Energy Management in Plug-in Hybrid Electric Vehicles: Convex Optimization Algorithms for Model Predictive Control

Sebastian East, Mark Cannon

***Abstract*—This paper details an investigation into the computational performance of algorithms used for solving a convex formulation of the optimization problem associated with model predictive control for energy management in hybrid electric vehicles with nonlinear losses. A projected interior point method is proposed, where the size and complexity of the Newton step matrix inversion is reduced by applying inequality constraints on the control input as a projection, and its properties are demonstrated through simulation in comparison with an alternating direction method of multipliers (ADMM) algorithm, and general purpose convex optimization software CVX. It is found that the ADMM algorithm has favourable properties when a solution with modest accuracy is required, whereas the projected interior point method is favourable when high accuracy is required, and that both are significantly faster than CVX.**

***Index Terms*—alternating direction method of multipliers, energy management, interior point method, model predictive control, plug-in hybrid electric vehicles.**

I. Introduction

INCREASED electricification of road vehicles has been identified as a key short term solution to important societal issues including climate change and air pollution [1]. Plug-in hybrid electic vehicles (PHEVs), where an electric propulsion system is complemented with an internal combustion engine, are currently a common configuration. Although the low energy density and lengthy recharge time of lithium ion batteries limits the viability of all-electric powertrains, analysis of daily driving behaviour reveals that 50% of internal combustion powered miles can be powered electrically using a hybrid vehicle with an all-electric range of just 40 miles [2]. The inclusion of an additional power source, however, introduces a challenging problem: at each instant during a given journey, how much power should be delivered from the motor, and how much should be delivered from the engine?

This is known as the energy management problem [3], and a simple heuristic is a charge depleting/charge sustaining strategy, where power is delivered from only the electric motor until the battery is sufficiently depleted, and then the vehicle is operated in a charge sustaining mode until the end of the journey [2]. It has, however, been demonstrated that significant savings in fuel consumption can be made by delivering power from both the motor and engine simultaneously, and modulating the fraction delivered from each throughout the journey in what is known as a 'blended mode' [4]. There are several methods for controlling the powertrain in this way, and the globally optimal solution can be obtained for complex, nonlinear models and/or integer conrol decisions (such as gear selection) using Dynamic Programming [5], [6], but this approach is too computationally demanding for an online solution. Other studies have investigated methods based on Pontryagin's Minimum Principle [7]–[9], although it is challenging to enforce complex constraints, such as the general state of charge constraint or engine switching, whilst still guaranteeing optimality.

Model predictive control (MPC) has shown promise in this application due to the inherent robustness to uncertainty in both the vehicle model and prediction of future driving behaviour [10], and nonlinear models of losses can be used throughout the hybrid powertrain for improved performance [11]. The associated online optimization problem is still computationally intractable when gear selection and engine switching are considered, so these elements are commonly removed from the problem or optimized externally [11]–[14], and it has been demonstrated that the power balance alone can be formulated, with nonlinear losses and without simplification, as a convex optimization problem with linear state dynamics [15].

For the last 30 years, the most popular algorithms for solving inequality constrained optimization problems have been interior point methods [16]. Originally formulated as 'primal' methods by approximating inequality constraints in an optimization problem with logarithmic barrier functions, their inherent ill-conditioning and numerical inefficiency rendered interior point methods ineffective until the publication of Karmakar's method [17] in 1984. The subsequently developed 'primal-dual' interior point methods [18] displayed excellent theoretical and practical properties, including polynomial complexity and a near constant number of iterations with variations in problem size, and today, a large volume of research output in the field of optimization for MPC is dedicated to the development and application of primal-dual interior point methods [19]–[21].

In the recently published literature, algorithms for solving convex formulations of the energy management problem have not been investigated and the optimization problem has normally been solved using general purpose convex optimization software [13], [14], [22], [23], with the exception of the alternating direction method of multipliers (ADMM) algorithm presented in [15]. The first contribution of this paper is a

S. East and M. Cannon are with the Department of Engineering Science, University of Oxford, Oxford, OX1 3PJ (e-mail: sebastian.east@eng.ox.ac.uk; mark.cannon@eng.ox.ac.uk)

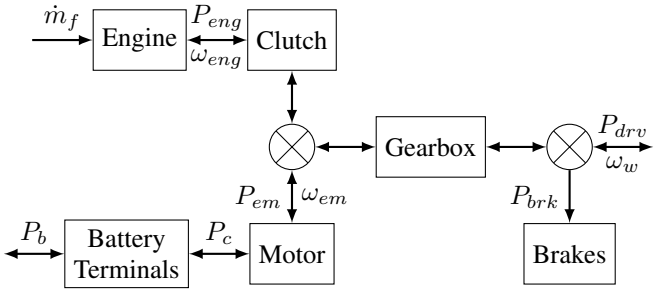

Fig. 1. A diagram of a simplified model of a parallel PHEV powertrain, labelled with the main flows of power and rotational speeds.

projected interior point solver for this problem, where the size of the matrix inversion associated with the primal-dual Newton step is reduced by enforcing the element-wise inequality constraints on the decision variable as a projection, thereby reducing the computational requirement of each iteration. The algorithm is not domain specific, and is applicable to any MPC optimization problem with linear dynamics, a separable convex cost function of the control variable, and upper and lower bounds on the control and state variables.

A primary motivation of this paper was to determine the relative computational benefits of second-order and first-order methods for the convex PHEV energy management formulation, and the second contribution is a set of numerical studies where the performance of the projected interior point algorithm is compared with the ADMM algorithm of [15]. In these studies we demonstrate that the projected interior point method has superior convergence properties, but requires more time to obtain a solution with modest accuracy, and consequently is only suitable for real-time solutions over shorter horizons (in this case fewer than 500 samples). We also demonstrate that both methods are significantly faster than CVX [24], and in ADMM (using an improved implementation from [15]) we demonstrate the first 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.

The paper is organised as follows: in section II the energy management problem, MPC framework, and convex reformulation are defined, and section III details the projected interior point method. The ADMM algorithm of [15] is stated in section IV, numerical experiments are presented in section V, and the paper is concluded in section VI.

II. Energy Management Problem & Model Predictive Control Framework

Fig. 1 shows a simplified diagram of a parallel PHEV powertrain, and illustrates the energy transfers that are considered as part of the energy management problem. At a given time, t, the mass flow rate of fuel delivered to the engine, $\dot{m}_f$, can be described by a time-varying function, f, of engine output power, P_{eng}, and engine shaft speed, ω_{eng}, as

$$\dot{m}_f(t) = f(P_{eng}(t), \omega_{eng}(t), \sigma(t), t), \tag{1}$$

where σ describes the state of the engine and clutch engagement by

$$\sigma(t) = \begin{cases} 1 & \text{engine on, clutch engaged} \\ 0 & \text{engine off, clutch disengaged} \end{cases}. \tag{2}$$

Similarly, the rate of consumption of the battery's internal chemical energy, P_b, can be described by a time varying function, g, of battery output power (i.e motor input power), P_c, which can in turn be described by a time varying function, h, of motor output power, P_{em}, and motor shaft speed, ω_{em}:

$$\begin{aligned} P_b(t) &= g(P_c(t), t) \\ P_c(t) &= h(P_{em}(t), \omega_{em}(t), t). \end{aligned} \tag{3}$$

Therefore, the state of charge of the battery, E, is given at time t by

$$E(t) = E(0) - \int_0^t P_b(t)\ \mathrm{d}t. \tag{4}$$

The engine output power that is delivered through the clutch is combined additively with power from the motor through a coupling device to drive the gearbox. Assuming that all drivetrain components are 100% mechanically efficient, the power delivered to the wheels (i.e the power demanded by the driver), P_{drv}, is given by

$$P_{drv}(t) = P_{em}(t) + P_{eng}(t) + P_{brk}(t), \tag{5}$$

where P_{brk} is the power extracted from the system by the mechanical brakes. Assuming a discrete variable transmission, the rotational velocities of the engine and motor shafts are given as a function of the rotational velocity of the wheels, ω_w, as

$$\omega_{em}(t) = r(t)\omega_w(t),\ \omega_{eng}(t) = \sigma(t)r(t)\omega_w(t) \tag{6}$$

where $r : \mathbb{R} \to \{r_1, \ldots, r_{N_g}\}$, and N_g is the number of available gear ratios.

The engine has upper and lower limits on torque, $\overline{T}_{eng}$ and $\underline{T}_{eng}$, that are functions of engine speed, so limits on engine power are given by

$$\begin{aligned} P_{eng}(t) \geq \underline{P}_{eng}(t) &= \underline{T}_{eng}(\omega_{eng}(t))\omega_{eng}(t) \\ P_{eng}(t) \leq \overline{P}_{eng}(t) &= \overline{T}_{eng}(\omega_{eng}(t))\omega_{eng}(t). \end{aligned} \tag{7}$$

The limits on motor power can be given similarly as

$$\begin{aligned} P_{em}(t) \geq \underline{P}_{em}(t) &= \underline{T}_{em}(\omega_{em}(t))\omega_{em}(t) \\ P_{em}(t) \leq \overline{P}_{em}(t) &= \overline{T}_{em}(\omega_{em}(t))\omega_{em}(t), \end{aligned} \tag{8}$$

where $\overline{T}_{em}$ and $\underline{T}_{em}$ are upper and lower limits on motor torque. The engine and motor have static limits on rotational speed, given by

$$\underline{\omega}_{em} \leq \omega_{em}(t) \leq \overline{\omega}_{em} \quad \text{and} \quad \underline{\omega}_{eng} \leq \omega_{eng}(t) \leq \overline{\omega}_{eng}, \tag{9}$$

and the battery has static limits on state-of-charge and rate of charge and discharge, given by

$$\underline{E} \leq E(t) \leq \overline{E} \quad \text{and} \quad \underline{P}_b \leq P_b(t) \leq \overline{P}_b. \tag{10}$$

The above system is under-constrained in three degrees of freedom: the fraction of total driver demand power delivered from motor, engine, and brakes; the engine switching and

clutch engagement; and the gear selection. Consequently, the parameters $P_{eng}(t)$, $P_{brk}(t)$, $\sigma(t)$, and $r(t)$ must be actively controlled ($P_{em}(t)$ is given in terms of $P_{eng}(t)$ and $P_{brk}(t)$ by (5)). The energy management problem can therefore be written as an open-loop optimal control problem for a journey of length T as

$$\begin{aligned}\min_{\sigma(t),r(t),P_{eng}(t),P_{brk}(t)} &\int_0^T \dot{m}_f(t)\ \mathrm{d}t \\ &\text{s.t. } (1)-(10)\ \forall t\in[0,T]\end{aligned} \tag{11}$$

Note that ω_w and P_{drv} (which determine the vehicle's speed and acceleration) are not affected in the control problem; the principle of the controller is to always meet the powertrain output demanded by the driver, and to not affect the overall driving behaviour of the vehicle.

A. MPC Framework

If implemented in a real vehicle, the solution found from (11) will be suboptimal as it is impossible to model the powertrain components with complete accuracy, and because the problem is dependent on future disturbance variables, P_{drv} and ω_w, that are impossible to exactly predict *a priori*. A MPC framework can be used to reduce these limitations, where instead of solving a single instance of the open-loop control problem, the control variables are repeatedly updated as the journey progresses [11]. This allows the predictions of driver behaviour to be improved as new information becomes available, and provides feedback on the vehicle state (i.e the battery state of charge), thus providing a degree of robustness to modelling and prediction errors. We describe a MPC framework for the energy management problem in this section.

Throughout the following text, the notation $\hat{x}$ is used for a variable, $x(t)$, to refer to the discretely sampled prediction used within the MPC framework, as opposed to the physical signals and states described in the previous section. At each control variable update instant, a discretely sampled prediction of demand power and wheel speed is made as

$$\hat{P}_{drv}=(\hat{P}_{drv,0},\dots,\hat{P}_{drv,N-1}),\ \hat{\omega}_w=(\hat{\omega}_{w,0},\dots,\hat{\omega}_{w,N-1})$$

for $k=0,\dots,N-1$, where the sampling period δ is assumed to be constant, and the prediction horizon is given by N. Although the prediction aspect of the energy management problem is still very much an open issue [25], the focus of this paper is on the subsequent optimization problem, so it is assumed that an accurate method is available to the controller and this aspect is not addressed further. The engine and motor loss maps can be approximated with quasi-static quadratic functions, $\hat{f}$ and $\hat{h}$, [11], [13], [14] as

$$\begin{aligned}\hat{\dot{m}}_{f,k}&=\hat{f}(\hat{P}_{eng,k},\hat{\omega}_{eng,k},\hat{\sigma}_k)\\ &=\hat{\sigma}_k[\alpha_2(\hat{\omega}_{eng,k})\hat{P}_{eng,k}{}^2+\alpha_1(\hat{\omega}_{eng,k})\hat{P}_{eng,k}+\alpha_0(\hat{\omega}_{eng,k})]\\ \hat{P}_{c,k}&=\hat{h}(\hat{P}_{em,k},\hat{\omega}_{em,k})\\ &=\beta_2(\hat{\omega}_{em,k})\hat{P}_{em,k}^2+\beta_1(\hat{\omega}_{em,k})\hat{P}_{em,k}+\beta_0(\hat{\omega}_{em,k})\end{aligned} \tag{12}$$

where $\alpha_2(\hat{\omega}_{eng,k})>0\ \forall\ \hat{\omega}_{eng,k}$, and $\beta_2(\hat{\omega}_{em,k})>0\ \forall\ \hat{\omega}_{em,k}$. The battery is commonly modelled as an equivalent circuit of internal resistance [11]–[15], [26], [27] as:

$$\begin{aligned}\hat{P}_{b,k}=&\hat{g}(\hat{P}_{em,k},\hat{\omega}_{em,k},k)\\ =&\frac{V_{oc,k}}{2R_k}\left(1-\sqrt{1-\frac{4R_k}{V_{oc,k}^2}\hat{h}(\hat{P}_{em,k},\hat{\omega}_{em,k})}\right)\end{aligned}$$

where $V_{oc,k}$ and R_k are the open circuit voltage and internal resistance. The limits on engine, motor, and discharge power (7,8) are now given in discrete time as

$$\begin{aligned}\underline{P}_{eng,k}&=\underline{T}_{eng}(\hat{\omega}_{eng,k})\hat{\omega}_{eng,k}\\ \overline{P}_{eng,k}&=\overline{T}_{eng}(\hat{\omega}_{eng,k})\hat{\omega}_{eng,k}\\ \underline{P}_{em,k}&=\underline{T}_{em}(\hat{\omega}_{em,k})\hat{\omega}_{em,k}\\ \overline{P}_{em,k}&=\overline{T}_{em}(\hat{\omega}_{em,k})\hat{\omega}_{em,k}\end{aligned}$$

and the MPC optimization at time t is then given as an approximation of (11) with Euler method integration of the state dynamics (4) as

$$\begin{aligned}\min_{\hat{\sigma},\hat{r},\hat{P}_{eng},\hat{P}_{brk}}\ &\sum_{k=0}^{N-1}\delta\hat{\dot{m}}_{f,k}\\ \text{s.t. }\ &\hat{E}_0=E(t)\\ &\left.\begin{aligned}&\hat{E}_{k+1}=\hat{E}_k-\delta\hat{g}(\hat{P}_{em,k},\hat{\omega}_{em,k},k)\\ &\hat{P}_{drv,k}=\hat{P}_{em,k}+\hat{P}_{eng,k}+\hat{P}_{brk,k}\\ &\hat{\omega}_{em,k}=\hat{r}_k\hat{\omega}_{w,k}\\ &\hat{\omega}_{eng,k}=\hat{\sigma}_k\hat{r}_k\hat{\omega}_{w,k}\\ &\hat{r}_k\in\{r_1,\dots,r_{N_g}\}\\ &\hat{\sigma}_k\in\{0,1\}\\ &\underline{P}_{eng,k}\le\hat{P}_{eng,k}\le\overline{P}_{eng,k}\\ &\underline{P}_{em,k}\le\hat{P}_{em,k}\le\overline{P}_{em,k}\\ &\underline{\omega}_{em}\le\hat{\omega}_{em,k}\le\overline{\omega}_{em}\\ &\underline{\omega}_{eng}\le\hat{\omega}_{eng,k}\le\overline{\omega}_{eng}\\ &\underline{P}_b\le\hat{P}_{b,k}\le\overline{P}_b\\ &\underline{E}\le\hat{E}_{k+1}\le\overline{E}\end{aligned}\right\}\forall k\end{aligned} \tag{13}$$

At each control variable update instance, the first elements of the vectors of optimization variables, $\hat{\sigma}_0^\star$, $\hat{r}_0^\star$, $\hat{P}_{eng,0}^\star$, and $\hat{P}_{brk,0}^\star$ are implemented as $\sigma(t)$, $r(t)$, $P_{eng}(t)$, and $P_{brk}(t)$, where $\hat{\sigma}^\star$, $\hat{r}^\star$, $\hat{P}_{eng}^\star$, and $\hat{P}_{brk}^\star$ are the minimizing arguments of (13).

B. Convex Reformulation

Whilst the MPC framework provides a degree of robustness to prediction and modelling errors, problem (13) is challenging to solve as the cost function is non-convex, it is subject to nonlinear, non-convex constraints, and it has $2N$ discrete decision variables (N is likely to be in the thousands for journeys longer than 15 mins, assuming an update frequency of approximately 1Hz). The global minimum of the problem can be found using Dynamic Programming, however this is a computationally demanding approach and not suitable for an online solution [15]. Instead, it is possible to reduce the

complexity of the problem by determining the three variables $\hat{\sigma}$, $\hat{r}$, and $\hat{P}_{brk}$ using heuristic rules [11] or an external optimization routine [13], leaving the power balance between the engine and motor as the only under-constrained parameter. It has previously been demonstrated in [15] that the resulting problem is convex when using the battery power, $\hat{P}_b$, as the decision variable. This is the approach that is taken in this paper, as discussed below.

Assuming an external method for estimating the engine switching behaviour, the set of timesteps at which the engine is switched on and the clutch is engaged is given by

$$\mathcal{P} = \{k : \hat{\sigma}(k) = 1\},$$

and also assuming a method for pre-determining the use of mechanical brake, we re-define the drive demand power from (5) as

$$\hat{P}_{drv,k} = \hat{P}_{em,k} + \hat{P}_{eng,k},$$

so that $P_{brk,k}$ is no longer an optimization variable in (13). Finally, by assuming that the gear selection, $\hat{r}$, is also pre-determined, $\hat{\omega}_{eng,k}$ and $\hat{\omega}_{em,k}$ are defined for all k, so the engine and motor models (12) can be reduced to time-varying polynomial functions of the form

$$\begin{aligned}\hat{m}_f(k) &= \hat{f}_k(\hat{P}_{eng,k}) \\ &= \alpha_{2,k}\hat{P}_{eng,k}{}^2 + \alpha_{1,k}\hat{P}_{eng,k} + \alpha_{0,k}, \\ \hat{P}_c(k) &= \hat{h}_k(\hat{P}_{em,k}) \\ &= \beta_{2,k}\hat{P}_{em,k}^2 + \beta_{1,k}\hat{P}_{em,k} + \beta_{0,k}.\end{aligned}$$

These functions are strictly convex ($\alpha_{2,k}, \beta_{2,k} > 0$), and we then ensure that $\hat{g}_k$ is also strictly convex by assuming that V_{oc} and R are independent of state of charge (and in this case, constant), so that the battery model is approximated by

$$\hat{P}_{b,k} = \hat{g}_k(\hat{P}_{em,k}) = \frac{V_{oc}^2}{2R}\left(1 - \sqrt{1 - \frac{4R}{V_{oc}^2}\hat{h}_k(\hat{P}_{em,k})}\right).$$

We update the limits on $\hat{P}_{eng,k}$ and $\hat{P}_{em,k}$ to ensure that $\hat{f}_k$, $\hat{h}_k$, and $\hat{g}_k$ are all non-decreasing and real-valued as

$$\begin{aligned}\underline{P}_{eng,k} &= \max\left\{\underline{P}_{eng,k}, -\frac{\alpha_{1,k}}{2\alpha_{2,k}}\right\} \\ \underline{P}_{em,k} &= \max\left\{\underline{P}_{em,k}, -\frac{\beta_{1,k}}{2\beta_{2,k}}\right\} \\ \overline{P}_{em,k} &= \min\left\{\overline{P}_{em,k}, \max\left\{x : 1 - \frac{4R}{V_{oc}^2}\hat{h}_k(x) = 0\right\}\right\}.\end{aligned}$$

This also ensures that $\hat{g}_k$ is a one-to-one function, so we can define

$$\hat{P}_{b,k} = \hat{g}_k(\hat{P}_{em,k}) \Leftrightarrow \hat{P}_{em,k} = \hat{g}_k{}^{-1}(\hat{P}_{b,k})$$

where

$$\hat{g}_k^{-1}(\hat{P}_{b,k}) = -\frac{\beta_{1,k}}{2\beta_{2,k}} + \sqrt{-\frac{R\hat{P}_{b,k}^2}{\beta_{2,k}V_{oc}^2} + \frac{\hat{P}_{b,k} - \beta_{0,k}}{\beta_{2,k}} + \frac{\beta_{1,k}{}^2}{4\beta_{2,k}{}^2}}$$

Using this definition of $\hat{g}_k^{-1}$, it is known that for $k \in \mathcal{P}$,

$$\hat{P}_{eng,k} = \hat{P}_{drv,k} - \hat{g}_k^{-1}(\hat{P}_{b,k})$$

where the corresponding limits on $\hat{P}_b$ are

$$\begin{aligned}\overline{P}_{b,k} &= \min\{\overline{P}_b, \hat{g}_k(\overline{P}_{em,k}), \hat{g}_k(\hat{P}_{drv,k} - \underline{P}_{eng,k})\} \\ \underline{P}_{b,k} &= \max\{\underline{P}_b, \hat{g}_k(\underline{P}_{em,k}), \hat{g}_k(\hat{P}_{drv,k} - \overline{P}_{eng,k})\}\end{aligned}$$

and it is known that for $k \notin \mathcal{P}$

$$\hat{f}_k(\hat{P}_{eng,k}) = 0, \quad \overline{P}_{b,k} = \underline{P}_{b,k} = \hat{g}_k(\hat{P}_{drv}(k)).$$

In [15], it is shown that given the properties of $\hat{g}_k(\cdot)$ (strictly convex, twice differentiable, non-decreasing, one-to-one) and $\hat{f}_k$ (convex and non-decreasing), the function $\hat{f}_k(\hat{P}_{drv,k} - \hat{g}_k^{-1}(\hat{P}_{b,k}))$ is convex, non-increasing, and twice differentiable, so the MPC problem (13) becomes the convex, linearly constrained optimization problem

$$\begin{aligned}\min_{\hat{P}_b} \; & \sum_{k\in\mathcal{P}} \delta\hat{f}_k(\hat{P}_{drv,k} - \hat{g}_k^{-1}(\hat{P}_{b,k})) \\ \text{s.t. } & \hat{E}_0 = E(t) \\ & \left.\begin{aligned}\hat{E}_{k+1} &= \hat{E}_k - \delta\hat{P}_{b,k} \\ \underline{E} &\le \hat{E}_{k+1} \le \overline{E}\end{aligned}\right\} k = 0, \ldots, N-1 \\ & \underline{P}_{b,k} \le \hat{P}_{b,k} \le \overline{P}_{b,k} \qquad k \in \mathcal{P} \\ & \underline{P}_{b,k} = \hat{P}_{b,k} = \overline{P}_{b,k} \qquad k \notin \mathcal{P}.\end{aligned} \tag{14}$$

For the sake of clarity in the following sections, we now revert to the commonly used notation for MPC problems, where u is the control input (the predicted battery power, $\hat{P}_b$), and x is the state variable (the predicted state of charge, $\hat{E}$), i.e

$$\begin{aligned}u &:= \hat{P}_b \in \mathbb{R}^N, \quad & \overline{u} &:= \underline{P}_b \in \mathbb{R}^N, \quad & \underline{u} &:= \underline{P}_b \in \mathbb{R}^N \\ x &:= \hat{E} \in \mathbb{R}^N, \quad & \overline{x} &:= \overline{E} \in \mathbb{R}, \quad & \underline{x} &:= \underline{E} \in \mathbb{R}.\end{aligned}$$

Then, by defining the non-increasing, separable, strictly convex cost function

$$F(u) = \sum_{k\in\mathcal{P}} \delta\hat{f}_k(\hat{P}_{drv,k} - \hat{g}_k^{-1}(u_k)),$$

we can express (14) equivalently as

$$\begin{aligned}\min_u \; & F(u) \\ \text{s.t. } & x = \Phi x_0 - \Psi u \\ & \Phi\underline{x} \le x \le \Phi\overline{x} \\ & \underline{u} \le u \le \overline{u},\end{aligned} \tag{15}$$

where Φ is a vector of N ones, and Ψ is an $N \times N$ lower triangular matrix where every non-zero element is equal to δ.

If we describe $\mathcal{F}_k$ as the feasible set of state of charge values at timestep k, then problem (14) is feasible if and only if $\mathcal{F}_k \neq \emptyset$ for all $k = 0, \ldots, N$, where $\mathcal{F}_k$ is defined recursively by

$$\begin{aligned}\mathcal{F}_{k+1} &= \{x_k - \delta u_k \in \mathcal{X} : x_k \in \mathcal{F}_k,\ u_k \in \mathcal{U}_k\} \\ &= \{\mathcal{F}_k \oplus -\delta\mathcal{U}_k\} \cap \mathcal{X},\end{aligned}$$

and

$$\begin{aligned}\mathcal{U}_k &= \{u_k : \underline{u}_k \le u_k \le \overline{u}_k\} \\ \mathcal{X} &= \{x_k : \underline{x} \le x_k \le \overline{x}\}.\end{aligned}$$

As $\mathcal{F}_k$ is a one-dimensional convex set, it can be parameterized by $\mathcal{F}_k = [\min \mathcal{F}_k, \max \mathcal{F}_k]$, and the sequence $\mathcal{F}_1, \dots, \mathcal{F}_N$ can be obtained iteratively from

$$\begin{aligned} \max \mathcal{F}_{k+1} &= \min\{\overline{x}, \max \mathcal{F}_k - \delta \underline{u}_k\} \\ \min \mathcal{F}_{k+1} &= \max\{\underline{x}, \min \mathcal{F}_k - \delta \overline{u}_k\} \end{aligned} \tag{16}$$

with $\mathcal{F}_0 = \{x_0\}$. The problem is therefore feasible if and only if $\max \mathcal{F}_{k+1}$ and $\min \mathcal{F}_{k+1}$ exist (i.e. $\max \mathcal{F}_{k+1} \geq \min \mathcal{F}_{k+1}$ when calculated from (16), and $\overline{u}_k \geq \underline{u}_k$) for $k \in \{0, \dots, N-1\}$.

The cost function in (15) is known to be non-increasing in u, so by inspection, if

$$\Phi \underline{x} \leq \Phi x_0 - \Psi \overline{u} \leq \Phi \overline{x},$$

then $\overline{u}$ is the minimizing argument.

III. Projected Interior Point Method

Problem (15) is in the form of a convex nonlinear program with affine equality and inequality constraints. Here we present a projected primal-dual interior point algorithm, where the element-wise bounds on the control variable are applied as a projection. This section begins with the definition of the barrier approximation and optimality conditions, followed by a statement of the initialization algorithm and main projected interior point algorithm with accompanying pseudocode. The section is then is then concluded with an analysis of the computational complexity of each iteration, and a discussion of convergence to the minimizing argument of (15).

Firstly, we introduce a slack variable $s \in \mathbb{R}^{2N}$, and write problem (15) equivalently as

$$\begin{aligned} \min_{u} \ & F(u) \\ \text{s.t. } & Au - b - s = 0 \\ & s \geq 0 \\ & \underline{u} \leq u \leq \overline{u}, \end{aligned} \tag{17}$$

where

$$A = \begin{bmatrix} \Psi \\ -\Psi \end{bmatrix}, \text{ and } \ b = \begin{bmatrix} \Phi(x_0 - \overline{x}) \\ -\Phi(x_0 - \underline{x}) \end{bmatrix}.$$

This can then be approximated with

$$\begin{aligned} \min_{u,s} \ & F(u) + B(s, \mu) \\ \text{s.t. } & Au - b - s = 0 \\ & \underline{u} \leq u \leq \overline{u}, \end{aligned} \tag{18}$$

where $B(s, \mu)$ is a log barrier function defined by

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

and $\mu > 0$ can be interpreted as the degree to which the log barrier function approximates the inequality constraint, $s \geq 0$. The Lagrangian function associated with problem (18) is

$$\begin{aligned} \mathcal{L}(u, s, \theta_1, \theta_2, \theta_3, \mu) = F(u) + B(s, \mu) - \theta_1^\top (Au - b - s) \\ - \theta_2^\top (u - \underline{u}) - \theta_3^\top (\overline{u} - u) \end{aligned}$$

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

$$\begin{aligned} \mathcal{A}^\circ = \{k : u_k^\circ = \underline{u}_k, \nabla_k F(u^\circ) - A_k^\top \theta_1^\circ > 0, \\ \text{or } u_k^\circ = \overline{u}_k, \ \nabla_k F(u^\circ) - A_k^\top \theta_1^\circ < 0\}, \end{aligned}$$

the necessary and sufficient conditions for the optimal values u°, s°, and θ_1° that minimize (18) are

$$\nabla_k F(u^\circ) - A_k^\top \theta_1^\circ = 0, \quad k \notin \mathcal{A}^\circ, \tag{19a}$$

$$S^\circ \theta_1^\circ = \frac{1}{\mu} \mathbf{1}, \tag{19b}$$

$$Au^\circ - b - s^\circ = 0, \tag{19c}$$

$$u^\circ - \underline{u} \geq 0, \tag{19d}$$

$$\overline{u} - u^\circ \geq 0, \tag{19e}$$

$$s^\circ > 0, \tag{19f}$$

$$\theta_1^\circ \geq 0, \tag{19g}$$

where $\nabla_k F(u)$ is the kth row of the gradient of F at u, A_k is the kth column of A, and $S = \text{diag}(s)$.

It can be demonstrated that to obtain the optimality conditions of (17), two changes must be made to (19): firstly, (19f) must become a non-strict inequality condition, and secondly, a vector of zeros must replace $\frac{1}{\mu}\mathbf{1}$ in the R.H.S of (19b). Therefore, it can be concluded that u° converges asymptotically to $u^\star$ as $\mu \to \infty$, where $u^\star$ is the minimizing argument of (17). The principle of the algorithm presented in this section is that conditions (19d-19g) hold at all iterations, whilst a projected Newton method [28] is used to obtain an approximation of the solutions of (19a-19c) for a fixed value of μ. This is then repeated for progressively larger values of μ, with progressively higher accuracy, to obtain $u^\star$.

A. Initialization

The projected interior point algorithm is initialized using Algorithm 1 to obtain the tube defined by the sequence $\mathcal{G}_0, \dots, \mathcal{G}_N$ where

$$\begin{aligned} \mathcal{G}_k &= \{x_{k+1} + \delta u_k \in \mathcal{F}_k : x_{k+1} \in \mathcal{G}_{k+1}, u_k \in \mathcal{U}_k\} \\ &= \mathcal{F}_k \cap \{\mathcal{G}_{k+1} \oplus \delta \mathcal{U}_k\}. \end{aligned}$$

The centerline of this tube is then used to obtain values for $u^{(0)}$, $s^{(0)}$, and $\theta_1^{(0)}$ that satisfy (19b-19g).

Proposition III.1. *The values of $u^{(0)}$, $s^{(0)}$, and $\theta_1^{(0)}$ obtained with Algorithm 1 will satisfy conditions (19b-19g), iff*

$$\begin{aligned} \max \mathcal{F}_k > \min \mathcal{X} \quad \textit{and} \\ \min \mathcal{F}_k < \max \mathcal{X} \end{aligned} \tag{20}$$

for $k \in 1, \dots, N$.

Proof. By induction. We start by defining the operation

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

for any given set $\mathcal{S}$, and note that for $\max \mathcal{F}_k$ and $\min \mathcal{F}_k$ to exist in (20) it is implied that the problem is feasible, so $|\mathcal{F}_k| \geq 0$ for $k \in \{1, \dots, N\}$. Let $k^\dagger$ be the smallest value of k such that $\max \mathcal{F}_k = \max \mathcal{X}$ and/or $\min \mathcal{F}_k = \min \mathcal{X}$, and suppose that $k^\dagger \leq N$. In this case we can use (20) to show that $\max \mathcal{F}_k > \min \mathcal{F}_k$ (i.e. $|\mathcal{F}_k| > 0$). Furthermore, we know

Algorithm 1 Initialization Algorithm

Set $\max \mathcal{F}_0 = \min \mathcal{F}_0 = x_0$
for $k = 0, \dots, N-1$ **do**
  $\max \mathcal{F}_{k+1} = \min\{\overline{x}, \max \mathcal{F}_k - \delta \underline{u}_k\}$
  $\min \mathcal{F}_{k+1} = \max\{\underline{x}, \min \mathcal{F}_k - \delta \overline{u}_k\}$
end for
$\mathcal{G}_N = \mathcal{F}_N$
$x_N^{(0)} = \frac{1}{2}(\max \mathcal{G}_N + \min \mathcal{G}_N)$
for $k = N-1, \dots, 0$ **do**
  $\max \mathcal{G}_k = \min\{\max \mathcal{G}_{k+1} + \delta \overline{u}_k, \max \mathcal{F}_k\}$
  $\min \mathcal{G}_k = \max\{\min \mathcal{G}_{k+1} + \delta \underline{u}_k, \min \mathcal{F}_k\}$
  $x_k^{(0)} = \frac{1}{2}(\max \mathcal{G}_k + \min \mathcal{G}_k)$
  $u_k^{(0)} = \frac{1}{\delta}(x_k^{(0)} - x_{k+1}^{(0)})$.
end for
$s^{(0)} = Au^{(0)} - b$
$\theta_1^{(0)} = \frac{1}{\mu}(S^{(0)})^{-1}\mathbf{1}$

that $|\mathcal{F}_{k+1}| \geq |\mathcal{F}_k|$ for any k where $\max \mathcal{F}_{k+1} \neq \max \mathcal{X}$, $\min \mathcal{F}_{k+1} \neq \min \mathcal{X}$, and $|\mathcal{U}_k| \geq 0$. Therefore $|\mathcal{F}_k| > 0$ for all $k \geq k^\dagger$. If we assume that $|\mathcal{F}_k| > 0$, and $|\mathcal{G}_{k+1}| > 0$, we can then show that

$$\begin{aligned} |\mathcal{F}_{k+1}| > 0 \quad &\Leftrightarrow \quad |\{\mathcal{F}_k \oplus -\delta \mathcal{U}_k\} \cap \mathcal{X}| > 0 \\ &\Rightarrow \quad |\mathcal{F}_k \cap \{\mathcal{F}_{k+1} \oplus \delta \mathcal{U}_k\}| > 0 \\ &\Rightarrow \quad |\mathcal{F}_k \cap \{\mathcal{G}_{k+1} \oplus \delta \mathcal{U}_k\}| > 0 \\ &\Rightarrow \quad |\mathcal{G}_k| > 0. \end{aligned} \tag{21}$$

In the case where $k^\dagger$ exists we know that $|\mathcal{F}_N| > 0$, so $|\mathcal{G}_N| > 0$, and the argument in (21) can then be made recursively to show that $|\mathcal{G}_k| > 0$ for $k = N-1, \dots, k^\dagger$. As $\mathcal{G}_k \subseteq \mathcal{F}_k \subseteq \mathcal{X}$, we now know that that $x_k^{(0)} \in \text{int}(\mathcal{X})$ for $k \in \{k^\dagger, \dots, N\}$.

For $k \in \{1, \dots, k^\dagger - 1\}$ we know that $|\mathcal{F}_k| \geq 0$ from feasibility, and using a similar method to (21) we can then show that $|\mathcal{F}_{k+1}| \geq 0 \Rightarrow |\mathcal{G}_k| \geq 0$. We also know that $\max \mathcal{F}_k \neq \max \mathcal{X}$ and $\min \mathcal{F}_k \neq \min \mathcal{X}$ for $k \in \{1, \dots, k^\dagger - 1\}$ (from the definition of $k^\dagger$), so as $\mathcal{G}_k \subseteq \mathcal{F}_k \subset \mathcal{X}$, it follows that $x_k^{(0)} \in \text{int}(\mathcal{X})$ for $k \in \{1, \dots, k^\dagger - 1\}$. A similar result can be shown for $k \in \{1, \dots, N\}$ if $k^\dagger$ does not exist.

The condition $x_k^{(0)} \in \text{int}(\mathcal{X})$ for $k \in \{1, \dots, N\}$ ensures that $Au^{(0)} - b > 0$, so $s^{(0)} = Au^{(0)} - b$ ensures (19c) and (19f), and $\theta_1^{(0)} = \frac{1}{\mu}(S^{(0)})^{-1}\mathbf{1}$ ensures (19b) and (19g). Finally, it can also be shown that

$$\begin{aligned} &\max u_k^{(0)} \\ =&\max \frac{1}{2\delta}\{\max \mathcal{G}_k + \min \mathcal{G}_k - \max \mathcal{G}_{k+1} - \min \mathcal{G}_{k+1}\} \\ =&\overline{u}_k \end{aligned}$$

and we can similarly show that $\min u_k^{(0)} = \underline{u}_k$, which therefore demonstrates (19d-19e) if (20) is true for $k \in \{1, \dots, N\}$.

Conversely, a subset of the above results are shown to be not true if (20) is not true for some $k \in \{1, \dots, N\}$. □

An example of the $x^{(0)}$ and $u^{(0)}$ values obtained by the initialization algorithm for a nominal, randomly generated example is shown in Fig. 2, demonstrating that $\underline{u}_k \leq u_k^{(0)} \leq \overline{u}_k \ \forall k$, and that $\underline{x} < x_k^{(0)} < \overline{x} \ \forall k$.

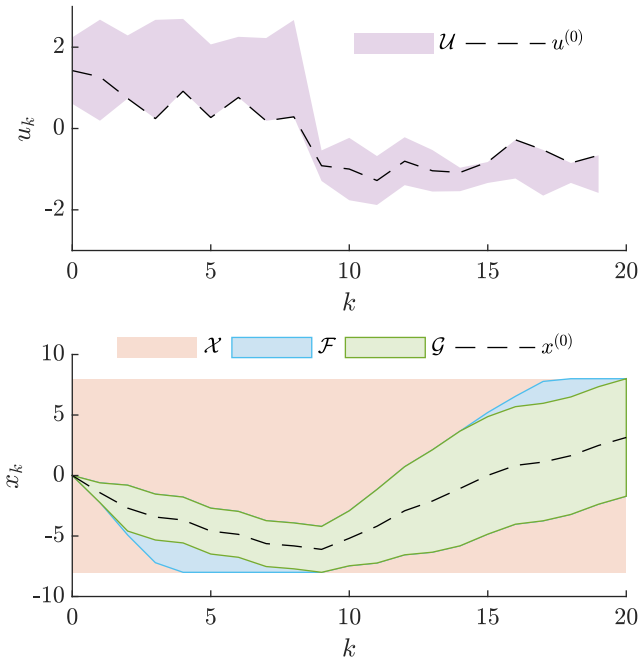

Fig. 2. An example of the tubes defined by $\mathcal{F}$ and $\mathcal{G}$, and the solutions obtained for $x^{(0)}$ and $u^{(0)}$ using Algorithm 1 for a nominal system (the values were chosen to illustrate the operation of the algorithm, and are not neccessarily representative of those observed in the energy management problem).

In addition to u, s, and θ_1, there are three further parameters that are initialized at the start of the algorithm: $\mu_0 > 0$, $\overline{\mu} \geq \mu_0$, $k_\mu > 1$, and $\tau \in (0, 1)$. These parameters can be assigned any value within the stated ranges, and their significance is discussed in the following subsection.

B. Algorithm

At each iteration, j, the elements k are partitioned into the sets

$$\begin{aligned} \mathcal{A}^{(j)} = \{&k : u_k^{(j)} = \underline{u}_k, \nabla_k F(u^{(j)}) - A_k^\top \theta_1^{(j)} > 0, \\ &\text{or } u_k^{(j)} = \overline{u}_k, \ \nabla_k F(u^{(j)}) - A_k^\top \theta_1^{(j)} < 0\} \\ \mathcal{D}^{(j)} = \{&k : k \notin \mathcal{A}^{(j)}\}, \end{aligned}$$

then an estimate is made of the solution to the equations (19a-19c) using a projected Newton method. Let

$$\begin{bmatrix} \nabla^2_{\mathcal{D}} F(u^{(j)}) & 0 & -w^\top \\ 0 & \Theta_1^{(j)} & S^{(j)} \\ w & -I & 0 \end{bmatrix} \begin{bmatrix} \Delta \tilde{u}^{(j)} \\ \Delta s^{(j)} \\ \Delta \theta_1^{(j)} \end{bmatrix} = \begin{bmatrix} -\nabla_{\mathcal{D}} F(u^{(j)}) + v^\top \theta_1^{(j)} \\ \frac{1}{\mu}\mathbf{1} - S^{(j)}\theta_1^{(j)} \\ -Au^{(j)} + b + s^{(j)} \end{bmatrix} \tag{22}$$

where $\nabla^2_{\mathcal{D}} F(u^{(j)})$ is the $k \in \mathcal{D}^{(j)}$ rows and columns of the Hessian of F evaluated at $u^{(j)}$, $\nabla_{\mathcal{D}} F(u^{(j)})$ is the $k \in \mathcal{D}^{(j)}$ elements of the gradient of F evaluated at $u^{(j)}$, and $\Theta = \text{diag}(\theta)$. We define $\hat{A}$ as the matrix A with the $k \in \{k : \underline{u}_k = u_k^{(j)} \text{ or } \overline{u}_k = u_k^{(j)}\}$ columns set to zero, so that v is the $k \in \mathcal{D}^{(j)}$ columns of A, and w is the $k \in \mathcal{D}^{(j)}$ columns of $\hat{A}$.

The search directions $\Delta\tilde{u}^{(j)}$, $\Delta\tilde{s}^{(j)}$, and $\Delta\theta_1^{(j)}$ are obtained from the reduced equations

$$\begin{aligned}\Delta\theta_1^{(j)} =& \left(w\left(\nabla^2_{\mathcal{D}}F(u^{(j)})\right)^{-1}w^\top + (\Theta_1^{(j)})^{-1}S^{(j)}\right)^{-1} \\ & \left(\frac{1}{\mu}(\Theta_1^{(j)})^{-1}\mathbf{1} - Au^{(j)} + b \right. \\ & \left. - w\left(\nabla^2_{\mathcal{D}}F(u^{(j)})\right)^{-1}(-\nabla F_{\mathcal{D}}(u^{(j)}) + v^\top\theta_1^{(j)})\right)\end{aligned} \tag{23a}$$

$$\begin{aligned}\Delta\tilde{u}^{(j)} =& -\left(\nabla^2 F_{\mathcal{D}}(u^{(j)})\right)^{-1} \\ & (\nabla_{\mathcal{D}}F(u^{(j)}) - v^\top\theta_1^{(j)} - w^\top\Delta\theta_1),\end{aligned} \tag{23b}$$

$$\Delta s^{(j)} = Au^{(j)} + w\Delta\tilde{u}^{(j)} - b - s^{(j)}, \tag{23c}$$

and search lengths α_s and α_θ are the determined from the 'fraction to the boundary' rule, [29, pp. 567] as

$$\alpha_s = \max\{\alpha \in (0,1] : s^{(j)} + \alpha\Delta s_1^{(j)} \geq (1-\tau)s^{(j)}\}, \tag{24a}$$

$$\alpha_\theta = \max\{\alpha \in (0,1] : \theta_1^{(j)} + \alpha\Delta\theta_1^{(j)} \geq (1-\tau)\theta_1^{(j)}\}, \tag{24b}$$

where $\tau \in (0,1)$ is fixed and arbitrary. The variables u, s, and θ_1 are then updated as

$$u_k^{(j+1)} = \begin{cases}\pi_k^u\left[u_k^{(j)} + \alpha_s\Delta\tilde{u}_{i(k)}^{(j)}\right] & k \in \mathcal{D}^{(j)} \\ 0 & k \in \mathcal{A}^{(j)}\end{cases}, \tag{25a}$$

$$s^{(j+1)} = s^{(j)} + \alpha_s\Delta\tilde{s}, \tag{25b}$$

$$\theta_1^{(j+1)} = \theta_1^{(j)} + \alpha_\theta\Delta\theta_1, \tag{25c}$$

where $\pi_k^u(u_k) = \min\{\overline{u}_k, \max\{\underline{u}_k, u\}\}$, and $i(k)$ is a function that returns the index of the element of $\mathcal{D}^{(j)}$ that is equal to k, assuming that the elements are ordered in a chronological, increasing sequence (e.g $i(7) = 2$ if $\mathcal{D}^{(j)} = \{1, 7, 12, \dots\}$). This iteration is performed repeatedly until the criterion

$$\begin{aligned}r_{\text{IP}}^{(j)} = \max\Big\{&\|\nabla_{\mathcal{D}}F(u^{(j)}) - v^\top\theta_1^{(j)}\|, \\ &\|\frac{1}{\mu}\mathbf{1} - S^{(j)}\theta_1^{(j)}\|, \|Au^{(j)} - b - s^{(j)}\|\Big\} < \frac{1}{\mu}\end{aligned} \tag{26}$$

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

$$\mu = \min\{\overline{\mu}, k_\mu\mu\}, \tag{27}$$

where $\overline{\mu} > 0$ is a pre-determined upper limit on the value of μ, and $k_\mu > 1$ is a pre-determined, arbitrary constant. The algorithm as a whole is then terminated when both conditions (26) and $\mu = \overline{\mu}$ are met. Algorithm 2 presents a pseudocode implementation of the above description.

C. Complexity

Lines 6-13 in Algorithm 2 constitute the recursive elements of the projected interior point algorithm, and Table I presents an analysis of the complexity of equations (23-26). The significance of enforcing the bounds on u as a projection is illustrated, as the complexity of the projected interior point operations is a function of the first dimension of A, which is $2N$ here. If the bounds on u were applied as log barrier functions (i.e $A = (\Psi, -\Psi, I, -I)$ and $b = (\Phi(x_0 - \overline{x}), -\Phi(x_0 - \underline{x}), \underline{u}, -\overline{u})$ in (17)) the relevant dimension would instead be $4N$, and the complexity of each update would become $\mathcal{O}(4^n N^n)$. It can be seen that (23a) is the most computationally demanding update due to the presence of both a dense matrix-matrix multiplication and a dense matrix inverse (note that *diagonal* matrix operations, e.g. $(\nabla^2_{\mathcal{D}}F(u^{(j)}))^{-1}$, are omitted from Table I). The computational complexity of each iteration of the projected interior point algorithm is therefore $\mathcal{O}(2^n N^n)$, where $n \leq 3$ is determined by the method used for matrix multiplication and inversion.

Algorithm 2 Projected Interior Point Method

1: Set parameters $\mu_0 > 0$, $\overline{\mu} \geq \mu_0$, $k_\mu > 1$, and $\tau \in (0,1)$
2: Initialize $u^{(0)}$, $s^{(0)}$, and $\theta_1^{(0)}$ using Algorithm 1
3: $\mu \leftarrow \mu_0$ and $j \leftarrow 0$
4: Determine $\mathcal{A}^{(0)}$ and $\mathcal{D}^{(0)}$
5: **repeat**
6: Calculate $\Delta\theta_1^{(j)}$, $\Delta\tilde{u}^{(j)}$, and Δs using (23a-23b)
7: Calculate α_s and α_θ from (24a-24b)
8: Update $u^{(j+1)}$, $s^{(j+1)}$, and $\theta_1^{(j+1)}$ with (25a-25c)
9: $j \leftarrow j + 1$
10: Determine $\mathcal{A}^{(j)}$ and $\mathcal{D}^{(j)}$
11: **if** $r_{\text{IP}}^{(j)} < \frac{1}{\mu}$ **then**
12: Update $\mu = \min\{\overline{\mu}, k_\mu\mu\}$
13: **end if**
14: **until** $\mu = \overline{\mu}$ and $r_{\text{IP}}^{(j)} < \frac{1}{\mu}$
15: $u^\star \leftarrow u^{(j)}$

TABLE I
SUMMARY OF THE MATRIX/VECTOR OPERATIONS PRESENT IN EACH EQUATION USED IN THE PROJECTED INTERIOR POINT ALGORITHM, WHERE V REFERS TO A VECTOR, AND M REFERS TO A NON-DIAGONAL MATRIX.

Equation(s)	M^{-1}	M·M	M·v	$\mathcal{O}(2^n N^n)$
(23a)	Yes	Yes	Yes	$n \leq 3$
(23b-23c)	No	No	Yes	$n \leq 2$
(24-25)	No	No	No	$n = 1$
(26)	No	No	Yes	$n \leq 2$

D. Convergence

The algorithm presented in Section III-B can be interpreted as a projected Newton method [28] used to obtain a stationary point, (u°, s°, θ_1°), of the function

$$\hat{\mathcal{L}}(u, s, \theta_1, \mu) = F(u) + B(s, \mu) - \theta_1^\top(Au - b - s) \tag{28}$$

for a given value of μ, subject to the constraint $\underline{u} \leq u \leq \overline{u}$. The strict inequality in the definition of $\mathcal{A}^\circ$ means that there is a region of (u, s, θ_1)-space close to $(u^\circ, s^\circ, \theta_1^\circ)$ where $\mathcal{A}^{(j)} = \mathcal{A}^\circ$, and a subset of this region will meet the conditions for local quadratic convergence of Newton's method for nonlinear equations [29, pp. 276]. This means that the R.H.S of (22) converges to 0 as $j \to \infty$, and the termination criterion (26) 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 [29, pp. 30] of an appropriate merit function, although the merit function from [28] cannot be used as the stationary point (u°, s°, θ_1°) is not a minimum of the function (28) in general. Despite this limitation, convergence

was demonstrated for all problem classes in the simulations that follow.

We have previously demonstrated that $u^\circ \to u^\star$ as μ is increased towards ∞, and the value of $u^{(j)}$ when the criterion (26) is met converges to u° as μ is increased towards ∞. Therefore, the value of $u^{(j)}$ when Algorithm 2 terminates can be made arbitrarily close to the minimizing argument of (17) by setting $\overline{\mu}$ arbitrarily high. The algorithm could be further optimised 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°, s°, and θ_1°, although in the numerical experiments that follow we demonstrate superlinear convergence for a broad class of problems using the simple update (27).

IV. Alternating Direction Method of Multipliers

We compare the performance of the projected interior point method in simulation with the ADMM algorithm proposed in [15], which is restated here with a new complexity analysis. We introduce a dummy variable, ζ, and rewrite (15) as

$$\begin{aligned} \min_{u} \; & F(u) + \Lambda(u, x) \\ \text{s.t. } & \zeta = -u \\ & x = \Phi x_0 + \Psi \zeta \end{aligned} \tag{29}$$

where the indicator function, Λ, is defined by

$$\Lambda(u, x) = \sum_{k=0}^{N-1} \Lambda_k^u(u_k) + \sum_{k=1}^{N} \Lambda_k^x(x_k)$$
$$\Lambda_k^z(z) = \begin{cases} 0 & \underline{z}_k \leq z_k \leq \overline{z}_k \\ \infty & \text{otherwise} \end{cases},$$

and the augmented Lagrangian associated with (29) is

$$\begin{aligned} L(u, \zeta, x, \lambda_1, \lambda_2) =& F(u) + \Lambda(u, x) + \frac{\rho_1}{2}\|u + \zeta + \lambda_1\|^2 \\ & + \frac{\rho_2}{2}\|\Phi x_0 + \Psi\zeta - x + \lambda_2\|^2 \end{aligned}$$

The ADMM algorithm is initialized with the values

$$\begin{aligned} & u^{(0)} = \overline{u},\ \zeta^{(0)} = -u^{(0)},\ x^{(0)} = \Pi^x\left(\Phi x_0 + \Psi\zeta^{(0)}\right) \\ & \lambda_1^{(0)} = 0,\ \lambda_2^{(0)} = \Phi x_0 + \Psi\zeta^{(0)} - x^{(0)} \end{aligned} \tag{30}$$

and by defining projection functions

$$\pi_k^z(z) = \min\{\overline{z}_k, \max\{\underline{z}_k, z\}\},$$
$$\Pi^z(z) = [\pi_1^z(z_1), \ldots, \pi_N^z(z_N)],$$

the iteration is given by

$$\begin{aligned} u_k^{(j+1)} =& \pi_k^u\Big[\arg\min_{u_k} \hat{f}_k(\hat{P}_{drv,k} - \hat{g}_k^{-1}(u_k)) \\ & + \frac{\rho_1}{2}(u_k + \zeta_k^{(j)} + \lambda_{1,k}^{(j)})^2\Big] && k \in \mathcal{P} \\ u_k^{(j+1)} =& \overline{u}_k && k \notin \mathcal{P} \\ x^{(j+1)} =& \Pi^x\big[\Phi x_0 + \Psi\zeta^{(j)} + \lambda_2^{(j)}\big] \\ \zeta^{(j+1)} =& (\rho_1 I + \rho_2\Psi^\top\Psi)^{-1}\big[-\rho_1(u^{(j+1)} + \lambda_1^{(j)}) \\ & - \rho_2\Psi^\top(\Phi x_0 - x^{(j+1)} + \lambda_2^{(j)})\big] \\ \lambda_1^{(j+1)} =& \lambda_1^{(j)} + u^{(j+1)} + \zeta^{(j+1)} \\ \lambda_2^{(j+1)} =& \lambda_2^{(j)} + \Phi x_0 + \Psi\zeta^{(j+1)} - x^{(j+1)} \end{aligned} \tag{31}$$

Problem (29) can be shown to be equivalent to the canonical ADMM form [30, equation (3.1)], for which the iteration is equivalent to (31). As $F(u) + \Lambda(u, x)$ is convex, it can therefore be concluded that iteration (31) will converge to the solution of (29) since the residuals defined by

$$r_{\text{P}}^{(j+1)} = \begin{bmatrix} I & 0 \\ 0 & -I \end{bmatrix}\begin{bmatrix} u^{(j+1)} \\ x^{(j+1)} \end{bmatrix} + \begin{bmatrix} I \\ \Psi \end{bmatrix}\zeta^{(j+1)} + \begin{bmatrix} 0 \\ \Phi x_0 \end{bmatrix}$$
$$r_{\text{D}}^{(j+1)} = \begin{bmatrix} \rho_1 I \\ -\rho_2\Psi \end{bmatrix}\left[\zeta^{(j)} - \zeta^{(j+1)}\right]$$

necessarily converge to zero. The algorithm is terminated when the conditions $\|r_{\text{P}}^{(j+1)}\| \leq \epsilon$ and $\|r_{\text{D}}^{(j+1)}\| \leq \epsilon$ are met, where ϵ is a pre-determined threshold.

Algorithm 3 Alternating Direction Method of Multipliers

1: Initialize $u^{(0)}$, $x^{(0)}$, $\zeta^{(0)}$, $\lambda_1^{(0)}$, $\lambda_2^{(0)}$ with (30)
2: $j \leftarrow 0$
3: **while** $\|r_{\text{P}}^{j+1}\| > \epsilon$ and $\|r_{\text{D}}^{j+1}\| > \epsilon$ **do**
4: Calculate $u^{(j+1)}$, $x^{(j+1)}$, $\zeta^{(j+1)}$, $\lambda_1^{(j+1)}$, $\lambda_2^{(j+1)}$ from (31)
5: $j \leftarrow j + 1$
6: **end while**
7: $u^\star \leftarrow u^{(j)}$

A. Complexity

Algorithm 3 shows a pseudocode implementation of the ADMM algorithm, and the computational complexity of each recursive variable update is presented in Table IV-A. Each u_k update is an unconstrained convex optimization problem that we solve here using a Newton method with a backtracking line search, so the u update therefore scales linearly with N if these updates are performed sequentially, or is constant if each k update can be performed in parallel. The matrix inversion in the ζ update can be computed offline as it involves no decision variables, so only a dense matrix multiplication is required. We note that multiplication by Ψ is not considered a matrix multiplication in the analysis presented in Table II, as it is the equivalent of a linear filtering operation and therefore scales linearly with N. This implies that the residual updates also scale linearly with N if they are analytically block multiplied. The complexity of the ADMM iteration is therefore $\mathcal{O}(N^n)$

TABLE II
SUMMARY OF THE MATRIX/VECOTR PRESENT IN THE ADMM VARIABLE UPDATES, WHERE 'V' REFERS TO A VECTOR, AND 'M' REFERS TO A NON-DIAGONAL MATRIX OTHER THAN Ψ.

Update	M^{-1}	M·M	M·v	$\mathcal{O}(N^n)$
u	No	No	No	$n=1$
x	No	No	No	$n=1$
ζ	No	No	Yes	$n\leq 2$
λ_1, λ_2	No	No	No	$n=1$
r_P, r_D	No	No	No	$n=1$

where $n \leq 2$ is determined by the method used for matrix multiplication.

V. NUMERICAL EXPERIMENTS

To compare the performance of the algorithms without reference to a particular PHEV powertrain, single-shot instances of problem (14) were created with randomly generated parameters. For each instance of the energy manangement problem, a nominal sampling frequency of 1Hz (i.e $\delta = 1\,\text{s}$) was assumed, and it was also assumed that the engine is always on and the clutch is engaged i.e $\hat{\sigma}(k) = 1$ (for the purposes of these experiments the switching heuristic is arbitrary). Using observations from previous experiments [11], predictions were made of driver power demand as $\hat{P}_{drv,k} \in [-2.5\times10^3, 10\times10^3]\,W$ and hardware parameters were generated from the distributions $\alpha_{2,k}, \beta_{2,k} \in [0.5\times10^{-5}, 1.5\times10^{-5}]\,W^{-1}$, $\alpha_{1,k}, \beta_{1,k} \in [0.5, 1.5]$, and $\alpha_{0,k}, \beta_{0,k} = 0\,W$, with $V_{oc} = 300\,V$ and $R = 0.1\,\Omega$. The limits on state and input were set at $\overline{x} = 10^5\,J$, $\underline{x} = 0\,J$, $\overline{u} = 15\times10^3\,W$ and $\underline{u} = -15\times10^3\,W$, and an initial state of charge of $x_0 = 0.9\overline{x}$ was assumed. These limits have little physical significance within the context of this experiment, and were chosen to ensure that the problems are feasible and that both the state and input constraints were active. Finally, the effect of varying τ was not investigated, and was set at $\tau = 0.995$ for all projected interior point solutions. The simulations were implemented in Matlab on a 2.6GHz Intel Core i7-6700HQ CPU.

A. Optimum

It is demonstrated in Section III that the output of Algorithm 2 can be made arbitrarily close to the solution of (17) 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 $u^\star$. 10 problems for each horizon length $N = 100, 200, 300, 400$ were generated for a total of 40 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 20 logarithmically spaced points (k_μ is not required as the algorithm will terminate when condition (26) is first met). For each $\bar{\mu}_i$, $i = 1, \ldots, 20$, the control input vector, u_i^*, at termination was recorded, forming the sequence $u_1^*, \ldots, u_{20}^*$ for each problem. Fig. 3 shows that as $\overline{\mu}$ was increased, the norm of the difference between the values of $u^{(j)}$ when the algorithm terminated with successive values of $\bar{\mu}$, $\|u_i^* - u_{i-1}^*\|$, decreased within an inverse band for all cases, and that at

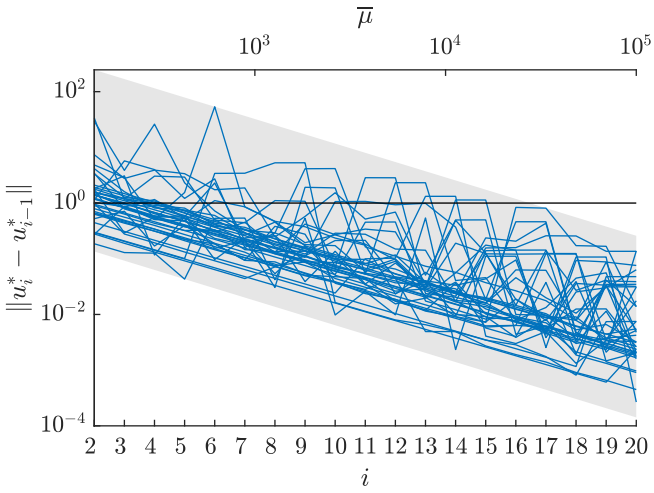

Fig. 3. Data showing the decrease in change of control vector obtained by the projected interior point method as $\overline{\mu}$ is increased from 10^0 to 10^5. The grey shaded area shows the minimum width linear band that contains all of the data points, and the dashed lines are simply used to highlight values on the vertical and horizontal axes.

$\overline{\mu} = 10^5$ this metric has reduced to less than 1 for all 40 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 therefore sufficiently large to provide a highly accurate solution to problem (17), and all future references to $u^\star$ refers to control inputs found using Algorithm 2 with $\mu_0 = \overline{\mu} = 10^5$.

B. 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 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, 20 new problems were generated for each horizon length of of $N = 100, 200, 300, 400$. These were solved using the projected interior point algorithm with $\overline{\mu} = 10^5$, $10^{-5} \leq \mu_0 \leq 10^1$, and $1 < k_\mu \leq 10^4/\mu_0$ (any value of k_μ greater than this would ensure that μ is projected onto $\overline{\mu}$ during the first update step (27)). For each problem instance, the number of iterations required for the algorithm to terminate were recorded, and the average for each combination of parameters is shown in Fig. 4. 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

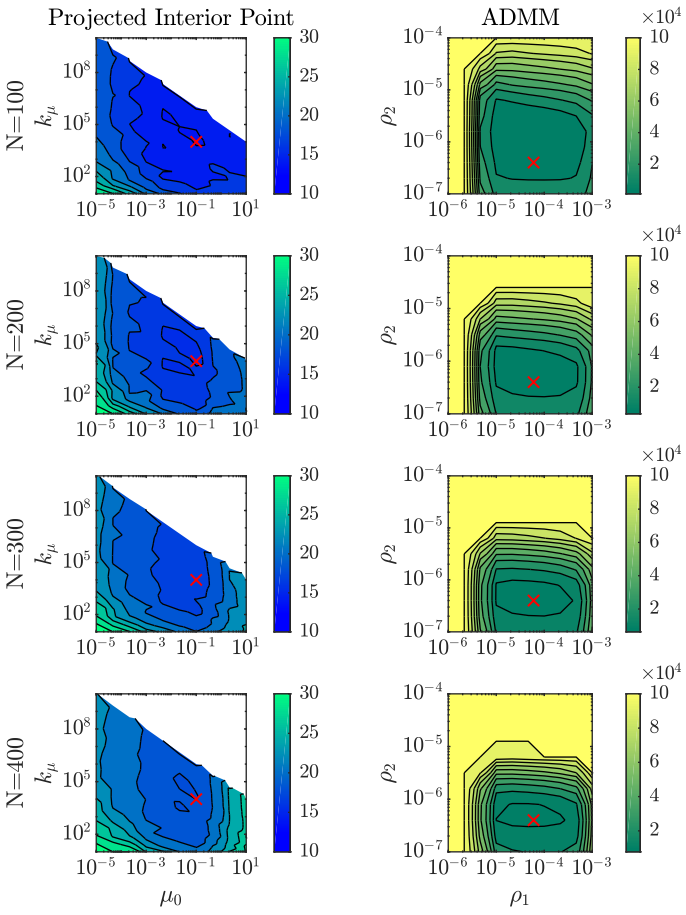

Fig. 4. Results of parameter tuning for both the projected interior point and ADMM algorithm. The projected interior point figures show the average number of iterations required to meet the termination criteria $\overline{\mu} = 10^5$ and are saturated at 30 iterations, whereas the ADMM figures show the average error measured by $\|u^{(100)} - u^\star\|$ and are saturated at 10×10^4. The red crosses show the chosen values for simulations described in Section V-C.

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 number of iterations required to achieve the same level of accuracy as the projected interior point algorithm with $\overline{\mu} = 10^5$ made a similar parameter search intractable. Instead, a total of 100 ADMM iterations were completed for each problem (ignoring the stated termination criteria) with $10^{-6} \leq \rho_1 \leq 10^{-2}$ and $10^{-8} \leq \rho_2 \leq 10^{-4}$. 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 Fig. 4, 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.

C. 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 criteria for a 'sufficiently' accurate solution was determined, then the variation in computational time with horizon length was investigated.

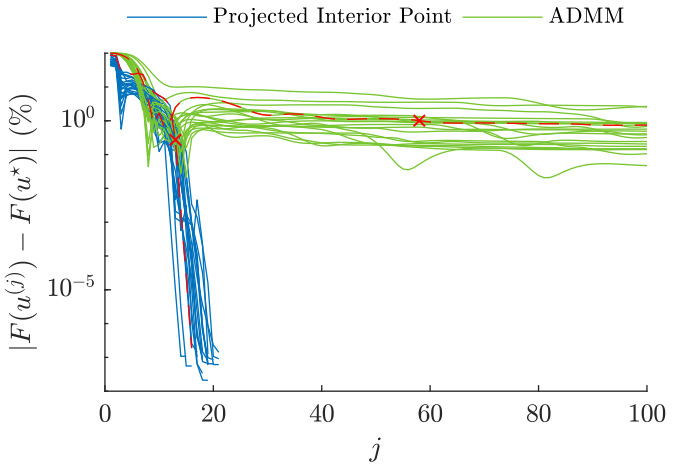

Fig. 5. Curves showing the normalised error between the cost evaluated at iteration j and the optimum, as a percentage, for 20 systems using both the projected interior point method and ADMM. The curves highlighted in red correspond to the system illustrated in Fig. 6, and the red cross shows the iteration from which those curves were taken.

A further 20 test cases were generated consisting of 5 cases for each of $N = 100, 200, 300, 400$, 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 100 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 Fig. 5. 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.

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 Fig. 6 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 Fig. 6: the state constraints are only guaranteed for both algorithms when the residuals, $r_{\text{IP}}, r_{\text{P}}$, and r_{D}, are precisely zero (this is also why we use the absolute error in Fig. 5, as the cost evaluated for each iteration of can be *lower* than $F(u^\star)$). Therefore, the termination criteria do not provide a guarantee of enforcing the state constraints, and we can see that for the final three

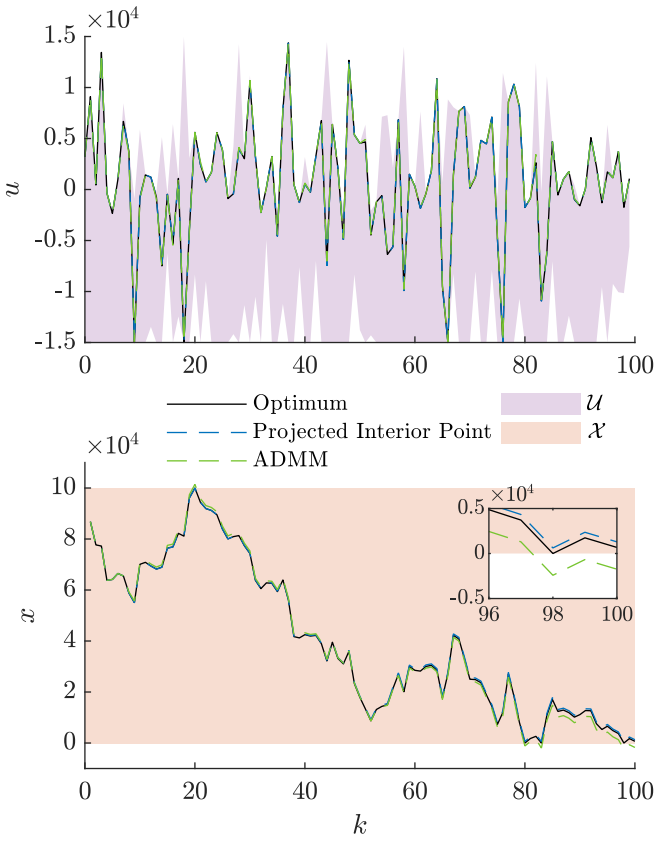

Fig. 6. The control and state vectors for the systems highlighted with red crosses in Fig. 5, plotted against the optimum $u^\star$ and $x^\star$. The feasible tubes $\mathcal{U}$ and $\mathcal{X}$ are also included, and note that the additional constraints enforced during the convex formulation specified in section II-B have significantly restricted the upper and lower bounds on $\mathcal{U}$ from the original $\pm 1.5 \times 10^4$.

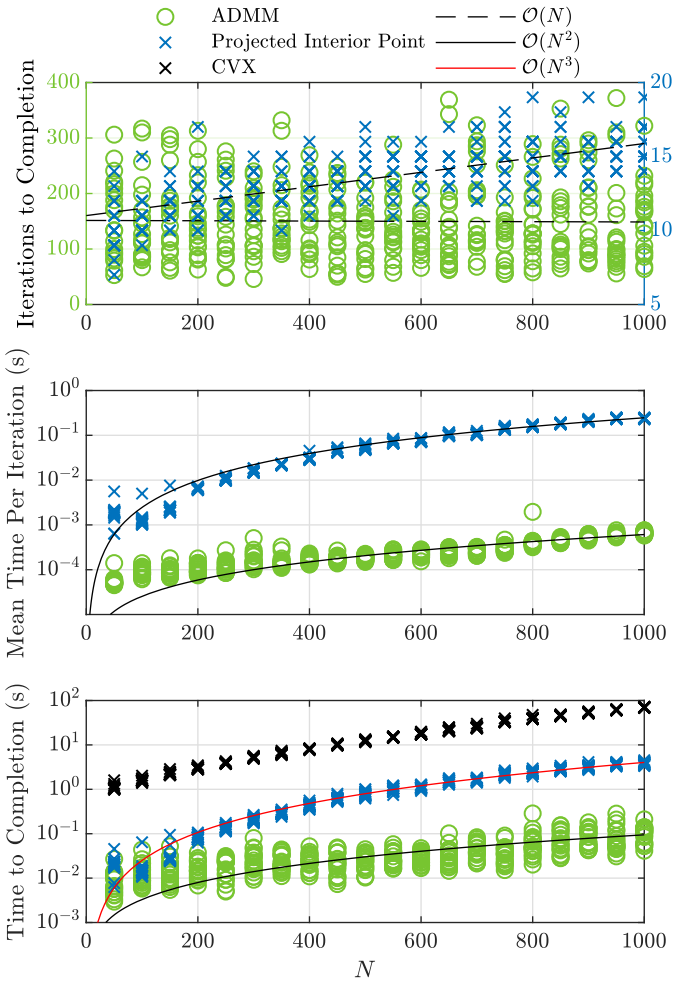

Fig. 7. Results showing the number of iterations required, mean time per iteration, and time to completion for 20 systems with $50 \leq N \leq 1000$, with linear, quadratic, and cubic trendlines.

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 Fig. 6, it was assumed that $\epsilon = 4 \times 10^3$ and $\overline{\mu} = 1$ enforce a 'sufficient' level of convergence. A further 20 problems were generated for horizons $50 \leq N \leq 1000$, and the iterations to completion, mean time taken per iteration, and time to completion 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 [31] 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 Fig. 7, 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 shown in sections III-C and IV-A that, as the horizon length is increased, the computational burden of the projected interior point algorithm is dominated by the $\Delta\theta_1$ update and the ADMM algorithm is dominated by the ζ update, so the methods used to perform these calculations 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 for both in Fig. 7, although as N was increased towards 1000, the projected interior point iterations took over two orders of magnitude longer than the ADMM iterations. It was therefore expected that the total time taken for the ADMM algorithm would scale quadratically 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 bottom plot in Fig. 7. It is also shown that (assuming an interval of 1 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 only required $\sim 0.1s$. From the previous scaling properties it can be assumed that ADMM is real time implementable for horizons significantly in excess of 1000 (the performance of the ADMM algorithm presented here exceeds that presented in [15] due to a fully vectorized software implementation). 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. If the accuracy requirement were tightened, however, it is likely that this performance relationship would change significantly.

In comparison, 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.

To conclude the numerical experiments, we note that the Newton method for the projected interior point requires the solution to non-diagonal linear systems of equations, which in turn is typically solved using BLAS [32]. This is not an issue when solving the problems on desktop hardware as demonstrated here, but may not be an option for the embedded hardware used for an online solution in a vehicle. In this case only the ADMM algorithm is suitable, as although it also requires a Newton method for the individual control variable updates, this can be performed element-wise and therefore does not require a matrix inversion step.

VI. Conclusion

This paper proposes a projected interior point method for the solution of a convex formulation of the optimization problem associated with nonlinear MPC for energy management in hybrid electric vehicles. The performance w.r.t the tailored ADMM algorithm of [15] is demonstrated through numerical experiments, and the projected interior point algorithm is shown to have faster convergence (superlinear) for the class of problems investigated, although the ADMM algorithm is shown to have superior numerical performance and scaling properties when a modest level of accuracy is required. Both algorithms are also shown to have superior computational performance to general purpose convex optimization software.

References

[1] M. Ehsani, Y. Gao, and A. Emadi, *Modern Electric, Hybrid Electric, and Fuel Cell Vehicles*, 2nd ed. CRC Press, 2009.

[2] 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.

[3] A. Sciarretta and L. Guzzella, "Control of hybrid electric vehicles," *IEEE Control Systems*, vol. 27, no. 2, pp. 60–70, 2007.

[4] C. Marina 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, 2017.

[5] O. Sundstrom, P. Soltic, and L. Guzzella, "A transmission-actuated energy-management strategy," *IEEE Transactions on Vehicular Technology*, vol. 59, no. 1, pp. 84–92, 2010.

[6] V. Larsson, L. Johannesson, and B. Egardt, "Analytic Solutions to the Dynamic Programming sub-problem in Hybrid Vehicle Energy Management," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 4, pp. 1458–1467, 2015.

[7] L. Serrao, A. Sciarretta, O. Grondin, A. Chasse, Y. Creff, D. Di Domenico, P. Pognant-Gros, C. Querel, and L. Thibault, "Open Issues in Supervisory Control of Hybrid Electric Vehicles: A Unified Approach Using Optimal Control Methods," *Oil & Gas Science and Technology Revue d'IFP Energies nouvelles*, vol. 68, no. 1, pp. 23–33, 2013.

[8] 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.

[9] R. Schmid, J. Buerger, and N. Bajcinca, "Efficient Optimal Control of Plug-in-Hybrid Electric Vehicles including explicit Engine on / off Decisions," in *2018 European Control Conference (ECC)*, Limassol, Cyprus, 2018, pp. 596–601.

[10] Y. Huang, H. Wang, A. Khajepour, H. He, and J. Ji, "Model predictive control power management strategies for HEVs: A review," *Journal of Power Sources*, vol. 341, pp. 91–106, 2017.

[11] 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 System Technology*, vol. 27, no. 3, pp. 1060–1070, 2019.

[12] P. Elbert, T. Nuesch, 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.

[13] 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.

[14] 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, pp. 834–856, 2014.

[15] 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)*, Miami Beach, FL, 2018, pp. 2641–2646.

[16] J. Gondzio, "Interior point methods 25 years later," *European Journal of Operational Research*, vol. 218, no. 3, pp. 587–601, 2012.

[17] N. Karmarkar, "A new polynomial-time algorithm for linear programming," *Combinatorica*, vol. 4, no. 4, pp. 373–395, 1984.

[18] F. A. Potra and S. J. Wright, "Interior-point methods," *Journal of Computational and Applied Mathematics*, vol. 124, no. 1-2, pp. 281–302, 2000.

[19] 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.

[20] X. Zhang, L. Ferranti, and T. Keviczky, "An improved primal-dual interior point solver for model predictive control," in *2017 IEEE Conference on Decision and Control (CDC)*, Melbourne, Australia, 2018, pp. 1126–1131.

[21] E. Klintberg and S. Gros, "A Parallelizable Interior Point Method for Two-Stage Robust MPC," *IEEE Transactions on Control Systems Technology*, vol. 25, no. 6, pp. 2087–2097, 2017.

[22] 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.

[23] 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.

[24] M. Grant and S. Boyd, "CVX: Matlab software for disciplined convex programming, version 2.0 beta," 2013. [Online]. Available: http://cvxr.com/cvx

[25] 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, no. October 2018, pp. 480–495, 2019.
[26] 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.
[27] C. Xiang, F. Ding, W. Wang, and W. He, "Energy management of a dual-mode power-split hybrid electric vehicle based on velocity prediction and nonlinear model predictive control," *Applied Energy*, vol. 189, pp. 640–653, 2017.
[28] D. Bertsekas, "Projected Newton Methods for Optimization Problems with Simple Constraints," *SIAM Journal on Control and Optimization*, vol. 20, no. 2, pp. 221–246, 1982.
[29] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed. Springer, 2006.
[30] S. Boyd, N. Parikh, E. Chu, J. Peleato, and J. Eckstein, "Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers," *Foundations and Trends in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.
[31] K. Toh, M. Todd, and R. Tutuncu, "SDPT3 — a Matlab software package for semidefinite programming,," *Optimization Methods and Software*, vol. 11, no. 1-4, pp. 545–581, 1999.
[32] "BLAS (Basic Linear Algebra Subprograms)." [Online]. Available: http://www.netlib.org/blas/

Sebastian East received the M.Eng. degree in mechanical engineering from the University of Bath, Bath, U.K., in 2015. He is currently a Doctoral Student with the University of Oxford, Oxford, U.K. His doctoral research is concerned with model predictive control and optimization, with applications to energy management problems in hybrid electric vehicles.

Mark Cannon Mark Cannon has an MEng (1993) in engineering science and a DPhil (1998) in control engineering, both from the University of Oxford, UK, and an MS (1995) in mechanical engineering from Massachusetts Institute of Technology, Cambridge, MA, USA. He is currently an Associate Professor of engineering science with the University of Oxford and a fellow of St Johns College, Oxford. His research interests are in robust and optimal control for constrained and uncertain systems, optimization for receding horizon control with robust constraints and stochastic uncertainty, and stochastic model predictive control.