received attention for its ability to exploit the structure of MPC problems [124, 125, 126].

4.2.1 ALTERNATING DIRECTION METHOD OF MULTIPLIERS

In this section an ADMM algorithm is proposed for the solution of (MPC.2). Firstly, the methods for initializing, iterating, and terminating the algorithm are presented, then the computational and memory complexity of the algorithm is analysed in detail. Problem (MPC.2) is the equivalent of the equality constrained optimization problem

$$\begin{aligned} \min_{u} \; & F(u) + \sum_{k=0}^{N-1} \left[\mathcal{I}_{\mathcal{U}_k}(u_k) + \mathcal{I}_{\mathcal{X}}(x_{k+1}) \right], \\ \text{s.t. } & u = \zeta \\ & x = \mathbf{1}x(t) - \Psi\zeta, \end{aligned} \tag{4.4}$$

where $\zeta \in \mathbb{R}^N$ is a vector of dummy variables, and the indicator functions are defined for a given set $\mathcal{S}$ by

$$\mathcal{I}_{\mathcal{S}} := \begin{cases} 0 & s \in \mathcal{S} \\ \infty & s \notin \mathcal{S} \end{cases}.$$

Problem 4.4 is in turn the equivalent of

$$\min_{u} \; F(u) + \sum_{k=0}^{N-1} \left[\mathcal{I}_{\mathcal{U}_k}(u_k) + \mathcal{I}_{\mathcal{X}}(x_{k+1}) \right] \quad \text{s.t. } \tilde{u} + B\zeta = c, \tag{4.5}$$

where

$$\tilde{u} := (u, x), \quad B := \left[\begin{smallmatrix} -I \\ \Psi \end{smallmatrix} \right], \quad c := (\mathbf{0}, \mathbf{1}x(t)).$$

The augmented Lagrangian function, $\mathcal{L}_\rho : \mathbb{R}^{2N} \times \mathbb{R}^N \times \mathbb{R}^{2N} \mapsto \mathbb{R}$, for (4.5) is

$$\mathcal{L}_\rho(\tilde{u}, \zeta, \lambda) := F(u) + \sum_{k=0}^{N-1} \left[\mathcal{I}_{\mathcal{U}_k}(u_k) + \mathcal{I}_{\mathcal{X}}(x_{k+1}) \right] + \frac{1}{2} \|\tilde{u} + B\zeta - c + \lambda\|_\rho^2 \tag{4.6}$$

where

$$\|x\|_\rho^2 := x^\top \rho x, \quad \rho := \text{diag}(\rho_1 \mathbf{1}, \rho_2 \mathbf{1}), \quad \rho_1, \rho_2 \in \mathbb{R}_+, \quad \lambda := (\lambda_1, \lambda_2), \quad \lambda_1, \lambda_2 \in \mathbb{R}^N.$$

The ADMM iteration is defined by

$$\tilde{u}^{(i+1)} := \underset{\tilde{u}}{\text{argmin}} \; \mathcal{L}(\tilde{u}, \zeta^{(i)}, \lambda^{(i)}), \tag{4.7a}$$

$$\zeta^{(i+1)} := \underset{\zeta}{\text{argmin}} \; \mathcal{L}(\tilde{u}^{(i+1)}, \zeta, \lambda^{(i)}), \tag{4.7b}$$

$$\lambda^{(i+1)} := \lambda^{(i)} + A\tilde{u} + B\zeta - c, \tag{4.7c}$$

and in [127] it was demonstrated that this algorithm converges to the solution of (4.5) as the residuals defined by

$$r^{(i+1)} := \tilde{u}^{(i+1)} + B\zeta - c, \quad \text{and} \quad s^{(i+1)} := \rho B(\zeta^{(i+1)} - \zeta^{(i)})$$

neccesarily converge to zero (the proof is presented for a scalar value of ρ, but can be trivially extended to a fixed positive diagonal matrix as presented here, and is also proven in Section 5.A.3). The algorithm is initialized with

$$\tilde{u}^{(0)} = \mathbf{0}, \quad \zeta^{(0)} = \mathbf{0}, \quad \lambda^{(0)} = \mathbf{0},$$

and terminated at the first iteration where the criterion

$$\max\{\|r^{(i+1)}\|, \|s^{(i+1)}\|\} \leq \epsilon \tag{4.8}$$

is met, where $\epsilon \in \mathbb{R}_+$ is a pre-determined convergence threshold. The ADMM iteration (4.7) is a particular case of 'Generalized ADMM', for which it was demonstrated in [128] that $\mathcal{O}(1/\epsilon)$ iterations are required to meet criterion (4.8) for a given problem. To the best of the author's knowledge there are no theoretical results that demonstrate how the iteration complexity varies with horizon length (i.e. problem size), but numerical studies suggest that the number of iterations required to meet a given tolerance ϵ is independent of horizon length [20, §5-C].

ALGORITHM COMPLEXITY

The Lagrangian function (4.6) is equivalent to

$$\mathcal{L}_\rho(\tilde{u}, \zeta, \lambda) := F(u) + \sum_{k=0}^{N-1} [\mathcal{I}_{\mathcal{U}_k}(u_k) + \mathcal{I}_{\mathcal{X}}(x_{k+1})] + \frac{\rho_1}{2}\|u - \zeta + \lambda_1\|_2^2 + \frac{\rho_2}{2}\|x + \Psi\zeta - \mathbf{1}x(t) + \lambda_2\|_2^2,$$

so update (4.7a) is the equivalent of

$$u_k^{(i+1)} := \begin{cases} \operatorname{argmin}_{u_k} \left[F_k(u_k) + \mathcal{I}_{\mathcal{U}_k}(u_k) + \frac{\rho_1}{2}\|u_k - \zeta_k^{(i)} + \lambda_{1,k}^{(i)}\|_2^2\right] & \forall k \in \mathcal{P} \\ \underline{u}_k & \forall k \in \mathcal{C} \end{cases}, \tag{4.9a}$$

$$x^{(i+1)} := \Pi_{\mathcal{X}^N}\left[\mathbf{1}x(t) - \Psi\zeta^{(i)} - \lambda_2^{(i)}\right], \tag{4.9b}$$

where $\mathcal{X}^N := \{x \in \mathbb{R}^N : \underline{x} \leq x_k \leq \overline{x},\ k \in \{1, \ldots, N\}\}$, and the projection on a set $\mathcal{S}$ is defined by $\Pi_{\mathcal{S}}(s) := \operatorname{argmin}_{\hat{s}\in\mathcal{S}} \|s - \hat{s}\|$. The $u^{(i+1)}$ update is separable w.r.t the elements

Table 4.1: Summary of the computational complexity of each of the ADMM variable updates.

$\mathcal{O}(\cdot)$	u	x	ζ	r	s	λ
Parallel	1	N	N	N	N	1
Sequential	N	N	N	N	N	N

k, so its computation scales linearly with the prediction horizon when these updates are performed sequentially, and is constant if they are performed in parallel. The individual update $u_k^{(i+1)}$ for $k \in \mathcal{P}$ is a one-dimensional constrained convex optimization problem that can be solved rapidly and reliably using Newton's method with a backtracking line-search [64, §3]. Note that if $F_k(\cdot)$ are approximated as quadratic functions then the update $u_k^{(i+1)}$ has an analytical solution for all $k \in \mathcal{P}$. Multiplication by Ψ is the equivalent of a cumulative sum (Ψ is a lower triangular matrix of ones), so the computation of the argument of the projection function in (4.9b) scales linearly with the prediction horizon and has no memory requirement. The projection $\Pi_{\mathcal{X}^N}$ can be performed element-wise, so the computation of (4.9b) scales linearly with N overall.

Update (4.7b) is equivalent to

$$\zeta^{(i+1)} = (\rho_1 I + \rho_2 \Psi^\top \Psi)^{-1}\Big[\rho_1(u^{(i+1)} + \lambda_1^{(i)}) - \rho_2 \Psi^\top (x^{(i+1)} - \mathbf{1}x(t) + \lambda_2^{(i)})\Big], \quad (4.10)$$

which is in turn the solution of the general system of linear equations $(kI + \Psi^\top \Psi)\mathbf{x} = \mathbf{b}$, and can be solved with $\mathcal{O}(N)$ computation and memory requirement (Proposition 4.A.1). The residual updates are the equivalent of

$$r^{(i+1)} = \begin{bmatrix} u^{(i+1)} - \zeta^{(i+1)} \\ x^{(i+1)} + \Psi\zeta^{(i+1)} - \mathbf{1}x(t) \end{bmatrix}, \quad s^{(i+1)} = \begin{bmatrix} \rho_1(\zeta^{(i)} - \zeta^{(i+1)}) \\ \rho_2 \Psi(\zeta^{(i+1)} - \zeta^{(i)}) \end{bmatrix}, \quad (4.11)$$

which scale linearly with N and have no additional memory requirement. Finally, (4.7c) is the equivalent of $\lambda^{(i+1)} = \lambda^{(i)} + r^{(i+1)}$.

The computational requirement of each variable update is summarized in Table 4.1, and the total memory requirement of the algorithm is $\mathcal{O}(N)$, as only a bandwidth two Cholesky factor requires storage (defined in Proposition 4.A.1). The algorithm as a whole is presented in pseudocode in Algorithm 2, for which the output $u^\dagger$ can be made arbitrarily close to the solution of (MPC.2) by setting ϵ arbitrarily close to zero.

Algorithm 2 ADMM Algorithm for Problem (MPC.2)

Input: $(x(t), \underline{u}, \overline{u}, \underline{x}, \overline{x}, F, \epsilon)$
Output: $u^\dagger$

(∼, Feasible) ← Algorithm 1.
if Feasible = False **then**
  return ∼
end if
$(u^{(0)}, x^{(0)}, \zeta^{(0)}, \lambda^{(0)}) \leftarrow \mathbf{0}$
$(r^{(0)}, s^{(0)}) \leftarrow \infty$
$i \leftarrow 0$
while $\max\{\|r^{(i+1)}\|, \|s^{(i+1)}\|\} > \epsilon$ **do**
  $u_k^{(i+1)} \leftarrow$ (4.9a) $\forall k \in \{0, \ldots, N-1\}$
  $x^{(i+1)} \leftarrow$ (4.9b)
  $\zeta^{(i+1)} \leftarrow$ (4.10)
  $(r^{(i+1)}, s^{(i+1)}) \leftarrow$ (4.11)
  $\lambda^{(i+1)} \leftarrow \lambda^{(i)} + r^{(i+1)}$
  $i \leftarrow i + 1$
end while
return $u^\dagger \leftarrow u^{(i)}$

4.2.2 PROJECTED INTERIOR POINT METHOD

This section presents a novel primal-dual interior point method algorithm where the element-wise bounds on the control variable are enforced by a projection in the Newton step instead of a logarithmic barrier function. The section begins with the formulation of the barrier approximation and optimality conditions, then a computationally efficient initialisation algorithm is presented. The main projected interior-point algorithm is then presented, and the section is concluded with an analysis of the computational complexity of each iteration, and a discussion of convergence to the minimizing argument of (MPC.2).

OPTIMALITY CONDITIONS

A slack variable $s \in \mathbb{R}^{2N}$ is introduced in (MPC.2) to obtain the equivalent problem

$$\begin{aligned} &\min_u F(u), \\ &\text{s.t. } Au - c - s = 0 \\ &\quad s \geq 0 \\ &\quad \underline{u} \leq u \leq \overline{u}, \end{aligned} \tag{IP.a}$$

where

$$A := \begin{bmatrix} \Psi \\ -\Psi \end{bmatrix}, \quad \text{and} \quad c := \begin{bmatrix} \mathbf{1}(x(t) - \overline{x}) \\ -\mathbf{1}(x(t) - \underline{x}) \end{bmatrix}.$$

Problem (IP.a) can then be approximated with

$$\begin{aligned} &\min_u F(u) + B_\mu(s), \\ &\text{s.t. } Au - c - s = 0 \\ &\quad \underline{u} \leq u \leq \overline{u}, \end{aligned} \tag{IP.b}$$

where $B_\mu : \mathbb{R}^{2N} \mapsto \mathbb{R}$ is a logarithmic barrier function defined by

$$B_\mu(s) := -\frac{1}{\mu} \sum_{k=1}^{2N} \log s_k,$$

and $\mu \in \mathbb{R}_+$ can be interpreted as the accuracy with which the barrier function approximates the original inequality constraint $s \geq 0$. The Lagrangian function, $\mathcal{L}_\mu : \mathbb{R}^{7N} \mapsto \mathbb{R}$, associated with (IP.b) for a given value of μ is

$$\mathcal{L}_\mu(u, s, \theta_1, \theta_2, \theta_3) := F(u) + B_\mu(s) - \theta_1^\top (Au - c - s) - \theta_2^\top (u - \underline{u}) - \theta_3^\top (\overline{u} - u),$$

where $\theta_1 \in \mathbb{R}^{2N}$ and $\theta_2, \theta_3 \in \mathbb{R}^N$ are vectors of Lagrange multipliers. By defining the set

$$\mathcal{A}^\circ := \{k : (u_k^\circ = \underline{u}_k \wedge F_k'(u_k^\circ) - A_{:,k}^\top > 0) \ \vee \ (u_k^\circ = \overline{u}_k \wedge F_k'(u_k^\circ) - A_{:,k}^\top < 0)\}$$

where $A_{:,k}$ is the kth column of A, the necessary and sufficient conditions for the (unique) values that minimize (IP.b) are

$$F_k'(u_k^\circ) - A_{:,k}^\top \theta_1^\circ = 0 \quad \forall k \notin \mathcal{A}^\circ, \tag{4.12a}$$

$$S^\circ \theta_1^\circ = \frac{1}{\mu}\mathbf{1}, \tag{4.12b}$$

$$Au^\circ - c - s^\circ = 0, \tag{4.12c}$$

$$u^\circ - \underline{u} \geq 0, \tag{4.12d}$$

$$\overline{u} - u^\circ \geq 0, \tag{4.12e}$$

$$s^\circ > 0, \tag{4.12f}$$

$$\theta_1^\circ \geq 0, \tag{4.12g}$$

where $S^\circ := \text{diag}(s^\circ)$.

It can be demonstrated that to obtain the optimality conditions of the original problem, (IP.a), two changes must be made to the optimality conditions, (4.12): first, (4.12f) must become a nonstrict inequality condition, and second, a vector of zeros must replace $\frac{1}{\mu}\mathbf{1}$ on the right-hand side of (4.12b). Therefore, it can be concluded that u° converges asymptotically to $u^\star$ as $\mu \to \infty$ (following what is known as the 'central path' [96, §11.2]), where $u^\star$ is the minimizing argument of (IP.a) (and therefore also the original problem, (MPC.2)). The principle of the algorithm presented in this section is that conditions (4.12d)-(4.12g) hold at all iterations, while a projected Newton method similar to that presented in Chapter 3 is used to obtain an approximation of the point satisfying (4.12a)-(4.12c) for a fixed value of μ. This is then repeated for progressively larger values of μ, with progressively higher accuracy, to obtain $u^\star$.

INITIALIZATION

The projected interior-point algorithm is initialized using the tube defined by the sequence $\{\mathcal{G}_0, \ldots, \mathcal{G}_N\}$, where $\mathcal{G}_N = \mathcal{F}_N$, and $\mathcal{G}_k$ is defined recursively by

$$\mathcal{G}_k := \mathcal{F}_k \cap \{\mathcal{G}_{k+1} \oplus \mathcal{U}_k\} \quad \forall k \in \{N-1, \ldots, 0\}.$$

If $\mathcal{G}_k$ is nonempty, then it is trivially closed and convex, and can be parametrized by

$$\mathcal{G}_k = [\min \mathcal{G}_k, \max \mathcal{G}_k] = \Big[\max\{\min \mathcal{G}_{k+1} + \underline{u}_k, \min \mathcal{F}_k\}, \min\{\max \mathcal{G}_{k+1} + \overline{u}_k, \max \mathcal{F}_k\} \Big].$$

The centreline of this tube, $x_k^{(0)} = \frac{1}{2}(\max \mathcal{G}_k + \min \mathcal{G}_k)$, is then used as an initial estimate of the optimal state-of-charge trajectory, from which an initial feasible estimate of the control vector, $u^{(0)}$, is obtained from $u_k^{(0)} = x_k^{(0)} - x_{k+1}^{(0)}\ \forall k \in \{N, \ldots, 1\}$. The other decision variables are then initialized as $s^{(0)} = Au^{(0)} - c$ and $\theta_1^{(0)} = \frac{1}{\mu}(S^{(0)})^{-1}\mathbf{1}$. The initialization algorithm is summarized in Algorithm 3, where $\max \mathcal{G}_k$ and $\min \mathcal{G}_k$ are treated as variables used to parameterize $\mathcal{G}_k$, and FLAG = TRUE is a certificate that the returned values of $u^{(0)}$, $s^{(0)}$, and $\theta_1^{(0)}$, satisfy (4.12c)-(4.12g).

Algorithm 3 Projected Interior Point Initialization Algorithm

Input: $(\underline{u}, \overline{u}, \underline{x}, \overline{x}, \mathcal{F})$
Output: $(u^{(0)}, s^{(0)}, \theta_1^{(0)}, \text{FLAG})$

$(\max \mathcal{G}_N, \min \mathcal{G}_N) \leftarrow (\max \mathcal{F}_N, \min \mathcal{F}_N)$
$x_N^{(0)} \leftarrow \frac{1}{2}(\max \mathcal{G}_N + \min \mathcal{G}_N)$
for $k = N-1, \ldots, 0$ **do**
  $\max \mathcal{G}_k \leftarrow \min\{\max \mathcal{G}_{k+1} + \overline{u}_k, \max \mathcal{F}_k\}$
  $\min \mathcal{G}_k \leftarrow \max\{\min \mathcal{G}_{k+1} + \underline{u}_k, \min \mathcal{F}_k\}$
  $x_k^{(0)} \leftarrow \frac{1}{2}(\max \mathcal{G}_k + \min \mathcal{G}_k)$
  $u_k^{(0)} \leftarrow x_k^{(0)} - x_{k+1}^{(0)}$
  if $\max \mathcal{F}_k = \min \mathcal{X}\ \vee\ \min \mathcal{F}_k = \max \mathcal{X}$ **then**
    return $(\sim, \sim, \sim, \text{FALSE})$
  end if
end for
$s^{(0)} = Au^{(0)} - c$
$\theta_1^{(0)} = \frac{1}{\mu}(S^{(0)})^{-1}\mathbf{1}$
return $(u^{(0)}, s^{(0)}, \theta_1^{(0)}, \text{TRUE})$

Proposition 4.2.1. *Assume that (MPC.2) is feasible. The values of $u^{(0)}$, $s^{(0)}$, and $\theta_1^{(0)}$ obtained using Algorithm 3 will satisfy conditions (4.12c)-(4.12g) iff*

$$\max \mathcal{F}_k > \min \mathcal{X}\ \wedge\ \min \mathcal{F}_k < \max \mathcal{X} \quad \forall k \in \{1, \ldots, N\}. \tag{4.13}$$

Proof. Let $|\mathcal{S}|$ denote

$$|\mathcal{S}| := \max \mathcal{S} - \min \mathcal{S}$$

for a given set $\mathcal{S}$. If (MPC.2) is feasible, then $|\mathcal{F}_k|$ exists $\forall k \in \{0, \ldots, N\}$ (Proposition (4.1.2) states that $\mathcal{F}_k$ is nonempty if (MPC.2) is feasible, and $\mathcal{F}_k$ is closed).

Equation (4.13) can be used to show that $\max \mathcal{F}_k > \min \mathcal{F}_k$ (i.e. $|\mathcal{F}_k| > 0$) for all k where $\max \mathcal{F}_k = \max \mathcal{X}$ and/or $\min \mathcal{F}_k = \min \mathcal{X}$, and from (4.2) it is clear that

$|\mathcal{F}_k| \leq |\mathcal{F}_{k+1}|$ if neither $\max \mathcal{F}_{k+1} = \max \mathcal{X}$ nor $\min \mathcal{F}_{k+1} = \min \mathcal{X}$. Suppose that $\max \mathcal{F}_k = \max \mathcal{X}$ and/or $\min \mathcal{F}_k = \min \mathcal{X}$ for some $k \in \{0, \dots, N\}$ and let $k^\dagger$ be the smallest such value. It can then be shown by induction that $|\mathcal{F}_k| > 0 \; \forall k \in \{k^\dagger, \dots, N\}$.

Now, note that

$$\mathcal{G}_k = \{x_{k+1} + u_k \in \mathcal{F}_k : x_k \in \mathcal{G}_{k+1}, u_k \in \mathcal{U}_k\}.$$

If it is assumed that $|\mathcal{F}_k| > 0$ and $|\mathcal{G}_{k+1}| > 0$, it can then be shown that

$$\begin{aligned} |\mathcal{F}_{k+1}| > 0 &\implies |\mathcal{F}_k \cap \{\mathcal{F}_{k+1} \oplus \mathcal{U}_k\}| > 0 && (\mathcal{F}_{k+1} = \{\mathcal{F}_k \oplus -\mathcal{U}_k\} \cap \mathcal{X}) \\ &\implies |\mathcal{F}_k \cap \{\mathcal{G}_{k+1} \oplus \mathcal{U}_k\}| > 0 && (\mathcal{G}_{k+1} \subseteq \mathcal{F}_{k+1} \text{ and } |\mathcal{G}_{k+1}| > 0) \\ &\implies |\mathcal{G}_k| > 0 && (\text{Definition of } \mathcal{G}_k). \end{aligned} \tag{4.14}$$

Since $k^\dagger \leq N$ by assumption, it is therefore known that $|\mathcal{F}_N| > 0$, so $|\mathcal{G}_N| > 0$, and the argument in (4.14) can be made recursively to show that $|\mathcal{G}_k| > 0 \; \forall k \in \{N, \dots, k^\dagger\}$. Therefore, as $\mathcal{G}_k \subseteq \mathcal{F}_k \subseteq \mathcal{X}$, it can be concluded that $x_k^{(0)} \in \text{int}(\mathcal{X}) \; \forall k \in \{k^\dagger, \dots, N\}$.

If (MPC.2) is feasible then $|\mathcal{F}_k| \geq 0$ for all k, and similar logic to (4.14) can be used to show that $|\mathcal{G}_k| \geq 0 \; \forall k \in \{k^\dagger - 1, \dots, 1\}$. From the definition of $k^\dagger$ it is known that $\max \mathcal{F}_k \neq \max \mathcal{X}$ and $\min \mathcal{F}_k \neq \min \mathcal{X} \; \forall k \in \{k^\dagger - 1, \dots, 1\}$, so as $\mathcal{G}_k \subseteq \mathcal{F}_k \subset \mathcal{X}$ it can be concluded that $x_k^{(0)} \in \text{int}(\mathcal{X}) \; \forall k \in \{k^\dagger - 1, \dots, 1\}$. The same result can also be shown for $\forall k \in \{N, \dots, 1\}$ if (4.13) holds and $k^\dagger$ does not exist.

The result that $x_k^{(0)} \in \text{int}(\mathcal{X}) \; \forall k \in \{1, \dots, N\}$ implies that $Au^{(0)} - b > 0$, so $s^{(0)} = Au^{(0)} - c$ ensures (4.12c) and (4.12f), and $\theta_1^{(0)} = \frac{1}{\mu}(S^{(0)})^{-1}\mathbf{1}$ ensures (4.12g).

Finally, it can be shown that

$$\max u_k^{(0)} = \frac{1}{2}\left(\max \mathcal{G}_k - \max \mathcal{G}_{k+1} + \min \mathcal{G}_k - \min \mathcal{G}_{k+1}\right) = \overline{u}_k,$$

and it can similarly be shown that $\min u_k^{(0)} = \underline{u}_k$, which therefore demonstrates (4.12c)-(4.12g) if (4.13) is satisfied. Conversely, a subset of the above results can be shown to be not true if (4.13) is not satisfied for any $k \in \{1, \dots, N\}$.

□

Remark 4.2.1. The computational and additional memory requirements of Algorithm 3 scale linearly with N.

Remark 4.2.2. Condition (4.13) holds if problem (MPC.2) satisfies Slater's condition [96, §5.2.3] (the converse statement is not necessarily true).

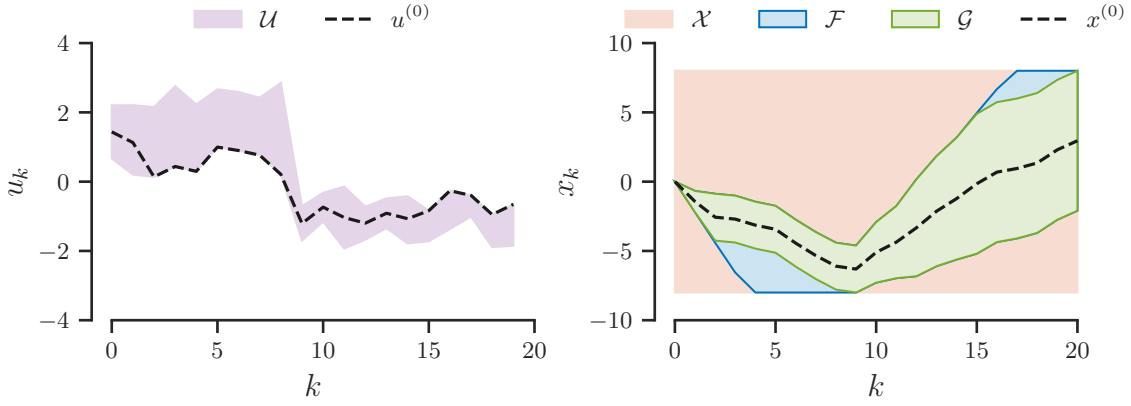


Figure 4.1: Example of the tubes defined by $\mathcal{F}$ and $\mathcal{G}$, and the solutions obtained for $x^{(0)}$ and $u^{(0)}$ using Algorithm 3 for a nominal system (the values were chosen to illustrate the operation of the algorithm, and are not necessarily representative of those observed in the energy management problem).

An example of the values of $x^{(0)}$ and $u^{(0)}$ obtained by Algorithm 3 for a representative example is shown in Figure 4.1. In addition to u, s, and θ_1, there are three further parameters that are initialized at the start of the algorithm: $\mu_0 \in \mathbb{R}_+$, $\overline{\mu} \in [\mu_0, \infty)$, $k_\mu \in (1, \infty)$, and $\tau \in (0, 1)$. These parameters can be assigned any value within the stated ranges, and their significance is discussed in the following section.

ALGORITHM

At each iteration, i, the elements k of $\mathcal{P}$ are partitioned into the the sets

$$\mathcal{A}^{(i)} := \{k \in \mathcal{P} : (u_k^{(i)} = \underline{u}_k \wedge F_k'(u_k^{(i)}) - A_{:,k}^\top \theta_1^{(i)} > 0) \\ \vee \ (u_k^{(i)} = \overline{u}_k \wedge F_k'(u_k^{(i)}) - A_{:,k}^\top \theta_1^{(i)} < 0)\},$$

$$\mathcal{D}^{(i)} := \mathcal{P} \setminus \mathcal{A}^{(i)},$$

then the solution to (4.12a)-(4.12c) is estimated using the Newton step

$$\begin{bmatrix} \nabla^2_{\mathcal{D}^{(i)}} F(u^{(i)}) & 0 & -w^\top \\ 0 & \Theta_1^{(i)} & S^{(i)} \\ w & -I & 0 \end{bmatrix} \begin{bmatrix} \Delta u^{(i)}_{\mathcal{D}^{(i)}} \\ \Delta s^{(i)} \\ \Delta \theta_1^{(i)} \end{bmatrix} := \begin{bmatrix} -\nabla_{\mathcal{D}^{(i)}} F(u^{(i)}) + v^\top \theta_1^{(i)} \\ \frac{1}{\mu} - S^{(i)} \theta_1^{(i)} \\ -Au^{(i)} + c + s^{(i)} \end{bmatrix}, \tag{4.15}$$

where $\nabla^2_{\mathcal{D}^{(i)}} F$ is a diagonal matrix of the $k \in \mathcal{D}^{(i)}$ diagonal elements of the Hessian of F, $\nabla_{\mathcal{D}^{(i)}} F$ is the $k \in \mathcal{D}^{(i)}$ elements of the gradient of F, and $\Theta_1 = \mathrm{diag}(\theta_1)$. Define $\hat{A}$ as the

matrix A with the $\{k \in \{0, \ldots, N-1\} : \underline{u}_k = u_k^{(i)} \vee \overline{u}_k = u_k^{(i)}\}$ columns set to zero, then v is defined as the $k \in \mathcal{D}^{(i)}$ columns of A, and w is defined as the $k \in \mathcal{D}^{(i)}$ columns of $\hat{A}$. The search directions are then obtained from the reduced equations

$$\begin{aligned}\Delta\theta_1^{(j)} = &\left[w\left(\nabla^2_{\mathcal{D}^{(i)}} F(u^{(i)})\right) w^\top + (\Theta_1^{(i)})^{-1} S^{(i)}\right]^{-1} \\ &\left(\frac{1}{\mu}\left(\Theta_1^{(i)}\right)^{-1}\mathbf{1} - Au^{(i)} + c - w\left(\nabla^2_{\mathcal{D}^{(i)}} F(u^{(i)})\right)^{-1}\right. \\ &\left.\left(-\nabla_{\mathcal{D}^{(i)}} F(u^{(i)}) + v^\top \theta_1^{(i)}\right)\right), \end{aligned} \tag{4.16a}$$

$$\Delta u^{(i)}_{\mathcal{D}^{(i)}} = -\left(\nabla^2_{\mathcal{D}^{(i)}} F(u^{(i)})\right)^{-1}\left(\nabla_{\mathcal{D}^{(i)}} F(u^{(i)}) - v^\top\theta_1^{(i)} - w^\top \Delta\theta_1^{(i)}\right), \tag{4.16b}$$

$$\Delta u_{\mathcal{A}^{(i)}} = \mathbf{0} \tag{4.16c}$$

$$\Delta s^{(i)} = Au^{(i)} + w\Delta u^{(i)}_{\mathcal{D}^{(i)}} - c - s^{(i)}, \tag{4.16d}$$

and step lengths are determined from the 'fraction to the boundary' rule [64, §19.2] as

$$\alpha_s := \max\left\{\alpha \in (0,1] : s^{(i)} + \alpha\Delta s^{(i)} \geq (1-\tau)s^{(i)}\right\}, \tag{4.17a}$$

$$\alpha_\theta := \max\left\{\alpha \in (0,1] : \theta^{(i)} + \alpha\Delta\theta_1^{(i)} \geq (1-\tau)\theta_1^{(i)}\right\}, \tag{4.17b}$$

where $\tau \in (0,1)$ is fixed and arbitrary within this range. The primal-dual interior point iteration is then defined by

$$u_k^{(i+1)} := \begin{cases} \Pi_{\mathcal{U}_k}\left[u_k^{(i)} + \alpha_s \Delta u_k^{(i)}\right] & \forall k \in \mathcal{D}^{(i)} \\ u^{(i)} & \forall k \in \mathcal{A}^{(i)} \end{cases}, \tag{4.18a}$$

$$s^{(i+1)} = s^{(i)} + \alpha_s \Delta s, \tag{4.18b}$$

$$\theta_1^{(i+1)} = \theta_1^{(i)} + \alpha_\theta \Delta\theta_1. \tag{4.18c}$$

Iteration (4.18) is performed repeatedly until the criterion

$$r_{\text{IP}}^{(i)} := \max\left\{\left\|-\nabla_{\mathcal{D}^{(i)}} F(u^{(i)}) + v^\top\theta_1^{(i)}\right\|, \left\|\frac{1}{\mu} - S^{(i)}\theta_1^{(i)}\right\|, \left\|-Au^{(i)} + c + s^{(i)}\right\|\right\} < \frac{1}{\mu} \tag{4.19}$$

is met, at which point the value of μ is updated using

$$\mu :\leftarrow \min\{\overline{\mu}, k_\mu \mu\}, \tag{4.20}$$

where $\overline{\mu} \in (0, \infty)$ is a predetermined upper limit on the value of μ, and $k_\mu \in (1, \infty)$ is a predetermined, arbitrary constant. The projected interior point algorithm is terminated at the first iteration where both of the conditions (4.19) and $\mu = \overline{\mu}$ are met. A pseudocode implementation of the projected interior point algorithm is presented in Algorithm 4.

Algorithm 4 Projected Interior Point Algorithm for Problem (MPC.2)

Input: $(x(t), \underline{u}, \overline{u}, \underline{x}, \overline{x}, F, \overline{\mu}, \mu_0, k_\mu, \tau)$
Output: $u^\dagger$

($\mathcal{F}$, Feasible) $\leftarrow$ Algorithm 1.
if Feasible = False **then**
  return $\sim$
end if
$(u^{(0)}, s^{(0)}, \theta_1^{(0)}, \text{FLAG}) \leftarrow$ Algorithm 3.
if FLAG = False **then**
  return $\sim$
end if
$\mu \leftarrow \mu_0$
$i \leftarrow 0$
$(\mathcal{A}^{(0)}, \mathcal{D}^{(0)}) \leftarrow$ (4.15)
while $\mu < \overline{\mu} \vee r_{\text{IP}}^{(i)} \geq \frac{1}{\mu}$ **do**
  $(\Delta u^{(i)}, \Delta s^{(i)}, \Delta \theta_1^{(i)}) \leftarrow$ (4.16)
  $(\alpha_s, \alpha_\theta) \leftarrow$ (4.17)
  $(u^{(i+1)}, s^{(i+1)}, \theta_1^{(i+1)}) \leftarrow$ (4.18)
  $i \leftarrow i + 1$
  $(\mathcal{A}^{(j)}, \mathcal{D}^{(j)}) \leftarrow$ (4.15)
  $r_{\text{IP}}^{(i)} \leftarrow$ (4.19)
  if $r_{\text{IP}}^{(i)} < \frac{1}{\mu}$ **then**
    $\mu \leftarrow \min\{\overline{\mu}, k_\mu \mu\}$
  end if
end while
return $u^\dagger \leftarrow u^{(j)}$

Table 4.2: Summary of the computational complexity of each of the interior point variable updates.

Equation(s)	$M^{-1} \cdot v$	$M \cdot M$	$M \cdot v$	$\mathcal{O}(2^n N^n)$
(4.16a)	Yes	Yes	Yes	$n \leq 3$
(4.16b - 4.16d), (4.19), (4.15)	No	No	Yes	$n \leq 2$
(4.17), (4.18)	No	No	No	$n = 1$

COMPLEXITY

Lines 13 to 18 in Algorithm 4 constitute the variable updates that are performed during each iteration of the interior point algorithm. The complexity of the operations required for each variable update are summarized in Table 4.2, considering dense systems of linear equations, dense matrix-matrix products, and dense matrix-vector products. The significance of enforcing the box constraints on u as a projection is highlighted, as the complexity of the projected interior-point operations is a function of the first dimension of A, which is $2N$ in this case. If the bounds on u were applied as a log-barrier function [i.e., $A = (\Psi, -\Psi, I - I)$ and $c = (\mathbf{1}(x(t) - \overline{x}), -\mathbf{1}(x(t) - \underline{x}), \underline{u}, \overline{u})$ in (IP.b)], the relevant dimension would instead be $4N$, and the computational complexity of each update would become $\mathcal{O}(4^n N^n)$. In the case of a system of linear equations, this could result in a reduction in worst cast computational time of almost an order of magnitude. It can be seen that (4.16a) is the most computationally demanding update due to the presence of both a dense matrix-matrix multiplication and a dense system of linear equations (note that *diagonal* matrix operations, e.g. $(\nabla^2_{\mathcal{D}^{(i)}} F(u^{(i)}))^{-1}$ are omitted from Table 4.2). The computational complexity of each iteration of the algorithm is therefore $\mathcal{O}(N^n)$, where $n \leq 3$ is determined by the method used for matrix multiplication and to solve the systems of linear equations. The worst case memory complexity is $\mathcal{O}(N^2)$, as the matrices A, v, and w must be stored in memory, although it may be possible to exploit the fact that A is constructed from Ψ blocks to reduce this requirement.

CONVERGENCE

The iteration (4.18) can be interpreted as as a projected Newton method used to obtain a stationary point $(u^\circ, s^\circ, \theta_1^\circ)$ of the function

$$\hat{\mathcal{L}}_\mu(u, s, \theta_1) := F(u) + B_\mu(s) - \theta_1^\top (Au - c - s) \tag{4.21}$$

subject to the constraint $\underline{u} \leq u \leq \overline{u}$. The strict inequality in the definition of $\mathcal{A}^\circ$ implies that there is a region of $(u^{(i)}, s^{(i)}, \theta_1^{(i)})$-space close to $(u^\circ, s^\circ, \theta_1^\circ)$ where $\mathcal{A}^{(i)} = \mathcal{A}^\circ$, and a subset of this region will meet the conditions for the local quadratic convergence of

Newton's method for nonlinear equations [64, Theorem 11.2]. This means that the right-hand-side of (4.15) converges to $\mathbf{0}$ as $i \to \infty$, and the termination criterion (4.19) will be met in a finite number of steps. Global convergence could be ensured by adapting the Δu step at each iteration with a line-search of an appropriate merit function, although the merit function from [119] cannot be used in the proposed form as the stationary point $(u^\circ, s^\circ, \theta_1^\circ)$ is not a minimum of the function (4.21) in general. Despite this limitation, convergence was demonstrated for all problem classes in the simulations that follow.

It was previously demonstrated that $u^\circ \to u^\star$ as μ is increased toward ∞, and the value of $u^{(i)}$ when criterion (4.19) is met converges to u° as μ is increased toward ∞. Therefore, the value of $u^\dagger$ returned when Algorithm 4 terminates can be made arbitrarily close to the minimizing argument of (IP.a) (and therefore (MPC.2)) by setting $\overline{\mu}$ arbitrarily high. The algorithm could be further optimized to update the value of μ possibly at every iteration, to ensure that the iterate remains in, or at least near to, the superlinearly convergent region around $(u^\circ, s^\circ, \theta_1^\circ)$, although superlinear convergence is demonstrated for a broad class of problems in the numerical experiments that follow using the update (4.20).

4.3 NUMERICAL EXPERIMENTS

This section presents two sets of numerical experiments. Initially, both of the algorithms are tested against randomly generated examples of problem (MPC.2) to determine their relative computational performance without reference to a specific powertrain. Then, a model predictive controller defined by the solution of (MPC.2) is implemented in closed-loop on the same driver behaviour data simulated in Section 3.4, in comparison with dynamic programming.

4.3.1 ALGORITHM COMPARISON

Using observations from the experiments in Section 3.4, predictions of driver power demand were generated from $p_k \in [-2.5 \times 10^3, 10^4]$, and hardware parameters generated from $\alpha_{2,k}, \beta_{2,k} \in [-5 \times 10^{-4}, 5 \times 10^{-4}]$, $\alpha_{1,k}, \beta_{1,k} \in [0.5, 1.5]$, and $\alpha_{0,k}, \beta_{0,k} \in \{0\}$ $\forall k \in \{0, \ldots, N-1\}$. All values were sampled uniformly within the stated ranges. The battery parameters were set at $V_k = 300$ and $R_k = 0.1\ \forall k \in \{0, \ldots, N-1\}$, and the limits on state and input were set at $\overline{x} = 10^5$, $\underline{x} = 0$, $\underline{u} = -\mathbf{1} \cdot 15 \times 10^3$ and $\overline{u} = \mathbf{1} \cdot 15 \times 10^3$. An initial state of charge of $x(t) = 0.9 \cdot \overline{x}$ was assumed for all experiments. The specified limits were chosen to approximate those obtained in Section 3.4, and to ensure that the problems were feasible and that both the state and input constraints were active at the solution. Finally, the effect of varying τ was not investigated and was set at $\tau = 0.995$ for all projected interior point solutions.

OPTIMUM

It was previously demonstrated that the output of Algorithm 4 can be made arbitrarily close to the solution of (MPC.2) by using a sufficiently large value of $\overline{\mu}$, and it is therefore necessary to determine a value of $\overline{\mu}$ that can be used to obtain a sufficiently accurate approximation of $u^\star$. Ten problems were generated for each horizon length of $N \in \{100, 200, 300, 400\}$ for a total of forty problems, and the projected interior point method was used to obtain a solution for each with the parameters $\mu_0 = \overline{\mu}$, and $\overline{\mu}$ iteratively increased from 10^2 to 10^5 in twenty logarithmically spaced points (k_μ is not required as the algorithm is terminated when condition (4.19) is first met). For each $\overline{\mu}_j$, $j \in \{1, \ldots, 20\}$, the control input vector at termination, $u_j^\dagger$, was recorded, forming the sequence $(u_1^\dagger, \ldots, u_{20}^\dagger)$ for each problem. Figure 4.2 shows that as $\overline{\mu}$ was increased, the norm of the difference between the value of $u^\dagger$ when the algorithm terminated with successive values of $\overline{\mu}$, $\|u_j^\dagger - u_{j-1}^\dagger\|$, decreased within an inverse band for all cases, and that at $\overline{\mu} = 10^5$, this metric had reduced to less than 1 for all problem cases. Given that the decision variable, u, can take a range of values in the order of 10^4, it was concluded that $\overline{\mu} = 10^5$ is sufficiently large to provide an accurate solution to problem (MPC.2), and all future references to $u^\star$ refer to control inputs found using Algorithm 4 with $\mu_0 = \overline{\mu} = 10^5$.

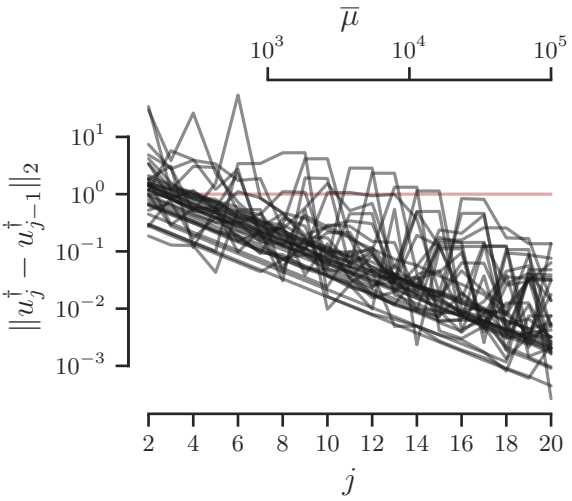


Figure 4.2: Data showing the decrease in the change of control vector obtained by the projected interior-point method as $\overline{\mu}$ is increased from 10^0 to 10^5. The red line is used to highlight the 10^0 threshold on the vertical axis.

ALGORITHM TUNING

Both algorithms have multiple parameters that must be tuned to provide computationally efficient solutions. For the ADMM algorithm, ρ_1 and ρ_2 (which can be loosely interpreted as the step length in a gradient descent algorithm) must be determined, whilst μ_0 and k_μ must be determined for the projected interior point method. The energy management MPC framework is commonly implemented with a shrinking horizon, and it is therefore important that the same set of parameters provide a similar level of performance for a broad class of problems over both long and short horizons. This section details the results of

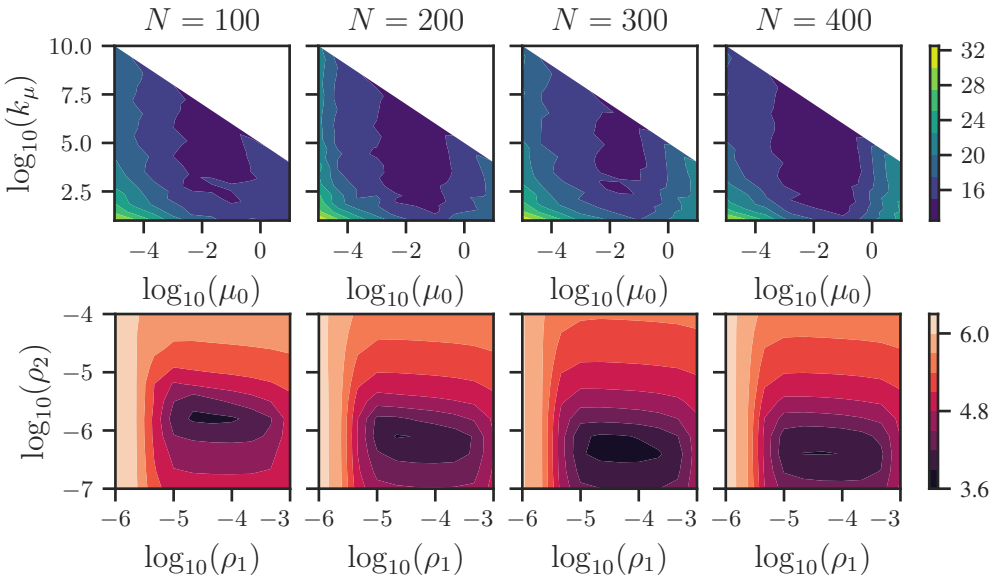


Figure 4.3: Results of parameter tuning for both the projected interior-point (top row) and ADMM (bottom row) algorithms. The projected interior-point figures show that average number of iterations required to meet the termination threshold $\overline{\mu} = 10^5$, whereas the ADMM figures show the average error measure by $\|u^{(100)} - u^\star\|_2^2$.

investigations to determine the most computationally efficient combination of parameters for each algorithm.

To determine the values of ρ_1, ρ_2, μ_0, and k_μ, that provide optimal convergence for the ADMM and projected interior point algorithms, twenty new problems were generated for each horizon length of of $N \in \{100, 200, 300, 400\}$. These were solved using the projected interior point algorithm with $\overline{\mu} = 10^5$, $10^{-5} \leq \mu_0 \leq 10^1$, and $1 < k_\mu \leq 10^5/\mu_0$ (any value of k_μ greater than this would ensure that μ is projected onto $\overline{\mu}$ during the first update step (4.18)). For each problem instance, the number of iterations required for the algorithm to terminate was recorded, and the average for each combination of parameters is shown in Figure 4.3. It can be seen that there is a vertically banded region at $\mu_0 \approx 10^{-1}$ that requires a minimum number of iterations for all horizon lengths, and that there is a profile to the search space that varies little with changes in horizon length. The values $\mu_0 = 10^{-1}$ and $k_\mu = 10^4$ were therefore selected as the optimal parameters.

For the ADMM algorithm a different approach was taken, as the significantly more iterations were required to achieve the same level of accuracy as the projected interior point algorithm with $\overline{\mu} = 10^5$. Instead, a total of 100 ADMM iterations were completed for each problem (ignoring the stated termination criteria) with $10^{-6} \leq \rho_1 \leq 10^{-3}$ and $10^{-7} \leq \rho_2 \leq 10^{-4}$. The average norm of the difference between the control input at

the 100th iteration of ADMM, $u^{(100)}$, and the optimum, $u^\star$, was recorded for each case. The results are shown in Figure 4.3, and there is a clear region within approximately two orders of magnitude of both ρ_1 and ρ_2 where the control vector has a minimum error relative to the optimum, and this region does not change significantly with horizon length. The values of $\rho_1 = 6 \times 10^{-5}$ and $\rho_2 = 4 \times 10^{-7}$ were therefore selected as the optimal parameters. To the best of the author's knowledge, there is no widely accepted method for tuning the ρ parameters, but note that ρ_1 corresponds to a constraint on u, which represents power, whereas ρ_2 corresponds to a constraint on Ψu, which represents energy. Typical magnitudes for the elements of $u^\star$ are $10^3 - 10^4$ W, and the limits on energy are set at 10^5, which may suggest the tuned ρ parameters work well because they scale the primal residuals to approximately the same magnitude. This is only conjecture, however, and requires further investigation.

COMPUTATIONAL PERFORMANCE

After tuning the parameters of both the projected interior point and ADMM algorithm to the class of problems being investigated, it was possible to analyse their comparative computational performance. This was achieved in two steps: firstly the termination criterion for a 'sufficiently' accurate solution was determined, then the variation in computational time with horizon length was investigated.

A further twenty test cases were generated consisting of 5 cases for each of $N \in \{100, 200, 300, 400\}$, and using the values of ρ_1, ρ_2, μ_0 and k_μ determined during the tuning phase, each problem was solved using ADMM for one hundred iterations, and using the projected interior point algorithm with $\overline{\mu} = 10^5$. The absolute difference between the cost evaluated at iteration j and the optimal cost, $|F(u^{(j)}) - F(u^\star)|$, is shown in Figure 4.4. The results clearly demonstrate sublinear convergence for the ADMM algorithm, whilst the projected interior point results show superlinear convergence. Therefore, the projected interior point algorithm can produce an extremely accurate solution within a few tens of iterations, whereas significantly more iterations are required for ADMM.

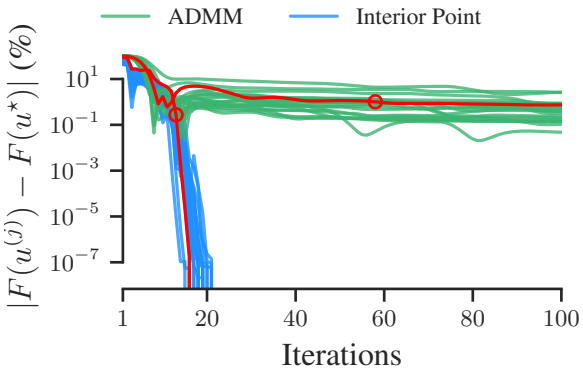


Figure 4.4: Curves showing the normalized error between the cost evaluated at iteration j and the optimum, as a percentage, for twenty systems using both the projected interior-point method and the ADMM. The curves highlighted in red correspond to the system shown in Figure 4.5, and the red circle shows the iteration from which those curves were taken.

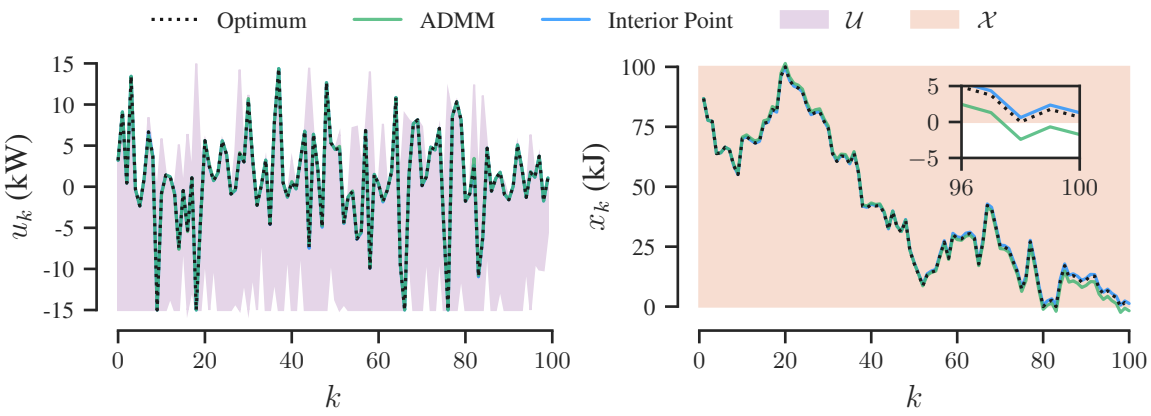


Figure 4.5: Control and state vectors for the systems highlighted with red circles in Figure 4.4, plotted against the optimum $u^\star$ and $x^\star$. The feasible tubes $\mathcal{U}$ and $\mathcal{X}$ are also included, and note that the additional constraints enforced during the convex formulation specified in Section 3.2.1 have significantly restricted the upper and lower bounds on $\mathcal{U}$ from the original $\pm 1.5 \times 10^4$.

A threshold of 1% was considered high enough for 'sufficient' accuracy, as this is likely to be lower than the level of uncertainty in state measurements used to formulate the problem, and the results shown in Figure 4.5 illustrate that the deviation between the control vectors obtained by both the ADMM and projected interior point algorithms and the optimum are almost imperceptible at this level of convergence. A slightly larger deviation is observed between the state trajectories, particularly that obtained with ADMM. However, this is because the state trajectory is a function of the integral of the control input, and as the cost is not a function of state-of-charge this does not necessarily indicate greater sub-optimality. A key property of the algorithms is also demonstrated in Figure 4.5: the state constraints are only guaranteed for both algorithms when the residuals, r and s, are precisely zero (this is also why the absolute error is presented in Figure 4.4, as the cost evaluated for each iteration of can be *lower* than $F(u^\star)$). Therefore, the termination criteria do not provide a strict guarantee of enforcing the state constraints, and we can see that for the final three timesteps the lower state limit is violated by $\approx 2.5\%$ of the feasible state band for the ADMM trajectory. This limitation can be reduced by tightening the algorithms' convergence thresholds, which makes it more significant for ADMM due to its sublinear rate of convergence.

Based on residuals for the projected interior point and ADMM trajectories shown in Figure 4.5, it was assumed that $\epsilon = 4 \times 10^3$ and $\overline{\mu} = 1$ enforce a 'sufficient' level of convergence. A further twenty problems were generated for horizons $50 \leq N \leq 1000$, and the iterations to completion, mean time taken per iteration, and time to completion

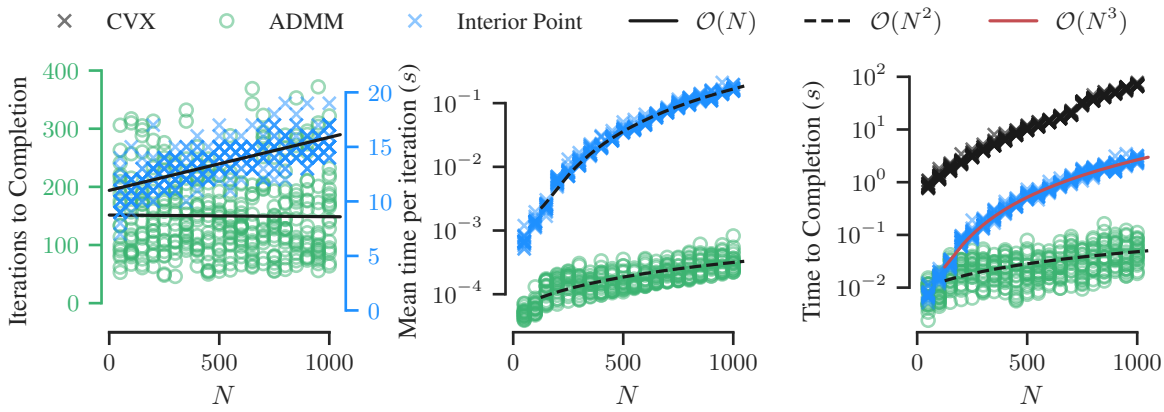


Figure 4.6: Axes showing the number of iterations required, mean time per iteration, and time to completion for twenty systems with $50 \leq N \leq 1000$*, with linear, quadratic, and cubic trend lines.*

were recorded for each using both ADMM and the projected interior point algorithm. For comparison, the problems were also solved using CVX with default solver SDPT3 v.4.0 and default error tolerance, for which only the total time was recorded (it is not possible to separate the total time from the individual iterations or the overhead required to parse the problem when using CVX). The results are shown in Figure 4.6, where it can be seen that whilst the uncertainty in the number of ADMM iterations is high (from as low as 50 to as high as 400), the band of uncertainty is near constant as the horizon is increased, so it can be assumed that the expected number of iterations is effectively constant with horizon length. The uncertainty for the number of projected interior point iterations is lower, and fewer iterations are required for all horizon lengths, however the number of iterations also increases linearly with horizon length from ~ 10 iterations at $N = 50$ to ~ 16 iterations at $N = 1000$.

It was previously shown that as the horizon length is increased the computational burden of the projected interior point algorithm is dominated by the $\Delta\theta_1$ update, so the methods used to perform this calculation will largely determine the time required per iteration. The Matlab operations `x=A\b` and `x=A*b` were used here, and a quadratic trendline is shown to have an approximate fit in Figure 4.6. The computation of each iteration of the ADMM algorithm was previously shown to scale linearly with horizon length, and this is also illustrated with a linear trendline in Figure 4.6. Additionally, the ADMM iteration was over two orders of magnitude faster than the interior point iteration as the horizon length was increased to 1000 samples.

Given the observed scaling properties of the number of iterations required for convergence and the computation of each iteration, it would therefore be expected that the total

time taken for the ADMM algorithm to terminate would scale linearly with horizon length whereas the time taken for the projected interior point method would scale cubically with horizon length, and this is supported by the results shown in the third plot in Figure 4.6. It is also shown that, with an assumed interval of one second between controller optimizations, the projected interior point is only suitable up to a horizon of $N \approx 500$, whereas even up to the maximum horizon length of $N = 1000$, the ADMM algorithm required less than $\sim 0.1s$, and from the previous scaling properties it can be assumed that ADMM is real time implementable for horizons significantly in excess of 1000. Therefore, whilst the projected interior point algorithm has been shown to converge to an extremely accurate solution in fewer iterations than the ADMM algorithm, for the hardware used in these experiments, less time is required for a moderate level of accuracy using ADMM, and the ADMM algorithm scales better with horizon length. It is worth noting, however, that if the accuracy requirement were significantly tightened, it is likely that this performance relationship could change. Furthermore, it may also be possible to further accelerate the interior point iteration using efficient factorizations such as those proposed in [120] and [121].

In comparison with the proposed algorithms, CVX was unable to obtain solutions in less than 1s for any horizon length, and was at least an order of magnitude slower than both algorithms over all horizon lengths; compared to ADMM it was a factor of 1000 slower for $N = 1000$. Although CVX is solving the problem to a different error tolerance, it would be expected that ADMM would still be faster were its termination threshold significantly tightened. This is the first demonstration of a method capable of solving the energy management problem in real time, over long horizons ($\geq$1000 samples) when nonlinear system dynamics are considered and hard limits on both power and state of charge are enforced over the entire horizon.

4.3.2 Energy Management

Control Algorithms

The same driver behaviour data as that presented in Section 3.4 was used for simulation, and for each journey, the state of charge of the battery was initialized at 60% and constrained between 40% and 100%. Four control algorithms were used:

1. **Dynamic Programming**. The gear selection, braking, and engine switching were controlled using the same heuristics as presented for MPC in Section 3.2, and the dynamic programming algorithm described in Section 2.2.1 was used with the exact engine, motor, and battery models to determine the globally optimal power split. The state space (the battery state x_t) was evenly discretized in 1000 intervals and

the control space (the battery power u_t) was evenly discretized in 100 intervals (this was found through trial-and-error to be sufficient for a highly accurate solution).

2. **MPC.** The state constrained MPC algorithm described in this chapter was used where it was assumed that the controller had perfect predictions of the future velocity and road gradients (as in Section 3.4). At each control variable update instant the power delivered from the battery was implemented as the first element of the solution of (MPC.2), for which the ADMM algorithm was used to obtain the solution. The ρ parameters were re-tuned as $\rho_1 = 2.34 \times 10^{-4}$ and $\rho_2 = 10^{-8}$, and the termination criterion, ϵ, was set at 5×10^4, which was found to eliminate the constraint violation issue illustrated in Figure 4.5.

3. **MPC (no electrical losses).** The same MPC algorithm as described above was used with the exception that all of the electrical losses in the predictive model were ignored (i.e. $f_k(p_k - g_k^{-1}(u_k)) = f_k(p_k - u_k)\ \forall k \in \mathcal{P}$). The ADMM algorithm was also used for the MPC optimization.

4. **CDCS.** In the initial charge-depleting phase the engine was switched off and all power was delivered from the motor until the lower threshold on the state-of-charge of the battery was met (the engine was also turned on to meet any power that exceeded the motor's power limit, but was then immediately turned off again). Once the lower limit on the battery's state of charge was met, the engine was turned on and kept on until the end of the journey, during which all positive power was delivered from the motor if the state-of-charge of the battery was above its lower limit, and from the engine if the lower state-of-charge limit was exceeded. All negative power was delivered from the battery.

RESULTS

Figure 4.7 shows the results for all forty-nine journeys using dynamic progamming and MPC both with and without the electrical losses, and it can now be observed that the state-of-charge of the battery satisfies its upper and lower constraints at all times. The route used for simulation consists of approximately 7km of uphill followed by 7km of downhill (it is a return journey), so a useful artefact is that the vehicle is regenerating for the second half of the route and ends the journey with the same battery state-of-charge across all control methods. This means that the relative efficiency of each control method can be determined by comparing the total fuel consumption alone (if this were not the case the terminal state-of-charge could be constrained by tightening the state-of-charge constraint for the final timestep with $X_T \leq x_N \leq X_T$). It is shown that the MPC controller that considers the electrical losses in the system achieves an extremely close approximation of the globally

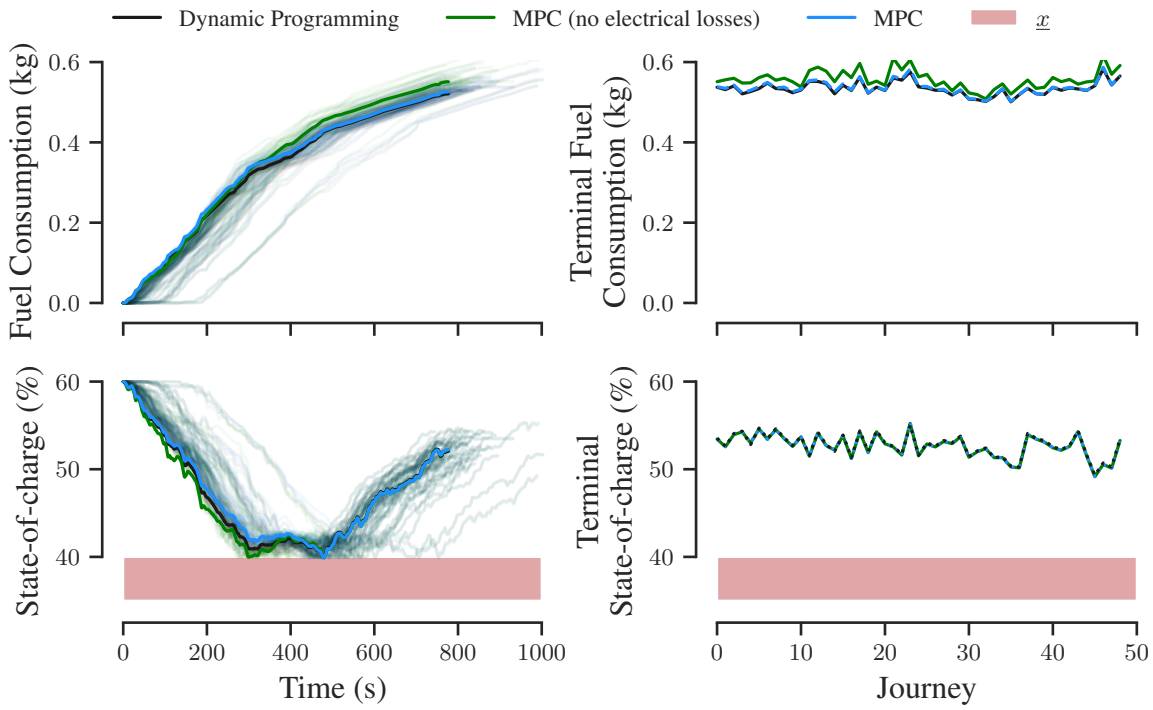


Figure 4.7: Closed loop results using dynamic programming, MPC, and MPC with the electrical losses ignored in the predicted dynamics. The hard lower bound on the state-of-charge of the battery is shown in the red area.

optimal solution obtained with dynamic programming, and that when the electrical losses are ignored the fuel consumption increases. On average, the fuel consumption increased relative to dynamic programming by 3.4% when using MPC without the electrical losses, but the fuel consumption was equal (to the nearest 0.1%) for both dynamic programming and MPC when using the full loss model. This provides a clear justification for using non-linear models of the electric elements of the powertrain in the MPC optimization. Figure 4.8 shows a histogram of the time taken using dynamic programming and the first MPC optimization (the shrinking horizon implementation means that the computational time reduces as the journey progresses) for both powertrain models across all 49 journeys. The average time taken for dynamic programming is 260 s, which is clearly too slow for a receding horizon implementation with an update frequency of 1s. Conversely, the time taken using the ADMM algorithm is significantly less that one second: the average where the electrical losses is considered is 0.10 s, and is reduced to 0.043 s when the electrical losses are ignored. This acceleration is achieved because the ADMM update (4.9a) has an

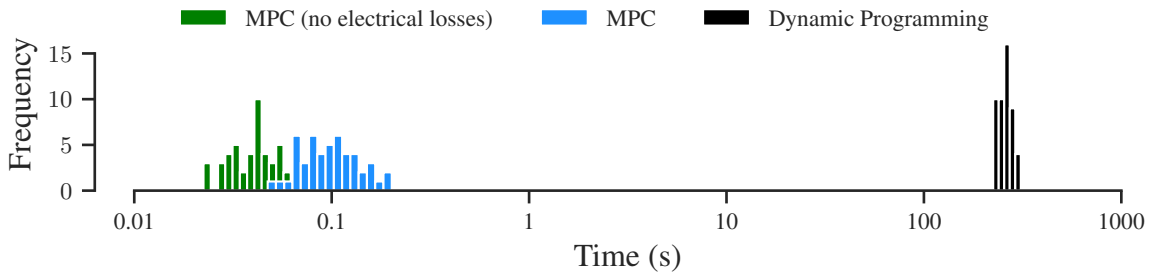


Figure 4.8: Time taken for the first MPC optimization (i.e. longest prediction horizon) for each journey using dynamic programming and both variations of MPC.

analytical solution when the electrical losses are ignored, although it comes at the price of the reduced closed loop performance illustrated in Figure 4.7.

Figure 4.9 shows the results using CDCS compared with the MPC results from Figure 4.7, and illustrates the potential limitation of using a suboptimal heuristic for engine switching control. The MPC controller only switches the engine off if the selected gear results in the engine speed being reduced below a minimum threshold (return to Section 3.2 for the details of the MPC heuristics), whereas the engine is off for a significant portion of the start of the journey using CDCS. Consequently, CDCS actually reduces fuel consumption by 1.1 % on average. This demonstrates that the engine switching control should also be considered in the MPC optimization in order for the resulting energy-management strategy to be optimal in terms of fuel consumption.

4.4 Concluding Remarks

In this chapter it was demonstrated that the convex MPC formulation presented in the Chapter 3 can be readily extended to consider the state-of-charge constraints across the entire prediction horizon. A computationally cheap algorithm was presented for obtaining a feasibility certificate for the problem, and then an ADMM algorithm and a novel projected interior point algorithm were presented for its solution. It was demonstrated in a set of numerical studies that the projected interior point method could obtain a highly accurate solution in fewer iterations, but that a sufficiently accurate solution could be obtained much faster using ADMM. The MPC optimization was implemented in closed-loop where it was shown to very closely approximate the globally optimal solution obtained using dynamic program, and that the performance of the MPC controller is reduced when the nonlinear losses in the battery are ignored. Finally, it was demonstrated that a poor engine control heuristic can render the optimized control inputs as poor as CDCS, so in the following chapter, the MPC optimization is extended to also consider engine switching.

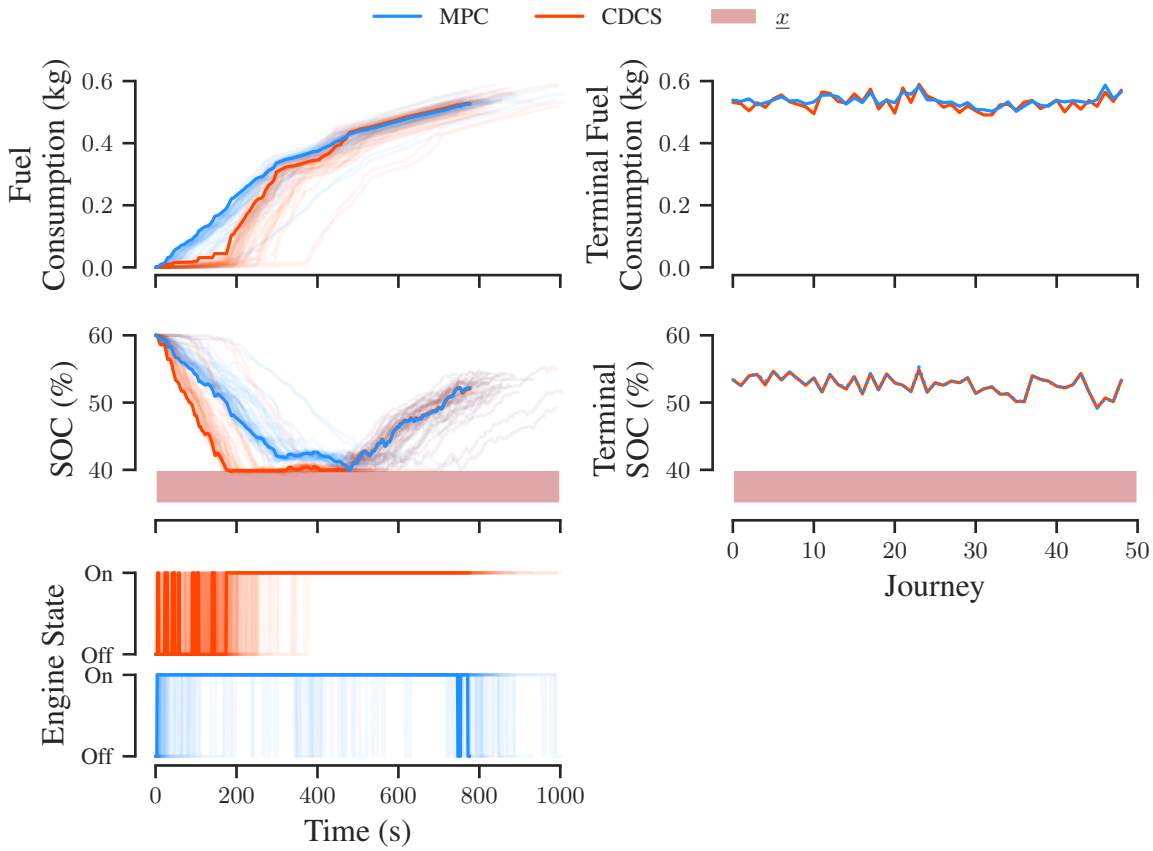


Figure 4.9: Closed loop results using MPC and CDCS.

APPENDICES

4.A ADMM SYSTEM OF LINEAR EQUATIONS

Proposition 4.A.1. *The system of linear equations*

$$(kI + \Psi^\top \Psi)\mathbf{x} = \mathbf{b}, \tag{4.22}$$

where $k \in \mathbb{R}_+$, $\mathbf{x}, \mathbf{b} \in \mathbb{R}^N$, Ψ is a $N \times N$ lower triangular matrix of ones, and I is the identity matrix, can be solved with $\mathcal{O}(N)$ computation and $\mathcal{O}(N)$ memory storage.

Proof. Firstly, note that

$$\Psi^{-1} = \begin{bmatrix} 1 & & & \\ -1 & 1 & & \\ & \ddots & \ddots & \\ & & -1 & 1 \end{bmatrix}, \quad \Psi^{-1}(\Psi^{-1})^\top = \begin{bmatrix} 1 & -1 & & \\ -1 & 2 & -1 & \\ & \ddots & \ddots & \ddots \\ & & -1 & 2 & -1 \\ & & & -1 & 2 \end{bmatrix},$$

and

$$(kI + \Psi^\top \Psi)\mathbf{x} = \mathbf{b} \quad \Longleftrightarrow \quad (k\Psi^{-1}(\Psi^{-1})^\top + I)\mathbf{x} = \Psi^{-1}(\Psi^{-1})^\top \mathbf{b}.$$

The matrix $(k\Psi^{-1}(\Psi^{-1})^\top + I)$ is diagonally dominant, so is positive definite from the Gershgorin circle Theorem [129, §6]. This implies that the Cholesky factorization

$$(k\Psi^{-1}(\Psi^{-1})^\top + I) = LL^\top$$

exists, where L is a lower diagonal matrix with nonzero entries on the diagonal and first subdiagonal only. The solution to (4.22) can therfore be obtained from

$$\mathbf{x} = L^\top \backslash L \backslash \Psi^{-1}(\Psi^{-1})^\top \mathbf{b},$$

where the backslash operator indicates a backwards/forwards substitution. Multiplications by Ψ^{-1} and $(\Psi^{-1})^\top$ are the equivalent of difference operations, and the backwards/forwards substitutions involve matrices with only nonzero entries on the diagonal and first subdiagonal/superdiagonal, so the total complexity of the operations is $\mathcal{O}(N)$. The only values that need storing in memory are the diagonal and first subdiagonal entries of L, which require $\mathcal{O}(N)$ storage. □

CHAPTER 5

ENGINE SWITCHING

IN the previous chapter it was demonstrated that if the engine switching control is determined by a poorly chosen heuristic then it is possible for the optimal power split to perform as badly as a CDCS strategy. Therefore, in this chapter, a convex optimization-based MPC approach to engine switching control is considered. A significant issue with optimizing the engine switching with convex optimization is that it introduces an integer decision variable that is necessarily nonconvex, so previous approaches (e.g. [107, 118, 106]) have considered algorithms that alternate between using PMP or dynamic programming to determine the optimal engine switching control, and convex optimization to determine the optimal power split. It has been demonstrated that this approach will converge to a point that satisfies the necessary conditions for optimality [118, §III.D], but globally optimal convergence cannot be proven in general. A further limitation with this approach is that it requires multiple solutions of the convex powersplit sub-problem, which may be computationally intractable on limited hardware (up to 155 s were required in [107]). Here, an ADMM algorithm is instead used to solve the problem directly in two phases: firstly it is used to obtain a globally optimal solution to a convex relaxation, then this solution is used to initialize the same algorithm applied to the original nonconvex problem. It is proven that the ADMM algorithm will return a power-split that is optimal with respect to the engine switching sequence at termination, and global convergence is encouraged using the initial convex phase. The performance of the algorithm is demonstrated in a set of numerical studies, where it is shown that the ADMM algorithm provides a very close approximation of the globally optimal solution of the mixed-integer MPC optimization, and that the algorithm provides a reduction in fuel consumption of 32 % relative to CDCS when used in closed-loop.

5.1 MODEL PREDICTIVE CONTROLLER

Recall from Section 3.1 that the engine control is defined by σ, where $\sigma_k = 0$ implies that the engine is off and $\sigma_k = 1$ implies that the engine is on at the kth timestep. Assuming that the transient dynamics of engine switching (and clutch engagement) are negligible, the switching variable σ_k is incorporated in the model by considering two effects: A) all of the power is delivered by the electric system when the engine is off B) the fuel consumption when the engine is off is zero. Effect A) is introduced to the model with the constraint

$$g_k(p_k) + \sigma_k\gamma_k \leq u_k \leq g_k(p_k) + \sigma_k\delta_k \quad \forall k \in \{0, \dots, N-1\}, \tag{5.1}$$

where $\gamma_k := \underline{u}_k - g_k(p_k)$ and $\delta_k := \overline{u}_k - g_k(p_k)$.

Proposition 5.1.1. *The constraint set defined by the inequalities in (5.1) is convex in* $(u_k, \sigma_k) \in \mathbb{R} \times [0, 1]$.

Proof. The proof immediately follows from the fact that $\{(u_k, \sigma_k) \in \mathbb{R} \times [0, 1] : g_k(p_k) + \sigma_k\gamma_k \leq u_k \leq g_k(p_k) + \sigma_k\delta_k\}$ is the intersection of the epigraph of an affine function and the hypograph of an affine function. □

Effect B is introduced by defining the fuel consumption as function, $\dot{m}_f : [\underline{u}_0, \overline{u}_0] \times \cdots \times [\underline{u}_{N-1}, \overline{u}_{N-1}] \times [0, 1]^N \mapsto \mathbb{R}$, of battery power and engine state, given by

$$\dot{m}_f(u, \sigma) := \sum_{k\in\mathcal{P}} \left[f_k(p_k - g_k^{-1}(u_k)) + (\sigma_k - 1)f_k(0)\right] + \sum_{k\in\mathcal{C}} \sigma_k f_k(0). \tag{5.2}$$

Note that it can be seen from (5.1) that $\sigma_k = 0 \implies f_k(p_k - g_k^{-1}(u_k)) = f_k(0)$.

Proposition 5.1.2. *Function (5.2) is convex.*

Proof. The proof immediately follows from the fact that $f_k(p_k - g_k^{-1}(u_k))$ is convex on the domain $u_k \in [\underline{u}_k, \overline{u}_k]$ for all $k \in \mathcal{P}$, and both $(\sigma_k - 1)f_k(0)$ and $\sigma_k f_k(0)$ are affine in σ_k. □

A quadratic 'driveability' cost, $d : [0, 1]^N \mapsto \mathbb{R}$, defined by

$$d(\sigma) := \frac{k_d}{2}\sigma^\top(\Psi^{-1})^\top\Psi^{-1}\sigma,$$

is included in the cost function of the MPC problem, where $k_d \in \mathbb{R}_{++}$ can be chosen either to accurately model the fuel consumed when switching the engine on, or chosen arbitrarily

to prevent the engine switching control from chattering. The MPC problem for optimizing the engine switching simultaneously with the power-split is therefore

$$\begin{aligned}
&\min_{(u,\sigma)} \sum_{i=0}^{N-1} \dot{m}_f(u,\sigma) + d(\sigma), \\
&\text{s.t. } x = \mathbf{1}x(t) - \Psi u, \\
&\quad \mathbf{1}\underline{x} \leq x \leq \mathbf{1}\overline{x}, \\
&\quad \underline{u} \leq u \leq \overline{u} \\
&\quad g_k(p_k) + \sigma_k\gamma_k \leq u_k \leq g_k(p_k) + \sigma\delta_k \quad \forall k \in \{0,\dots,N-1\}, \\
&\quad \sigma_k \in \mathcal{S} \quad \forall k \in \{0,\dots,N-1\}.
\end{aligned} \tag{MPC.3}$$

From the properties of the cost function and constraints demonstrated above, this problem is nonconvex when the engine switching variable σ_k is an integer decision variable, i.e. $\mathcal{S} = \{0,1\}$. However, if the constraint set for σ_k is relaxed to the interval $\mathcal{S} = [0,1]$, then problem (MPC.3) becomes convex.

Proposition 5.1.3. *If the constraint set of (MPC.3) is nonempty, then the solution exists. Additionally, if $\mathcal{S} = [0,1]$, it is unique.*

Proof. The objective of (MPC.3) is continuous, and the constraint set is bounded and closed (for both $\mathcal{S} = [0,1]$ and $\mathcal{S} = \{0,1\}$) and is therefore also compact. The first result therefore follows from Weierstrass' Theorem for continuous functions [91, p.237]. For the second result, observe that $\dot{m}_f(u,\sigma)$ is separable w.r.t. u and σ, strictly convex in u (this follows from Theorem 3.2.1 for all $k \in \mathcal{P}$, and is trivially true for all $k \in \mathcal{C}$ as $\underline{u}_k = \overline{u}_k$), and affine in σ_k. Additionally, Ψ^{-1} is nonsingluar and $k_d > 0$, which implies that $\frac{k_d}{2}(\Psi^{-1})^\top\Psi^{-1} \succ 0$. Therefore, the objective of (MPC.3) is strictly convex, so the second result follows from Proposition 2.2.1. □

CONVEX MIXED INTEGER PROGRAMMING

Note that (MPC.3) is of the form of a convex mixed-integer nonlinear program (when $\mathcal{S} = \{0,1\}$), a general class of problems for which several optimization algorithms have been extensively investigated: for a review of algorithms and software see [130] and/or [131]. A major issue with solving mixed-integer problems is that, even when the objective and constraints are otherwise convex, they are $\mathcal{NP}$-hard in general, and exact solution algorithms are therefore often too computationally intensive for an online solution. For example, in the worst case, a branch-and-bound algorithm would require the solution of 2^N convex relaxations to generate a solution for (MPC.3). This issue motivates the use of

ADMM as a heuristic for solving (MPC.3) as presented in this chapter, and was previously investigated for PHEV energy management in this way in [132]. That approach, however, considered a very simple powertrain model and limited hardware constraints (the state-of-charge constraint, in particular, was ignored) compared to (MPC.3), and neither the convergence of the algorithm nor the optimality of the returned values were addressed.

5.1.1 FEASIBILITY

This section considers the feasibility of (MPC.3) under the following assumption.

Assumption 5.1.1. *$g_k(p_k) \geq \underline{u}_k$ for all $k \in \{0, \dots, N-1\}$.*

The physical interpretation of the case in which $g_k(p_k) < \underline{u}_k$ is that the vehicle is braking and the braking power cannot be delivered entirely by the motor. Assumption 5.1.1 implicitly requires that if this case occurs for any $k \in \{0, \dots, N-1\}$, the magnitude of the mechanical braking power ($b_k \in \mathbb{R}_-$) is increased until $g_k(p_k) = g_k(d_k - b_k) \geq \underline{u}_k$ (where d_k is the demand power) for all k. Note that it is not required that $g_k(p_k) \leq \overline{u}_k$: the physical interpretation of this case is that both the motor and engine are required to meet the driver's power demand, and the intersection of (5.1) and constraint $u_k \leq \overline{u}_k$ ensures that $\sigma_k = 1$ (i.e. the engine is on).

Proposition 5.1.4. *For both $\mathcal{S} = \{0, 1\}$ and $\mathcal{S} = [0, 1]$, the constraint set in (MPC.3) is nonempty if and only if $\mathcal{F}_k \neq \emptyset \; \forall k \in \{1, \dots, N\}$, where $\mathcal{F}_0 := \{x(t)\}$, $\mathcal{F}_{k+1} := \{\mathcal{F}_k \oplus -\mathcal{U}_k\} \cap \mathcal{X} \; \forall k \in \{0, \dots, N-1\}$, $\mathcal{U}_k := [\underline{u}_k, \overline{u}_k]$, and $\mathcal{X} := [\underline{x}, \overline{x}]$.*

Proof. The projection of the constraint set in (MPC.3) onto (u, x)-space is equivalent to the constraint set for (MPC.2), so the proof immediately follows from the proof of proposition 4.1.2. □

Proposition 5.1.4 implies that Algorithm 1 can be also be used to generate a feasibility certificate for (MPC.3). This is because the set of all trajectories for u_k and x_k for all possible engine switching sequences lies within the set of trajectories that can be obtained with $\sigma_k = 1$ for all k, so only the case where $\sigma = \mathbf{1}$ needs to be checked to determine feasibility.

5.2 OPTIMIZATION

5.2.1 ALGORITHM

An extension of the ADMM algorithm presented in Chapter 4 is proposed for the solution of (MPC.3) in two phases: firstly, it is used to find the global solution to the convex

relaxation of (MPC.3) with $\mathcal{S} = [0, 1]$, then this solution is used as the initialization point for the same algorithm with $\mathcal{S} = \{0, 1\}$, which is shown to be locally convergent. Three dummy variables, $\zeta \in \mathbb{R}^N$, $\eta \in \mathbb{R}^N$, and $\kappa \in \mathbb{R}^N$, are included, so that (MPC.3) is equivalent to

$$\begin{aligned} \min_{(\kappa,u,x,\sigma,\eta,\zeta)} \; & \dot{m}_f(u,\sigma) + d(\kappa) + \sum_{k=0}^{N-1} \left[\mathcal{I}_{\mathcal{U}_k}(u_k) + \mathcal{I}_{\mathcal{X}}(x_{k+1}) + \mathcal{I}_{\mathcal{S}}(\sigma_k) + \mathcal{I}_{\mathcal{R}_k}(\eta_k, \sigma_k)\right], \\ \text{s.t. } & x = \mathbf{1}x(t) - \Psi u, \\ & \zeta = u = \eta \\ & \kappa = \sigma, \end{aligned} \tag{5.3}$$

where

$$\mathcal{R}_k := \left\{(u_k, \sigma_k) \in \mathbb{R}^2 : g_k(p_k) + \sigma_k\gamma_k \leq u_k \leq g_k(p_k) + \sigma_k\delta_k\right\}.$$

Problem (5.3) is in turn equivalent to

$$\begin{aligned} \min_{(\tilde{u},\tilde{x})} \; & \tilde{f}(\tilde{u}) + \tilde{g}(\tilde{x}) \\ \text{s.t. } & A\tilde{u} + B\tilde{x} = c \end{aligned} \tag{5.4}$$

where

$$\begin{aligned} \tilde{u} := (\kappa, u, x), \quad \tilde{x} := (\sigma, \eta, \zeta), \\ \tilde{f}(\tilde{u}) := \sum_{k\in\mathcal{P}} f_k(p_k - g_k^{-1}(u_k)) + d(\kappa) + \sum_{k=0}^{N-1} \left[\mathcal{I}_{\mathcal{X}}(x_{k+1}) + \mathcal{I}_{\mathcal{U}_k}(u_k)\right] \\ \tilde{g}(\tilde{x}) := \sum_{k=0}^{N-1} \left[\sigma_k f_k(0) + \mathcal{I}_{\mathcal{S}}(\sigma_k) + \mathcal{I}_{\mathcal{R}_k}(\eta_k, \sigma_k)\right], \\ A := \begin{bmatrix} 0 & 0 & -I \\ 0 & I & 0 \\ 0 & I & 0 \\ I & 0 & 0 \end{bmatrix}, \quad B := \begin{bmatrix} 0 & 0 & -\Psi \\ 0 & 0 & -I \\ 0 & -I & 0 \\ -I & 0 & 0 \end{bmatrix}, \quad c := (-\mathbf{1}x(t), \mathbf{0}, \mathbf{0}, \mathbf{0}). \end{aligned} \tag{5.5}$$

The augmented Lagrangian associated with (5.4) is

$$\mathcal{L}_\rho(\tilde{u}, \tilde{x}, \lambda) := \tilde{f}(\tilde{u}) + \tilde{g}(\tilde{x}) + \frac{1}{2}\|A\tilde{u} + B\tilde{x} - c + \lambda\|_\rho^2 \tag{5.6}$$

where

$$\begin{aligned} \|x\|_\rho^2 := x^\top \rho x, \quad \rho := \text{diag}(\rho_1\mathbf{1}, \rho_2\mathbf{1}, \rho_3\mathbf{1}, \rho_4\mathbf{1}), \quad \rho_1, \rho_2, \rho_3, \rho_4 \in \mathbb{R}_{++}, \\ \lambda := (\lambda_1, \lambda_2, \lambda_3, \lambda_4), \quad \lambda_1, \lambda_2, \lambda_3, \lambda_4 \in \mathbb{R}^N. \end{aligned}$$

The ADMM iteration is then defined by

$$\tilde{u}^{(i+1)} := \underset{\tilde{u}}{\operatorname{argmin}}\ \mathcal{L}(\tilde{u}, \tilde{x}^{(i)}, \lambda^{(i)}), \tag{5.7a}$$

$$\tilde{x}^{(i+1)} \in \underset{\tilde{x}}{\operatorname{argmin}}\ \mathcal{L}(\tilde{u}^{(i+1)}, \tilde{x}, \lambda^{(i)}), \tag{5.7b}$$

$$\lambda^{(i+1)} := \lambda^{(i)} + A\tilde{u}^{(i+1)} + B\tilde{x}^{(i+1)} - c. \tag{5.7c}$$

The notation $\in$ is used in (5.7b) because the argmin may not be unique; a tie-breaker criterion is described for the algorithm in the following section. The algorithm is initialized with the values $(\tilde{u}^{(0)}, \tilde{x}^{(0)}, \lambda^{(0)}) = \mathbf{0}$ and the set $\mathcal{S} = [0, 1]$, and iteration (5.7) is repeated until the criterion

$$\max\left\{\|r^{(i+1)}\|, \|s^{(i+1)}\|\right\} \leq \epsilon \tag{5.8}$$

is met, where $\epsilon \in \mathbb{R}_{++}$ is a fixed convergence threshold and the residuals $r^{(i+1)}$ and $s^{(i+1)}$ are defined by

$$r^{(i+1)} := A\tilde{u}^{(i+1)} + B\tilde{x}^{(i+1)} - c \quad \text{and} \quad s^{(i+1)} := A^{\top}\rho B(\tilde{x}^{(i+1)} - \tilde{x}^{(i)}). \tag{5.9}$$

The set $\mathcal{S}$ is then updated with $\{0, 1\}$, and the iteration (5.7) is continued until the criterion (5.8) is met again. The ADMM algorithm as a whole is presented in pseudocode in Algorithm (5) (the generation of a feasibility certificate using Algorithm 1 is omitted for the sake of conciseness).

5.2.2 Convergence & Optimality

Problem (5.4) is the form of optimization problem addressed in the appendix to this chapter, in which it is demonstrated that when $\mathcal{S} = \{0, 1\}$, every point in (u, x, y)-space where the battery power allocation is globally optimal w.r.t the engine switching sequence is a stationary point of iteration (5.7), and these points each have a convergent neighbourhood. Additionally, it is demonstrated that these are the *only* stationary points of iteration (5.7). Therefore, the solution returned when Algorithm 5 terminates is the equivalent of the solution of (MPC.2) given the 'optimized' engine switching sequence. A limitation of this approach is that it cannot be guaranteed that the algorithm will converge to a 'good' engine switching sequence: for example, the engine switching sequence obtained from the heuristic in the previous Chapter is an attractor for (5.7). A further limitation is that no lower bound is provided on the size of the basin of attraction at each point (so convergence is not guaranteed from an arbitrary point). However, a locally convergent region exists for *every possible* engine switching sequence, so (5.7) has 2^N attractors. This may explain why the algorithm reliably converges in the numerical simulations that follow. The principle of Algorithm 5 is to encourage both convergence (in the sense that the iterates do

Algorithm 5 ADMM Algorithm for Problem (MPC.3)

Input: $(x(t), \underline{u}, \overline{u}, \gamma, \delta, F, \epsilon)$
Output: $(u^\dagger, \sigma^\dagger)$

$(\kappa^{(0)}, u^{(0)}, x^{(0)}, \sigma^{(0)}, \eta^{(0)}, \zeta^{(0)}, \lambda^{(0)}) \leftarrow \mathbf{0}$
$(r^{(0)}, s^{(0)}) \leftarrow \infty$
$\mathcal{S} \leftarrow [0, 1]$
$i \leftarrow 0$
while $\max\{\|r^{(i+1)}\|, \|s^{(i+1)}\|\} > \epsilon$ **do**
  $\tilde{u}^{(i+1)} \leftarrow$ (5.7a) $\quad \forall k \in \{0, \ldots, N-1\}$
  $\tilde{x}^{(i+1)} \leftarrow$ (5.7b)
  $(r^{(i+1)}, s^{(i+1)}) \leftarrow$ (5.9)
  $\lambda^{(i+1)} \leftarrow \lambda^{(i)} + r^{(i+1)}$
  $i \leftarrow i + 1$
end while
$\mathcal{S} \leftarrow \{0, 1\}$
while $\max\{\|r^{(i+1)}\|, \|s^{(i+1)}\|\} > \epsilon$ **do**
  $\tilde{u}^{(i+1)} \leftarrow$ (5.7a) $\quad \forall k \in \{0, \ldots, N-1\}$
  $\tilde{x}^{(i+1)} \leftarrow$ (5.7b)
  $(r^{(i+1)}, s^{(i+1)}) \leftarrow$ (5.9)
  $\lambda^{(i+1)} \leftarrow \lambda^{(i)} + r^{(i+1)}$
  $i \leftarrow i + 1$
end while
return $(u^\dagger, \sigma^\dagger) \leftarrow (u^{(i)}, \sigma^{(i)})$

converge to a fixed point and will satisfy the criterion (5.8) in a finite number of iterations) and optimality (in the sense that the iterates also converge to the globally optimal engine switching sequence) by initializing the algorithm with the globally optimal solution of the convex relaxation $\mathcal{S} = [0, 1]$ (for which the ADMM algorithm is globally convergent). Convergence can be guaranteed more generally by fixing $\sigma^{(i+1)}$ after a finite number of iterations, but this was not found to be necessary in the following simulations.

5.2.3 VARIABLE UPDATES & COMPLEXITY

Firstly, note that the augmented Lagrangian (5.6) is equivalent to

$$\begin{aligned}\mathcal{L}_\rho(\tilde{u}, \tilde{x}, \lambda) = &\sum_{k\in\mathcal{P}} f_k(p_k - g_k^{-1}(u_k)) + d(\kappa) + \sum_{k=0}^{N-1} [\mathcal{I}_{\mathcal{X}}(x_{k+1}) + \mathcal{I}_{\mathcal{U}_k}(u_k)] \\ &+ \sum_{k=0}^{N-1} [\sigma_k f_k(0) + \mathcal{I}_{\mathcal{S}}(\sigma_k) + \mathcal{I}_{\mathcal{R}_k}(\eta_k, \sigma_k)] + \frac{\rho_1}{2}\|\mathbf{1}x(t) - x - \Psi\zeta + \lambda_1\|^2 \\ &+ \frac{\rho_2}{2}\|u - \zeta + \lambda_2\|^2 + \frac{\rho_3}{2}\|u - \eta + \lambda_3\|^2 + \frac{\rho_4}{2}\|\kappa - \sigma + \lambda_4\|^2 \quad (5.10)\end{aligned}$$

The update (5.7a) considers the variables κ, u, and x, which are completely decoupled in (5.10). Therefore, the solution of (5.7a) can be obtained by minimizing (5.10) w.r.t each of κ, u, and x separately, and is therefore equivalent to

$$\kappa^{(i+1)} = \left(k_d(\Psi^{-1})^\top\Psi^{-1} + \rho_4 I\right)^{-1}\rho_4(\sigma^{(i)} - \lambda_4^{(i)}) \quad (5.11a)$$

$$\begin{aligned}u_k^{(i+1)} = \underset{u_k}{\operatorname{argmin}} \Big(&F_k(u_k) + \mathcal{I}_{\mathcal{U}_k}(u_k) + \frac{\rho_2}{2}(u_k - \zeta_k^{(i)} + \lambda_{2,k}^{(i)})^2 \\ &+ \frac{\rho_3}{2}(u_k - \eta_k^{(i)} + \lambda_{3,k})^2\Big) \quad \forall k \in \mathcal{P} \quad (5.11b)\end{aligned}$$

$$u_k^{(i+1)} = \frac{\rho_2(\zeta_k^{(i)} - \lambda_{2,k}^{(i)}) + \rho_3(\eta_k^{(i)} - \lambda_{3,k}^{(i)})}{\rho_2 + \rho_3} \quad \forall k \in \mathcal{C} \quad (5.11c)$$

$$x^{(i+1)} = \Pi_{\mathcal{X}^N}\left[\mathbf{1}x(t) - \Psi\zeta^{(i)} + \lambda_1^{(i)}\right] \quad (5.11d)$$

The κ update is the solution of the general linear system of equations $M\mathbf{x} = \mathbf{b}$ where the matrix M is tridiagonal, diagonally dominant, and does not change between iterations. The proof of Proposition 4.A.1 can be used to demonstrate that the Cholesky factorization of M exists, has $\mathcal{O}(N)$ memory requirement, and that the solution of (5.11a) has $\mathcal{O}(N)$ computational requirement at each iteration. The u_k update is equivalent to the u_k update

described for the ADMM algorithm in Section 4.2.1 for all $k \in \mathcal{P}$[1], and has an analytical solution for all $k \in \mathcal{C}$. The x update is exactly equal to the x update described in Section 4.2.1.

The update (5.7b) considers the variables σ, η, and ζ, of which σ and η are coupled in (5.10). Therefore, the solution of (5.7b) is obtained by minimizing (5.10) w.r.t σ and η together and ζ separately, and is therefore equivalent to

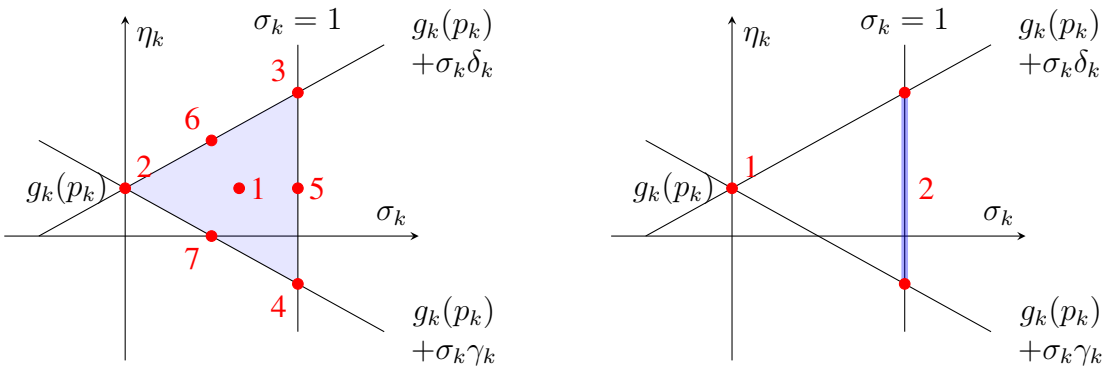
$$(\eta_k, \sigma_k)^{(i+1)} \in \underset{(\sigma,\eta)}{\operatorname{argmin}} \Big(\sigma_k f_k(0) + \mathcal{I}_{\mathcal{S}}(\sigma_k) + \mathcal{I}_{\mathcal{R}_k}(\eta_k, \sigma_k) + \frac{\rho_3}{2} \|u^{(i+1)} - \eta + \lambda_3^{(i)}\|^2 + \frac{\rho_4}{2} \|\kappa^{(i+1)} - \sigma + \lambda_4^{(i)}\|^2 \Big) \quad \forall k \in \mathcal{P} \tag{5.12a}$$

$$(\eta_k, \sigma_k)^{(i+1)} = (g_k(p_k), 0) \quad \forall k \in \mathcal{C} \tag{5.12b}$$

$$\zeta^{(i+1)} = \Big(\rho_2 I + \rho_1 \Psi^\top \Psi \Big)^{-1} \Big(\rho_2 (u^{(i+1)} + \lambda_2^{(i)}) + \rho_1 \Psi^\top (\mathbf{1} x(t) - x + \lambda_1) \Big) \tag{5.12c}$$

The joint (η_k, σ_k) update (5.12a) is the solution of a two-dimensional optimization problem with a quadratic objective function. When $\mathcal{S} = [0, 1]$ it is a convex inequality constrained quadratic program for which there are 7 possible solutions as shown in Figure 5.1: the unconstrained minimum 1, three at the vertices 2-4, and three on the edges 5-7. Each of these has an analytical solution, so the solution of (5.12a) is obtained by evaluating each of the seven candidate solutions against the objective function to determine which is the minimizing argument (when $\mathcal{S} = [0, 1]$ this point is unique). When $\mathcal{S} = \{0, 1\}$, the constraints become nonconvex, and there are two candidate solutions as shown in Figure 5.1: either $(\eta_k, \sigma_k)^{(i+1)} = (g_k(p_k), 0)$ or $(\eta_k, \sigma_k)^{(i+1)} = (\Pi_{\mathcal{U}_k}(u_k^{(i+1)} + \lambda_{3,k}), 1)$. In the case that both of these points have the same value when evaluated against the objective function of (5.12a), $(\eta_k, \sigma_k)^{(i+1)} = (g_k(p_k), 0)$ is always chosen as the solution. The joint (σ_k, η_k) update is separable w.r.t the elements of σ and η and so its computational requirement depends linearly on N if each k update is performed sequentially, or is independent of N if performed in parallel. The $\zeta^{(i+1)}$ update is equivalent to the update in the previous ADMM algorithm (4.10), so its computation and memory requirement scales linearly with prediction horizon. The residual updates (5.9) are equivalent to

[1] One of the motivations for including both sides of the constraint $\underline{u} \leq u \leq \overline{u}$ in (MPC.3) is to ensure that (5.11b) is convex, as F_k is generally nonconvex outside of the domain represented by $\mathcal{I}_{\mathcal{U}_k}(u_k)$.



(a) $\mathcal{S} = [0, 1]$ (b) $\mathcal{S} = \{0, 1\}$

Figure 5.1: Constraint sets defined by $\mathcal{I}_{\mathcal{S}}(\sigma_k) + \mathcal{I}_{\mathcal{R}_k}(\eta_k, \sigma_k)$.

Table 5.1: Summary of the computational complexity of each of the ADMM variable updates.

$\mathcal{O}(\cdot)$	κ	u	x	(η, σ)	ζ	r	s	λ
Parallel	N	1	N	1	N	N	N	1
Sequential	N	N	N	N	N	N	N	N

$$r^{(i+1)} := \begin{bmatrix} \mathbf{1}x(t) - x^{(i+1)} - \zeta^{(i+1)} \\ u^{(i+1)} - \zeta^{(i+1)} \\ u^{(i+1)} - \eta^{(i+1)} \\ \kappa^{(i+1)} - \sigma^{(i+1)} \end{bmatrix}$$

$$s^{(i+1)} := \begin{bmatrix} \rho_4(\sigma^{(i+1)} - \sigma^{(i)}) \\ -\rho_3(\eta^{(i+1)} - \eta^{(i)}) - \rho_2(\zeta^{(i+1)} - \zeta^{(i)}) \\ \rho_4\Psi(\zeta^{(i+1)} - \zeta^{(i)}) \end{bmatrix}$$

for which the computational and memory requirements scale linearly with N, and update (5.7c) is equivalent to $\lambda^{(i+1)} = \lambda^{(i)} + r^{(i+1)}$. The computational complexity of the elements of each instance of iteration (5.7) is summarized in Table 5.1, and the memory requirement of the iteration is $\mathcal{O}(N)$.

5.3 NUMERICAL EXPERIMENTS

The experiments were separated into two phases: firstly, only the first instance of the MPC optimization (MPC.3) was considered for each journey, where the output of the ADMM algorithm was compared against the globally optimal solution obtained using dynamic programming to determine the suboptimality caused by local convergence. Secondly, the MPC controller defined by the approximate solution of (MPC.3) obtained using the ADMM algorithm was simulated in closed loop in comparison with a CDCS baseline.

5.3.1 OPTIMIZATION PERFORMANCE

Figure 5.2 shows the solution of (MPC.3) obtained for each journey at time $t = 0$ using both the ADMM algorithm and dynamic programming. It is demonstrated in the appendix to this chapter that the ADMM algorithm will terminate with the optimal power split given the engine switching sequence, so suboptimality is only introduced through adverse engine switching decisions. It can be seen in Figure 5.2 the the engine switching control obtained using ADMM closely matches the solutions obtained using dynamic programming, although ADMM does introduce more switching decisions for the highlighted example (journey 14), particularly around $k = 400$. This is reflected in the bottom right axis, where it is shown that the ADMM algorithm introduces 43% more engine switches than dynamic programming across all journeys. Broadly speaking, however, the ADMM algorithm correctly identifies the periods in which it is optimal for the engine to be switched off, and consequently the fuel consumption and state-of-charge trajectories closely match those obtained with dynamic programming. The terminal state-of-charge is closely matched between the ADMM and dynamic programming solutions (as observed in the previous section), so the performance can be determined by directly comparing the total fuel consumption: the ADMM algorithm increases the predicted fuel consumption relative to DP by 5.3 % on average.

Figure 5.3 shows histograms of the time taken to completion using dynamic programming and ADMM for all journeys. The mean time taken was 1600 s for dynamic programming and 0.23 s for ADMM; a reduction of more than three orders of magnitude. The ADMM algorithm was implemented in vectorised Matlab code here, and it is therefore likely that the absolute performance could be increased further with a compiled software implementation, where a subset of the decision variables are possibly updated in parallel. Overall, the results demonstrate that the ADMM algorithm is suitable for a real-time implementation.

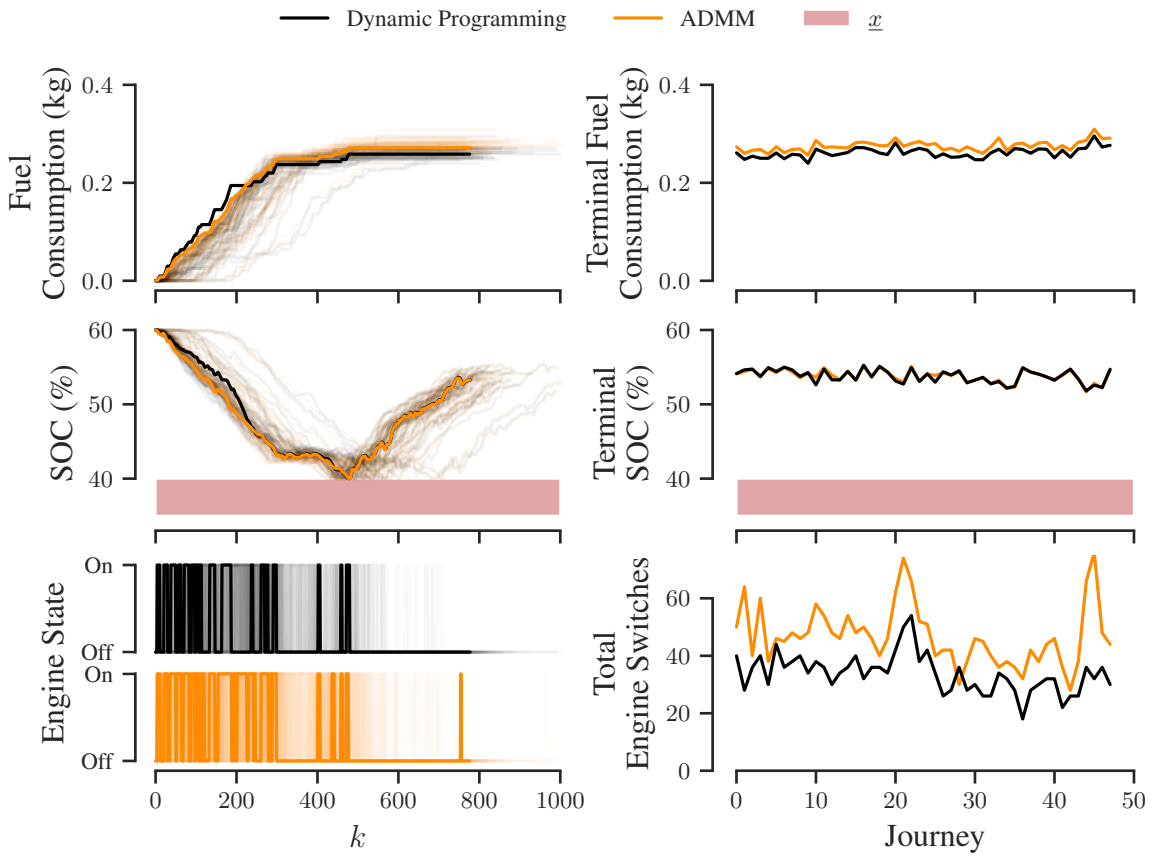


Figure 5.2: Solution of (MPC.3) at time $t = 0$ obtained using ADMM and dynamic programming.

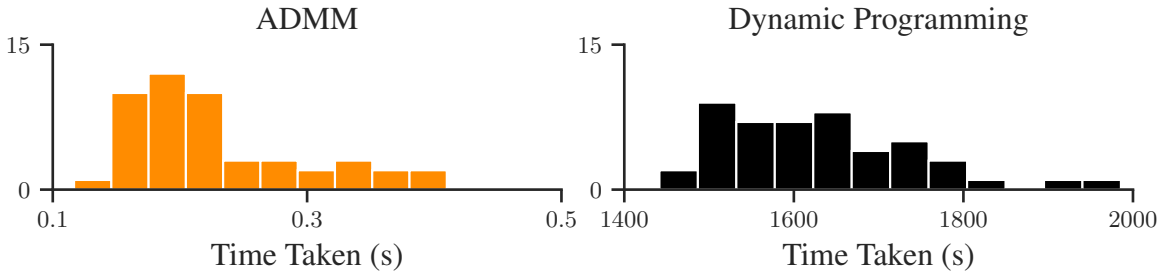


Figure 5.3: Time taken for each of the solutions in Figure 5.2 using ADMM and dynamic programming.

5.3.2 CLOSED-LOOP CONTROL

Figure 5.4 shows simulations results using MPC where the control variable update is defined by the approximate solution of (MPC.3) obtained using the ADMM algorithm, in comparison with the CDCS results previously presented in Figure 5.2. It can now be seen that MPC provides a clear and consistent reduction in fuel consumption relative to CDCS across all journeys, with an average reduction of 32 % (recall that the MPC results in section 4.3.2 performed *worse* than CDCS due to a poor engine switching heuristic). It can also be seen that the closed-loop engine switching sequence for the highlighted example does not match the predictions of the first optimization highlighted in Figure 5.2, and particularly that additional switching decisions are introduced between 400 and 700 seconds. This is reflected in the total number of engine switches across all journeys, where the average was 47 for the first MPC optimizations shown in Figure 5.2, whereas an average of 73 is found in Figure 5.4. The increased engine switching is in turn reflected in fuel consumption, which was predicted to average 0.28 kg in Figure 5.2, but was realized as an average of 0.36 kg in Figure 5.4. This may be partly due to the modelling errors, but will also be caused by the engine being switched on for a larger fraction of the journey. Figure 5.5 shows the battery power trajectories for the highlighted solutions in Figures 5.2 and 5.4, which provide a clearer illustration of the differences each method. Of particular significance is the period between approximately 200 and 500 seconds, where the CDCS controller is required to deliver all of the power from the engine, resulting in a significant increase in fuel consumption.

5.4 CONCLUDING REMARKS

In this section the MPC energy management controller was extended from the form presented in Chapter 4 to consider engine switching. Problem (MPC.2) was extended to include the binary switching variable, which resulted in a convex mixed-integer nonlinear program. An ADMM algorithm was proposed for the solution of the new MPC optimization problem, which is neccessarily convex due to the integer decision variable. Local convergence of the ADMM algorithm was proven, so the algorithm was initialized using the solution of a convex relaxation of the problem to encourage convergence to the globally minimizing argument. The ADMM algorithm was compared against dynamic programming on a set of single-shot optimization problems, where it was shown to obtain a close approximation of the globally minimizing argument, and reduced the computational burden from approximately three hours to a fraction of a second. The MPC framework and ADMM algorithm were then simulated in closed loop against a CDCS strategy, where they were shown to reduce fuel consumption by 32 %.

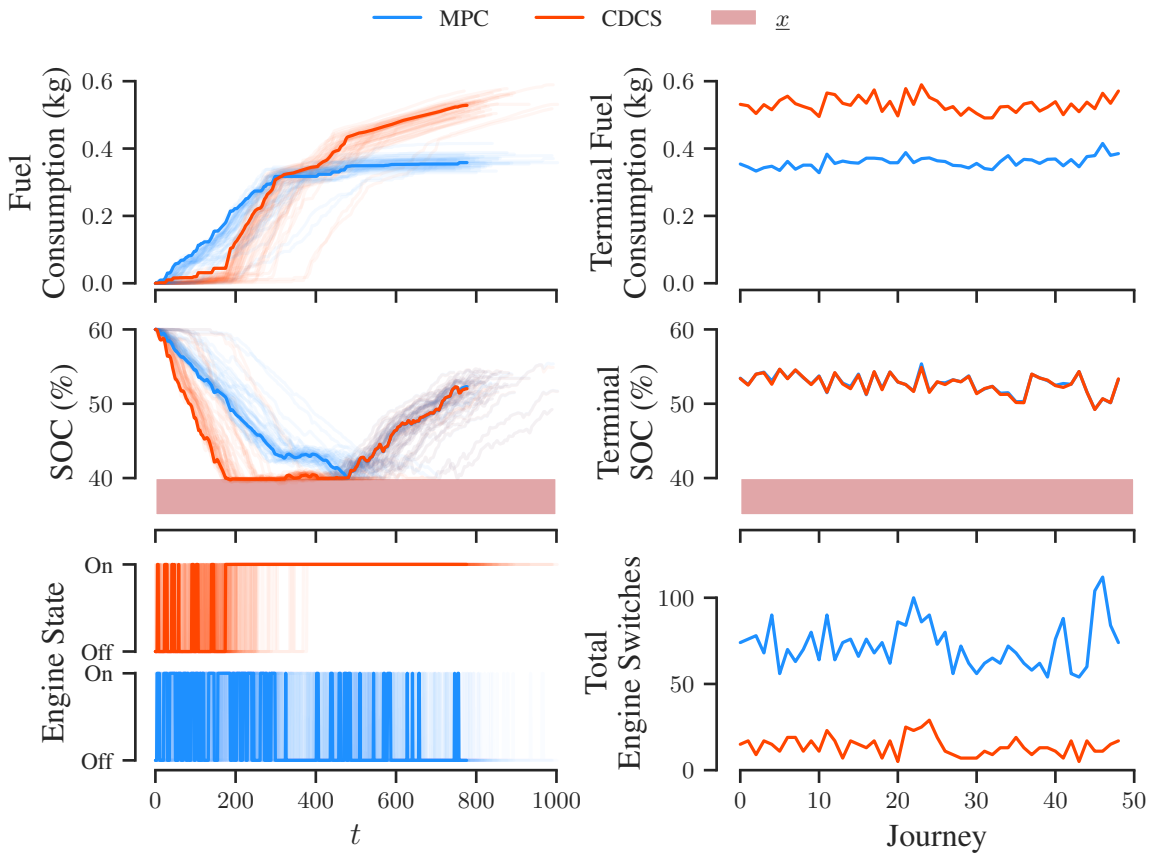


Figure 5.4: Closed-loop simulation using MPC with an approximate solution obtained using ADMM, compared against CDCS.

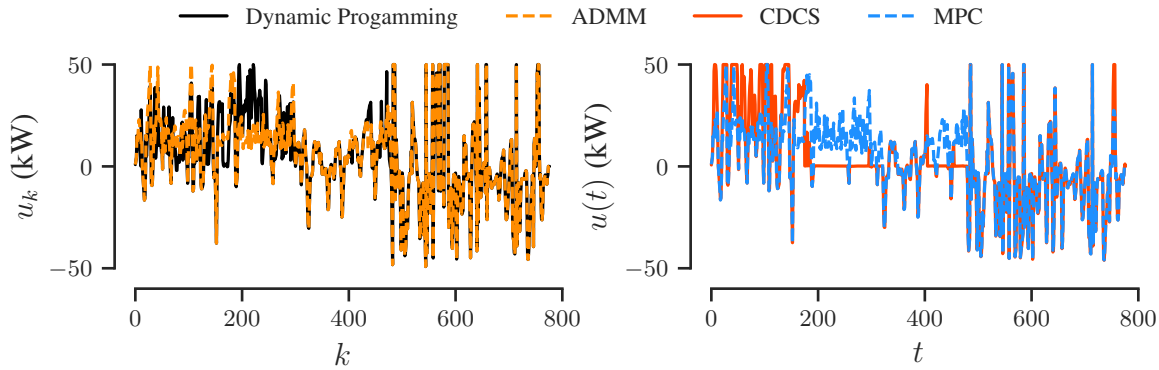


Figure 5.5: Battery power (u) allocation for the highlighted examples in Figures 5.2 and 5.4 (note that the results in the left-hand plot correspond to single shot optimizations, where the results in the right-had plot are the results from closed loop simulations).

APPENDICES

5.A OPTIMALITY AND CONVERGENCE OF ALGORITHM 5

This appendix addresses the convergence properties of Algorithm 5, and characterizes the optimality of the stationary point(s) that it may converge to. The appendix is structured as follows: in Section 5.A.1 the problem is formalized mathematically, and the main result is stated in Proposition 5.A.1 with a sketch of the proof. The proof is then provided in detail in the following three sections.

5.A.1 PROBLEM STATEMENT

Recall that problem (5.4) is of the form

$$\begin{aligned} &\min_{(u,x)} \ f(u) + g(x) \\ &\text{s.t. } Au + Bx = c. \end{aligned} \tag{5.13}$$

For the sake of readability in the analysis that follows, the 'tilde' accents used in (5.4) have not been included in (5.13), and all notation used within this appendix can be considered separately from the rest of this thesis. The full definitions of each of the terms in (5.13) is provided in (5.5), but the important properties relevant to the following proof are also recalled here:

- The function f is the sum of continuous convex functions and indicator functions for closed and convex polyhedral constraint sets.
- The function g is the sum of a linear function of the first N elements of x, an indicator function for a closed and convex polyhedral constraint set, and an indicator function for the set $\{0,1\}^N \times \mathbb{R}^{2N}$ (i.e. the first N elements of x are binary decision variables).
- The matrices A and B have full column rank.

Importantly, problem (5.13) is nonconvex due to the binary decision component of x *only*. To construct the following proof, decompositions of x into the vectors $x_1 \in \mathbb{R}^N$ and $x_2 \in \mathbb{R}^{2N}$ and B into the matrices $B_1 \in \mathbb{R}^{4N \times N}$ and $B_2 \in \mathbb{R}^{4N \times 2N}$ are defined as $(x_1, x_2) := x$ and $[B_1, B_2] := B$. In other words, x_1 is the binary decision component of x, and B_1 and B_2 are the blocks of B that correspond to x_1 and x_2 under matrix-vector multiplication.

The augmented Lagrangian function corresponding to (5.13) is defined by

$$\mathcal{L}_\rho(u,x,y) := f(u) + g(x) + y^\top(Au + Bx - c) + \frac{1}{2}\|Au + Bx - c\|_\rho^2 \tag{5.14}$$

where $y \in \mathbb{R}^{4N}$ is a vector of Lagrange multipliers, and $\rho := \text{diag}(\rho_1, \ldots, \rho_{4N})$ such that $\rho_i \in \mathbb{R}_{++}\ \forall i \in \{1, \ldots, 4N\}$. Note that (5.14) is the *scaled form* of (5.6), and is equivalent using the change of variables $\rho\lambda^{(i)} = y^{(i)}$ (see [127, §3.1.1] for more details). The ADMM iteration for problem (5.13) is

$$u^{(i+1)} = \underset{u}{\text{argmin}} \left\{ f(u) + (y^{(i)})^\top(Au + Bx^{(i)} - c) + \frac{1}{2}\|Au + Bx^{(i)} - c\|_\rho^2 \right\} \tag{5.15a}$$

$$x^{(i+1)} = \underset{x}{\text{argmin}} \left\{ g(x) + (y^{(i)})^\top(Au^{(i+1)} + Bx - c) + \frac{1}{2}\|Au^{(i+1)} + Bx - c\|_\rho^2 \right\} \tag{5.15b}$$

$$y^{(i+1)} := y^{(i)} + \rho(Au^{(i+1)} + Bx^{(i+1)} - c) \tag{5.15c}$$

which is equivalent to the iteration (5.7) for problem (5.4) with $y^{(i)} = \rho\lambda^{(i)}$. It can readily be shown that the objective functions of (5.15a) and (5.15b) are closed and coercive, and under the assumption that (5.13) is feasible, they are proper, so at least one minimizing argument always exists for the objectives in (5.15a) and (5.15a) (Weierstrass's Theorem [91, p. 119]).

Proposition 5.A.1. *Assume that there exists a stationary point of the ADMM iteration (5.15), denoted $(u', x', y') \in \mathbb{R}^{(m+n+l)}$, such that*

$$(u^{(j)}, x^{(j)}, y^{(j)}) = (u', x', y') \implies (u^{(j+1)}, x^{(j+1)}, y^{(j+1)}) = (u', x', y').$$

Also assume that $\text{argmin}_x\, \mathcal{L}_\rho(u', x, y')$ is unique. It then follows that

a. *There exists a neighbourhood of $(u^{(j)}, x^{(j)}, y^{(j)})$-space close to (u', x', y') where the values of $(u^{(j+1)}, x^{(j+1)}, y^{(j+1)})$ defined by iteration (5.15) converge to (u', x', y').*

b. *Any given (u', x', y') minimizes (5.13) for fixed binary decision vector x_1', and any point (u, x, y) that would minimize (5.13) if the binary decision vector x_1 were fixed is a stationary point of iteration (5.15).*

Proof. A sketch of the proof is presented here; the full proof is provided in following sections.

1. In Section 5.A.2 it is demonstrated that there exists a neighbourhood of $(u^{(i)}, x^{(i)}, y^{(i)})$-space close to (u', x', y'), denoted $\mathcal{B}_\tau$, in which the ADMM algorithm (5.15) is equivalent to another ADMM algorithm applied to a convex problem; namely problem (MPC.2).

2. In Section 5.A.3 it is demonstrated that the equivalent ADMM algorithm necessarily converges to the minimizing argument of (MPC.2), denoted $(u^\star, x^\star, y^\star)$. Additionally, it demonstrated that if the algorithm is initialised within a bounded neighbourhood of $(u^\star, x^\star, y^\star)$, denoted $\mathcal{S}_k$, then the iterates always remain within this neighbourhood as $i \to \infty$.

3. The results from the previous two sections are combined in Section 5.A.4. It is demonstrated that points satisfying the criteria to be denoted (u', x', y') and $(u^\star, x^\star, y^\star)$ are equivalent, hence k can be chosen such that $\mathcal{S}_k \subseteq \mathcal{B}_\tau$. Consequently, it is shown that if $(u^{(j)}, x^{(j)}, y^{(j)}) \in \mathcal{S}_k$ at any iteration, the iterates remain within $\mathcal{S}_k$ (and therefore also $\mathcal{B}_\tau$) and converge to $(u^\star, x^\star, y^\star)$ as $i \to \infty$.

□

5.A.2 EQUIVALENT CONVEX PROBLEM

Lemma 5.A.1. *There exists a neighbourhood of* $(u^{(j+1)}, y^{(j)})$*-space close to* (u', y') *where* $x^{(j+1)} = x'$.

Proof. It is assumed in Proposition 5.A.1 that $\operatorname{argmin}_x \mathcal{L}_\rho(u', x, y')$ is unique, so it follows from the definition of the stationary point that

$$\min_{x_2} \mathcal{L}_\rho(u', (x_1', x_2), y') < \min_{x_2} \mathcal{L}_\rho(u', (x_1^\dagger, x_2), y') \quad \forall x_1^\dagger \in \{0,1\}^N, \quad x_1^\dagger \neq x_1'. \tag{5.16}$$

Now define $x_2^\dagger := \operatorname{argmin}_{x_2} \mathcal{L}_\rho(u', (x_1^\dagger, x_2), y')$ for all fixed $x_1^\dagger \in \{0,1\}^N$. Recall that $g(x)$ is the sum of a linear function of x_1, an indicator function enforcing a binary decision constraint on x_1, and an indicator function enforcing a convex closed polyhedral constraint on x. Consequently, it can be shown that

$$\begin{aligned} x_2^\dagger = \operatorname*{argmin}_{x_2} \; & (u', y', \rho(B_1 x_1^\dagger + c))^\top F x_2 + \frac{1}{2} x_2^\top H x_2 \\ & \text{s.t. } x_2 \in \mathcal{X}(x_1^\dagger), \end{aligned} \tag{5.17}$$

where

$$F := \begin{bmatrix} A^\top \rho \\ I \\ I \end{bmatrix} \rho B_2, \quad H := B_2^\top \rho B_2,$$

and $\mathcal{X}(x_1^\dagger)$ is a closed convex polyhedral constraint set dependant on the fixed value of $x_1^\dagger$. The optimization problem in (5.17) returns a unique minimizing argument as $H \succ 0$ and the constraint set is closed, so (5.17) is a *multiparametric quadratic program* in terms of the vector $(u', y', \rho(B_1 x_1^\dagger + c))$. A change of variables[2] can be used to obtain $x_2^\dagger$ from $x_2^\dagger := z^\star - H^{-1}F^\top(u', y', \rho(B_1 x_1^\dagger + c))$, where $z^\star$ is the minimizing argument of

$$\begin{aligned} &\min_z \ z^\top H z \\ &\text{s.t. } z \in \mathcal{Z}, \end{aligned} \tag{5.18}$$

such that $\mathcal{Z}$ is also a polyhedral constraint set. It is proven in [69, Theorem 4] that the minimizing value of (5.18) varies continuously with u' and y' (using the fact that $H \succ 0$), and to return to $\min_{x_2} \mathcal{L}_\rho(u', (x_1^\dagger, x_2), y')'$ from this minimizing value, one must only add additional terms that also vary continuously with u' and y'. Hence $\min_{x_2} \mathcal{L}_\rho(u', (x_1^\dagger, x_2), y')$ is continuous w.r.t u' and y', and the inequality (5.16) also holds within a neighbourhood of u' and y'. This concludes the proof for any pair (u', y') such that u' is in the relative interior of the constraint set enforced by the indicator function in $f(u)$. In the case where u' is a boundary point of this set, there exists a sequence of points $\{u_k\}$ that converge to u' such that $\mathcal{L}_\rho(u_k, x', y') = \infty$ for all x' and y'. Therefore, the 'neighbourhood' proposed in Lemma 5.A.1 does not exist because $x^{(i+1)}$ does not exist for these points. For the analysis presented in this section we can sidestep this issue by choosing to redefine $x^{(i+1)}$ such that it is always equal to x' for the values of $u^{(i+1)}$ where $f(u^{(i+1)}) = \infty$, safe in the knowledge that the iteration (5.15) can never enter this region. □

It is now possible to define

$$\mathcal{B}_\tau := \{(u, x, y) \in \mathbb{R}^{3N} : \|u - u'\|_2^2 \leq \tau, \|y - y'\|_2^2 \leq \tau\},$$

where $\tau \in \mathbb{R}_{++}$ can be (and is) chosen such that the projection of $\mathcal{B}_\tau$ onto $(u^{(j)}, y^{(j)})$-space is entirely contained with the neighbourhood of (u', y') where $x_1^{(j+1)} = x_1'$. It then follows from Lemma 5.A.1 that, given $(u^{(j+1)}, x^{(j)}, y^{(j)}) \in \mathcal{B}_\tau$, the updates in (5.15) (starting with

[2] See [69, §4] for more details.

$x^{(j+1)}$) are equivalent to

$$x_1^{(i+1)} = x_1' \tag{5.19a}$$

$$x_2^{(i+1)} \in \underset{x_2}{\operatorname{argmin}} \Bigg\{ g\big((x_1', x_2)\big) + (y^{(i)})^\top (Au^{(i+1)} + B_2 x_2 + B_1 x_1' - c) + \frac{1}{2}\|Au^{(i+1)} + B_2 x_2 + B_1 x_1' - c\|_\rho^2 \Bigg\} \tag{5.19b}$$

$$y^{(i+1)} := y^{(i)} + \rho(Au^{(i+1)} + B_2 x^{(i+1)} + B_1 x_1' - c) \tag{5.19c}$$

$$u^{(i+2)} \in \underset{u}{\operatorname{argmin}} \Bigg\{ f(u) + (y^{(i+1)})^\top (Au + B_2 x_2^{(i+1)} + B_1 x_1' - c) + \frac{1}{2}\|Au + B_2 x_2^{(i+1)} + B_1 x_1' - c\|_\rho^2 \Bigg\}. \tag{5.19d}$$

Hence the updates (5.19) are also the equivalent to an ADMM algorithm applied to the convex problem

$$\begin{aligned} \min_{(u,x_2)} \; & f(u) + g_2(x_2) \\ \text{s.t. } & Au + B_2 x_2 = c_2, \end{aligned} \tag{5.20}$$

where $g_2(x_2) := g\big((x_1', x_2)\big)$ for fixed x_1', and $c_2 := c - B_1 x_1'$. In the context of the energy management problem considered in this chapter, problem (5.20) equates to problem (MPC.3) with the engine switching sequence σ fixed, and is therefore also equivalent to the convex formulation considered in Chapter 4; problem (MPC.2). Consequently, (5.20) has a unique minimizing argument, denoted $(u^\star, x^\star, y^\star)$, for which the first order necessary and sufficient conditions are

$$\mathbf{0} = Au^\star + B_2 x_2^\star - c_2, \tag{5.21a}$$

$$\mathbf{0} \in \partial f(u^\star) + A^\top y^\star, \tag{5.21b}$$

$$\mathbf{0} \in \partial g_2(x_2^\star) + B_2^\top y^\star, \tag{5.21c}$$

5.A.3 Optimality and Convergence of the Convex ADMM Iteration

This section considers ADMM applied to problem (5.20) *only*; in the following section the convergence of ADMM for problem (5.20) is then connected back to the original nonconvex problem (5.13). It is a well known result that ADMM will converge from an arbitrary

initial point to a globally minimizing argument of (5.20) for very general classes of convex functions f and g [127]. The convergence result in [127] relies on the construction of a quadratic Lyapunov function that bounds $x^{(i+1)}$ and $y^{(i+1)}$ with respect to $x^\star$ and $y^\star$ as $i \to \infty$. In this section a slight variation on this approach is taken: the Lyapunov function is constructed to *also* bound $u^{(i+1)}$ relative to $u^\star$ as $i \to \infty$. The useful consequence of this variation is that the level sets of the Lyapunov function can be used to bound the magnitude of the iterate in all directions of (u, x, y)-space; a result that is then exploited in the following section.

Before continuing, recall that the ADMM iteration for problem (5.20) is given in (5.19), with the caveat that the iteration starts with the $u^{(i+1)}$ update. Additionally, recall that the $u^{(i+1)}$ and $x^{(i+1)}$ updates are the minimizing arguments of the augmented Lagrangian function corresponding to problem (5.20), which is defined as $\hat{\mathcal{L}}_\rho(u, x_2, y)$ and is equal to (5.14) with fixed $x_1 = x'$.

Lemma 5.A.2. *$(u^{(i+1)}, x_2^{(i+1)}, y^{(i+1)})$ satisfies the necessary conditions (5.21) iff $r^{(i+1)} = \mathbf{0}$ and $s^{(i+1)} = \mathbf{0}$, where*

$$\begin{aligned} r^{(i+1)} &:= Au^{(i+1)} + B_2 x_2^{(i+1)} - c_2, \\ s^{(i+1)} &:= A^\top \rho B_2 (x_2^{(i)} - x_2^{(i+1)}). \end{aligned}$$

Proof. Firstly consider condition (5.21c). Since $x_2 = x_2^{(i+1)}$ is by definition the minimizer of $\hat{\mathcal{L}}_\rho(u^{(i+1)}, x_2, y^{(i)})$, it follows that

$$\mathbf{0} \in \partial g_2(x_2^{(i+1)}) + B_2^\top y^{(i)} + B_2^\top \rho(Au^{(i+1)} + B_2 x_2^{(i+1)} - c_2), \tag{5.22}$$

so the update law $y^{(i+1)} = y^{(i)} + \rho(Au^{(i+1)} + B_2 x_2^{(i+1)} - c_2)$ ensures that

$$\mathbf{0} \in \partial g_2(x_2^{(i+1)}) + B_2^\top y^{(i+1)} \quad \forall (i+1) \in \mathbb{Z}_+, \tag{5.23}$$

Next, consider (5.21b). By definition $u = u^{(i+1)}$ minimizes $\hat{\mathcal{L}}_\rho(u, x_2^{(i)}, y^{(i)})$, so

$$\mathbf{0} \in \partial f(u^{(i+1)}) + A^\top (y^{(i+1)} + \rho(Au^{(i)} + B_2 x_2^{(i)} - c_2)).$$

Hence the update law $y^{(i+1)} = y^{(i)} + \rho(Au^{(i+1)} + Bx_2^{(i+1)} - c_2)$ gives

$$\mathbf{0} \in \partial f(u^{(i+1)}) + A^\top (y^{(i+1)} + \rho B_2 (x_2^{(i)} - x_2^{(i+1)})), \tag{5.24}$$

which in turn implies

$$\mathbf{0} \in \partial f(u^{(i+1)}) + A^\top (y^{(i+1)}) + s^{(i+1)} \quad \forall (i+1) \in \mathbb{Z}_{++}. \tag{5.25}$$

Therefore, (5.23) and (5.25) imply that $(u^{(i+1)}, x^{(i+1)}, y^{(i+1)})$ satisfies the conditions (5.21) if and only if $r^{(i+1)} = \mathbf{0}$ and $s^{(i+1)} = \mathbf{0}$. □

Lemma 5.A.3. *Define the set*

$$\mathcal{S}_k := \left\{(u, x_2, y) \in \mathbb{R}^{(9N)} : \|y - y^\star\|^2_{\rho^{-1}} + \|B_2(x_2 - x_2^\star)\|^2_\rho + \|r(u, x_2)\|^2_\rho \leq k\right\},$$

where $r(u, x_2) := Au + B_2x_2 - c_2$. *Choose* $k \in \mathbb{R}_{++}$, *then choose* $(u^{(0)}, x_2^{(0)}, y^{(0)}) \in \mathcal{S}_k$. *It then follows that,* $(u^{(i+1)}, x_2^{(i+1)}, u^{(i+1)}) \in \mathcal{S}_k$ *as* $i \to \infty$, *and both* $r^{(i+1)} \to \mathbf{0}$ *and* $s^{(i+1)} \to \mathbf{0}$ *as* $i \to \infty$.

Proof. It is first shown that $V^{(i)} := V(u^{(i)}, x_2^{(i)}, y^{(i)})$ is a Lyapunov function, where

$$V(u, x_2, y) := \|y - y^\star\|^2_{\rho^{-1}} + \|B_2(x_2 - x_2^\star)\|^2_\rho + \|r(u, x_2)\|^2_\rho.$$

This part of the proof is split into three steps:

1. (5.24) demonstrates that the $u^{(i)}$ update (5.15a) can be equivalently written as

$$u^{(i+1)} \in \underset{u}{\operatorname{argmin}} \left\{ f(u) + \left(y^{(i+1)} + \rho B(x_2^{(i)} - x_2^{(i+1)})\right)^\top Au \right\},$$

which implies that

$$f(u^{(i+1)}) + \left(y^{(i+1)} + \rho B_2(x_2^{(i)} - x_2^{(i+1)})\right)^\top Au^{(i+1)} \leq f(u^\star) + \left(y^{(i+1)} + \rho B_2(x_2^{(i)} - x_2^{(i+1)})\right)^\top Au^\star.$$

Similarly, (5.23) implies that

$$x_2^{(i+1)} = \underset{x}{\operatorname{argmin}} \left\{ g_2(x_2) + \left(y^{(i+1)}\right)^\top B_2x_2 \right\},$$

so

$$g_2(x_2^{(i+1)}) + \left(y^{(i+1)}\right)^\top B_2x_2^{(i+1)} \leq g_2(x_2^\star) + \left(y^{(i+1)}\right)^\top B_2x_2^\star.$$

Therefore, by defining $p^{(i+1)} := f(u^{(i+1)}) + g_2(x_2^{(i+1)})$ and $p^\star := f(u^\star) + g_2(x_2^\star)$, it can be shown that

$$\begin{aligned} p^{(i+1)} - p^\star &= f(u^{(i+1)}) - f(u^\star) + g_2(x_2^{(i+1)}) - g_2(x_2^\star) \\ &\leq \left(y^{(i+1)}\right)^\top \left(B_2(x_2^\star - x_2^{(i+1)}) + A(u^\star - u^{(i+1)})\right) \\ &\quad + (x_2^{(i)} - x_2^{(i+1)})^\top B_2^\top \rho A(u^\star - u^{(i+1)}) \qquad (5.26) \\ &= -\left(y^{(i+1)}\right)^\top r^{(i+1)} \\ &\quad + (x_2^{(i)} - x_2^{(i+1)})^\top B_2^\top \rho \left(-r^{(i+1)} + B_2(x_2^{(i+1)} - x_2^\star)\right) \end{aligned}$$

2. Since $(u^\star, x^\star, y^\star)$ must satisfy the first order necessary conditions (5.21), it follows that $u = u^\star$ is a minimizer of $f(u) + (y^\star)^\top Au$ and $x_2 = x_2^\star$ is a minimizer of $g_2(x_2) + (y^\star)^\top B_2 x_2$. Therefore,

$$f(u^\star) + (y^\star)^\top Au^\star \leq f(u) + (y^\star)^\top Au$$
$$g_2(x_2^\star) + (y^\star)^\top B_2 x_2^\star \leq g_2(x_2) + (y^\star)^\top B_2 x_2$$

for all u and x_2, so $f(u^\star) + g_2(x_2^\star) \leq f(u) + g_2(x_2) + (y^\star)^\top (Au + B_2 x_2 - c_2)$ and hence

$$p^\star - p^{(i+1)} \leq (y^\star)^\top r^{(i+1)} \tag{5.27}$$

3. Combining the bounds on $p^\star - p^{(i+1)}$ in (5.26) and (5.27) yields

$$\left(y^{(i+1)} - y^\star\right)^\top r^{(i+1)} - (x_2^{(i+1)} - x_2^{(i)})^\top B_2^\top \rho r^{(i+1)} + (x_2^{(i+1)} - x_2^{(i)})^\top B_2^\top \rho B_2 (x_2^{(i+1)} - x_2^\star) \leq 0. \tag{5.28}$$

The first term in (5.28) can be simplified using the update law $y^{(i+1)} = y^{(i)} + \rho r^{(i+1)}$ and completing the square:

$$\begin{aligned}
&2\left(y^{(i+1)} - y^\star\right)^\top r^{(i+1)} \\
&= 2\left(y^{(i)} - y^\star\right)^\top r^{(i+1)} + 2\|r^{(i+1)}\|_\rho^2 \\
&= \|y^{(i)} - y^\star + \rho r^{(i+1)}\|_{\rho^{-1}}^2 - \|y^{(i)} - y^\star\|_{\rho^{-1}}^2 + \|r^{(i+1)}\|_\rho^2 \\
&= \|y^{(i)} - y^\star + y^{(i+1)} - y^{(i)}\|_{\rho^{-1}}^2 - \|y^{(i)} - y^\star\|_{\rho^{-1}}^2 + \|r^{(i+1)}\|_\rho^2 \\
&= \|y^{(i+1)} - y^\star\|_{\rho^{-1}}^2 - \|y^{(i)} - y^\star\|_{\rho^{-1}}^2 + \|r^{(i+1)}\|_\rho^2.
\end{aligned}$$

The second and third terms in (5.28) can then be simplified as

$$\begin{aligned}
&\|r^{(i+1)}\|_\rho^2 - 2(x_2^{(i+1)} - x_2^{(i)})^\top B_2^\top \rho r^{(i+1)} + 2(x_2^{(i+1)} - x_2^{(i)})^\top B_2^\top \rho B_2 (x_2^{(i+1)} - x_2^\star) \\
&= \|r^{(i+1)} - B_2(x_2^{(i+1)} - x_2^{(i)})\|_\rho^2 + \|B_2(x_2^{(i+1)} - x_2^\star)\|_\rho^2 - \|B_2(x_2^{(i)} - x_2^\star)\|_\rho^2.
\end{aligned}$$

Therefore, (5.28) is equivalent to

$$\|y^{(i+1)} - y^\star\|_{\rho^{-1}}^2 - \|y^{(i)} - y^\star\|_{\rho^{-1}}^2 + \|r^{(i+1)} - B_2(x_2^{(i+1)} - x_2^{(i)})\|_\rho^2 + \|B_2(x_2^{(i+1)} - x_2^\star)\|_\rho^2 - \|B_2(x_2^{(i)} - x_2^\star)\|_\rho^2 \leq 0,$$

which, using the definition of $V^{(i)} = V(u^{(i)}, x^{(i)}, y^{(i)})$, is equivalent to

$$\begin{aligned} V^{(i+1)} - V^{(i)} &\leq -\|r^{(i+1)} - B_2(x_2^{(i+1)} - x_2^{(i)})\|_\rho^2 - \|r^{(i)}\|_\rho^2 + \|r^{(i+1)}\|_\rho^2 \\ &= -\|r^{(i)}\|_\rho^2 - \|B_2(x_2^{(i+1)} - x_2^{(i)})\|_\rho^2 + 2(r^{(i+1)})^\top \rho B_2(x_2^{(i+1)} - x_2^{(i)}) \\ &= -\|r^{(i)}\|_\rho^2 - \|B_2(x_2^{(i+1)} - x_2^{(i)})\|_\rho^2 + 2(y^{(i+1)} - y^{(i)})^\top B_2(x_2^{(i+1)} - x_2^{(i)}). \end{aligned}$$

To bound the right hand side of this expression, recall from 1. that $x_2 = x_2^{(i)}$ and $x_2 = x_2^{(i+1)}$ are minimizers of $g_2(x_2) + \left(y^{(i)}\right)^\top B_2 x_2$ and $g_2(x_2) + \left(y^{(i+1)}\right)^\top B_2 x_2$, so that

$$g_2(x_2^{(i)}) + (y^{(i)})^\top B_2 x_2^{(i)} \leq g_2(x_2^{(i+1)})^\top + (y^{(i)})^\top B_2 x_2^{(i+1)}$$

and

$$g(x^{(i+1)}) + (y^{(i+1)})^\top B x^{(i+1)} \leq g(x^{(i)})^\top + (y^{(i+1)})^\top B x^{(i)}.$$

Summing these two inequalities yields

$$(y^{(i+1)} - y^{(i)}) B (x^{(i+1)} - x^{(i)}) \leq 0,$$

and it follows that

$$V^{(i+1)} - V^{(i)} \leq -\|r^{(i)}\|_\rho^2 - \|B_2(x_2^{(i+1)} - x_2^{(i)})\|_\rho^2.$$

The positive invariance of $\mathcal{S}_k$ follows from its definition as a sublevel set of $V(u, x, y)$ and from the fact that $V^{(i)}$ is monotonically non-increasing with i. Asymptotic convergence of the residuals $r^{(i+1)}$ and $s^{(i+1)}$ can be established by summing both sides of the inequality satisfied by $V^{(i+1)} - V^{(i)}$ over all $i \geq 0$:

$$\sum_{i=0}^{\infty} \left(\|r^{(i)}\|_\rho^2 + \|B_2(x_2^{(i+1)} - x_2^{(i)})\|_\rho^2 \right) \leq V^{(0)}.$$

It can be concluded from this bound that, as $i \to \infty$, $r^{(i)} \to \mathbf{0}$ and $B_2(x_2^{(i+1)} - x_2^{(i)}) \to \mathbf{0}$, and hence also $s^{(i)} \to \mathbf{0}$. □

5.A.4 Optimality and Convergence of the Nonconvex ADMM Iteration

The results from Sections 5.A.2 and 5.A.3 are now combined to complete the proof of Proposition 5.A.1. The principle of the approach is that there is an equivalence between (u', x', y') and $(u^\star, (x_1', x_2^\star), y^\star)$, and it can then shown that there must exist a convergent neighbourhood of $(u^\star, (x', x^\star), y^\star)$.

Lemma 5.A.4. *Any stationary point, (u', x', y'), of iteration (5.15), is also a minimizing argument of (5.20), given x_1'. Conversely, for any fixed x_1', the point $(u^\star, (x_1', x_2^\star), y^\star)$ is a stationary point of iteration (5.15).*

Proof. For the first statement, if $x_1^{(i)} = x_1'$ for all i, then iteration (5.15) is always equivalent to iteration (5.19), for which the only stationary point is $(u^\star, (x_1', x_2^\star), y^\star)$. For the second statement, the point $(u^\star, (x_1', x_2^\star), y^\star)$ is necessarily a fixed point of iteration (5.19), which is therefore also a stationary point of iteration (5.15). □

Lemma 5.A.4 has two significant consequences. Firstly, in the context of the energy management problem discussed in this chapter, *every* stationary point of the ADMM iteration (5.15) results in the minimum fuel consumption given the engine switching sequence. Additionally, it means that (u', x', y') and $(u^\star, x^\star, u^\star)$ (where $x^\star := (x_1', x_2^\star)$) can be used interchangeably, most importantly in the definitions of $\mathcal{B}_\tau$ and $\mathcal{S}_k$. The proof is now completed by showing that there is a locally convergent neighbourhood around every $(u^\star, x^\star, u^\star)$.

Lemma 5.A.5. *Define the set*

$$\hat{\mathcal{S}}_k := \left\{(u, x, y) \in \mathbb{R}^{9N} : (u, x_2, y) \in \mathcal{S}_k\right\}.$$

For every $\tau > 0$, there exists $k > 0$ such that $\hat{\mathcal{S}}_k \subset \mathcal{B}_\tau$, and for every $k > 0$ there exists a neighbourhood of (u, x, y)-space centred on $(u^\star, x^\star, y^\star)$ that is entirely contained within $\hat{\mathcal{S}}_k$.

Proof. Note that $r(u, x) = A(u - u^\star) + B(x_2 - x_2^\star)$, so the left hand side of the inequality

$$\|y - y^\star\|^2_{\rho^{-1}} + \|B_2(x_2 - x_2^\star)\|^2_\rho + \|r(u, x)\|^2_\rho \leq k \tag{5.29}$$

in the definition of $\mathcal{S}_k$ is equivalent to

$$\|y - y^\star\|^2_{\rho^{-1}} + \begin{bmatrix} A(u - u^\star) \\ B_2(x_2 - x_2^\star) \end{bmatrix}^\top \underbrace{\begin{bmatrix} \rho & \rho \\ \rho & 2\rho \end{bmatrix}}_{M} \begin{bmatrix} A(u - u^\star) \\ B_2(x_2 - x_2^\star) \end{bmatrix}.$$

The eigendecomposition of the matrix M is

$$\begin{bmatrix} I & \\ I & I \end{bmatrix} \begin{bmatrix} \rho & \\ & \rho \end{bmatrix} \begin{bmatrix} I & I \\ & I \end{bmatrix}$$

so M is positive definite. Both A and B_2 have full column rank, so (5.29) defines an ellipsoid centred on (u, x_2, y) that bounds the magnitude of $u - u^\star$, $x - x^\star$, and $y - y^\star$ in all directions. Therefore, there exists a $k > 0$ such that $\hat{\mathcal{S}}_k \subset \mathcal{B}_\tau$ for any $\tau > 0$. Additionally, for any $k > 0$ there exists a 2-norm ball centred on (u, x, y) that is a subset of $\hat{\mathcal{S}}_k$, which completes the proof. □

Lemma 5.A.6. *Choose $k > 0$ such that $\hat{\mathcal{S}}_k \subset \mathcal{B}_\tau$, and choose arbitrary $(u^{(0)}, x^{(0)}, y^{(0)})$. Define $S_1 := \{(u^{(i+1)}, x^{(i+1)}, y^{(i+1)})\}$ as the sequence generated using ADMM update (5.15) as $i \to \infty$, starting from $i = 0$. Assume that there exists at least one $i \in \mathbb{Z}_+$ such that $(u^{(i)}, x^{(i)}, y^{(i)}) \in \hat{\mathcal{S}}_k$, and let $\underline{i}$ be the first such value. Now define $S_2 := \{(u^{(i+1)}, x^{(i+1)}, y^{(i+1)})\}$ as a second sequence generated using the ADMM update (5.15) for $i \in \{0, \dots, \underline{i} - 1\}$ only, and then using iteration (5.19) (such that the iteration starts with $u^{(i+1)}$) as $i \to \infty$. It then follows that*

- $S_1 = S_2$
- $(u^{(i)}, x^{(i)}, y^{(i)}) \to (u^*, x^*, y^*)$ *as* $i \to \infty$.

Proof. To begin, it is clear that the elements of the sequences S_1 and S_2 are equal for all $i \in \{0, \dots, \underline{i} - 1\}$. Now consider the sequence S_2. If $(u^{(\underline{i})}, x^{(\underline{i})}, y^{(\underline{i})}) \in \hat{\mathcal{S}}_k$, then $(u^{(i)}, x^{(i)}, y^{(i)}) \in \hat{\mathcal{S}}_k$ for all $i \geq \underline{i}$ when generated using iteration (5.19). If $(u^{(i)}, x^{(i)}, y^{(i)}) \in \hat{\mathcal{S}}_k$ and $(u^{(i+1)}, x^{(i+1)}, y^{(i+1)}) \in \hat{\mathcal{S}}_k$, then both of these points are also elements of $\mathcal{B}_\tau$, and it follows that $(u^{(i+1)}, x^{(i)}, y^{(i)})$ is also an element of $\mathcal{B}_\tau$. Therefore, the ADMM iteration (5.19) is equivalent to iteration (5.15) for all $i \geq \underline{i}$, so $S_2 = S_1$. Additionally, the sequence S_2 necessarily converges to (u^*, x^*, y^*), which completes the proof. □

The immediate consequence of Lemma 5.A.6 is that there is a neighbourhood close to every $(u^\star, x^\star, y^\star)$ in which the iteration (5.19) necessarily converges to $(u^\star, x^\star, y^\star)$. Additionally, due to Lemma 5.A.4, iteration (5.19) is *only* stationary at points that satisfy the first order conditions 5.21 (i.e. points that satisfy the conditions to be considered $(u^\star, x^\star, y^\star)$). As a final comment on the result of this appendix, note from the definitions of A and B in (5.5) that the residual $s^{(i+1)}$ used to terminate the algorithm in (5.8) is the concatenation of $\rho_4(x_1^{(i+1)} - x_1^{(i)})$ and the residual $s^{(i+1)}$ used in Section 5.A.3. Therefore, a sufficiently small termination threshold ϵ (specifically, $\epsilon < \rho_4$ if the 2-norm is used in (5.8)) provides a certificate that $x_1^{(i+1)} - x_2^{(i)} = \mathbf{0}$. Additionally, the residual $r^{(i+1)}$ used to terminate the algorithm in (5.8) is equivalent to the residual $r^{(i+1)}$ used in Section 5.A.3, so a small value of ϵ also provides a certificate that $(u^{(i+1)}, x^{(i+1)}, y^{(i+1)})$ is 'close' to $(u^\star, x^\star, y^\star)$.

CHAPTER 6

DRIVER BEHAVIOUR UNCERTAINTY

THUS far, it has been assumed that the driver's future behaviour is known to the controller with complete precision, whereas in reality it is subject to considerable uncertainty. For each of the frameworks presented in the previous chapters, it is possible to generate predictions that are sufficiently inaccurate for the MPC to perform as badly as (or worse than) CDCS. For example, if the predictions estimate that the journey will be shorter than the all-electric range of the vehicle (when in reality the journey is longer than the all-electric range), then the MPC strategy will behave very similarly to CDCS. In this chapter the robustness of MPC is increased by considering multiple predictions of future driver behaviour, possibly (although not necessarily) drawn from a database of previous journeys on a given route. In particular, it is demonstrated that the ADMM algorithm used in previous chapters can be readily extended to a new formulation that takes into account multiple scenarios of future driver behaviour.

6.1 MODEL PREDICTIVE CONTROLLER

The convex formulation presented in Chapter 4 is considered for the basis of the approach, and the engine switching control action is not considered to be a decision variable in the MPC optimization. This approach is taken so that the suboptimality in the simulations that follow is attributable only to the inaccuracies in the predictions, as opposed to the suboptimality of the predictions *and* the local convergence of the ADMM algorithm. The approach presented here can, however, also be extended to the case where engine switching is also considered in Chapter 5.

Recall from Chapter 4 that the convex MPC controller is defined at each control variable update instance as the first element of the minimizing argument of (MPC.2):

$$\begin{aligned}
\min_{u} \ & \sum_{k\in\mathcal{P}} f_k(p_k - g_k^{-1}(u_k)), \\
\text{s.t. } & x = \mathbf{1}x(t) - \Psi u, \\
& \mathbf{1}\underline{x} \leq x \leq \mathbf{1}\overline{x}, \\
& \underline{u} \leq u \leq \overline{u},
\end{aligned}$$

where the parameters of the functions f_k and g_k^{-1}, the horizon length N, the elements of the set $\mathcal{P}$, and the values of $\underline{u}$ and $\overline{u}$ are dependent on a single, deterministic prediction of the future road gradient, θ, and vehicle velocity, v.

Now, consider the case where the predictions of v and θ are drawn from sets $\mathcal{V}$ and Θ according to an associated probability distribution. The objective of the optimization problem may now be to minimize the expected cost across all possible realizations of the uncertainty, and the MPC could be defined by the minimizing argument of the stochastic program

$$\begin{aligned}
\min_{u} \ & \mathbb{E}\left[\sum_{k\in\mathcal{P}(v,\theta)} f_{v_k,\theta_k}(p(v_k,\theta_k) - g_{v_k,\theta_k}^{-1}(u_k))\right], \\
\text{s.t. } & x = \mathbf{1}x(t) - \Psi u, \\
& \mathbf{1}\underline{x} \leq x \leq \mathbf{1}\overline{x}, \\
& \underline{u}(v,\theta) \leq u \leq \overline{u}(v,\theta), \\
& v \in \mathcal{V}, \quad \theta \in \Theta.
\end{aligned} \tag{6.1}$$

This approach would increase the robustness of the controller as it is more likely that the resulting driver behaviour will have been explicitly considered by the MPC optimization, but still has two significant issues. Firstly, in order for the controller to minimize the expected fuel consumption when implemented in closed loop, the probability measure associated with $\mathcal{V}$ and Θ must still closely match real driver behaviour, a phenomenon that is extremely challenging to model explicitly. Secondly, even in the case where an accurate representation of driver behaviour is available and the associated probability density function is 'simple', problem (6.1) is nonconvex and intractable in general.

Therefore, instead consider the case where $S \in \mathbb{Z}_{++}$ individual samples of v and θ are obtained from the uncertainty sets, where each sampled pair $(v,\theta) \in \mathcal{V} \times \Theta$ is referred to as a *scenario*. It is again assumed that each of the scenarios is sampled at 1 Hz over the remaining duration of the journey, although as before, this approach can be extended to an

arbitrary sampling interval. Additionally, it is assumed that each of the predictions are of the same length, N, as although the remaining duration of the journey may vary for each scenario, zeros can be appended to the shorter predictions until they equal the length of the longest prediction. Therefore, the predictions are given by $v \in \mathbb{R}^{N\times S}$ and $\theta \in \mathbb{R}^{N\times S}$, where $v_{:,s} \in \mathcal{V}$ and $\theta_{:,s} \in \Theta \; \forall s \in \{1,\dots,S\}$. The approach detailed in Chapter 3 can then be used to obtain the optimization parameters across all of the predictions, now given by $\underline{u} \in \mathbb{R}^{N\times S}$, $\overline{u} \in \mathbb{R}^{N\times S}$, and $p \in \mathbb{R}^{N\times S}$. Additionally, the set $\mathcal{P}$ is defined for each scenario $s \in \{1,\dots,S\}$ as $\mathcal{P}_s$, and the functions $f_{k,s}(\cdot)$ and $g_{k,s}^{-1}(\cdot)$ are defined for all elements of the prediction horizon $k \in \{0,\dots,N-1\}$, and scenarios $s \in \{1,\dots,S\}$. The stochastic program (6.1) may then be be approximated with

$$
\begin{aligned}
&\min_{(u,\tau)} \frac{1}{S}\sum_{s=1}^{S}\Bigg(\sum_{k\in\mathcal{P}_s} f_{k,s}(p_{k,s} - g_{k,s}^{-1}(u_{k,s}))\Bigg),\\
&\text{s.t. } x_{:,s} = \mathbf{1}x(t) - \Psi u_{:,s},\\
&\quad \mathbf{1}\underline{x} \le x_{:,s} \le \mathbf{1}\overline{x},\\
&\quad \underline{u}_{:,s} \le u_{:,s} \le \overline{u}_{:,s}\\
&\quad u_{0,s} = \tau,\\
&\quad \forall s \in \{1,\dots,S\}.
\end{aligned}
\tag{MPC.4}
$$

From a computational perspective, (MPC.4) has two significant advantages relative to (6.1). Firstly, (MPC.4) is convex and deterministic, and can therefore be readily solved using standard convex optimization algorithms. Secondly, (MPC.4) does not require the driver's behaviour to be modelled explicitly, and the scenarios can be drawn from a database obtained from previous examples of a given route (or its constituent parts, assembled from available data). An additional advantage of (MPC.4) from a control perspective is that it allows a separate control vector for each scenario (i.e. $u \in \mathbb{R}^{N\times S}$) rather than a single control vector that minimizes the objective across all predictions. This is preferable as at any given instant along the prediction horizon one scenario may predict high speed and/or acceleration whereas another may predict that the vehicle is moving slowly or is stationary (this places conflicting requirements on the predicted control input which cannot satisfy both simultaneously). The predicted control trajectories in (MPC.4) can therefore be interpreted as samples of a control probability density function, which is a more general representation than that considered in (6.1). The solution of (MPC.4) is then made meaningful in the context of MPC by constraining the first element of each predicted control sequence to be equal, so that the feedback law is defined by $u(t) := u_{0,1} = u_{0,2} = \cdots = u_{0,S} = \tau$.

A limitation of (MPC.4) is that the cost can only be considered equal to the expected cost in the limit as $S \to \infty$ (the error term between the expected cost and the solution of

(MPC.4) is $\mathcal{O}(S^{-1/2})$ [133]). This may become an issue if both a large number of scenarios and a long prediction horizon is required, as (MPC.4) has $\mathcal{O}(NS)$ decision variables. An efficient ADMM algorithm for the solution of (MPC.4) is presented in the following to address this issue. A further limitation is that the solution of (MPC.4) represents S deterministic, open loop policies, so the optimal value of (MPC.4) does not necessarily converge to the resulting closed-loop cost as $S \to \infty$. Modifications to (MPC.4) that reduce this optimality gap are an interesting direction for future work.

6.1.1 FEASIBILITY

This section addresses the feasibility of (MPC.4) when implemented in closed loop. In particular, results from scenario MPC are used to obtain a bound on the number of scenarios, S, required to bound the confidence that the one-step-ahead instance of (MPC.4) will be feasible with a given probability.

Assumption 6.1.1. *The first element of the predictions of velocity and road gradient are accurate measurements of the vehicle's current velocity and the current road gradient, and are therefore equal for each scenario, i.e* $v_{0,s} = v(t)$ *and* $\theta_{0,s} = \theta(t)$ *for all* $s \in \{1, \dots, S\}$.

Under Assumption 6.1.1, $\underline{u}_{0,s} = \underline{u}(t)$ and $\overline{u}_{0,s} = \overline{u}(t)$ for all $s \in \{1, \dots, S\}$ (where $\underline{u}(t) \in \mathbb{R}$ and $\overline{u}(t) \in \mathbb{R}$ are the lower and upper limits on battery power at time t), so feasibility for a given instance of problem (MPC.4) can be determined by applying Algorithm 1 to each scenario individually.

Assumption 6.1.2. *The scenarios are generated so that*

$$x(t) - \overline{u}(t) - \sum_{i=1}^{k} \overline{u}_{i,s} \leq \overline{x} \tag{6.2}$$

and

$$x(t) - \underline{u}(t) - \sum_{i=1}^{k} \underline{u}_{i,s} \geq \underline{x} \tag{6.3}$$

for all $k \in \{1, \dots, N-1\}$ *and* $s \in \{1, \dots, S\}$.

Remark 6.1.1. Assumption 6.1.2 reflects hard constraints on vehicle operation that in turn also constrain driver behaviour, so every $(v, \theta) \in \mathcal{V} \times \Theta$ must satisfy (6.2) and (6.3). In practice, Assumption 6.1.2 can be enforced by rejecting any scenario for which Algorithm 1 returns an infeasibility certificate.

Using the definition of the feasible tube, $\{\mathcal{F}_0, \ldots, \mathcal{F}_N\}$, proposed in 4.1.2, problem (MPC.4) is feasible under Assumption 6.1.2 if and only if

$$x(t) - \overline{u}(t) \leq \overline{x} \quad \text{and} \quad x(t) - \underline{u}(t) \geq \underline{x}.$$

Hence, (MPC.4) is feasible at time $t+1$ if and only if

$$x(t) - \tau^\star(t) - \overline{u}(t+1) \leq \overline{x} \quad \text{and} \quad x(t) - \tau^\star(t) - \underline{u}(t+1) \geq \underline{x}, \tag{6.4}$$

where $(u^\star(t), \tau^\star(t))$ is the solution of (MPC.4) at time t. Now, note that problem (MPC.4) is equivalent to

$$\min_{(u,\tau)\in\mathcal{C}} \frac{1}{S}\sum_{s=1}^{S}\left(\sum_{k\in\mathcal{P}_s} f_{k,s}(p_{k,s} - g_{k,s}^{-1}(u_{k,s}))\right), \tag{6.5a}$$

$$\text{s.t. } x(t) - \tau \in [\underline{x} + \underline{u}_{1,s}, \overline{x} + \overline{u}_{1,s}] \tag{6.5b}$$

$$\forall s \in \{1, \ldots, S\},$$

where $\mathcal{C}$ is a convex constraint set, and is the form of problem considered in [100] and [99]. To apply the results of these papers, assume that (6.5) is feasible, and define the violation probability for the one-step-ahead feasibility condition (6.4) as

$$V(\tau^\star(t)) := \mathbb{P}\{x(t) - \tau^\star(t) \notin [\underline{x} + \underline{u}(t+1), \overline{x} + \overline{u}(t+1)]\},$$

where $\underline{u}(t+1)$ and $\overline{u}(t+1)$ can be interpreted as an unseen sample of the random variables $\underline{u}_{1,s}$ and $\overline{u}_{1,s}$ in (6.5). The support rank of constraint (6.5b) is one[1], since the constraint fixes a single degree of freedom of the decision variable (u, τ) (see Definition 10 in [100]). Therefore, Lemma 13 in [100] implies the confidence bound

$$\mathbb{P}\{V(\tau^\star(t)) > \hat{\epsilon}\} \leq B(\epsilon; S, 0) = (1-\hat{\epsilon})^S \tag{6.6}$$

on the violation (i.e. one-step-ahead infeasibility) probability. More generally, if R samples are discarded (using any criteria, e.g. a greedy strategy), then the right-hand-side of (6.6) can be replaced with $B(\hat{\epsilon}; S, R)$, where B is the beta distribution.

Let the confidence that the violation probability is no greater than $\hat{\epsilon}$ be $\hat{\beta}$, so that

$$\mathbb{P}\{V(\tau^\star(t)) \leq \hat{\epsilon}\} = \hat{\beta} \geq 1 - (1-\hat{\epsilon})^S, \tag{6.7}$$

[1]Note that, whist the set $\mathcal{C}$ depends on the S random samples of $\mathcal{V}$ and Θ and is therefore 'stochastic', the problem of interest here is whether the solution of (MPC.4) is still feasible with respect to an additional randomly sampled constraint of the form of (6.5b) *only*, and not every constraint from (MPC.4) (i.e. $\mathcal{C}$ does not change when the new scenario is sampled). Therefore, $\mathcal{C}$ can be considered deterministic, and the relevant support rank is that of constraint (6.5b).

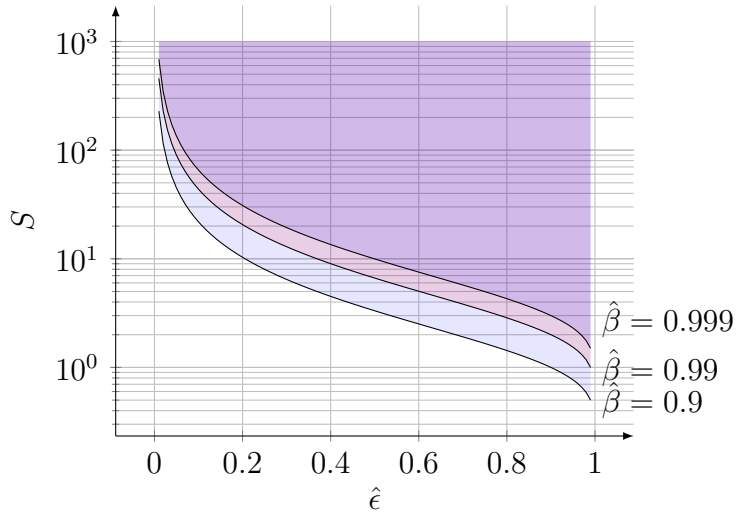


Figure 6.1: Illustration of the values of S that satisfy (6.8) for $\hat{\beta} \in \{0.9, 0.99, 0.999\}$ and $\hat{\epsilon} \in [0.01, 0.99]$.

then the lower bound on the number of scenarios required to meet a given confidence of violation probability is

$$S \geq \frac{\log\left(1-\hat{\beta}\right)}{\log\left(1-\hat{\epsilon}\right)}. \tag{6.8}$$

This bound is illustrated in Figure 6.1. One-step-ahead feasibility (and by induction, recursive feasibility) is still only guaranteed with complete certainty in the limit as $\mathcal{S} \to \infty$. However, note that (6.7) is a *lower bound* on the confidence of constraint violation probability, which will only be tight when the state-of-charge of the battery is 'close' to its limits (i.e. one-step-ahead feasibility is guaranteed when the battery's state-of-charge is 'far' from its limits, as it will not be possible to exceed the limits within two sampling intervals). It is not common for the state-of-charge of the battery to approach its upper or lower bounds more than once or twice during a given journey (see for example [84, §5], [47, §5], [113, §5], and the previous experiments in this thesis), so high confidence of feasibility throughout a given journey may be obtained with a modest number of scenarios. For example, a bound on constraint violation of $\hat{\epsilon} = 0.1$ can be achieved with confidence of $\hat{\beta} \geq 0.9$ using just $S = 22$ scenarios. This can be improved to $V(\tau^\star(t)) \leq 0.01$ with a confidence of $\hat{\beta} \geq 0.99$ by using $S = 459$ scenarios.

Note that whilst the energy management problem considered in Section 5 can be extended, in principle, to consider multiple predictions of future driver behaviour in the same way as presented in this section, the bound (6.6) would not necessarily hold for the new

problem as it would not be convex (to apply the results from [100] and [99], it is also required that $\tau^\star$ is the minimizing argument, which is not guaranteed using the ADMM algorithm proposed in Section 5).

6.2 OPTIMIZATION

An ADMM algorithm similar to those presented in previous chapters is now proposed for the solution of (MPC.4).

ALGORITHM

Problem (MPC.4) is equivalent to the equality constrained problem

$$\begin{aligned}
&\min_{u} \frac{1}{S}\sum_{s=1}^{S}\left(\sum_{k\in\mathcal{P}_s}\left(f_{k,s}(p_{k,s}-g_{k,s}^{-1}(u_{k,s}))\right)+\sum_{k=0}^{N-1}\left(\mathcal{I}_{\mathcal{U}_{k,s}}(u_{k,s})+\mathcal{I}_{\mathcal{X}}(x_{k+1,s})\right)\right),\\
&\text{s.t. } u=\zeta\\
&\quad x=\Phi x(t)-\Psi\zeta,\\
&\quad \mathbf{1}\tau=u_{0,:}^{\top},
\end{aligned} \tag{6.9}$$

which is in turn equivalent to

$$\min_{\tilde{u},\tilde{x}}\ \tilde{f}(\tilde{u})\quad \text{s.t } A\tilde{u}+B\tilde{x}=c, \tag{6.10}$$

where

$$\begin{aligned}
\tilde{u}&:=(\operatorname{vec} u,\operatorname{vec} x),\quad \tilde{x}:=(\tau,\operatorname{vec}\zeta)\\
\tilde{f}(\tilde{u})&:=\frac{1}{S}\sum_{s=1}^{S}\left(\sum_{k\in\mathcal{P}_s}\left(f_{k,s}(p_{k,s}-g_{k,s}^{-1}(u_{k,s}))\right)+\sum_{k=0}^{N-1}\left(\mathcal{I}_{\mathcal{U}_{k,s}}(u_{k,s})+\mathcal{I}_{\mathcal{X}}(x_{k+1,s})\right)\right),\\
A&:=\begin{bmatrix} I & O\\ O & I\\ -(\mathbf{1}\otimes[1,0,\ldots,0]) & O\end{bmatrix},\quad B:=\begin{bmatrix} O & -I\\ O & (I\otimes\Psi)\\ \mathbf{1} & O\end{bmatrix},\quad c:=(\mathbf{0},\mathbf{1}x(t),\mathbf{0}).
\end{aligned}$$

Problem (6.10) is the canonical ADMM form (5.4) presented in Chapter 5, and the same ADMM algorithm is therefore used. As problem (MPC.4) is strictly convex (this result trivially follows from Theorem 3.2.1), the iteration (5.7) can be made to terminate at a point that can be made arbitrarily close to the unique minimizing argument of (MPC.4) by setting the termination threshold, ϵ, arbitrarily close to zero.

6.2.1 VARIABLE UPDATES & COMPLEXITY

The augmented Lagrangian function for (6.10) is

$$\mathcal{L}_\rho(\tilde{u},\tilde{x},\lambda) := \frac{1}{S}\sum_{s=1}^{S}\Bigg(\sum_{k\in\mathcal{P}_s}\Big(f_{k,s}(p_{k,s}-g_{k,s}^{-1}(u_{k,s}))\Big)+ \sum_{k=0}^{N-1}\big(\mathcal{I}_{\mathcal{U}_{k,s}}(u_{k,s})+\mathcal{I}_{\mathcal{X}}(x_{k+1,s})\big)+\frac{\rho_1}{2}\|u_{:,s}-\zeta_{:,s}+\nu_{:,s}\|^2 + \frac{\rho_2}{2}\|x_{:,s}+\Psi\zeta_{:,s}-\mathbf{1}x(t)+\psi_{:,s}\|^2+\frac{\rho_3}{2}(\tau-u_{0,s}+\phi_s)^2\Bigg), \quad (6.11)$$

where the vector of Lagrange multipliers is now defined by $\lambda := (\text{vec}\ \nu, \text{vec}\ \psi, \phi)$, and $\nu \in \mathbb{R}^{N\times S}$, $\psi \in \mathbb{R}^{N\times S}$, and $\phi \in \mathbb{R}^S$ (different symbols are used to increase the readability of subscripts). The u and x variables are completely decoupled in (6.11), so the $\tilde{u}$ ADMM update (5.7a) is equivalent to

$$\begin{aligned}
u_{0,s}^{(i+1)} &= \underset{u_{0,s}}{\text{argmin}}\Big(f_{0,s}(p_{0,s}-g_{0,s}^{-1}(u_{0,s}))+\mathcal{I}_{\mathcal{U}_{0,s}}(u_{0,s})+\frac{\rho_1}{2}(u_{0,s}-\zeta_{0,s}^{(i)}+\nu_{0,s}^{(i)})^2 \\
&\qquad\qquad +\frac{\rho_3}{2}(\tau^{(i)}-u_{0,s}+\phi_s^{(i)})^2\Big) \quad \forall s\in\{0,\dots,S\} \\
u_{k,s}^{(i+1)} &= \underset{u_{k,s}}{\text{argmin}}\Big(f_{k,s}(p_{k,s}-g_{k,s}^{-1}(u_{k,s}))+\mathcal{I}_{\mathcal{U}_{k,s}}(u_{k,s})+\frac{\rho_1}{2}(u_{k,s}-\zeta_{k,s}^{(i)}+\nu_{k,s}^{(i)})^2\Big) \\
&\qquad\qquad \forall s\in\{0,\dots,S\},\ k\in\mathcal{P}_s\setminus\{0\} \\
u_{k,s}^{(i+1)} &= \underline{u}_{k,s} \quad \forall s\in\{0,\dots,S\}, k\in\mathcal{C}_s \\
x_{:,s}^{(i+1)} &= \Pi_{\mathcal{X}^N}\left[\mathbf{1}x(t)-\Psi\zeta_{:,s}^{(i)}-\psi_{:,s}^{(i)}\right]
\end{aligned}$$

The update $u_{k,s}^{(i+1)}$ is of the equivalent form to (4.9a) for all $s\in\{0,\dots,S\}$ and $k\in\mathcal{P}_s$ (i.e. a 1-dimensional convex inequality constrained optimization problem) which can be solved using Newton's method. This update is separable w.r.t both the scenarios and the prediction horizon, and so the computational requirement is constant as both are increased if the updates are performed in parallel (so long as $\sum_{s=1}^{S}|\mathcal{P}_s|$ threads are available). Otherwise, the computational complexity is $\mathcal{O}(NS)$ if the updates are performed sequentially. The update for $x^{(i+1)}$ is equivalent to the deterministic case (4.9b) for each scenario, so the complexity is $\mathcal{O}(N)$ using parallel processing, and $\mathcal{O}(NS)$ using sequential processing. No additional memory is required for either the $u^{(i+1)}$ or $x^{(i+1)}$ update.

The τ and ζ variables are also completely decoupled in (6.11), so the $\tilde{x}$ ADMM update (5.7b) is equivalent to

$$\tau^{(i+1)} = \frac{1}{S}\sum_{s=1}^{S}(u_{0,s}^{(i+1)} + \phi_s^{(i)})$$
$$\zeta_s^{(i+1)} = (\rho_1 I + \rho_2 \Psi^\top \Psi)^{-1}\Big(\rho_1(u_{:,s} + \nu_{:,s}) - \rho_2 \Psi^\top (x_{:,s} - \mathbf{1}x(t) + \psi_{:,s})\Big).$$

The computation of $\tau^{(i+1)}$ scales linearly with S regardless of whether parallel processing is available, but is only a scalar sum, so will typically be computationally inexpensive. The $\zeta^{(i+1)}$ is equivalent to the deterministic case (4.10) for each scenario, so the computational complexity is $\mathcal{O}(N)$ if the S updates are performed in parallel, or $\mathcal{O}(NS)$ if they are performed sequentially. The Cholesky factors required for the solution of the linear system of equations are the same for each scenario so only need storing once. Therefore, the memory cost of the update is $\mathcal{O}(N)$.

The residual updates (5.9) are equivalent to

$$r^{(i+1)} = \begin{bmatrix} \operatorname{vec} u^{(i+1)} - \operatorname{vec} \zeta^{(i+1)} \\ \operatorname{vec} x - \mathbf{1}x(t) + (\Psi\zeta_{:,1}^{(i+1)}, \dots, \Psi\zeta_{:,S}^{(i+1)}) \\ \mathbf{1}\tau^{(i+1)} - (u_{0,:}^{(i+1)})^\top \end{bmatrix}$$
$$s^{(i+1)} = \begin{bmatrix} \rho_1 \operatorname{vec}(\zeta^{(i)} - \zeta^{(i+1)}) - \rho_3(\mathbf{1}_S \otimes [1, 0, \dots, 0]^\top)(\tau^{(i)} - \tau^{(i+1)}) \\ \rho_2\Big(\Psi(\zeta_{:,1}^{(i)} - \zeta_{:,1}^{(i+1)}), \dots, \Psi(\zeta_{:,S}^{(i)} - \zeta_{:,S}^{(i+1)})\Big) \end{bmatrix},$$

for which the computation scales linearly with N and S (the only non-trivial computations are multiplication by Ψ, which is equivalent to a cumulative sum, and the Kronecker product $(\mathbf{1}_S \otimes [1, 0, \dots, 0]^\top)$, which does not involve any decision variables and can be performed offline). $\mathcal{O}(NS)$ memory storage is required to store the variables $\tau^{(i)}$ and $\zeta^{(i)}$ for the computation of $s^{(i+1)}$ at the following iteration. The λ update is then obtained again from $\lambda^{(i+1)} = \lambda^{(i)} + r^{(i+1)}$.

The computational complexity of each iteration of the ADMM algorithm is summarized in Table 6.1. A feature of particular note is that the complexity of the u update, which may be computationally expensive due to the lack of closed-form solution, is reduced from $\mathcal{O}(NS)$ to $\mathcal{O}(1)$ when a sufficient number of parallel threads are available.

Table 6.1: Summary of the computational complexity of each of the ADMM variable updates.

$\mathcal{O}(\cdot)$	u	x	τ	ζ	r	s	λ
Parallel	1	N	S	N	NS	NS	1
Sequential	NS	NS	S	NS	NS	NS	NS

6.3 NUMERICAL EXPERIMENTS

6.3.1 CONTROL ALGORITHMS

The driver behaviour data presented in Section 3.4 was used for simulation, however the start and end of each journey was trimmed so that each represented the exact same route (each of the journeys depicted in Figure 3.4 start and end at slightly different locations within the Johannes Kepler University Linz campus, hence the variations of journey length). For each journey, the state of charge of the battery was initialized at 60% and constrained between 40% and 100%. Two control algorithms were used:

1. **Nominal MPC.** The MPC controller defined by (MPC.2) was used where the controller had perfectly accurate predictions of future driver behaviour.

2. **Scenario MPC.** An MPC controller defined by $u(t) := \tau$ was used where τ was obtained from the solution of (MPC.4) using the ADMM algorithm described in Section 6.2. The method used for scenario generation is explained in detail in the following section.

6.3.2 SCENARIO GENERATION

A database was available to the controller where arrays of velocity, road gradient, and time data, indexed by the longitudinal position within the route, l, and sampled at a resolution of 1m, was available for all of the 49 journeys used for simulation. Before the start of each simulated journey, the position indexed data for $S = 48$ journeys (i.e. every journey *except* the one representing the actual journey to be simulated) was retrieved. The data used for simulation was indexed by time (as discussed in Chapter 3), and at each sample time $t \in \{0, \dots, T-1\}$ during a simulated journey, the scenarios were generated using the following steps:

- The longitudinal position of the vehicle was obtained from $l(t) = \sum_{i=0}^{t-1} v(t)$.

- For each of the $S = 48$ journeys that had been retrieved from the database, a scenario was generated by:
  - Re-sampling the position indexed velocity and road gradient data for the remaining simulation distance in terms of time at a rate of 1 Hz (given by $\hat{v}$ and $\hat{\theta}$).
  - Calculating the scenario road gradient and velocity values using the 'blending' coefficient $e^{-0.25k}$ from[2]

$$\begin{bmatrix}\theta_{k,s}\\ v_{k,s}\end{bmatrix} := \begin{bmatrix}\theta(t)\\ v(t)\end{bmatrix} e^{-0.25k} + \begin{bmatrix}\hat{\theta}_k\\ \hat{v}_k\end{bmatrix} (1 - e^{-0.25k}) \quad \forall k \in \{0, \dots, N-1\}. \tag{6.12}$$

- The parameters of problem (MPC.4) were generated, after which Algorithm 1 was used to determine if any of the scenarios were infeasible, and if so they were discarded.

The principle of this approach is that the predictions are made *entirely* using other data 'previously' recorded from the route, and that the controller has no additional information about the journey currently being used for simulation. The 'blending' process (6.12) was included as it was found that the near horizon predictions (i.e. $k \in \{0, \dots, 5\}$) were otherwise highly inaccurate due to the uncertainty in the velocity at any given point in the journey, as shown in Figure 3.4. An example of the predictions generated using the above process is illustrated in Figure 6.2. It can be observed in Figure 3.4 that whilst there is uncertainty in the velocity at any given point on the route, there is also some discernible structure: for example, there is a roundabout at the midpoint of the journey, and the velocity in all examples drops at this point. It is interesting to note that the long-horizon predictions presented in the top plot of Figure 6.2 do not retain this structure, and the predictions at the later time steps do not correlate with the simulated velocities at all. It can, however, also be seen that as the vehicle approaches certain points in the journey (corresponding to locations at which the data has reduced variability), the mid-horizon predictions begin to accurately approximate the simulated velocity. For example, it can be seen in the middle plot that the first $\sim$100s of the predictions accurately match the simulated profile.

[2]The acceleration values used in the longitudinal power model (3.4) were obtained from $\dot{v}_{k,s} := \dot{v}(t)e^{-0.25k} + \dot{\hat{v}}_k(1 - e^{-0.25k})$ for all $k \in \{0, \dots, N-1\}$, where $\dot{v}(t)$ is the vehicle's simulated acceleration at time t and $\dot{\hat{v}}_k$ is obtained from numerical differentiation of $\hat{v}$. The values of $v_{:,s}$ were not numerically differentiated directly as additional acceleration may have been introduced by (6.12), which would therefore distort the predicted power demand.

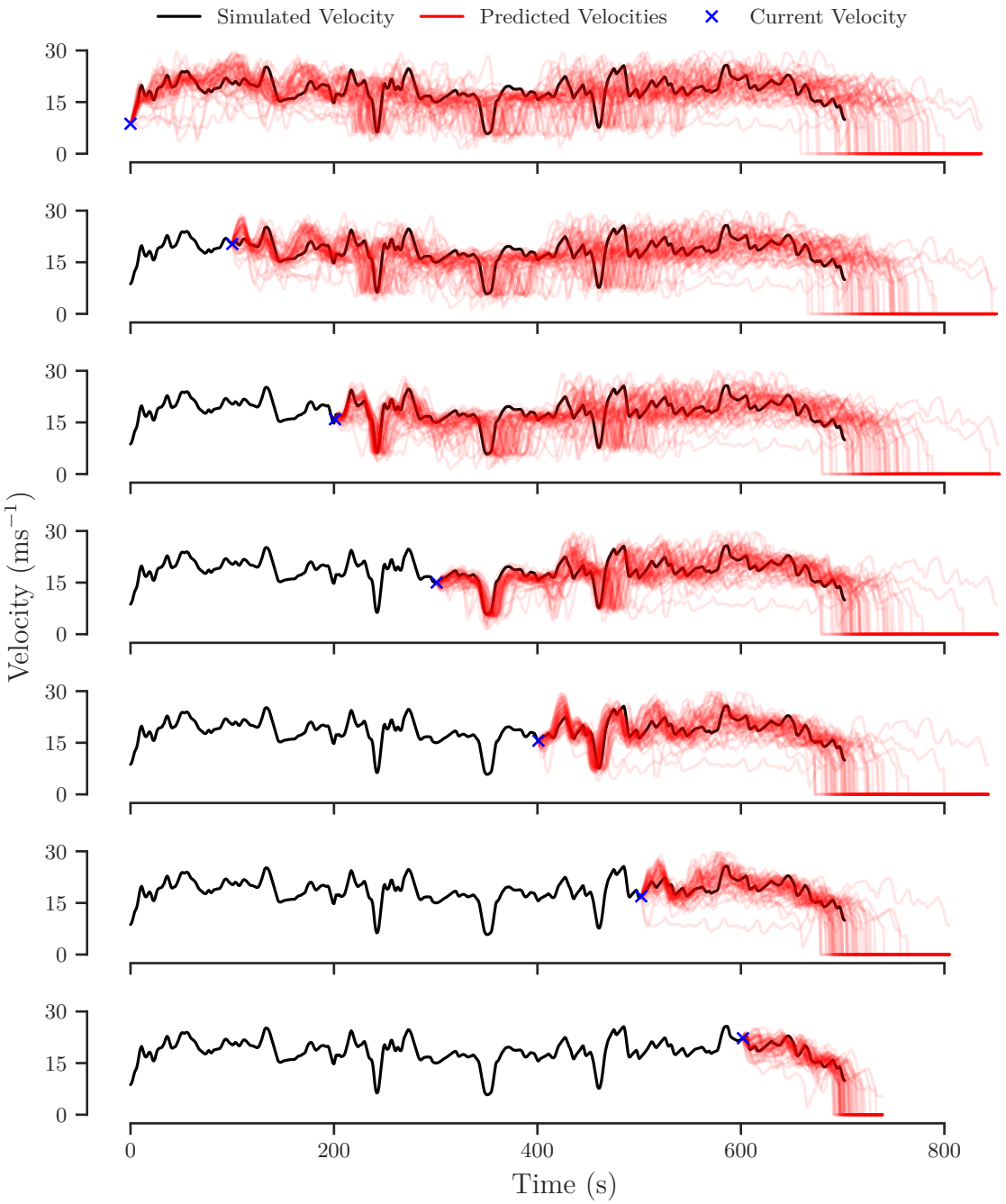


Figure 6.2: An example of the velocity predictions generated at seven instants during an individual journey.

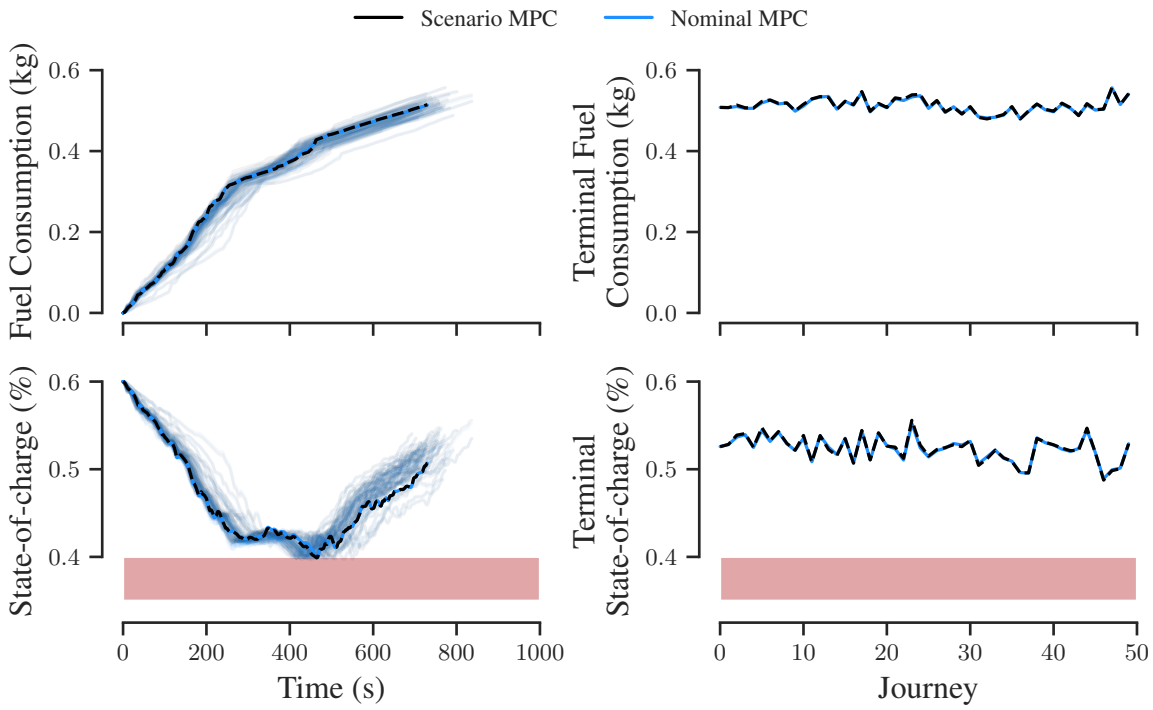


Figure 6.3: Closed loop results using nominal and scenario MPC.

6.3.3 RESULTS

Figure 6.3 shows the closed-loop simulation results using both nominal MPC controller and scenario MPC. The results are remarkable: the scenario MPC state-of-charge trajectories and fuel consumption closely match those obtained using nominal MPC, and the total fuel consumption is equal to within meaningful precision across all 49 journeys (the scenario MPC therefore also provides a close approximation of the globally optimal controls, as illustrated in Figure 4.7). This is made more remarkable by the fact that the driver behaviour is collected from 4 different drivers, which implies that modelling an individual driver's behaviour may not be necessary to achieve optimal performance. The explanation for this performance is that, although the optimal state-of-charge trajectories vary with respect to time, they match extremely closely with respect to distance. This is illustrated in Figure 6.4a, where the closed-loop state-of-charge trajectories from Figure 6.3 are resampled with respect to the completed distance of the journey, and can be seen to match closely during the first half of the journey when the car is driving uphill and drive power is required from the engine. The trajectories do also, however, diverge in the second half of the journey, but this is because the vehicle is descending a hill and is either braking

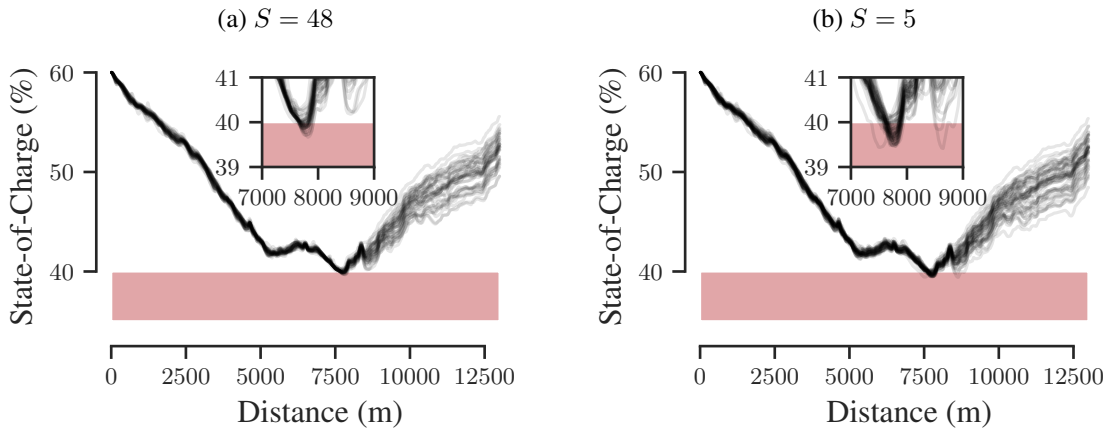


Figure 6.4: Closed loop results using scenario MPC re-sampled w.r.t completed distance. a) shows the results for $S = 48$, and b) shows the results for $S = 5$.

or coasting, so the energy that can be regenerated depends on losses such as aerodynamic drag. It is possible that a route that is not as evenly partitioned into an ascending phase and a descending phase may cause the scenario controller to behave more poorly. A systematic investigation into performance on different routes is left to future work.

The results in Figure 6.4a also show that the lower state-of-charge constraint is violated by up to $\sim 0.2\%$. The bound (6.7) provides a confidence of $\hat{\beta} \geq 0.92$ for a violation probability of $\hat{\epsilon} = 0.05$ when using $S = 48$ scenarios, so a portion of this constraint violation can be attributed to an insufficient number of predictions. This is supported by the results in Figure 6.4b, which show the results from 6.4a repeated with only $S = 5$ scenarios, and for which the constraint violation increases up to a maximum of $\sim 0.5\%$ (more scenarios could not be tested due to a lack of data). It is worth noting, however, that the bound (6.7) is dependent on the scenarios being sampled from the uncertainty set $\mathcal{V} \times \Theta$, and the one-step-ahead predictions obtained using the blending process (6.12) are not empirical. Therefore, it may also be possible to improve the performance by sampling the one-step-ahead predictions alone from a separate distribution, possibly conditioned on the current road gradient, and vehicle velocity and acceleration. Nonetheless, the observed constraint violation is likely to be lower than the uncertainty in the battery state estimate, so may not become an issue in practice.

Figure 6.5 shows the time taken for the first MPC optimization (i.e. the longest horizon) of each journey using both nominal and scenario MPC. The averages for nominal and scenario MPC were 0.098 s and 4.8 s, so the scenario approach does not meet the real-time requirement for the hardware used for these experiments. However, the ADMM iteration

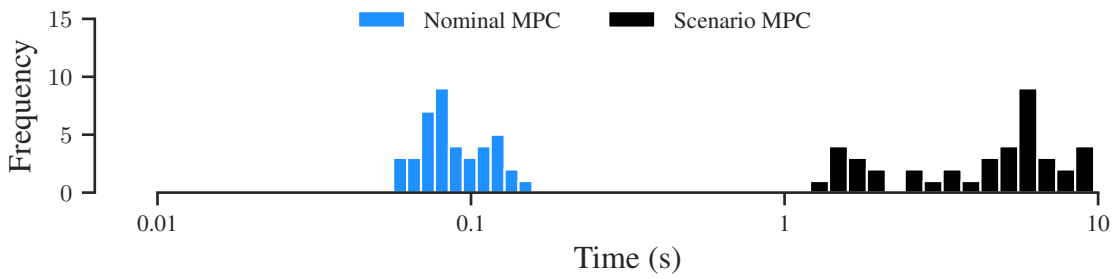


Figure 6.5: Histograms of time taken to solve the first MPC optimization for both nominal MPC and scenario MPC.

can be accelerated using parallel processing (as discussed in Section 6.2.1), and it has been demonstrated that the computational time of a similar ADMM algorithm can be significantly accelerated using a graphical processing unit (GPU) [134] (the algorithm was implemented in vectorized sequential Matlab code here; the Matlab gpuArray data type was investigated but not found to provide a performance benefit). Therefore, it is possible that the algorithm could be accelerated to meet the real-time requirement when implemented on suitable hardware. Additionally, the overall state-of-charge trajectories in Figure 6.4b closely match those in 6.4a, so it may be possible to accelerate the algorithm and reduce the occurrence of constraint violations without negatively affecting performance (in terms of increased fuel consumption) by using fewer scenarios for the long-horizon predictions, and more for the one-step-ahead predictions. A systematic investigation into these possibilities is left to future work.

6.4 Concluding Remarks

In this chapter, the MPC energy management framework from Chapter 4 was extended to consider multiple predictions of future driver behaviour at each control variable update instant. This allows the MPC to explicitly account for uncertainty by accessing previous examples of drivers completing a route, thereby removing the need to analytically model driver behaviour, which can be challenging. Additionally, it allows the use of results from scenario MPC to provide a confidence bound on a given probability of constraint violation (which in this case translates to one-step-ahead feasability). It was demonstrated that the ADMM algorithm considered in Chapter 4 can be extended to this case, and that the iteration cost scales linearly with the number of scenarios considered (and can, as before, be accelerated using parallel processing). A scenario generation method was proposed where predictions of future driver behaviour are generated from a database of previous

examples of a driver completing a route, and was simulated in comparison with a nominal MPC controller that had full route and velocity preview. It was found that the scenario MPC performance was indistinguishable from the nominal MPC, due to the fact that the optimal state-of-charge trajectories were in close agreement between journeys with respect to completed distance along the route. It was shown that the proposed ADMM algorithm required an average of 4 s to solve the optimization when implemented on the CPU used for these experiments, but that the algorithm may be accelerated to meet the real-time requirement using parallel processing hardware such as a GPU.

CHAPTER 7

BATTERY/SUPERCAPACITOR ELECTRIC VEHICLES

IN this chapter the optimization techniques used for the PHEV energy management problem are extended to all-electric vehicles where the energy storage system consists of both a conventional battery and a supercapacitor. The issues motivating the use of such a system were presented earlier in Chapter 1, but are re-stated here for the sake of readability.

A significant challenge facing the large scale adoption of electric vehicles is the high cost of lithium-ion batteries, which can reach 50% of the total cost of a vehicle [5, §3]. This limitation is exacerbated by the batteries' low power density and cycle life ($\leq$2000 W/kg and $\sim$2000 cycles [6, Table B1]), which imply that the battery may need to be sized above its total energy requirement in order to meet its power requirement, and also require replacement at a higher frequency than other powertrain components. A hybrid energy storage system consisting of a conventional battery and a supercapacitor can be used to reduce the effect of these limitations. The operational principle is that short term fluctuations and large spikes in power demand are delivered by the supercapacitor whilst the battery delivers power at a reduced and more constant level (see [135] for a review of battery/supercapacitor vehicle architectures). Supercapacitors are appropriate for this architecture as they have a power density of up to 23,500 W/kg and an effectively infinite cycle life [6, Table B1]).

One of the factors that determines the effectiveness of the hybrid storage system in reducing battery usage is the control system used to determine the power allocation between the battery and supercapacitor (see [11, §4] for a comprehensive review of control methods). A common rule-based approach uses a low pass filter to allocate the low frequency components of the power demand signal to the battery and the high frequency components to the supercapacitor [136, 137, 138]. A drawback of this approach is that it does not account for hardware constraints such as electrical storage capacity, so the vehicle will be forced to allocate charge/discharge demands to the battery during periods when the super-

capacitor is completely charged/discharged. Furthermore, it is unlikely that a rule-based controller is optimal (in the sense of minimizing energy consumption or battery degradation) even when hardware limits are not active.

Optimization-based control strategies aim to achieve the best possible power allocation for a given performance criterion. This approach typically has greater computational requirements than rule-based heuristics and assumes knowledge of the predicted power demand over a future horizon, but it is potentially much more effective. Dynamic programming can be used to determine the globally optimal control inputs for arbitrary cost functions, system dynamics, and hardware constraints. However, as demonstrated for the PHEV energy management problem, dynamic programming is computationally inefficient, and can typically only be used offline as a benchmark [139, 140, 137]. A possible use case of dynamic programming is to generate target data for tuning real-time control methods; a rule-based controller was tuned to dynamic programming results in [141], and an artificial neural network was trained on dynamic programming results in [142]. A real-time dynamic programming control strategy was obtained in [143], where the optimal state-feedback law obtained using stochastic dynamic programming was saved as a look-up table, but this approach has a potentially prohibitive memory requirement (12Gb).

Pontryagin's Minimum Principle was investigated in [137] and [144] to develop a real-time optimization-based controller, but these problem formulations considered very limited hardware constraints in the optimal control formulation. Hard constraint satisfaction can be addressed using MPC, and in [145, 140] and [146] the hybrid energy storage system was modelled as a linear system so that the MPC optimization problem reduced to a quadratic program, although the linear approximation of the system dynamics renders the obtained control inputs suboptimal in general.

In this chapter, the limitations of the aforementioned optimization-based control approaches are addressed using convex optimization. This approach allows the use of nonlinear models of powertrain losses and more general system dynamics than linear-quadratic MPC, whilst guaranteeing constraint satisfaction. Previous studies have also investigated convex optimization for this application: in [147] a real-time convex optimization based controller was presented, although constraints on battery power and energy were not considered, and only a one-step prediction horizon was used. A primal-dual interior point algorithm for convex optimization was also implemented in [148], but this did not optimize the power split in regenerative mode, the algorithm was only tested on a single drive cycle, and the computational performance of the algorithm was not reported. Additionally, combined sizing and control of supercapacitor-based hybrid energy storage systems was approached using convex optimization in [104] and [105], but the optimization problems were solved using general purpose optimization software that is not suitable for real-time, embedded control.

CONTRIBUTIONS

This chapter makes three novel contributions to the electric vehicle energy management problem using convex optimization.

1. A general convex optimization framework is presented for optimal electric vehicle power allocation. The framework considers losses in each of the battery, supercapacitor, and powertrain, and enforces hard constraints on instantaneous power delivered by the battery, supercapacitor, and powertrain, and total energy stored in both the battery and supercapacitor.

2. A computationally efficient ADMM algorithm is presented for the solution of the convex optimization problem. The algorithm exploits the structure of the problem in a similar way to the ADMM algorithms presented in previous chapters, so that the iteration and memory costs are $\mathcal{O}(T)$ (where T is the length of the journey). If parallel processing is available, the computation of a subset of the variable updates can be reduced to $\mathcal{O}(1)$.

3. A set of numerical experiments are presented in which the proposed ADMM algorithm is compared with a low-pass filter against an all-battery baseline on 49 examples of real driver behaviour. It is demonstrated that the convex formulation significantly reduces several metrics of battery use (Root-mean-square (RMS) battery power, peak battery power, total power throughput, and energy consumption) relative to the low-pass filter, and that the ADMM algorithm solves the convex optimization problem in an average of 0.38s (even for prediction horizons of up to 1003 samples), compared to an average of 63s using CVX.

7.1 PROBLEM FORMULATION

ENERGY & POWER

Figure 7.1 shows a simplified diagram of the power flows in the powertrain used to formulate the energy management problem, where it is assumed that the storage system is an active architecture in which the power delivered by the battery and supercapacitor can be controlled individually (a discussion of the power electronics required for this is presented in [135]). The powertrain is modelled in discrete time with an assumed sampling interval of 1 s, so that $t \in \{0, \dots, T\}$, where $T \in Z_{++}$ is the duration of the journey under consideration (although the methods presented here can be readily extended to an arbitrary sampling interval). The variable $u := (u_0, \dots, u_{T-1}) \in \mathbb{R}^T$ represents the rate of change

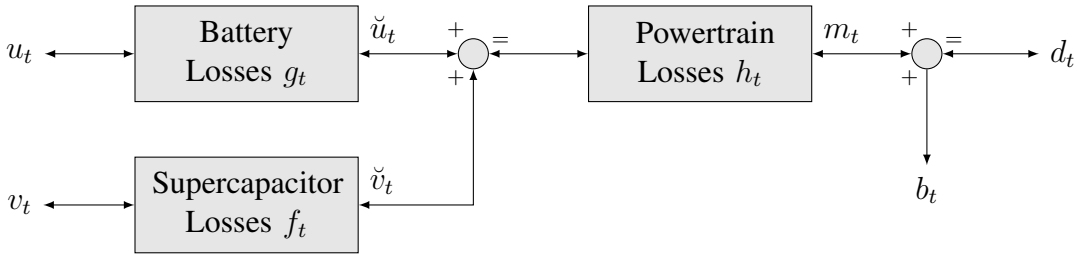


Figure 7.1: Power flow diagram for electric vehicle with hybrid energy storage system.

of the internal energy of the battery, and by assuming that the power values are constant between sampling intervals, the energy stored in the battery, $x := (x_1, \ldots, x_T) \in \mathbb{R}^T$, is given by

$$x_t := x_0 - \sum_{i=0}^{t-1} u_i \quad \text{for} \quad t \in \{1, \ldots, T\},$$

where $x_0 \in \mathbb{R}$ is the battery's internal energy at the start of the journey. Similarly, the variable $v := (v_0, \ldots, v_{T-1}) \in \mathbb{R}^T$ represents the rate of change of internal energy in the supercapacitor, so the energy stored in the supercapacitor, $y := (y_1, \ldots, y_T) \in \mathbb{R}^T$, is given by

$$y_t := y_0 - \sum_{i=0}^{t-1} v_i \quad \text{for} \quad t \in \{1, \ldots, T\},$$

where $y_0 \in \mathbb{R}$ is the supercapacitor's internal energy at the start of the journey. The variable $\breve{u} := (\breve{u}_0, \ldots, \breve{u}_{T-1}) \in \mathbb{R}^T$ represents the electrical power delivered by the battery after losses, $\breve{v} := (\breve{v}_0, \ldots, \breve{v}_{T-1}) \in \mathbb{R}^T$ is the electrical power delivered by the supercapacitor after losses, $m := (m_0, \ldots m_{T-1}) \in \mathbb{R}^T$ is the mechanical power delivered by the powertrain to the wheels, $b := (b_0, \ldots, b_{T-1}) \in \mathbb{R}_+^T$ is the mechanical braking power, and $d := (d_0, \ldots, d_{T-1})$ represents the power demanded by the driver throughout the journey.

POWERTRAIN LOSSES

Battery losses are represented by the time-varying function g_t, while powertrain losses are modelled with the time varying function h_t, and the supercapacitor losses are modelled with the time varying function f_t. It is assumed that the losses in the battery and motor are independent of states such as temperature and state of charge; this assumption is also implicitly made in the predictive model used for the PHEV problem in the previous chapters, and is extended to the simulation model in this chapter because the same model is used for both simulation and prediction. Under the above assumptions, the electrical power

delivered by the battery is given by $\breve{u}_t := g_t(u_t)\ \forall t \in \{0,\dots,T-1\}$ and the electrical power delivered by the supercapacitor is $\breve{v}_t := f_t(v_t)\ \forall t \in \{0,\dots,T-1\}$. These are then combined additively and delivered to the powertrain so that $m_t := h_t(g_t(u_t)+f_t(v_t))\ \forall t \in \{0,\dots,T-1\}$. Finally, the power delivered by the powertrain is combined additively with the power from the brakes to meet the driver demand power

$$d_t := b_t + h_t(g_t(u_t) + f_t(v_t)) \quad \forall t \in \{0,\dots,T-1\}. \tag{7.1}$$

Assumption 7.1.1. *For all $t \in \{0,\dots,T-1\}$, $f_t(\cdot)$, $g_t(\cdot)$, and $h_t(\cdot)$ are strictly increasing functions of their arguments.*

Assumption 7.1.2. *For all $t \in \{0,\dots,T-1\}$, $f_t(\cdot)$ and $g_t(\cdot)$ are concave functions of their arguments.*

Assumptions 7.1.1 and 7.1.2 are justified by a physical interpretation of a loss function: it would be expected that an increase in output power would require an increase in input power, which implies Assumption 7.1.1, and it would be expected that the losses would increase as the magnitude of the input/output power increases, which implies Assumption 7.1.2. In order for the modelled system to be valid, it is also required that $f_t(x) \leq x$, $g_t(x) \leq x$, and $h_t(x) \leq x\ \forall x,t$ (i.e. the system cannot create energy), but this condition is not explicitly required to obtain a convex formulation.

Assumption 7.1.3. *For all $t \in \{0,\dots,T-1\}$, $f_t(\cdot)$, $g_t(\cdot)$, and $h_t(\cdot)$ are surjective functions.*

Assumption 7.1.3 is a technicality required for Assumption 7.1.1 to imply that $f_t(\cdot)$, $g_t(\cdot)$, and $h_t(\cdot)$ are bijective $\forall t \in \{0,\dots,T-1\}$, which therefore implies that $f_t^{-1}(\cdot)$, $g_t^{-1}(\cdot)$, and $h_t^{-1}(\cdot)$ exist $\forall t \in \{0,\dots,T-1\}$.

Note that whilst the focus of this chapter is on battery/supercapacitor electric vehicles, the proposed approach can be used for any hybrid energy storage system in which the loss functions satisfy Assumptions 7.1.1-7.1.3.

7.1.1 Optimal Control Problem

The power delivered by the battery, power delivered by the supercapacitor, total electrical power delivered to the powertrain, total battery energy, and total supercapacitor energy are all subject to upper and lower bounds, and the braking power is constrained to be nonpositive:

$$\underline{u}_t \leq u_t \leq \overline{u}_t, \quad \underline{v}_t \leq v_t \leq \overline{v}_t, \quad \underline{e}_t \leq g_t(u_t) + f_t(v_t) \leq \overline{e}_t,$$
$$\underline{x}_t \leq x_t \leq \overline{x}_t, \quad \underline{y}_t \leq y_t \leq \overline{y}_t, \quad b_t \leq 0,$$

$\forall t \in \{0, \ldots, T-1\}$. The cost function

$$\sum_{t=0}^{T-1} [u_t + v_t - h_t(g_t(u_t) + f_t(v_t)) - b_t]$$

represents the total energy loss in the system (although any convex function of u, v, x, and/or y can be used in the proposed framework). This cost represents the sum over T steps of the braking energy, $-b_t$, plus the difference between the combined energy delivered by the battery and supercapacitor, $(u_t + v_t)$, and the total energy delivered by the powertrain, $h_t(g_t(u_t) + f_t(v_t))$. Therefore, the optimal control sequences, denoted $(u, v, b)^\star$, are obtained as the minimizing argument of

$$\begin{aligned}
\min_{(u,v,b)} \ & \sum_{t=0}^{T-1} [u_t + v_t - h_t(g_t(u_t) + f_t(v_t)) - b_t] \\
\text{s.t.} \ & \left.\begin{aligned}
& d_t = b_t + h_t(g_t(u_t) + f_t(v_t)) \\
& b_t \leq 0 \\
& \underline{u}_t \leq u_t \leq \overline{u}_t \\
& \underline{v}_t \leq v_t \leq \overline{v}_t \\
& \underline{e}_t \leq g_t(u_t) + f_t(v_t) \leq \overline{e}_t \\
& x_{t+1} = x_0 - \textstyle\sum_{i=0}^{t} u_i \\
& \underline{x}_{t+1} \leq x_{t+1} \leq \overline{x}_{t+1} \\
& y_{t+1} = y_0 - \textstyle\sum_{i=0}^{t} v_i \\
& \underline{y}_{t+1} \leq y_{t+1} \leq \overline{y}_{t+1}
\end{aligned} \right\} \forall t \in \{0, \ldots, T-1\}.
\end{aligned} \tag{7.2}$$

Problem (7.2) is generally nonconvex if any of $f_t(\cdot)$, $g_t(\cdot)$, or $h_t(\cdot)$ are nonlinear for any $t \in \{0, \ldots, T-1\}$.

7.1.2 CONVEX FORMULATION

The constraint $b_t \leq 0 \ \forall t \in \{0, \ldots, T-1\}$ can be combined with (7.1) to obtain $d_t \leq h_t(g_t(u_t) + f_t(v_t))$, which under Assumption 7.1.1 is equivalent to

$$h_t^{-1}(d_t) \leq g_t(u_t) + f_t(v_t).$$

This is combined with the constraint $\underline{e}_t \leq g_t(u_t) + f_t(v_t)$ as

$$\hat{e}_t \leq g_t(u_t) + f_t(v_t)$$

where $\hat{e}_t := \max\{\underline{e}_t, h_t^{-1}(d_t)\}$. The set $\{(u_t, v_t) \in \mathbb{R}^2 : \hat{e}_t \leq g_t(u_t) + f_t(v_t)\}$ is convex under Assumption 7.1.2. The constraint $g_t(u_t) + f_t(v_t) \leq \overline{e}_t$ is in general nonconvex, and is linearly approximated by

$$\hat{g}_t(u_t) + \hat{f}_t(v_t) \leq \overline{e}_t, \quad \hat{g}_t(u_t) := g_t(\hat{u}_t) + g_t'(\hat{u}_t)(u_t - \hat{u}_t),$$
$$\hat{f}_t(v_t) := f_t(\hat{v}_t) + f_t'(\hat{v}_t)(v_t - \hat{v}_t),$$

where $g_t'(\cdot)$ and $f_t'(\cdot)$ are the derivatives of $g_t(\cdot)$ and $f_t(\cdot)$[1], and $\hat{u}_t \in \mathbb{R}$ and $\hat{v}_t \in \mathbb{R}$ are fixed linearization points[2] chosen such that $g_t'(\hat{u}_t) \neq 0$ and $f_t(\hat{v}_t) \neq 0$. The concavity of g_t and f_t (Assumption 7.1.2) implies that $\{(v_t, u_t) \in \mathbb{R}^2 : \hat{g}_t(u_t) + \hat{f}_t(v_t) \leq \overline{e}_t\} \subseteq \{(v_t, u_t) \in \mathbb{R}^2 : g_t(u_t) + f_t(v_t) \leq \overline{e}_t\}$, so the linear approximation guarantees that the original constraint is enforced, and the set $\{(v_t, u_t) \in \mathbb{R}^2 : \hat{g}_t(u_t) + \hat{f}_t(v_t) \leq \overline{e}_t\}$ is convex. The approximation does also, however, potentially introduce some conservatism to the solution, but note that $g_t(u_t) + f_t(v_t) \leq \overline{e}_t$ is the upper bound on powertrain power (typically imposed by the torque limits of the powertrain components). Assuming that the driver power demand does not require this upper bound to be exceeded (i.e. that d_t is such that $h_t^{-1}(d_t) \leq \overline{e}_t$ for all t), the constraint $g_t(u_t) + f_t(v_t) \leq \overline{e}_t$ can only be active if the mechanical brakes are active (i.e. $b_t < 0$) while the battery and supercapacitor deliver power in excess of the demand power. Given that the cost function penalizes energy loss, it is unlikely that this constraint is active at the optimal solution of (7.2), although it is in principle possible in particular operating conditions (the issue is discussed further in Appendix 7.A).

Using (7.1), the objective of (7.2) can be simplified:

$$u_t + v_t - h_t(g_t(u_t) + f_t(v_t)) - b_t = u_t + v_t - d_t,$$

where d is independent of the decision variables. Problem (7.2) can therefore be approximated by the convex optimization problem

$$\begin{aligned}
&\min_{(u,v)} \ \mathbf{1}^\top (u + v) \\
&\text{s.t. } x = \mathbf{1}x_0 - \Psi u \quad x \in \mathcal{X}, \\
&\quad y = \mathbf{1}y_0 - \Psi v \quad y \in \mathcal{Y}, \\
&\quad \left.\begin{array}{l} (u_t, v_t) \in \mathcal{C}_t \\ u_t \in \mathcal{U}_t \\ v_t \in \mathcal{V}_t \end{array}\right\} \ \forall t \in \{0, \ldots, T-1\},
\end{aligned} \tag{MPC.5}$$

[1] if $f_t(\cdot)$ or $g_t(\cdot)$ are nondifferentiable, then any non-zero element from their subdifferentials can be used in place of $f_t'(\cdot)$ and $g_t'(\cdot)$.

[2] The choice of $\hat{u}_t$ and $\hat{v}_t$ is discussed further in Remark 7.C.1 in Appendix 7.C.

where Ψ is a $T \times T$ lower triangular matrix of ones, and

$$\begin{aligned}
\mathcal{C}_t &:= \{(u_t, v_t) \in \mathbb{R}^2 : \hat{e}_t \leq g_t(u_t) + f_t(v_t), \\
&\qquad \hat{g}_t(u_t) + \hat{f}_t(v_t) \leq \overline{e}_t\}, \\
\mathcal{U}_t &:= \{u_t \in \mathbb{R} : \underline{u}_t \leq u_t, \leq \overline{u}_t\}, \\
\mathcal{V}_t &:= \{v_t \in \mathbb{R} : \underline{v}_t \leq v_t, \leq \overline{v}_t\}, \\
\mathcal{X} &:= \{x \in \mathbb{R}^T : \underline{x}_t \leq x_t, \leq \overline{x}_t \; \forall t \in \{1, \ldots, T\}\}, \\
\mathcal{Y} &:= \{y \in \mathbb{R}^T : \underline{y}_t \leq y_t, \leq \overline{y}_t \; \forall t \in \{1, \ldots, T\}\}.
\end{aligned}$$

The optimal braking power sequence, $b^\star$, is then obtained from $b_t^\star = d_t - h_t(g_t(u_t^\star) + f_t(v_t^\star)) \; \forall t \in \{0, \ldots, T-1\}$, where $u^\star$ and $v^\star$ are the minimizing arguments of (MPC.5).

7.2 OPTIMIZATION ALGORITHM

Problem (MPC.5) can be solved using general purpose convex optimization software (e.g. CVX), but these tools are typically computationally intensive. In this section an ADMM algorithm is proposed that is tailored to the structure of (MPC.5).

7.2.1 ALGORITHM

Problem (MPC.5) is equivalent to the equality constrained problem

$$\begin{aligned}
\min_{(u,v)} \; & \mathbf{1}^\top (u + v) + \sum_{t=0}^{T-1} \left[\mathcal{I}_{\mathcal{C}_t}(u_t, v_t) + \mathcal{I}_{\mathcal{U}_t}(u_t) + \mathcal{I}_{\mathcal{V}_t}(v_t)\right] \\
& + \mathcal{I}_{\mathcal{X}}(x) + \mathcal{I}_{\mathcal{Y}}(y), \\
\text{s.t. } & u = \zeta, \\
& v = \eta, \\
& x = \mathbf{1}x_0 - \Psi\zeta, \\
& y = \mathbf{1}y_0 - \Psi\eta,
\end{aligned} \tag{7.3}$$

where $\zeta \in \mathbb{R}^T$ and $\eta \in \mathbb{R}^T$ are vectors of dummy variables. Problem (7.3) is in turn the equivalent of

$$\underset{(u,v)}{\text{argmin}} \; \tilde{f}(\tilde{u}) \quad \text{s.t. } A\tilde{u} + B\tilde{x} = c, \tag{7.4}$$

where

$$\tilde{u} := (u, v, x, y), \quad \tilde{x} := (\zeta, \eta),$$
$$\tilde{f}(\tilde{u}) := \mathbf{1}^\top (u + v) + \sum_{t=1}^{T-1} [\mathcal{I}_{\mathcal{C}_t}(u_t, v_t) + \mathcal{I}_{\mathcal{U}_t}(u_t) + \mathcal{I}_{\mathcal{V}_t}(v_t)] + \mathcal{I}_{\mathcal{X}}(x) + \mathcal{I}_{\mathcal{Y}}(y),$$
$$A := \begin{bmatrix} I & I & & \\ & & I & \end{bmatrix}, \quad B := \begin{bmatrix} -I & \\ \Psi & -I \\ & \Psi \end{bmatrix}, \quad c := (\mathbf{0}, \mathbf{0}, \mathbf{1}x_0, \mathbf{1}y_0).$$

The augmented Lagrangian function for (7.4) is

$$\mathcal{L}(\tilde{u}, \tilde{x}, \lambda) := \tilde{f}(\tilde{u}) + \frac{1}{2} \|A\tilde{u} + B\tilde{x} - c + \lambda\|_\rho^2 \tag{7.5}$$

where

$$\|x\|_\rho^2 := x^\top \rho x, \quad \rho := \text{diag}(\rho_1 \mathbf{1}, \rho_2 \mathbf{1}, \rho_3 \mathbf{1}, \rho_4 \mathbf{1}), \quad \lambda := (\lambda_1, \lambda_2, \lambda_3, \lambda_4),$$
$$\rho_i \in \mathbb{R}_+ \; \forall i \in \{1, 2, 3, 4\}, \quad \lambda_i \in \mathbb{R}^T \; \forall i \in \{1, 2, 3, 4\}.$$

for which the ADMM iteration is defined by

$$\tilde{u}^{(j+1)} := \underset{\tilde{u}}{\text{argmin}} \; \mathcal{L}(\tilde{u}, \tilde{x}^j, \lambda^j), \tag{7.6a}$$
$$\tilde{x}^{(j+1)} := \underset{\tilde{x}}{\text{argmin}} \; \mathcal{L}(\tilde{u}^{(j+1)}, \tilde{x}, \lambda^j), \tag{7.6b}$$
$$\lambda^{(j+1)} := \lambda^j + A\tilde{u}^{(j+1)} + B\tilde{x}^{(j+1)} - c. \tag{7.6c}$$

(note that (7.5) is strongly convex in $\tilde{u}$ and $\tilde{x}$ so the updates (7.6a) and (7.6b) exist and are unique). The algorithm is initialized with the values

$$\hat{u}^{(0)} = \mathbf{0}, \quad \hat{x}^{(0)} = \mathbf{0}, \quad \lambda^{(0)} = \mathbf{0},$$

and terminated when the criterion

$$\max\{\|r^{(j+1)}\|, \|s^{(j+1)}\|\} \leq \epsilon \tag{7.7}$$

is met, where $\epsilon \in \mathbb{R}_+$ is a pre-determined convergence threshold and

$$r^{(j+1)} := A\hat{u}^{(j+1)} + B\hat{x}^{(j+1)} - c,$$
$$s^{(j+1)} := A^\top \rho B(\hat{x}^{(j+1)} - \hat{x}^{(j)}).$$

The convergence results for this algorithm are discussed extensively in Section (5.A), and global convergence is guaranteed due to the convexity of (MPC.5).

7.2.2 VARIABLE UPDATES & ALGORITHM COMPLEXITY

Firstly, note that the augmented Lagrangian (7.5) is equivalent to

$$\begin{aligned}\mathcal{L}(\tilde{u},\tilde{x},\lambda) := \mathbf{1}^\top(u+v) + \sum_{t=1}^{T-1}[\mathcal{I}_{\mathcal{C}_t}(u_t,v_t) + \mathcal{I}_{\mathcal{U}_t}(u_t) + \mathcal{I}_{\mathcal{V}_t}(v_t)] \\ + \mathcal{I}_{\mathcal{X}}(x) + \mathcal{I}_{\mathcal{Y}}(y) + \frac{\rho_1}{2}\|u-\zeta+\lambda_1\|_2^2 + \frac{\rho_2}{2}\|v-\eta+\lambda_2\|_2^2 \\ + \frac{\rho_3}{2}\|x+\Psi\zeta-\mathbf{1}x_0+\lambda_3\|_2^2 + \frac{\rho_4}{2}\|y+\Psi\eta-\mathbf{1}y_0+\lambda_4\|_2^2\end{aligned}$$

Therefore, update (7.6a) is equivalent to

$$\begin{aligned}(u_t,v_t)^{(j+1)} := \underset{(u_t,v_t)}{\operatorname{argmin}}\Big[&\mathcal{I}_{\mathcal{C}_t}(u_t,v_t) + \mathcal{I}_{\mathcal{U}_t}(u_t) + \mathcal{I}_{\mathcal{V}_t}(v_t) + \\ &\frac{\rho_1}{2}(u_t-\zeta_t^{(j)}+\lambda_{1,t}^{(j)})^2 + \frac{\rho_2}{2}(v_t-\eta_t^{(j)}+\lambda_{2,t}^{(j)})^2\Big] \qquad (7.8a)\end{aligned}$$

$$x^{(j+1)} := \Pi_{\mathcal{X}}\left[\mathbf{1}x_0 - \Psi\zeta^{(j)} - \lambda_3^{(j)}\right], \qquad (7.8b)$$

$$y^{(j+1)} := \Pi_{\mathcal{Y}}\left[\mathbf{1}y_0 - \Psi\eta^{(j)} - \lambda_4^{(j)}\right]. \qquad (7.8c)$$

The combined (u_t, v_t) update in (7.8a) is a convex optimization problem subject to inequality constraints, and a method is presented in Appendix 7.C for solving this problem from a finite set of candidate solutions, thereby avoiding the use of a general inequality constrained convex optimization algorithm (e.g. interior-point algorithm), which could increase the computational complexity of the ADMM algorithm as a whole. The update for (u, v) is separable w.r.t. each element (u_t, v_t), and therefore its computational complexity scales linearly with T when each update is performed sequentially for $t \in \{0, \dots, T-1\}$, or is independent of T when they are performed in parallel. The computation of the argument of the projection in (7.8b) scales linearly with T because multiplication by Ψ is the equivalent to a cumulative sum (and has no memory requirement). The projection $\Pi_{\mathcal{X}}$ can then be performed element-wise, so (7.8b) scales linearly with T overall, and the same is true for $y^{(j+1)}$.

Update (7.6b) is equivalent to

$$\zeta^{(j+1)} := (\rho_1 I + \rho_3\Psi^\top\Psi)^{-1}\left[\rho_1(u^{(j+1)} + \lambda_1^{(j)}) - \rho_3\Psi^\top(x^{(j+1)} - \mathbf{1}x_0 + \lambda_3^{(j)})\right] \qquad (7.9a)$$

$$\eta^{(j+1)} := (\rho_2 I + \rho_4\Psi^\top\Psi)^{-1}\left[\rho_2(v^{(j+1)} + \lambda_2^{(j)}) - \rho_4\Psi^\top(y^{(j+1)} - \mathbf{1}y_0 + \lambda_4^{(j)})\right] \qquad (7.9b)$$

where equations (7.9a) and (7.9b) are the solutions of the general system of linear equations $(kI + \Psi^\top\Psi)\mathbf{x} = \mathbf{b}$. In Proposition 4.A.1 it is demonstrated that these solutions can

Table 7.1: Summary of the complexity of each of the ADMM variable updates w.r.t. horizon length T.

$\mathcal{O}(\cdot)$	(u, v)	x	y	ζ	η	r	s	λ
Parallel	1	T	T	T	T	T	T	1
Sequential	T	T	T	T	T	T	T	T

be obtained with $\mathcal{O}(T)$ computation and memory requirement. The residual updates are the equivalent of

$$r^{(j+1)} = \begin{bmatrix} u^{(j+1)} - \zeta^{(j+1)} \\ v^{(j+1)} - \eta^{(j+1)} \\ x^{(j+1)} + \Psi\zeta^{(j+1)} - \mathbf{1}x_0 \\ y^{(j+1)} + \Psi\eta^{(j+1)} - \mathbf{1}y_0 \end{bmatrix} \quad s^{(j+1)} = \begin{bmatrix} \rho_1(\zeta^{(j)} - \zeta^{(j+1)}) \\ \rho_2(\eta^{(j)} - \eta^{(j+1)}) \\ \rho_3\Psi(\zeta^{(j+1)} - \zeta^{(j)}) \\ \rho_4\Psi(\eta^{(j+1)} - \eta^{(j)}) \end{bmatrix}$$

which scale linearly with T and have no additional memory requirement (multiplication by Ψ is the equivalent of a cumulative sum). Finally, update (7.6c) is equivalent to

$$\lambda^{(j+1)} := \lambda^{(j)} + r^{(j+1)}.$$

The overall scaling properties w.r.t. horizon T of the computation of each iteration are summarised in Table 1, and the memory requirement of the algorithm is $\mathcal{O}(T)$, as only two bandwidth 2 matrices (defined in Proposition 4.A.1) and the variables themselves require storage.

7.3 NUMERICAL EXPERIMENTS

The approach to the numerical experiments presented here is a departure from the approach used in earlier chapters for the PHEV energy management problem. The focus is on demonstrating that, under similar model approximations to those proposed in the previous chapters, the energy management problem for electric vehicles with a hybrid energy storage system can be reduced to a convex optimization problem that can be solved rapidly. Moreover, the battery degradation mechanisms of interest (e.g. lithium plating) are not realized in the models used for PHEV simulation, and so are of limited use in terms of ascertaining controller performance. Therefore, only the approximate model is considered for both prediction and simulation, and an investigation into higher fidelity simulation models (in particular, detailed battery ageing models) is left for future work.

VEHICLE & DRIVER MODEL

The 49 driver velocity and road gradient trajectories used in previous chapters (and presented in Section 3.4) were used again for simulation here, from which the driver power demand was also calculated using the longitudinal model (3.4).

POWERTRAIN MODEL

The vehicle was modelled with a single-speed transmission so that the rotational speed of the motor, $\omega_m \in \mathbb{R}^T$, was calculated at each timestep from $\omega_{m,t} = \frac{v_t}{r_w r_d}\ \forall t \in \{0, \dots, T-1\}$, where $r_w \in \mathbb{R}$ is the effective radius of the wheel and $r_d \in \mathbb{R}$ is the final drive ratio of the transmission. The losses in the motor were modelled as a quadratic function mapping the output power, m_t, to motor input power, $g_t(u_t) + f_t(v_t)$, with parameters dependent on the motor speed, $\omega_{m,t}$:

$$g_t(u_t) + f_t(v_t) := \beta_2(\omega_{m,t}) m_t^2 + \beta_1(\omega_{m,t}) m_t + \beta_0(\omega_{m,t})$$

where $\beta_2(\omega_{m,t}) > 0\ \forall \omega_{m,t}$ (i.e. the predictive model used for PHEV energy management detailed in Section 3.2 was used for both simulation and prediction). It was assumed that all drivetrain components other than the motor were 100% efficient, so that the inverse powertrain losses were modelled using the sampled coefficients, $\beta_{i,t} = \beta_i(\omega_{m,t})\ \forall i \in \{0, 1, 2\}$, for each sample of $\omega_{m,t}$ as

$$h_t^{-1}(m_t) := \beta_{2,t} m_t^2 + \beta_{1,t} m_t + \beta_{0,t} \quad \forall t \in \{0, \dots, T-1\},$$

which is invertible on the domain $\left[-\frac{\beta_{1,k}}{2\beta_{2,k}}, \infty\right]$ and range $\left[\beta_{0,k} - \frac{\beta_{1,k}^2}{4\beta_{2,k}}, \infty\right]$. As a result the powertrain loss function,

$$h_t(x) := \frac{-\beta_{1,t} + \sqrt{\beta_{1,t}^2 - 4\beta_{2,t}(\beta_{0,t} - x)}}{2\beta_{2,t}},$$

satisfies assumptions 7.1.1 and 7.1.3. The motor was also subject to upper and lower bounds on power due to its torque limits, so the overall power limits were

$$\underline{e}_k := \max\left\{\beta_{0,k} - \frac{\beta_{1,k}^2}{4\beta_{2,k}}, \underline{T}\omega_{m,k}\right\}, \quad \overline{e}_k := \overline{T}\omega_{m,k},$$

where $\underline{T}$ and $\overline{T}$ are lower and upper torque limits (set at $\pm$250 Nm).

BATTERY & MODEL

The battery was modelled as an equivalent circuit with fixed internal resistance R (0.1 Ω) and open circuit voltage V (300 V), so that

$$g_t(u_t) := \frac{V^2 - (V - \frac{2Ru_t}{V})^2}{4R} \quad \forall t \in \{0, \dots, T-1\},$$

which satisfies Assumptions 7.1.1, 7.1.2, and 7.1.3 on the domain $\left[-\infty, \frac{V^2}{2R}\right]$ and range $\left[-\infty, \frac{V^2}{4R}\right]$. The linearization point was set at $\hat{u}_t = 0 \; \forall t \in \{0, \dots, T-1\}$.

The battery was subject to upper and lower bounds on power, $\overline{P}$ and $\underline{P}$ (set at $\pm$70 kW), so that

$$\underline{u}_t := \underline{P} \quad \overline{u}_t := \min\left\{\overline{P}, \frac{V^2}{2R}\right\}$$

$\forall t \in \{0, \dots, T-1\}$, and the battery upper limit was set to 22 kWh (80 MJ) to model a typical electric vehicle battery capacity (e.g. 2015 Renault Fluence ZE [149, Table 2]). It was assumed that the battery's power electronics (e.g. DC-DC converter) were 100% efficient.

SUPERCAPACITOR MODEL

It was assumed that the supercapacitor and associated power electronics were 100% efficient so that

$$f_t(v_t) = v_t \quad \forall t \in \{0, \dots, T-1\},$$

which trivially satisfies Assumptions 7.1.1, 7.1.2, and 7.1.3 (and does not require linearization, so the choice of $\hat{v}_t$ is arbitrary). The power limits were infinite so that $\underline{v}_t = -\infty$ and $\overline{v}_t = \infty \; \forall t \in \{0, \dots, T-1\}$. There are currently no commercially available battery/supercapacitor electric vehicles, so the supercapacitor energy limit was set at 300 Wh (1.08 MJ) to reflect the parameters used in similar studies (441.5 Wh was used in [150, Table 2]; 203 Wh in [142, §II.B.]; and 0.8 MJ in [143, Table IV]).

ADMM PARAMETERS

The ρ parameters detailed in Section 7.2.1 require tuning; a two-dimensional parameter search similar to that detailed in Section 4.3.1 was used to optimize the parameters ρ_1 and ρ_3, with $\rho_2 = \rho_1$ and $\rho_4 = \rho_3$, as these parameters correspond to constraints of similar magnitude. The chosen values were $\rho_1 = 5 \times 10^{-5}$ and $\rho_3 = 1 \times 10^{-8}$, which were

used for all simulations (the results in [20] and [22] suggest that the ρ values are tuned to hardware characteristics, and fixed values perform well across a diversity of drive cycles).

The termination criterion (7.7) enforces an upper bound on a measure of constraint violation ($\|r^{(j+1)}\|$) and sub-optimality ($\|s^{(j+1)}\|$). In the experiments presented here the termination criterion $\epsilon = 100$ was used, which can be loosely interpreted as a 0.1 % upper bound on solution error (as u and v took values of the order of magnitude 10^5). The solutions obtained using CVX and ADMM were indistinguishable using this criterion.

CONTROL ALGORITHMS

Each of the 49 journeys were simulated using three alternative power allocation methods:

1. **All-battery** - all positive (and negative power) was delivered from (to) the battery, unless the upper bound on the battery energy was active and the power demand was negative, in which case the excess power was delivered by the brakes.

2. **Low-pass Filter** - A first-order low-pass filter with a bandwidth of 0.01 Hz was used to separate the power demand frequencies. The filtered signal was allocated to the battery, and the remaining power demand was allocated to the supercapacitor, unless the supercapacitor limits were active, in which case the excess power was also delivered by the battery.

3. **Optimal** - The battery and supercapacitor were controlled using the optimal controls obtained from the solution of (MPC.5) in open-loop (no uncertainty was modelled in the driver behaviour predictions or vehicle model so the full-horizon open-loop control exactly matches the shrinking horizon MPC). The solution was obtained for each journey using both ADMM and CVX to determine the relative computational performance. The ADMM algorithm was programmed in Matlab, the default solver and tolerance was used for CVX, and a 2.60GHz Intel Core i7-9750H CPU was used for both.

7.3.1 RESULTS

Figure 7.2 shows the battery power, supercapacitor power, battery energy, and supercapacitor energy obtained using each of the algorithms detailed in Section 7.3 for a single journey. Qualitatively, it can be seen that the low-pass filter reduces the amplitude and frequency of the peaks in the battery control signal relative to the all-battery controller, whilst the optimal controls obtained from the solution of (7.3) result in a piecewise constant battery control signal (this characteristic of the solution is explained further in Appendix 7.B). The optimal solution suggests that it is challenging to approximate the optimal controls

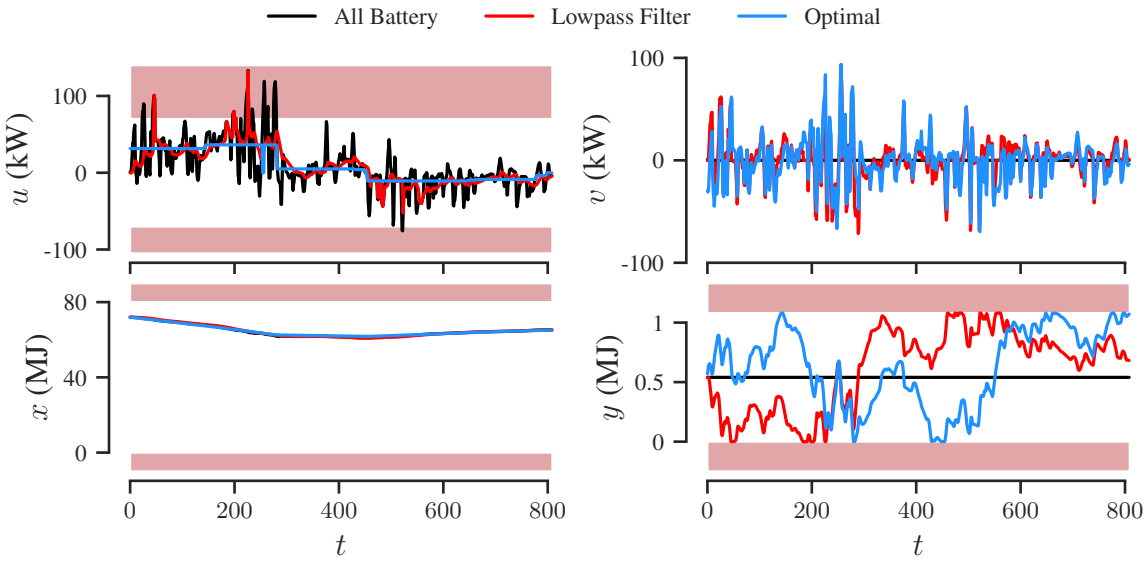


Figure 7.2: Trajectories for battery power, supercapacitor power, battery energy, and supercapacitor energy for a single journey using all-battery mode, a low-pass filter, and optimal controls. Hard constraints are shown in red areas.

using a linear filter, as the presence of both hard discontinuities and periods of constant output place conflicting requirements on the bandwidth of the filter. The battery power constraint is violated for the all-battery mode as the battery is the only power source, but it is also violated using the low pass filter. In particular, it is violated at two large peaks at $\sim$50 s and $\sim$200s, where the supercapacitor is completely empty and the battery is forced to meet the positive power demand alone. Conversely, the battery power constraint is satisfied at all times using the optimal controller, for which the only hard constraints that are active are the upper and lower limits on supercapacitor energy. It was found that by tightening the battery power or energy limits to ensure that they were active at the solution, the problem generally becomes infeasible.

An aim of the energy management optimization was to minimize battery degradation, a complex phenomenon caused by a multitude of factors (see [151] for a comprehensive review of lithium-ion battery ageing mechanisms). There is a range of explicit models of battery degradation (for a review of modelling methods see [152]), but these typically model the battery at a level that is inconsistent with the resolution of the simulations performed here (e.g. the current distribution on a cellular level would depend on the battery management system, which is not modelled here). Therefore, three metrics of battery

Table 7.2: Approximate measures of battery degradation for the trajectories shown in Figure 7.2. The percentage values are the improvements relative to the all-battery baseline.

	All Battery	Low-pass Filter	Optimal
RMS(u) (kW)	31.8	25.2 (-20.8%)	21.0 (-33.9%)
$\max \lvert u \rvert$ (kW)	133.3	132.1 (-0.9%)	36.4 (-72.7%)
$\sum \lvert u \rvert$ (MJ)	17.9	15.9 (-11.3%)	13.6 (-24.2%)
$\sum (u + v)$ (MJ)	6.8	6.6 (-3.7%)	6.4 (-5.5%)

power were instead used as approximate measures of battery degradation: the Root-Mean-Squared (RMS) battery power, RMS(u), the peak battery power, $\max |u|$, and total power throughput, $\sum |u|$. Additionally, the total energy consumption, $\sum(u+v)$, was used to determine relative efficiency of each control method (this was the optimization objective in (MPC.5)). Note that RMS(u), $\max |u|$, and $\sum |u|$ could also be included as objectives in a convex optimization problem, but the results show that the proposed objective function is a good heuristic for minimizing all four quantities. Table 7.2 shows each of these measures for the trajectories shown in Figure 7.2, and it can be seen that the optimal controller provides a significant improvement over the low-pass filter for all three degradation metrics and energy consumption. In particular, the low-pass filter provides almost no reduction in peak battery power due to the instances where a high positive power is demanded and the supercapacitor is empty, whereas the optimal controller reduces the peak battery power from 133.3 kW to 36.4 kW. This clearly demonstrates the benefits of a predictive controller that can ensure the supercapacitor has sufficient charge available for high power events. Furthermore, the RMS battery power and total battery throughput are also significantly reduced using the optimal controller, and the supercapacitor state constraints are only active for a small fraction of the journey. This suggests that the filter is inherently suboptimal, even when detrimental control decisions are not being forced by the state constraints.

Figure 7.3 shows the RMS battery power, peak battery power, battery power throughput, and energy consumption for each control method on all 49 journeys, and the averages are summarized in Table 7.3. It can be clearly seen that the optimal controller provides a significant and consistent improvement over the low-pass filter across all four metrics and every journey. High temperatures and thermal gradients have been identified as factors contributing to battery degradation [153, §3], and the optimal controller provides the greatest overall reduction in peak current, which will have an impact on reducing both battery temperature (there will be a lag between heat being generated within individual cells and being sensed by the cooling system) and temperature gradients within each cell (the increased temperature will initially be localised to the core and/or terminals). Conversely,

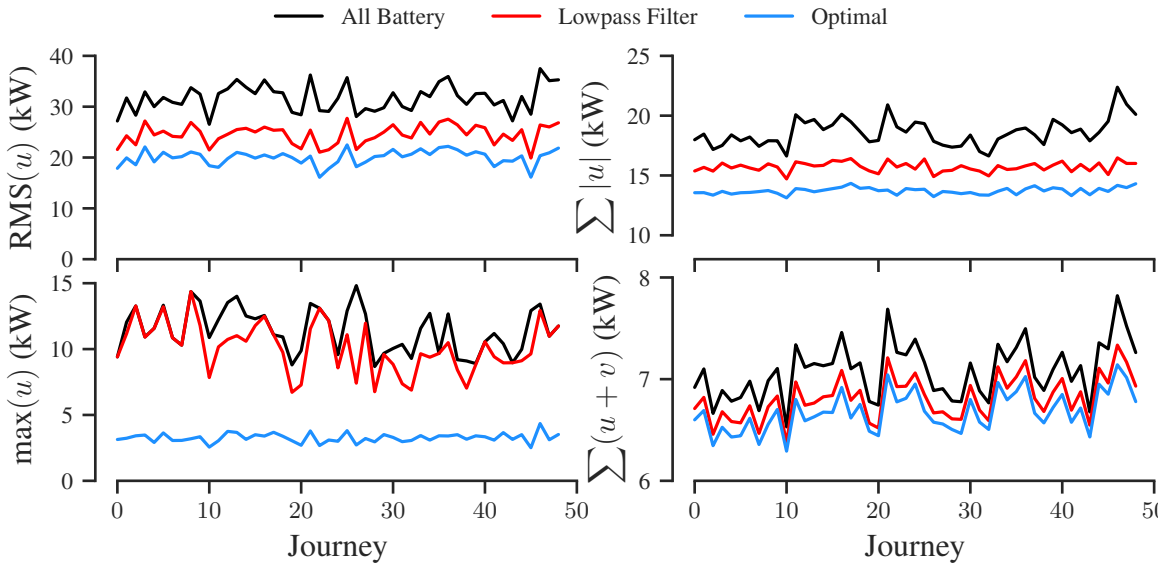


Figure 7.3: Approximate measures of battery degradation for all 49 journeys, using all-battery control, the low-pass filter, and optimal controls.

for a significant number of journeys the low-pass filter provides no perceptible reduction in peak battery power, as shown for the journey in Figure 7.2. The optimal controller also significantly reduces the RMS battery power, which will also reduce thermal degradation, and total power throughput, which will reduce degradation from battery cycling. Finally, the optimal controller also increases the efficiency of the powertrain over the low-pass filter, from a reduction of 3.8% to 5.7% relative to the all battery mode, providing an increase in the range available to the vehicle from full charge.

Table 7.3: Averages of the approximate measures of battery degradation for all 49 journeys and all control methods. The stated percentages are the average of the percentage improvements relative to the all-battery baseline.

	All Battery	Low-Pass Filter	Optimal
$\overline{\text{RMS}(u)}$ (kW)	31.7	24.6 (-22.5%)	20.0 (-36.8%)
$\overline{\max \lvert u \rvert}$ (kW)	114.1	101.5 (-11.0%)	32.7 (-71.4%)
$\overline{\sum \lvert u \rvert}$ (MJ)	18.6	15.7 (-15.6%)	13.7 (-26.4%)
$\overline{\sum (u+v)}$ (MJ)	7.1	6.8 (-3.8%)	6.7 (-5.7%)

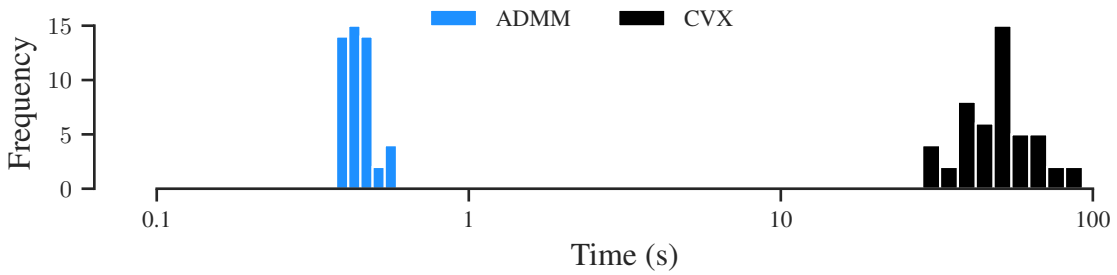


Figure 7.4: Histograms of solution time using ADMM and CVX for all 49 journeys.

Figure 7.4 shows histograms of the solution times using ADMM and CVX. The average horizon length of T was 815, with a maximum of 1003, for which the average and maximum solution times using CVX were 51 s and 93 s, and the average and maximum solution times using ADMM were 0.45 s and 0.59 s. The ADMM algorithm was implemented sequentially using vectorized Matlab code in these experiments, so a compiled implementation where the combined (u, v) updates are performed in parallel will improve the absolute performance further.

These results have important implications for electric vehicle powertrain control and design. The speed of computation suggests that the ADMM algorithm is a promising candidate for a real-time, online, receding-horizon MPC implementation, which could significantly reduce the battery degradation and energy consumption characteristics of a given electric powertrain design. Furthermore, the algorithm could also be used to determine the optimal size of the powertrain components (a problem considered in [141] and [139]). In this case, the speed of computation could allow a brute-force approach, where every possible combination of a discrete set of powertrain parameters is evaluated against a set of candidate drive-cycles.

One of the limitations of the proposed approach is that it does not address uncertainty in the predictions of driver behaviour, and it is possible that sufficiently inaccurate predictions could be generated so that the performance of optimization-based controllers becomes worse than a low-pass filter. It is, however, worth highlighting that the convex formulation permits the use of scenario MPC, which could be implemented using a similar method to the previous chapter. A systematic investigation of the robustness of the proposed method to prediction errors is left for future work.

7.4 Concluding Remarks

In this chapter, the approach from the previous chapters was extended to consider electric vehicles with a hybrid energy storage system containing both battery and supercapacitor components. The objective in this case was to minimize energy consumption and battery degradation, as opposed to minimizing fuel consumption. A general framework for energy management of hybrid energy storage systems was formulated, and it was demonstrated that the corresponding optimization problem is convex when the upper limit on the motor's power is linearized, and that this approximation guarantees that the original nonlinear constraint is also satisfied. The structure of the convex optimization problem is similar to that considered for PHEV energy management, so the ADMM algorithm that had shown good performance in the previous chapters was applied to the new problem. The optimization-based controller was compared in simulation with a low-pass filtering rule-based strategy against an all-electric baseline, where it was shown to provide significant improvements in battery degradation, inferred through three metrics of battery use. It was also shown that the ADMM algorithm solved the optimization problem in a fraction of a second, compared to approximately 60 seconds using CVX. One of the limitations of the proposed method is that feasibility has not been addressed. The approach previously used for PHEV energy management does not translate directly as the BSEV problem has two state variables. In this case, therefore, it may be preferrable to use known results for infeasability detection in ADMM (e.g. [154]) rather than developing a dedicated pre-optimization algorithm. This aspect of the problem is left to future work.

APPENDICES

7.A ANALYSIS OF PROBLEM (MPC.5)

The Lagrangian equation for problem (MPC.5) is

$$\begin{aligned}\mathcal{L}(u,v) := u + v - \lambda_1^\top(u-\underline{u}) - \lambda_2^\top(\overline{u}-u) - \lambda_3^\top(v-\overline{v}) \\ - \lambda_4^\top(\overline{v}-v) - \lambda_5^\top(\mathbf{1}x_0 - \Psi u - \underline{x}) - \lambda_6^\top(\overline{x} - \mathbf{1}x_0 + \Psi u) \\ - \lambda_7^\top(\mathbf{1}y_0 - \Psi v - \underline{y}) - \lambda_8^\top(\overline{y} - \mathbf{1}y_0 + \Psi v) \\ - \lambda_9^\top(g(u) + f(v) - \overline{e}) - \lambda_{10}^\top(\overline{e} - g(u) - f(v)),\end{aligned}$$

where

$$\begin{aligned}g(u) &:= (g_0(u_0), \ldots, g_{T-1}(u_{T-1})) \\ f(v) &:= (f_0(v_0), \ldots, f_{T-1}(v_{T-1})).\end{aligned}$$

The first-order necessary conditions for optimality imply that $\lambda_j^\star \geq 0 \; \forall j \in \{0, \ldots, 10\}$, and $\nabla_u \mathcal{L} = 0$ and $\nabla_v \mathcal{L} = 0$ imply that

$$\lambda_{1,t}^\star - \lambda_{2,t}^\star + \sum_{i=t}^{T-1}[\lambda_{5,i}^\star - \lambda_{6,i}^\star] + g_t'(u_t^\star)(\lambda_{10,t}^\star - \lambda_{9,t}^\star) = -1$$

$$\lambda_{3,t}^\star - \lambda_{4,t}^\star + \sum_{i=t}^{T-1}[\lambda_{7,i}^\star - \lambda_{8,i}^\star] + f_t'(v_t^\star)(\lambda_{10,t}^\star - \lambda_{9,t}^\star) = -1$$

$\forall t \in \{0, \ldots, T-1\}$. Suppose that the constraint $\overline{e}_t - g_t(u_t) - f_t(v_t) \geq 0$ is active for some t, then $\lambda_{10,t} > 0$ and $\lambda_{9,t} = 0$ so

$$g_t'(u_t^\star)\lambda_{10,t}^\star = -1 + \lambda_{1,t}^\star - \lambda_{2,t}^\star - \sum_{i=t}^{T-1}[\lambda_{5,i}^\star - \lambda_{6,i}^\star], \tag{7.10a}$$

$$f_t'(v_t^\star)\lambda_{10,t}^\star = -1 + \lambda_{3,t}^\star - \lambda_{4,t}^\star - \sum_{i=t}^{T-1}[\lambda_{7,i}^\star - \lambda_{8,i}^\star], \tag{7.10b}$$

Assumption 7.1.1 implies that $g_t'(u^\star) \geq 0$ and $f_t'(v^\star) \geq 0$, so $g_t'(u_t^\star)\lambda_{10,t}^\star \geq 0$ and $f_t'(v_t^\star)\lambda_{10,t}^\star \geq 0$. In the case where the bounds on battery and supercapacitor energy and power are not considered (i.e. $\lambda_j^\star = 0 \; \forall j \in \{1, \ldots, 6\}$), this contradicts (7.10a) and (7.10b) so the constraint $\overline{e}_t - g_t(u_t) - f_t(v_t) \geq 0$ cannot be active at the solution. In the case where these bounds are included in the problem formulation, the same conclusion cannot be reached as $\lambda_{1,t}^\star$ and/or $\lambda_{6,i}^\star$ for some $i \in \{t, \ldots, T-1\}$ may be nonzero.

7.B PIECEWISE CONSTANT SOLUTION OF (MPC.5)

A similar approach as used in Appendix 7.A can be used to show that, under the assumption that the constraints on battery power are inactive (i.e. $\lambda_{1,t} = 0$ and $\lambda_{2,t} = 0\ \forall t$) and that the upper constraint on powertrain power is inactive (i.e. $\lambda_{10,t} = 0\ \forall t$), then the solution to (MPC.5) satisfies

$$-g_t'(u_t^\star)\lambda_{9,t}^\star = -1 - \sum_{i=t}^{T-1}[\lambda_{5,i}^\star - \lambda_{6,i}^\star], \tag{7.11a}$$

$$-f_t'(v_t^\star)\lambda_{9,t}^\star = -1 - \sum_{i=t}^{T-1}[\lambda_{7,i}^\star - \lambda_{8,i}^\star], \tag{7.11b}$$

Consider a set $\{\underline{t}, \dots, \bar{t}\}$ where the upper and lower bounds on x and y are inactive, i.e. $\lambda_{5,t}^\star = 0$, $\lambda_{6,t}^\star = 0$, $\lambda_{7,t}^\star = 0$, and $\lambda_{8,t}^\star = 0\ \forall t \in \{\underline{t}, \dots, \bar{t}\}$, then $-1 - \sum_{i=t}^{T-1}[\lambda_{5,i}^\star - \lambda_{6,i}^\star] = c_1$ and $-1 - \sum_{i=t}^{T-1}[\lambda_{7,i}^\star - \lambda_{8,i}^\star] = c_2\ \forall t \in \{\underline{t}, \dots, \bar{t}\}$. Therefore, $u_t^\star$, $v_t^\star$, and $\lambda_{9,t}^\star$ are given by

$$-g_t'(u_t^\star)\lambda_{9,t}^\star = c_1 \quad \text{and} \quad -f_t'(v_t^\star)\lambda_{9,t}^\star = c_2.$$

For the system models presented in Section 7.3, $f_t'(v_t) = 1$ and $g_t'(u_t) = 1 - \frac{2Ru_t}{V^2}\ \forall t \in \{0, \dots, T-1\}$, so $\lambda_{9,t}^\star = -c_2\ \forall t \in \{\underline{t}, \dots, \bar{t}\}$, and

$$u_t = \frac{V^2}{2R}\left(1 - \frac{c_1}{c_2}\right)\ \forall t \in \{\underline{t}, \dots, \bar{t}\},$$

which implies that the optimal control input is constant on the interval $\{\underline{t}, \dots, \bar{t}\}$.

7.C COMBINED u AND v UPDATE

The optimization problem in (7.8a) is equivalent to

$$\begin{aligned} \underset{(u_t, v_t)}{\text{argmin}}\ & \frac{\rho_1}{2}(u_t - \zeta_t^{(j)} + \lambda_{1,t}^{(j)})^2 + \frac{\rho_2}{2}(v_t - \eta_t^{(j)} + \lambda_{2,t}^{(j)})^2 \\ \text{s.t. } & \underline{u}_t \le u_t \le \overline{u}_t, \quad \underline{v}_t \le v_t \le \overline{v}_t, \\ & \hat{e}_t \le g_t(u_t) + f_t(v_t), \quad \hat{g}_t(u_t) + \hat{f}_t(v_t) \le \overline{e}_t, \end{aligned} \tag{7.12}$$

which is a convex (quadratic) inequality constrained optimization problem, and Figure 7.5 shows the constraint set for an illustrative example. A rigorous treatment of the proposed

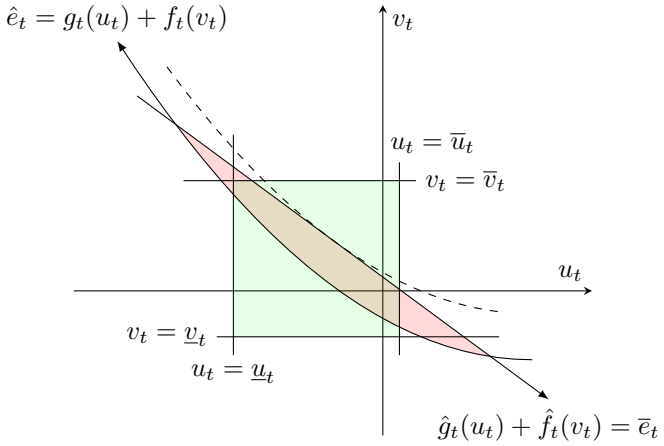


Figure 7.5: Illustration of constraint sets defined by $\mathcal{C}_t$ (in red) and $\mathcal{U}_t \cap \mathcal{V}_t$ (in green).

approach is provided below, but the principle is that there are three candidate solutions that can be obtained from simpler optimization problems: (a) problem (7.12) with the constraints $\hat{e}_t \leq g_t(u_t) + f_t(v_t)$ and $\hat{g}_t(u_t) + \hat{f}_t(v_t) \leq \overline{e}_t$ discarded, (b) problem (7.12) with equality constraint $\hat{g}_t(u_t) + \hat{f}_t(v_t) = \overline{e}_t$, and (c) problem (7.12) with equality constraint $\hat{e}_t = g_t(u_t) + f_t(v_t)$. Problems (a) and (b) have analytical solutions, and (c) reduces to a one-dimensional problem that can be solved numerically. Each of these cases will now be considered in detail; it is assumed throughout that (7.12) is feasible.

(a) The constraints $\hat{e}_t \leq g_t(u_t) + f_t(v_t)$ and $\hat{g}_t(u_t) + \hat{f}_t(v_t) \leq \overline{e}_t$ (i.e. $(u_t, v_t) \in \mathcal{C}_t$) are discarded, and a candidate solution $(u_t^{\dagger 1}, v_t^{\dagger 1})$ to (7.12) is obtained from

$$\begin{aligned} \operatorname*{argmin}_{(u_t, v_t)} \; & \frac{\rho_1}{2}(u_t - \zeta_t^{(j)} + \lambda_{1,t}^{(j)})^2 + \frac{\rho_2}{2}(v_t - \eta_t^{(j)} + \lambda_{2,t}^{(j)})^2 \\ \text{s.t. } & \underline{u}_t \leq u_t \leq \overline{u}_t, \quad \underline{v}_t \leq v_t \leq \overline{v}_t, \end{aligned}$$

as $u_t^{\dagger 1} = \min\{\overline{u}_t, \max\{\underline{u}_t, \zeta_t^{(j)} - \lambda_{1,t}^{(j)}\}\}$ and $v_t^{\dagger 1} = \min\{\overline{v}_t, \max\{\underline{u}_t, \eta_t^{(j)} - \lambda_{2,t}^{(j)}\}\}$. If the discarded constraints are satisfied for the candidate solution, then this is the actual solution to (7.12). If not, then the solution must be further constrained by $\hat{e}_t = g_t(u_t) + f_t(v_t)$ and/or $\hat{g}_t(u_t) + \hat{f}_t(v_t) = \overline{e}_t$. Therefore, two further candidate solutions are obtained by tightening each of $\hat{e}_t \leq g_t(u_t) + f_t(v_t)$ and $\hat{g}_t(u_t) + \hat{f}_t(v_t) \geq \overline{e}_t$ in (7.12) to equality constraints.

(b) A candidate solution $(u_t^{\dagger 2}, v_t^{\dagger 2})$ is obtained from

$$\begin{aligned} \underset{(u_t,v_t)}{\text{argmin}} \; & \frac{\rho_1}{2}(u_t - \zeta_t^{(j)} + \lambda_{1,t}^{(j)})^2 + \frac{\rho_2}{2}(v_t - \eta_t^{(j)} + \lambda_{2,t}^{(j)})^2 \\ \text{s.t. } & \underline{u}_t \leq u_t \leq \overline{u}_t, \quad \underline{v}_t \leq v_t \leq \overline{v}_t, \\ & \hat{e}_t \leq g_t(u_t) + f_t(v_t), \quad \hat{g}_t(u_t) + \hat{f}_t(v_t) = \overline{e}_t. \end{aligned} \tag{7.13}$$

In section 7.1.2 it was specified that $\hat{u}$ and $\hat{v}$ were chosen so that $g'(\hat{u})$ and $f'(\hat{v})$ are nonzero (and under Assumption 7.1.1 must be positive), so $\hat{g}^{-1}$ and $\hat{f}^{-1}$ exist and are both affine and increasing.

Proposition 7.C.1. *Define the set*

$$\hat{\mathcal{U}}_t := \{u_t \in \mathbb{R} : \hat{f}_t^{-1}(\overline{e}_t - \hat{g}_t(u_t)) \geq f_t^{-1}(\hat{e}_t - g_t(u_t))\},$$

and define $u_t^{\cap 1} := \inf \hat{\mathcal{U}}_t$ *and* $u_t^{\cap 2} := \sup \hat{\mathcal{U}}_t$*. Then*

$$\begin{aligned} &\{(u_t, v_t) \in \mathbb{R}^2 : \hat{g}_t(u_t) + \hat{f}_t(v_t) = \overline{e}_t, \hat{e}_t \leq g_t(u_t) + f_t(v_t)\} \\ =&\{(u_t, v_t) \in \mathbb{R}^2 : \hat{g}_t(u_t) + \hat{f}_t(v_t) = \overline{e}_t, u_t \in [u_t^{\cap 1}, u_t^{\cap 2}] \cap \mathbb{R}\}. \end{aligned}$$

Proof. The set $\{(u_t, v_t) \in \mathbb{R}^2 : \hat{e}_t \leq g_t(u_t) + f_t(v_t)\}$ is convex under Assumption 7.1.2 and is the epigraph of the function $v_t = f_t^{-1}(\hat{e}_t - g_t(u_t))$, which therefore must also be a convex function. Additionally, $\hat{g}_t(u_t) + \hat{f}_t(v_t) = \overline{e}_t \Leftrightarrow v_t = \hat{f}_t^{-1}(\overline{e}_t - \hat{g}_t(u_t))$, so

$$\begin{aligned} &\{(u_t, v_t) \in \mathbb{R}^2 : \hat{g}_t(u_t) + \hat{f}_t(v_t) = \overline{e}_t, \hat{e}_t \leq g_t(u_t) + f_t(v_t)\} \\ =&\{(u_t, v_t) \in \mathbb{R}^2 : \hat{g}_t(u_t) + \hat{f}_t(v_t) = \overline{e}_t, \\ &\qquad \hat{f}_t^{-1}(\overline{e}_t - \hat{g}_t(u_t)) \geq f_t^{-1}(\hat{e}_t - g_t(u_t))\} \end{aligned}$$

The set $\hat{\mathcal{U}}_t$ therefore defines the values of u_t where an affine function is greater than or equal to a convex function, so is convex.

Consider the illustration of $\hat{g}_t(u_t) + \hat{f}_t(v_t) = \overline{e}_t$ and $\hat{e}_t = g_t(u_t) + f_t(v_t)$ in Figure 7.5, and assume that the function $f_t^{-1}(\hat{e}_t - g_t(u_t))$ is strongly convex (which is the case for the models specified in Sections 7.3 and 7.3). This implies that $\hat{\mathcal{U}}_t$ is closed and $\hat{\mathcal{U}}_t = [u_t^{\cap 1}, u_t^{\cap 2}]$ (i.e. $u_t^{\cap 1} \in \mathbb{R}$ and $u_t^{\cap 2} \in \mathbb{R}$ are the 'lower' and 'upper' intersection points of the functions $\hat{g}_t(u_t) + \hat{f}_t(v_t) = \overline{e}_t$ and $\hat{e}_t = g_t(u_t) + f_t(v_t)$).

Now assume that $f_t^{-1}(\hat{e}_t - g_t(u_t))$ is not strongly convex. Under the assumptions on $f_t(\cdot)$ and $g_t(\cdot)$ it is possible to construct cases in which there are no intersection points, or only an 'upper' or 'lower' intersection point (e.g. if $f_t(\cdot)$ and $g_t(\cdot)$ are piecewise affine). In these cases $\hat{\mathcal{U}}_t = [u_t^{\cap 1}, u_t^{\cap 2}]$ where $u^{\cap 1} \in \{-\infty, \mathbb{R}\}$ and $u^{\cap 2} \in \{\mathbb{R}, \infty\}$. □

Remark 7.C.1. If $\hat{e}_t = g_t(u_t) + f_t(v_t)$ is strongly convex and twice continuously differentiable (which is the case for the models specified in Sections 7.3 and 7.3), then $\overline{e}_t = \hat{e}_t \implies u_t^{\cap 1} = u_t^{\cap 2}$, and $\mathcal{C}_t$ has a single element that is trivially the solution to (7.12). This also implies that $u_t^{\cap 1} = u_t^\star$, so the linearization points $\hat{u}_t$ and $\hat{v}_t$ should be chosen so that the power consumed by the powertrain is zero when $\overline{e}_t = \hat{e}_t$, as this occurs when the vehicle is stationary.

Proposition 7.C.2.

$$\begin{aligned}&\{(u_t, v_t) \in \mathbb{R}^2 : \hat{g}_t(u_t) + \hat{f}_t(v_t) = \overline{e}_t, a \le v_t \le b\} \\ &= \{(u_t, v_t) \in \mathbb{R}^2 : \hat{g}_t(u_t) + \hat{f}_t(v_t) = \overline{e}_t, \hat{g}_t^{-1}(\overline{e}_t - \hat{f}_t(b)) \le u_t \le \hat{g}_t^{-1}(\overline{e}_t - \hat{f}_t(a))\}\end{aligned}$$

Proof. In section 7.1.2 it was specified that $\hat{u}$ and $\hat{v}$ were both chosen so that $g'(\hat{u})$ and $f'(\hat{v})$ were non-zero (which under Assumption 7.1.1 implies that they are greater than zero), so $\hat{f}_t(\cdot)$ and $\hat{g}_t(\cdot)$ are both increasing, and it can then be shown that

$$\begin{aligned}v_t \le b &\Rightarrow \overline{e}_t - \hat{f}_t(v_t) \ge \overline{e}_t - \hat{f}_t(b) \\ &\Rightarrow u_t \ge \hat{g}_t^{-1}(\overline{e}_t - \hat{f}_t(b)).\end{aligned}$$

It can similarly be shown that $u_t \le \hat{g}_t^{-1}(\overline{e}_t - \hat{g}_t(a))$. □

Propositions 7.C.1 and 7.C.2 imply that (7.13) is equivalent to

$$\begin{aligned}\underset{u_t}{\text{argmin}}\ &\frac{\rho_1}{2}(u_t - \zeta_t^{(j)} + \lambda_{1,t}^{(j)})^2 + \frac{\rho_2}{2}(\hat{f}_t^{-1}(\overline{e}_t - \hat{g}_t(u_t)) - \eta_t^{(j)} + \lambda_{2,t}^{(j)})^2 \\ \text{s.t. } &u_t \ge \max\{\underline{u}_t, u_t^{\cap 1}, \hat{g}_t^{-1}(\overline{e}_t - \hat{f}_t(\overline{v}_t))\} \\ &u_t \le \min\{\overline{u}_t, u_t^{\cap 2}, \hat{g}_t^{-1}(\overline{e}_t - \hat{f}_t(\underline{v}_t))\}\end{aligned} \tag{7.14}$$

which is a one-dimensional constrained quadratic optimization problem for $\hat{u}_t$, with the constraint set illustrated in blue in Figure 7.6. The solution to the unconstrained problem can be obtained analytically and projected onto the upper and lower bounds on $\hat{u}_t$ to obtain $u_t^{\dagger 2}$, then the corresponding value of v_t can then be returned from $v_t^{\dagger 2} = \hat{f}_t^{-1}(\overline{e}_t - \hat{g}_t(u_t^{\dagger 2}))$.

(c) The final candidate solution $(u_t^{\dagger 3}, v_t^{\dagger 3})$ is obtained from

$$\begin{aligned}\underset{(u_t, v_t)}{\text{argmin}}\ &\frac{\rho_1}{2}(u_t - \zeta_t^{(j)} + \lambda_{1,t}^{(j)})^2 + \frac{\rho_2}{2}(v_t - \eta_t^{(j)} + \lambda_{2,t}^{(j)})^2 \\ \text{s.t. } &\underline{u}_t \le u_t \le \overline{u}_t, \quad \underline{v}_t \le v_t \le \overline{v}_t, \\ &\hat{e}_t = g_t(u_t) + f_t(v_t), \quad \hat{g}_t(u_t) + \hat{f}_t(v_t) \le \overline{e}_t,\end{aligned}$$

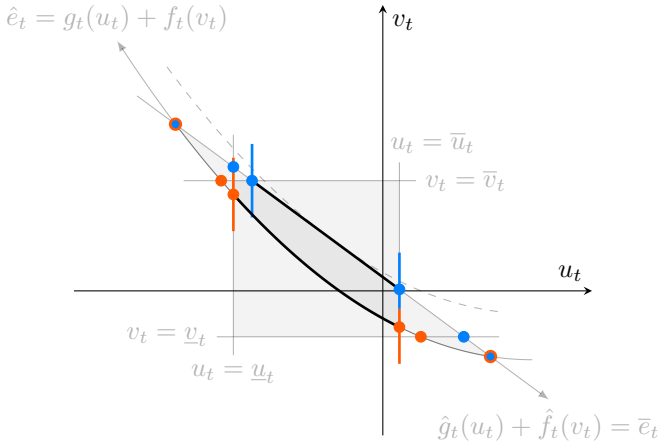


Figure 7.6: Illustration of constraint sets for reduced problems (7.14) and (7.15).

which can be shown to be equivalent to

$$\begin{aligned} \underset{u_t}{\text{argmin}}\ & \frac{\rho_1}{2}(u_t - \zeta_t^{(j)} + \lambda_{1,t}^{(j)})^2 + \frac{\rho_2}{2}(f_t^{-1}(\overline{e}_t - g_t(u_t)) - \eta_t^{(j)} + \lambda_{2,t}^{(j)})^2 \\ \text{s.t. } & u_t \geq \max\{\underline{u}_t, u_t^{\cap 1}, \hat{g}_t^{-1}(\hat{e}_t - \hat{f}_t(\overline{v}_t))\} \\ & u_t \leq \min\{\overline{u}_t, u_t^{\cap 2}, \hat{g}_t^{-1}(\hat{e}_t - \hat{f}_t(\underline{v}_t))\} \end{aligned} \tag{7.15}$$

using the same approach as for (b). The constraint set for (7.15) is illustrated in red in Figure 7.6. The function $f_t^{-1}(\hat{e}_t - g_t(u_t))$ is nonlinear, so the cost function in (7.15) is nonconvex in general, but a stationary point can be obtained without the inequality constraints using an iterative algorithm (e.g. Newton's method)[3] and then projected onto the bounds on u_t.

The stationary point of (7.15) and $(u_t^{\dagger 2}, v_t^{\dagger 2})$ are then evaluated against the cost function of (7.12) to determine which is the minimizing argument. If the constraint $\hat{e}_t \leq g_t(u_t) + f_t(v_t)$ is strongly active at the solution then the stationary point of (7.15) corresponds to the global minimum of (7.12). If the constraint $\hat{e}_t \leq g_t(u_t) + f_t(v_t)$ is not strongly active at the solution to (7.12) then problem (7.15) may have multiple stationary points that may not be minimal for (7.12), but in this case the constraint $\hat{g}_t(u_t) + \hat{f}_t(v_t) \leq \overline{e}_t$ will be strongly active at the solution to (7.12), so $(u_t^{\dagger 2}, v_t^{\dagger 2})$ will be the minimizing argument of (7.12).

The overall algorithm for update (7.8a) is presented in Algorithm 6.

[3]For the experiments detailed in Section 7.3, the function $f_t^{-1}(\overline{e}_t - g_t(u_t))$ is quadratic, so the cost function in (7.15) is quartic, and the stationary points can be found from the roots of a cubic equation.

Algorithm 6 Update (7.8a)

$u_t^{\dagger 1} \leftarrow \min\{\overline{u}_t, \max\{\underline{u}_t, \zeta_t^{(j)} - \lambda_{1,t}^{(j)}\}\}$
$v_t^{\dagger 1} \leftarrow \min\{\overline{v}_t, \max\{\underline{u}_t, \eta_t^{(j)} - \lambda_{2,t}^{(j)}\}\}$
if $(u_t^{\dagger 1}, v_t^{\dagger 1}) \in \mathcal{C}_t$ **then**
    $(u_t, v_t)^{j+1} \leftarrow (u_t^{\dagger 1}, v_t^{\dagger 1})$
else
    $u_t^{\dagger 2} \leftarrow$ Solution to (7.14)
    $v_t^{\dagger 2} \leftarrow \hat{f}_t^{-1}(\overline{e}_t - \hat{g}_t(u_t^{\dagger 2}))$
    $u_t^{\dagger 3} \leftarrow$ Stationary point of (7.15)
    $v_t^{\dagger 3} \leftarrow f_t^{-1}(\hat{e}_t - g_t(u_t^{\dagger 3}))$
    Evaluate $(u_t^{\dagger 2}, v_t^{\dagger 2})$ and $(u_t^{\dagger 3}, v_t^{\dagger 3})$ against cost function of (7.8a)
    $(u_t, v_t)^{j+1} \leftarrow$ minimizing argument
end if

CHAPTER 8
CONCLUSION

THIS thesis presents an investigation into convex optimization for model predictive energy management in plug-in hybrid and battery/supercapacitor electric vehicles. Several problem formulations were considered, for which tailored optimization algorithms were generated, and the closed-loop performance of the model predictive controllers was presented through numerical studies. Overall, it was shown that convex optimization can be used to generate model predictive controllers that provide an extremely close approximation of the globally optimal control inputs, even when an exact prediction of future driver behaviour is not available, and that the associated MPC optimization problems can be solved reliably in real time. The individual contributions of each of the chapters is now discussed.

Chapter 2 introduced the PHEV energy management problem and provided an illustration of the common optimization-based approaches for its solution. It was highlighted that one of the limitations associated with model predictive energy management is that it requires the repeated on-line solution of a challenging optimization problem, and that this limitation could be addressed using convex optimization algorithms. In particular, it was observed that no previous systematic investigation into convex optimization algorithms for model predictive energy management existed.

Chapter 3 began with a formal presentation of the mathematical PHEV model used for simulation, and the set of real-world driving data used to evaluate each controller. The first MPC formulation was presented, which considered nonlinear losses and power constraints for the engine, motor, and battery, and a terminal state-of-charge constraint for the battery. It was shown that, under negligible further approximation, the MPC optimization could be reformulated as a convex optimization problem when considered in terms of the battery power instead of the engine output power (as commonly used in the energy management literature). In simulations it was demonstrated that the MPC obtained a very close approximation of the controls obtained using PMP, but that the lack of general state-of-charge constraints limited the performance of the MPC, for example when the road was not flat.

In light of the limitations of the MPC proposed in Chapter 3, Chapter 4 extended the controller to consider upper and lower bounds on the state-of-charge of the battery across the entire prediction horizon. Two algorithms were proposed for the solution of the new MPC optimization: an alternating direction method of multipliers algorithm and a novel projected interior point method. It was demonstrated through numerical experiments that the projected interior point algorithm obtained a highly accurate solution in a few iterations, but that the ADMM algorithm obtained a sufficiently accurate solution much faster in general (and that both were significantly faster than CVX). It was then demonstrated that the closed-loop convex MPC controller provided an accurate approximation of the globally optimal power-split obtained using dynamic programming, but that the engine switching heuristic caused the overall performance to be worse than a simple charge-depleting/charge-sustaining strategy.

Consequently, in Chapter 5 the MPC optimization and ADMM optimization algorithm were extended to also consider engine switching. The engine switching control is an integer decision variable, so the associated MPC optimization problem is necessarily nonconvex and only local convergence of ADMM could be guaranteed, but it was also demonstrated that ADMM will always return the optimal power split given the engine switching control sequence at termination. Global convergence was therefore encouraged by initializing the nonconvex ADMM with the solution of a convex relaxation. The ADMM algorithm was compared against dynamic programming where it shown to reduce the computational requirement from $\sim$3 hours to a fraction of a second, with only a 5.3% increase in predicted fuel consumption. The MPC framework and ADMM algorithm were then implemented in closed loop, where they achieved a 32 % improvement in fuel consumption relative to the CDCS baseline.

Up to this point, it had been assumed that the MPC could predict the driver's future behaviour with complete accuracy, whereas in reality it is likely to be subject to considerable uncertainty. Therefore, in Chapter 6 driver behaviour uncertainty was explicitly considered. In particular, a scenario MPC framework was proposed that allowed the predictions of future driver behaviour to be drawn from previously recorded examples of a route being driven, thereby avoiding the challenging process of generating an explicit model of driver behaviour. This approach also allowed the use of results from scenario MPC to generate a bound on the required number of scenarios for a given confidence of one-step-ahead feasibility. The ADMM algorithm proposed in Chapter 4 was extended to the scenario MPC problem, and it was demonstrated that the iteration cost scaled well with both prediction horizon length and number of scenarios, particularly when parallel processing is available. The scenario MPC framework was then simulated in closed-loop, where it was shown to achieve a close approximation of the nominal MPC without perfect knowledge of the driver's future behaviour.

In Chapter 7 a different problem was considered: energy management in all-electric vehicles with a hybrid energy store consisting of both batteries and supercapacitors. In this case the control objective is to maximize the efficiency of the powertrain and minimize battery degradation, rather than fuel consumption as in the PHEV problem. A general framework for BSEV energy management was proposed, that considered losses and power limits on the battery, supercapacitor, and motor, and limits on energy stored in the battery and supercapacitor. It was then shown that this optimal control problem could be reformulated as a convex optimization problem under minor further approximation, and an ADMM algorithm was proposed for its solution. The convex formulation was simulated in open-loop in comparison with a low-pass filter-based controller against an all electric baseline, and was shown to provide significant improvements in energy efficiency and battery degradation (inferred through three measures of battery ageing). Additionally, it was shown that the ADMM algorithm solved the optimization problem in a fraction of a second in all cases.

Overall, the results have several important implications for electric vehicle control and design. For the PHEV case it has been demonstrated that the fuel consumption of a given powertrain can be significantly reduced using the proposed MPC controllers, and the solution times suggest that real-time embedded implementations are possible. This has the potential to improve the economic viability of PHEVs, which may increase the adoption of electrified vehicles in the coming years, and consequently reduce the tailpipe emissions of global transportation. In particular, the results are the most relevant for heavy-duty vehicles that repeatedly drive the same route, such as lorries or buses. In these cases all-electric powertrains have not reached the point of practical or economic viability, and it is likely that hybrid technology will be required for the foreseeable future. Additionally, cases in which the vehicle is used repeatedly for the same route would make it feasible to generate a database of previous driver behaviour to generate predictions as presented in Chapter 6. The results in Chapter 7 extend the possible impact of convex optimization-based predictive energy management to all-electric powertrains. The proposed optimal control framework could be used to increase the life of a given battery design, or could instead be used to reduce the battery size required to reach a given life-span. This will then in turn also improve the economic and practical viability of all-electric passenger vehicles. It is also worth noting that the proposed framework can readily be extended to applications other than road vehicles: for example, the framework can be readily applied to energy storage systems used for renewable energy storage (e.g. [155]) by replacing the motor model with the associated power electronics.

FUTURE WORK

The presented research could be extended in the following directions to aid the development of optimization-based energy management systems.

Embedded Control: One of the limitations of the experiments presented in this thesis is that they have been entirely limited to a simulation environment. The necessary next step is to start transitioning the experiments in to the real world. This may initially take the form of hardware-in-the-loop experiments where the MPC optimization alone is solved on a representative electronic control unit, and then transitioned onto experiments in a test vehicle. One of the reasons that ADMM algorithms have been extensively used in this thesis is that the iteration cost can be substantially reduced using parallelization. This makes them ideal for implementation on hardware such as field-programmable gate arrays (FPGAs), which may be able to exceed the computational performance demonstrated in the experiments presented here (which were all implemented on a central processing unit (CPU) using vectorized sequential Matlab code).

Driver Behaviour Prediction: The database approach to driver behaviour prediction presented in this thesis is particularly suited to vehicles that repeatedly operate on a small number of individual routes. This approach may not be possible for vehicles that can be driven to an arbitrary destination, as a database of driving examples for every possible segment of road would be required (and the driver would need to make the vehicle aware of their intended route at the start of the journey). A possible solution for vehicles that are primarily used for commuting is that the controller records each route that the vehicle is used for, and then a clustering algorithm is used to identify the route that is about to be driven at the start of each journey. If the correct journey has been identified, then the previously recorded examples of that journey can be used for prediction within the framework presented here (a similar approach was previously investigated in [38]). Driver behaviour prediction is very much still an open problem, and significant further work is required in this direction.

BIBLIOGRAPHY

[1] M. Ehsani, Y. Gao, and A. Emadi, *Modern Electric, Hybrid Electric, and Fuel Cell Vehicles*, 2nd ed. CRC Press, 2009.

[2] Deloitte. (2019) New market. New entrants. New challenges. Battery electric vehicles. (Date last accessed 10-July-2020). [Online]. Available: https://www2.deloitte.com/content/dam/Deloitte/uk/Documents/manufacturing/deloitte-uk-battery-electric-vehicles.pdf

[3] G. Zubi, R. Dufo-López, M. Carvalho, and G. Pasaoglu, "The lithium-ion battery: State of the art and future perspectives," *Renewable and Sustainable Energy Reviews*, vol. 89, pp. 292 – 308, 2018.

[4] J. B. Heywood, *Internal Combustion Engine Fundamentals*, 2nd ed. McGraw Hill, 2018.

[5] "An Overview of Costs for Vehicle Components, Fuels, Greenhouse Gas Emissions and Total Cost of Ownership Update 2017," 2017, (Date last accessed 10-July-2020). [Online]. Available: https://steps.ucdavis.edu/wp-content/uploads/2018/02/FRIES-MICHAEL-An-Overview-of-Costs-for-Vehicle-Components-Fuels-Greenhouse-Gas-Emissions-and-Total-Cost-of-Ownership-Update-2017-.pdf

[6] B. Zakeri and S. Syri, "Electrical energy storage systems: A comparative life cycle cost analysis," *Renewable and Sustainable Energy Reviews*, vol. 42, pp. 569–596, 2015.

[7] M. M. Kabir and D. E. Demirocak, "Degradation mechanisms in Li-ion batteries: a state-of-the-art review," *International Journal of Energy Research*, vol. 41, no. 14, pp. 1963–1986, 2017.

[8] S. Vazquez, S. M. Lukic, E. Galvan, L. G. Franquelo, and J. M. Carrasco, "Energy storage systems for transport and grid applications," *IEEE Transactions on Industrial Electronics*, vol. 57, no. 12, pp. 3881–3895, 2010.

[9] T. Markel and A. Simpson, "Cost-benefit analysis of plug-in hybrid electric vehicle technology," *World Electric Vehicle Journal*, vol. 1, no. 1, p. 294–301, Dec 2007.

[10] C. M. Martinez, X. Hu, D. Cao, E. Velenis, B. Gao, and M. Wellers, "Energy management in plug-in hybrid electric vehicles: Recent progress and a connected vehicles perspective," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 6, pp. 4534–4549, 2017.

[11] S. F. Tie and C. W. Tan, "A review of energy sources and energy management system in electric vehicles," *Renewable and Sustainable Energy Reviews*, vol. 20, pp. 82 – 102, 2013.

[12] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd, "OSQP: An operator splitting solver for quadratic programs," arXiv:1711.08013, 2017.

[13] Y. Nesterov and A. Nemirovskii, *Interior-Point Polynomial Algorithms in Convex Programming*. Society for Industrial and Applied Mathematics, 1994.

[14] B. Egardt, N. Murgovski, M. Pourabdollah, and L. Johannesson Mardh, "Electromobility studies based on convex optimization: Design and control issues regarding vehicle electrification," *IEEE Control Systems Magazine*, vol. 34, no. 2, pp. 32–49, 2014.

[15] M. Grant and S. Boyd, "CVX: Matlab software for disciplined convex programming, version 2.0 beta," 2013. [Online]. Available: http://cvxr.com/cvx

[16] ——, "Graph implementations for nonsmooth convex programs," in *Recent Advances in Learning and Control*, ser. Lecture Notes in Control and Information Sciences, V. Blondel, S. Boyd, and H. Kimura, Eds. Springer-Verlag Limited, 2008, pp. 95–110.

[17] N. Murgovski, L. Johannesson, J. Sjöberg, and B. Egardt, "Component sizing of a plug-in hybrid electric powertrain via convex optimization," *Mechatronics*, vol. 22, no. 1, pp. 106 – 120, 2012.

[18] S. East and M. Cannon, "An ADMM algorithm for MPC-based energy management in hybrid electric vehicles with nonlinear losses," in *2018 IEEE Conference on Decision and Control (CDC)*, 2018, pp. 2641–2646.

[19] J. Buerger, S. East, and M. Cannon, "Fast dual-loop nonlinear receding horizon control for energy management in hybrid electric vehicles," *IEEE Transactions on Control Systems Technology*, vol. 27, no. 3, pp. 1060–1070, 2019.

[20] S. East and M. Cannon, "Energy management in plug-in hybrid electric vehicles: Convex optimization algorithms for model predictive control (early access)," *IEEE Transactions on Control Systems Technology*, vol. -, pp. 1–13, 2019.

[21] ——, "Fast optimal energy management with engine on/off decisions for plug-in hybrid electric vehicles," *IEEE Control Systems Letters*, vol. 3, no. 4, pp. 1074–1079, 2019.

[22] ——, "ADMM for MPC with state and input constraints, and input nonlinearity," in *2018 Annual American Control Conference (ACC)*, 2018, pp. 4514–4519.

[23] ——, "Optimal power allocation in battery/supercapacitor electric vehicles using convex optimization," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 11, pp. 12 751–12 762, 2020.

[24] Y. Gao and M. Ehsani, "Design and control methodology of plug-in hybrid electric vehicles," *IEEE Transactions on Industrial Electronics*, vol. 57, no. 2, pp. 633–640, 2010.

[25] L. Serrao, A. Sciarretta, O. Grondin, A. Chasse, Y. Creff, D. D. Domenico, P. Pognant-Gros, C. Quérel, and L. Thibault, "Open issues in supervisory control of hybrid electric vehicles: A unified approach using optimal control methods," *Oil Gas Sci. Technol. - Rev. IFP Energies nouvelles*, vol. 68, pp. 23–33, 2013.

[26] R. Bellman and E. Lee, "History and development of dynamic programming," *IEEE Control Systems Magazine*, vol. 4, no. 4, pp. 24–28, 1984.

[27] D. Liberzon, *Calculus of Variations and Optimal Control Theory: A Concise Introduction*. USA: Princeton University Press, 2011.

[28] R. Bellman, "On the theory of dynamic programming," *Proceedings of the National Academy of Sciences*, vol. 38, no. 8, pp. 716–719, 1952.

[29] C.-C. Lin, H. Peng, J. W. Grizzle, and J.-M. Kang, "Power management strategy for a parallel hybrid electric truck," *IEEE Transactions on Control Systems Technology*, vol. 11, no. 6, pp. 839–849, 2003.

[30] Q. Gong, Y. Li, and Z. Peng, "Trip-based optimal power management of plug-in hybrid electric vehicles," *IEEE Transactions on Vehicular Technology*, vol. 57, no. 6, pp. 3393–3401, 2008.

[31] C. Zhang, A. Vahidi, P. Pisu, X. Li, and K. Tennant, "Role of terrain preview in energy management of hybrid electric vehicles," *IEEE Transactions on Vehicular Technology*, vol. 59, no. 3, pp. 1139–1147, 2010.

[32] S. J. Moura, H. K. Fathy, D. S. Callaway, and J. L. Stein, "A stochastic optimal control approach for power management in plug-in hybrid electric vehicles," *IEEE Transactions on Control Systems Technology*, vol. 19, no. 3, pp. 545–555, 2011.

[33] V. Ngo, T. Hofman, M. Steinbuch, and A. Serrarens, "Optimal control of the gearshift command for hybrid electric vehicles," *IEEE Transactions on Vehicular Technology*, vol. 61, no. 8, pp. 3531–3543, 2012.

[34] Z. Yuan, L. Teng, S. Fengchun, and H. Peng, "Comparative study of dynamic programming and Pontryagin's minimum principle on energy management for a parallel hybrid electric vehicle," *Energies*, vol. 6, no. 4, pp. 1–14, 2013.

[35] L. Li, C. Yang, Y. Zhang, L. Zhang, and J. Song, "Correctional DP-based energy management strategy of plug-in hybrid electric bus for city-bus route," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 7, pp. 2792–2803, 2015.

[36] X. Zeng and J. Wang, "A parallel hybrid electric vehicle energy management strategy using stochastic model predictive control with road grade preview," *IEEE Transactions on Control Systems Technology*, vol. 23, no. 6, pp. 2416–2423, 2015.

[37] V. Larsson, L. Johannesson, and B. Egardt, "Analytic solutions to the dynamic programming subproblem in hybrid vehicle energy management," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 4, pp. 1458–1467, 2015.

[38] V. Larsson, L. Johannesson Mårdh, B. Egardt, and S. Karlsson, "Commuter route optimized energy management of hybrid electric vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 3, pp. 1145–1154, 2014.

[39] Z. Chen, C. C. Mi, J. Xu, X. Gong, and C. You, "Energy management for a power-split plug-in hybrid electric vehicle based on dynamic programming and neural networks," *IEEE Transactions on Vehicular Technology*, vol. 63, no. 4, pp. 1567–1580, 2014.

[40] L. S. Pontryagin, V. G. Boltyanskii, R. V. Gamkrelidze, and E. F. Mishchenko, *The mathematical theory of optimal processes*. New York, NY: Interscience, 1962.

[41] Pesch, Hans, Plail, and Michael, "The maximum principle of optimal control: A history of ingenious ideas and missed opportunities," *Control and Cybernetics*, vol. 38, no. 4A, pp. 973–995, 2009.

[42] R. V. Gamkrelidze, "History of the discovery of the Pontryagin maximum principle," *Proceedings of the Steklov Institute of Mathematics*, vol. 304, no. 1, pp. 1–7, Jan. 2019.

[43] L. Serrao and G. Rizzoni, "Optimal control of power split for a hybrid electric refuse vehicle," in *2008 American Control Conference*, 2008, pp. 4498–4503.

[44] L. Tribioli and S. Onori, "Analysis of energy management strategies in plug-in hybrid electric vehicles: Application to the GM chevrolet volt," in *2013 American Control Conference*, 2013, pp. 5966–5971.

[45] Z. Chen, C. C. Mi, B. Xia, and C. You, "Energy management of power-split plug-in hybrid electric vehicles based on simulated annealing and Pontryagin's minimum principle," *Journal of Power Sources*, vol. 272, pp. 160 – 168, 2014.

[46] L. Tribioli, M. Barbieri, R. Capata, E. Sciubba, E. Jannelli, and G. Bella, "A real time energy management strategy for plug-in hybrid electric vehicles based on optimal control theory," *Energy Procedia*, vol. 45, pp. 949 – 958, 2014, aTI 2013 - 68th Conference of the Italian Thermal Machines Engineering Association.

[47] S. Stockar, V. Marano, G. Rizzoni, and L. Guzzella, "Optimal control for plug-in hybrid electric vehicle applications," in *Proceedings of the 2010 American Control Conference*, 2010, pp. 5024–5030.

[48] S. Stockar, V. Marano, M. Canova, G. Rizzoni, and L. Guzzella, "Energy-optimal control of plug-in hybrid electric vehicles for real-world driving cycles," *IEEE Transactions on Vehicular Technology*, vol. 60, no. 7, pp. 2949–2962, 2011.

[49] N. Kim, S. Cha, and H. Peng, "Optimal control of hybrid electric vehicles based on Pontryagin's minimum principle," *IEEE Transactions on Control Systems Technology*, vol. 19, no. 5, pp. 1279–1287, 2011.

[50] L. Tang, G. Rizzoni, and S. Onori, "Energy management strategy for HEVs including battery life optimization," *IEEE Transactions on Transportation Electrification*, vol. 1, no. 3, pp. 211–222, 2015.

[51] L. Serrao, S. Onori, A. Sciarretta, Y. Guezennec, and G. Rizzoni, "Optimal energy management of hybrid electric vehicles including battery aging," in *Proceedings of the 2011 American Control Conference*, 2011, pp. 2125–2130.

[52] G. Paganelli, S. Delprat, T. M. Guerra, J. Rimaux, and J. J. Santin, "Equivalent consumption minimization strategy for parallel hybrid powertrains," in *Vehicular Technology Conference. IEEE 55th Vehicular Technology Conference. VTC Spring 2002 (Cat. No.02CH37367)*, vol. 4, 2002, pp. 2076–2081 vol.4.

[53] L. Serrao, S. Onori, and G. Rizzoni, "ECMS as a realization of Pontryagin's minimum principle for HEV control," in *2009 American Control Conference*, 2009, pp. 3964–3969.

[54] C. Musardo, G. Rizzoni, Y. Guezennec, and B. Staccia, "A-ECMS: An adaptive algorithm for hybrid electric vehicle energy management," *European Journal of Control*, vol. 11, no. 4, pp. 509 – 524, 2005.

[55] P. Pisu and G. Rizzoni, "A comparative study of supervisory control strategies for hybrid electric vehicles," *IEEE Transactions on Control Systems Technology*, vol. 15, no. 3, pp. 506–518, 2007.

[56] A. Sciarretta, L. Serrao, P. Dewangan, P. Tona, E. Bergshoeff, C. Bordons, L. Charmpa, P. Elbert, L. Eriksson, T. Hofman, M. Hubacher, P. Isenegger, F. Lacandia, A. Laveau, H. Li, D. Marcos, T. Nüesch, S. Onori, P. Pisu, J. Rios, E. Silvas, M. Sivertsson, L. Tribioli, A.-J. van der Hoeven, and M. Wu, "A control benchmark on the energy management of a plug-in hybrid electric vehicle," *Control Engineering Practice*, vol. 29, pp. 287 – 298, 2014.

[57] J. Liu and H. Peng, "Modeling and control of a power-split hybrid vehicle," *IEEE Transactions on Control Systems Technology*, vol. 16, no. 6, pp. 1242–1251, 2008.

[58] P. Tulpule, V. Marano, and G. Rizzoni, "Effects of different PHEV control strategies on vehicle performance," in *2009 American Control Conference*, 2009, pp. 3950–3955.

[59] F. Tianheng, Y. Lin, G. Qing, H. Yanqing, Y. Ting, and Y. Bin, "A supervisory control strategy for plug-in hybrid electric vehicles based on energy demand prediction and route preview," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 5, pp. 1691–1700, 2015.

[60] F. Zhang, J. Xi, and R. Langari, "Real-time energy management strategy based on velocity forecasts using V2V and V2I communications," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 2, pp. 416–430, 2017.

[61] C. Zhang and A. Vahidi, "Route preview in energy management of plug-in hybrid vehicles," *IEEE Transactions on Control Systems Technology*, vol. 20, no. 2, pp. 546–553, 2012.

[62] A. Sciarretta and L. Guzzella, "Control of hybrid electric vehicles," *IEEE Control Systems Magazine*, vol. 27, no. 2, pp. 60–70, 2007.

[63] R. E. Kálmán, "Contributions to the theory of optimal control," in *Boletin de la Sociedad Matematica Mexicana*, vol. 5, no. 2, 1960, pp. 102–119.

[64] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed. New York, NY, USA: Springer, 2006.

[65] H. J. Ferreau, C. Kirches, A. Potschka, H. G. Bock, and M. Diehl, "qpOASES: a parametric active-set algorithm for quadratic programming," *Mathematical Programming Computation*, vol. 6, pp. 327–363, 2014.

[66] B. Hermans, A. Themelis, and P. Patrinos, "QPALM: A Newton-type proximal augmented Lagrangian method for quadratic programs," in *2019 IEEE 58th Conference on Decision and Control (CDC)*, 2019, pp. 4325–4330.

[67] A. Bemporad and P. Patrinos, "Simple and certifiable quadratic programming algorithms for embedded linear model predictive control," *IFAC Proceedings Volumes*, vol. 45, no. 17, pp. 14 – 20, 2012, 4th IFAC Conference on Nonlinear Model Predictive Control.

[68] J. L. Jerez, G. A. Constantinides, and E. C. Kerrigan, "An FPGA implementation of a sparse quadratic programming solver for constrained predictive control," in *Proceedings of the 19th ACM/SIGDA International Symposium on Field Programmable Gate Arrays (FPGA '11)*, 2011, p. 209–218.

[69] A. Bemporad, M. Morari, V. Dua, and E. N. Pistikopoulos, "The explicit linear quadratic regulator for constrained systems," *Automatica*, vol. 38, no. 1, pp. 3 – 20, 2002.

[70] J. H. Lee, "Model predictive control: Review of the three decades of development," *International Journal of Control, Automation and Systems*, vol. 9, no. 3, p. 415–424, Jun 2011.

[71] D. Mayne, J. Rawlings, C. Rao, and P. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789 – 814, 2000.

[72] J. Rawlings and D. Mayne, *Model predictive control: theory, computation, and design*, 2nd ed. Nob Hill Publishing, 2017.

[73] A. Bemporad and M. Morari, "Robust model predictive control: A survey," in *Robustness in Identification and Control*, A. Garulli and A. Tesi, Eds. London: Springer London, 1999, pp. 207–226.

[74] M. Farina, L. Giulioni, and R. Scattolini, "Stochastic linear model predictive control with chance constraints – a review," *Journal of Process Control*, vol. 44, pp. 53 – 67, 2016.

[75] B. Kouvaritakis and M. Cannon, *Model predictive control: classical, robust and stochastic*. Springer, 2016.

[76] S. Kermanil, S. Delprat, T. M. Guerra, and R. Trigui, "Predictive control for HEV energy management: experimental results," in *2009 IEEE Vehicle Power and Propulsion Conference*, 2009, pp. 364–369.

[77] F. Yan, J. Wang, and K. Huang, "Hybrid electric vehicle model predictive control torque-split strategy incorporating engine transient characteristics," *IEEE Transactions on Vehicular Technology*, vol. 61, no. 6, pp. 2458–2467, 2012.

[78] H. Borhan, A. Vahidi, A. M. Phillips, M. L. Kuang, I. V. Kolmanovsky, and S. Di Cairano, "MPC-based energy management of a power-split hybrid electric vehicle," *IEEE Transactions on Control Systems Technology*, vol. 20, no. 3, pp. 593–603, 2012.

[79] S. D. Cairano, D. Bernardini, A. Bemporad, and I. V. Kolmanovsky, "Stochastic MPC with learning for driver-predictive vehicle control and its application to HEV energy management," *IEEE Transactions on Control Systems Technology*, vol. 22, no. 3, pp. 1018–1031, 2014.

[80] M. Joševski, A. Katriniok, and D. Abel, "Scenario MPC for fuel economy optimization of hybrid electric powertrains on real-world driving cycles," in *2017 American Control Conference (ACC)*, 2017, pp. 5629–5635.

[81] M. Joševski and D. Abel, "Gear shifting and engine on/off optimal control in hybrid electric vehicles using partial outer convexification," in *2016 IEEE Conference on Control Applications (CCA)*, 2016, pp. 562–568.

[82] S. Xie, X. Hu, Z. Xin, and L. Li, "Time-efficient stochastic model predictive energy management for a plug-in hybrid electric bus with an adaptive reference state-of-charge advisory," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 7, pp. 5671–5682, 2018.

[83] G. Li, J. Zhang, and H. He, "Battery SOC constraint comparison for predictive energy management of plug-in hybrid electric bus," *Applied Energy*, vol. 194, pp. 578 – 587, 2017.

[84] C. Sun, S. J. Moura, X. Hu, J. K. Hedrick, and F. Sun, "Dynamic traffic feedback data enabled energy management in plug-in hybrid electric vehicles," *IEEE Transactions on Control Systems Technology*, vol. 23, no. 3, pp. 1075–1086, 2015.

[85] M. Koot, J. T. B. A. Kessels, B. de Jager, W. P. M. H. Heemels, P. P. J. van den Bosch, and M. Steinbuch, "Energy management strategies for vehicular electric power systems," *IEEE Transactions on Vehicular Technology*, vol. 54, no. 3, pp. 771–782, 2005.

[86] Z. Pu, X. Jiao, C. Yang, and S. Fang, "An adaptive stochastic model predictive control strategy for plug-in hybrid electric bus during vehicle-following scenario," *IEEE Access*, vol. 8, pp. 13 887–13 897, 2020.

[87] Y. Zhou, A. Ravey, and M.-C. Péra, "A survey on driving prediction techniques for predictive energy management of plug-in hybrid electric vehicles," *Journal of Power Sources*, vol. 412, pp. 480 – 495, 2019.

[88] C. Sun, X. Hu, S. J. Moura, and F. Sun, "Velocity predictors for predictive energy management in hybrid electric vehicles," *IEEE Transactions on Control Systems Technology*, vol. 23, no. 3, pp. 1197–1204, 2015.

[89] R. E. Bixby, "A brief history of linear and mixed-integer programming computation," *Documenta Mathematica*, pp. 107–121, 2012.

[90] M. Wright, "The interior-point revolution in optimization: History, recent developments, and lasting consequences," *Bulletin of the American Mathematical Society*, vol. 42, no. 1, pp. 39–56, Jan. 2005.

[91] D. P. Bertsekas, *Convex Optimization Theory*. Athenea Scientific, 2009.

[92] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan, *Linear Matrix Inequalities in System and Control Theory*, ser. Studies in Applied Mathematics. Philadelphia, PA: SIAM, Jun. 1994, vol. 15.

[93] X. Liu, P. Lu, and B. Pan, "Survey of convex optimization for aerospace applications," *Astrodynamics*, vol. 1, pp. 23–40, 2017.

[94] P. P. Khargonekar and M. A. Rotea, "Mixed H_2/H_∞ control: a convex optimization approach," *IEEE Transactions on Automatic Control*, vol. 36, no. 7, pp. 824–837, 1991.

[95] S. Prajna, P. A. Parrilo, and A. Rantzer, "Nonlinear control synthesis by convex optimization," *IEEE Transactions on Automatic Control*, vol. 49, no. 2, pp. 310–314, 2004.

[96] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[97] P. András, *Stochastic Programming*. Kluwer Academic Publishers, 1995.

[98] M. C. Campi and S. Garatti, "The exact feasibility of randomized solutions of uncertain convex programs," *SIAM Journal on Optimization*, vol. 19, no. 3, pp. 1211–1230, 2008.

[99] G. Schildbach, L. Fagiano, and M. Morari, "Randomized solutions to convex programs with multiple chance constraints," *SIAM Journal on Optimization*, vol. 23, no. 4, pp. 2479–2501, 2013.

[100] G. Schildbach, L. Fagiano, C. Frei, and M. Morari, "The scenario approach for stochastic model predictive control with bounds on closed-loop constraint violations," *Automatica*, vol. 50, no. 12, pp. 3009 – 3018, 2014.

[101] G. C. Calafiore and M. C. Campi, "The scenario approach to robust control design," *IEEE Transactions on Automatic Control*, vol. 51, no. 5, pp. 742–753, 2006.

[102] G. C. Calafiore and L. Fagiano, "Robust model predictive control via scenario optimization," *IEEE Transactions on Automatic Control*, vol. 58, no. 1, pp. 219–224, 2013.

[103] N. Murgovski, L. Johannesson, X. Hu, B. Egardt, and J. Sjöberg, "Convex relaxations in the optimal control of electrified vehicles," in *2015 American Control Conference (ACC)*, 2015, pp. 2292–2298.

[104] N. Murgovski, L. Johannesson, and J. Sjöberg, "Convex modeling of energy buffers in power control applications," *IFAC Proceedings Volumes*, vol. 45, no. 30, pp. 92 – 99, 2012, 3rd IFAC Workshop on Engine and Powertrain Control, Simulation and Modeling.

[105] X. Hu, N. Murgovski, L. M. Johannesson, and B. Egardt, "Comparison of three electrochemical energy buffers applied to a hybrid bus powertrain with simultaneous optimal sizing and energy management," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 3, pp. 1193–1205, 2014.

[106] N. Murgovski, L. M. Johannesson, and J. Sjöberg, "Engine on/off control for dimensioning hybrid electric powertrains via convex optimization," *IEEE Transactions on Vehicular Technology*, vol. 62, no. 7, pp. 2949–2962, 2013.

[107] T. Nüesch, P. Elbert, M. Flankl, C. Onder, and L. Guzzella, "Convex optimization for the energy management of hybrid electric vehicles considering engine start and gearshift costs," *Energies*, vol. 7, no. 2, p. 834–856, Feb 2014.

[108] M. Pourabdollah, N. Murgovski, A. Grauers, and B. Egardt, "An iterative dynamic programming/convex optimization procedure for optimal sizing and energy management of PHEVs," *IFAC Proceedings Volumes*, vol. 47, no. 3, pp. 6606 – 6611, 2014, 19th IFAC World Congress.

[109] N. Murgovski, L. M. Johannesson, and B. Egardt, "Optimal battery dimensioning and control of a CVT PHEV powertrain," *IEEE Transactions on Vehicular Technology*, vol. 63, no. 5, pp. 2151–2161, 2014.

[110] Z. Ma, N. Murgovski, B. Egardt, and S. Cui, "Convex modeling for optimal battery sizing and control of an electric variable transmission powertrain," *Oil Gas Sci. Technol. - Rev. IFP Energies nouvelles*, vol. 74, p. 25, 2019.

[111] M. Pourabdollah, N. Murgovski, A. Grauers, and B. Egardt, "Optimal sizing of a parallel PHEV powertrain," *IEEE Transactions on Vehicular Technology*, vol. 62, no. 6, pp. 2469–2480, 2013.

[112] X. Hu, S. J. Moura, N. Murgovski, B. Egardt, and D. Cao, "Integrated optimization of battery sizing, charging, and power management in plug-in hybrid electric vehicles," *IEEE Transactions on Control Systems Technology*, vol. 24, no. 3, pp. 1036–1043, 2016.

[113] X. Hu, N. Murgovski, L. Johannesson, and B. Egardt, "Energy efficiency analysis of a series plug-in hybrid electric bus with different energy management strategies and battery sizes," *Applied Energy*, vol. 111, pp. 1001 – 1009, 2013.

[114] S. Hadj-Said, G. Colin, A. Ketfi-Cherif, and Y. Chamaillard, "Convex optimization for energy management of parallel hybrid electric vehicles," *IFAC-PapersOnLine*,

vol. 49, no. 11, pp. 271 – 276, 2016, 8th IFAC Symposium on Advances in Automotive Control AAC 2016.

[115] K. C. Toh, M. J. Todd, and R. H. Tütüncü, "SDPT3 — a Matlab software package for semidefinite programming, version 1.3," *Optimization Methods and Software*, vol. 11, no. 1-4, pp. 545–581, 1999.

[116] R. S. Vadamalu, D. Buch, H. Xiao, and C. Beidl, "Energy management of hybrid electric powertrain using predictive trajectory planning based on direct optimal control," *IFAC-PapersOnLine*, vol. 48, no. 25, pp. 236 – 241, 2015, 16th IFAC Workshop on Control Applications of Optimization 2015.

[117] R. S. Vadamalu and C. Beidl, "Online MPC based PHEV energy management using conic interior-point methods," in *2016 IEEE Intelligent Vehicles Symposium (IV)*, 2016, pp. 466–471.

[118] P. Elbert, T. Nüesch, A. Ritter, N. Murgovski, and L. Guzzella, "Engine on/off control for the energy management of a serial hybrid electric bus via convex optimization," *IEEE Transactions on Vehicular Technology*, vol. 63, no. 8, pp. 3549–3559, 2014.

[119] D. P. Bertsekas, "Projected Newton methods for optimization problems with simple constraints," *SIAM Journal on Control and Optimization*, vol. 20, no. 2, pp. 221–246, 1982.

[120] Y. Wang and S. Boyd, "Fast model predictive control using online optimization," *IEEE Transactions on Control Systems Technology*, vol. 18, no. 2, pp. 267–278, 2010.

[121] A. Domahidi, A. U. Zgraggen, M. N. Zeilinger, M. Morari, and C. N. Jones, "Efficient interior point methods for multistage problems arising in receding horizon control," in *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, 2012, pp. 668–674.

[122] C. V. Rao, S. J. Wright, and J. B. Rawlings, "Application of interior-point methods to model predictive control," *Journal of Optimization Theory and Applications*, vol. 99, pp. 723–757, 1998.

[123] R. A. Bartlett, A. Wachter, and L. T. Biegler, "Active set vs. interior point strategies for model predictive control," in *Proceedings of the 2000 American Control Conference (ACC)*, vol. 6, 2000, pp. 4229–4233 vol.6.

[124] F. Rey, P. Hokayem, and J. Lygeros, "A tailored ADMM approach for power coordination in variable speed drives," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 7403 – 7408, 2017, 20th IFAC World Congress.

[125] R. Rostami, G. Costantini, and D. Görges, "ADMM-based distributed model predictive control: Primal and dual approaches," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, 2017, pp. 6598–6603.

[126] T. V. Dang, K. V. Ling, and J. M. Maciejowski, "Embedded ADMM-based QP solver for MPC with polytopic constraints," in *2015 European Control Conference (ECC)*, 2015, pp. 3446–3451.

[127] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, p. 1–122, Jan. 2011.

[128] V. A. Adona, M. L. N. Gonçalves, and J. G. Melo, "Iteration-complexity analysis of a generalized alternating direction method of multipliers," *Journal of Global Optimization*, vol. 73, pp. 331–348, 2017.

[129] R. A. Horn and C. R. Johnson, *Matrix Analysis*, 2nd ed. USA: Cambridge University Press, 2012.

[130] P. Bonami, M. Kilinç, and J. Linderoth, "Algorithms and software for convex mixed integer nonlinear programs," in *Mixed Integer Nonlinear Programming*, J. Lee and S. Leyffer, Eds. New York, NY: Springer New York, 2012, pp. 1–39.

[131] J. Kronqvist, D. E. Bernal, A. Lundell, and I. E. Grossmann, "A review and comparison of solvers for convex MINLP," *Optimization and Engineering*, vol. 20, pp. 397–455, 2019.

[132] R. Takapoui, N. Moehle, S. Boyd, and A. Bemporad, "A simple effective heuristic for embedded mixed-integer quadratic programming," *International Journal of Control*, vol. 93, no. 1, pp. 2–12, 2020.

[133] A. Shapiro, "Asymptotic behavior of optimal solutions in stochastic programming," *Mathematics of Operations Research*, vol. 18, no. 4, pp. 829–845, 1993.

[134] Z. Qureshi, S. East, and M. Cannon, "Parallel ADMM for robust quadratic optimal resource allocation problems," in *2019 American Control Conference (ACC)*, 2019, pp. 3402–3407.

[135] F. Ju, Q. Zhang, W. Deng, and J. Li, "Review of structures and control of battery-supercapacitor hybrid energy storage system for electric vehicles," *Advances in Battery Manufacturing, Services, and Management Systems*, pp. 303–318, 2016.

[136] A. L. Allègre, A. Bouscayrol, and R. Trigui, "Flexible real-time control of a hybrid energy storage system for electric vehicles," *IET Electrical Systems in Transportation*, vol. 3, no. 3, pp. 79–85, 2013.

[137] B. Nguyen, R. German, J. P. F. Trovão, and A. Bouscayrol, "Real-time energy management of battery/supercapacitor electric vehicles based on an adaptation of Pontryagin's minimum principle," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 1, pp. 203–212, 2019.

[138] S. Dusmez and A. Khaligh, "A supervisory power-splitting approach for a new ultracapacitor-battery vehicle deploying two propulsion machines," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 3, pp. 1960–1971, 2014.

[139] M. Masih-Tehrani, M. R. Ha'iri-Yazdi, V. Esfahanian, and A. Safaei, "Optimum sizing and optimum energy management of a hybrid energy storage system for lithium battery life improvement," *Journal of Power Sources*, vol. 244, pp. 2–10, 2013.

[140] A. Santucci, A. Sorniotti, and C. Lekakou, "Power split strategies for hybrid energy storage systems for vehicular applications," *Journal of Power Sources*, vol. 258, pp. 395–407, 2014.

[141] Z. Song, H. Hofmann, J. Li, X. Han, and M. Ouyang, "Optimization for a hybrid energy storage system in electric vehicles using dynamic programing approach," *Applied Energy*, vol. 139, no. Dc, pp. 151–162, 2015.

[142] J. Shen and A. Khaligh, "A supervisory energy management control strategy in a battery/ultracapacitor hybrid energy storage system," *IEEE Transactions on Transportation Electrification*, vol. 1, no. 3, pp. 223–231, 2015.

[143] C. Romaus, K. Gathmann, and J. Böcker, "Optimal energy management for a hybrid energy storage system for electric vehicles based on stochastic dynamic programming," *2010 IEEE Vehicle Power and Propulsion Conference (VPPC)*, 2010.

[144] C. Zheng, W. Li, and Q. Liang, "An energy management strategy of hybrid energy storage systems for electric vehicle applications," *IEEE Transactions on Sustainable Energy*, vol. 9, no. 4, pp. 1880–1888, 2018.

[145] L. Wang, Y. Wang, C. Liu, D. Yang, and Z. Chen, "A power distribution strategy for hybrid energy storage system using adaptive model predictive control," *IEEE Transactions on Power Electronics*, vol. 35, no. 6, pp. 5897–5906, 2020.

[146] B. Hredzak, V. G. Agelidis, and M. Jang, "A model predictive control system for a hybrid battery-ultracapacitor power source," *IEEE Transactions on Power Electronics*, vol. 29, no. 3, pp. 1469–1479, 2014.

[147] M. Choi, J. Lee, and S. Seo, "Real-time optimization for power management systems of a battery/supercapacitor hybrid energy storage system in electric vehicles," *IEEE Transactions on Vehicular Technology*, vol. 63, no. 8, pp. 3600–3611, 2014.

[148] R. Wang, J. Peng, Y. Zhou, H. Liao, and Z. Huang, "Primal-dual interior-point method based energy distribution optimization for semi-active hybrid energy storage system," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 14 477 – 14 482, 2017.

[149] A. Mahmoudzadeh Andwari, A. Pesiridis, S. Rajoo, R. Martinez-Botas, and V. Esfahanian, "A review of battery electric vehicle technology and readiness levels," *Renewable and Sustainable Energy Reviews*, vol. 78, pp. 414 – 430, 2017.

[150] M. Wieczorek and M. Lewandowski, "A mathematical representation of an energy management strategy for hybrid energy storage system in electric vehicle and real time optimization using a genetic algorithm," *Applied Energy*, vol. 192, pp. 222 – 233, 2017.

[151] A. Barré, B. Deguilhem, S. Grolleau, M. Gérard, F. Suard, and D. Riu, "A review on lithium-ion battery ageing mechanisms and estimations for automotive applications," *Journal of Power Sources*, vol. 241, pp. 680 – 689, 2013.

[152] J. M. Reniers, G. Mulder, and D. A. Howey, "Review and performance comparison of mechanical-chemical degradation models for lithium-ion batteries," *Journal of The Electrochemical Society*, vol. 166, no. 14, pp. A3189–A3200, 2019.

[153] A. Tomaszewska, Z. Chu, X. Feng, S. O'Kane, X. Liu, J. Chen, C. Ji, E. Endler, R. Li, L. Liu, Y. Li, S. Zheng, S. Vetterlein, M. Gao, J. Du, M. Parkes, M. Ouyang, M. Marinescu, G. Offer, and B. Wu, "Lithium-ion battery fast charging: A review," *eTransportation*, vol. 1, p. 100011, 2019.

[154] G. Banjac, P. Goulart, B. Stellato, and S. Boyd, "Infeasibility detection in the alternating direction method of multipliers for convex optimization," *Journal of Optimization Theory and Applications*, vol. 183, no. 2, pp. 490–519, 2019.

[155] T. Bocklisch, "Hybrid energy storage systems for renewable energy applications," *Energy Procedia*, vol. 73, pp. 103 – 111, 2015, 9th International Renewable Energy Storage Conference (IRES).