Stochastic Dynamic Programming in the Real-World Control of Hybrid Electric Vehicles

Christopher Vagg, Sam Akehurst, Chris J. Brace, and Lloyd Ash

Abstract—Stochastic dynamic programming (SDP) is applied to the optimal control of a hybrid electric vehicle in a concerted attempt to deploy and evaluate such a controller in the real world. Practical considerations for robust implementation of the SDP algorithm are addressed, such as the choice of discount factor used and how charge sustaining characteristics of the SDP controller can be examined and adjusted. A novel cost function is used incorporating the square of battery charge (C-rate) as an indicator of electrical powertrain stress, with the aim of lessening the affliction of real-world concerns such as battery health and motor temperature, while allowing short spells of operation toward the system peak power limits where advantageous. This paper presents the simulation and chassis dynamometer results over the LA92 drive cycle, as well as the results of testing on open roads. The hybrid system is operated at several levels of aggressivity, allowing the tradeoff between fuel savings and electrical powertrain stress to be evaluated. In dynamometer testing, this approach yielded a 13% reduction in electrical powertrain stress without sacrificing any fuel savings, compared with a controller that does not consider aggressivity in its optimization.

Index Terms—Battery stress, energy management, equivalent consumption minimization strategy (ECMS), hybrid vehicles, optimal control, predictive control, stochastic dynamic programming (SDP), vehicle driving.

I. INTRODUCTION

The optimal control of hybrid electric vehicle (HEV) powertrains in real-world conditions is nontrivial because the solution depends on the future use case: this determines whether there are likely to be effective opportunities for hybrid operation as well as the availability of recoverable energy.

The present literature on the optimal control of hybrid vehicles advocates stochastic dynamic programming (SDP) [1]–[8], which aims to use Bellman’s principle of optimality applied to a statistical model of what the future is likely to entail, rather than deterministic dynamic programming (DDP), which may only be implemented over a drive cycle that is known precisely in advance. While DDP and SDP are both global optimization techniques, DDP produces a control policy that is time variant, whereas SDP enables a stationary (time-invariant) optimal control policy to be found as a function of vehicle state; the result is globally optimal if run for an infinite period of time over a drive cycle with the same statistical distributions as the data on which it was based. The SDP controller would likely be out performed by a controller with perfect knowledge of the future and freedom to be time varying; however, it is the stationary policy that will yield the best possible result when operated indefinitely, over a drive cycle matching its probability model. In the absence of perfect knowledge of the future, this SDP controller is therefore extremely attractive. However, it does require a collection of representative driving data on which to be based, which may not be easily available. Furthermore, its implementation is not straightforward and so requires time investment during design, the control map is difficult to examine because it is multidimensional, and the embedded real-time operation may be memory intensive (though not computationally intensive).

A. Implementation of SDP

The existing literature highlights the potential of the SDP algorithm, though there is a lack of best practice guidelines for its practical implementation. In particular, the algorithm depends heavily on the use of a discount factor that assures convergence of an infinite sum for cost and effectively determines how far into the future the controller considers by defining how quickly future costs are discounted. However, the effect of changing the value of the discount factor, 0 < λ < 1, and how a final decision on its value should be reached are not adequately explored. In two worked examples, Kolmanovsky et al. [1] state that they used discount factors of 0.95 and 0.98, though without further explanation. Johannesson et al. [3] used 0.995, while Lin et al. [2] used 0.95, whose example was followed by Tate Jr. et al. [4] and Moura et al. [7], with Moura noting that the choice of discount factor is a matter open for further research. Shortest path SDP has been used as an alternative [4], [9], [10] as it does not use a discount factor, but instead considers that each trip has a finite probability of ending. Since the vehicle studied here may not be charged from the grid, we consider
that subsequent trips are tantamount to continuations of the present, and so remain with the classic SDP formulation.

To achieve charge sustenance, a quadratic cost for deviation from a nominal state of charge (SOC) is often used. In this work, no such cost is used, giving the controller greater freedom to deviate in SOC. The charge sustenance properties of this approach are explored.

Most SDP controllers reported in the literature are tested in simulation only, with the notable exceptions of Opila et al. [10] who report dynamometer testing and Leroy et al. [11] who report road testing. This paper expands the body of data by providing extensive hardware test results over a drive cycle known to be representative of the naturalistic driving data on which the SDP controller is based.

### B. Real-World Relevance

Optimal control design techniques such as SDP rely heavily on model-based design to assess and optimize the performance of a control strategy. This approach is enormously powerful; however, computer models always require simplifications and assumptions to be made. One such noteworthy simplification made frequently is that the electric powertrain has a fixed rated power limit at which it may be operated as much or as little as desired. In fact, this is far from the truth, since electric motors can often deliver considerably more than their continuous rated power in short bursts, usually limited by heating. Similarly, batteries may be operated at high charge/discharge rates, but very often the battery management system will struggle to maintain even temperature and SOC distributions between cells. Maximum charge and discharge rates of a battery pack are continuously updated and are always defined by the extrema values of cell voltage and temperature within it. The development of imbalances between cells therefore limits the operating window of the pack [12]. Uneven heat generation in battery packs is also problematic as hot spots introduce the risk of thermal runaway, and so may require the battery power to be temporarily limited; for this reason, battery pack cooling is important [13]. If not properly managed and protected, cells’ usable life span may also suffer as a result.

Since the great majority of the work in this area is based solely in simulation, the significance of these simplifications has been understated, though these concerns are familiar in industry and to those working with hardware [14], [15].

Considering battery health, Moura [16] and Moura et al. [17] proposed two cost functions aimed at improving long-term health: 1) one based on an electrochemical model of aging and 2) another on battery energy throughput. The relative mathematical simplicity of throughput models makes them better candidates for control optimization purposes. Ebbesen et al. [18] and Serrao et al. [19] have demonstrated controllers incorporating throughput aging models. In both examples, the rate of aging is acknowledged to increase at high battery currents, and so the cost function amounts to a nonlinear function of instantaneous current, while [19] also accounts for temperature and SOC. Load-leveling control strategies have been proposed to reduce battery stress [20], [21] as well as hybrid energy storage systems using supercapacitors [22].
modify any standard components. For this reason, the design incorporates an electric traction motor, connected by a belt to a pulley sandwiched between the prop shaft and the final drive. The system controller obtains vehicle data nonintrusively via the vehicle on-board diagnostic (OBD) port. In addition to its easy installation, this design has the advantage of being entirely failsafe; even in the worst failure modes, the belt can simply break, mechanically disconnecting the retrofit system from the standard vehicle, leaving it in a usable condition.

A custom computational model of the vehicle powertrain was built in SIMULINK for the purposes of design evaluation and control strategy design. The model is structured as a backward model such that time traces of vehicle speed and gear selection (the drive cycle) are fed in and assumed to be followed perfectly. The tractive force required to achieve the speed trace is calculated, and the control strategy determines the torque split between the engine and the electric motor. Engine and electric motor speeds are both constrained by the drive cycle, and the vehicle uses a manual transmission with gear selected by the driver. Since the model is bespoke, each component of the powertrain was modeled to a high degree of fidelity based on a combination of experimental work and theory and the complete model validated against chassis dynamometer data. The modeling of each component is briefly described in the following sections, and the key vehicle characteristics are presented in Table I.

### A. Vehicle Dynamics

Forces acting on the vehicle are assumed to include only aerodynamic drag, rolling resistance, and inertia. For the purposes of simulation and dynamometer testing, forces arising due to road gradient are not considered, though the controller was adapted for the road test phase of the study. Steady-state road forces were modeled as a second-order polynomial of vehicle speed, as is common in dynamometers. Inertial forces are proportional to the vehicle acceleration, with the greatest by far being due to the vehicle mass; however, the simulation also included the minor contributions of the rotational inertia of the wheels and driveline components using approximate equivalent masses suggested by Miller [25]. Combining all these forces

$$F_{TR} = F_2 \cdot v^2 + F_1 + \frac{dv}{dt}(m_v + m_w + m_d).$$  (1)

In this approach, $F_{TR}$ is the total tractive force, $v$ is the vehicle speed, $F_2$ is a constant resulting from the vehicle aerodynamic properties, $F_1$ is a constant rolling resistance force, $m_v$ is the vehicle mass, and $m_w$ and $m_d$ are the equivalent inertial masses of the wheels and driveline components, respectively.

### B. Combustion Engine

The fuel consumption of the combustion engine was mapped in each gear using a chassis dynamometer in speed control mode. For each gear, the road speed was adjusted to achieve a set of engine speeds that was maintained regardless of the engine load. A mechanism was used to actuate the accelerator pedal such that fuel consumption samples could be taken in steady state for a set of engine loads. The sample points collected were used to fit a polynomial surface of fuel consumption in each gear with the Model-Based Calibration toolbox in MATLAB.

The advantage of this engine mapping technique is that a direct mapping may be made between any speed and tractive force, and the resulting fuel consumption. Measurements include all steady-state losses, for example, bearing, gear, and tire losses, so there are no losses to be accounted for.

### C. Traction Motor

A power electronic inverter and a traction motor were mapped collectively, and as in Section III-B, this approach of modeling systems at the highest level possible, rather than separating them, ensures that all losses as well as any system interdependencies are accounted for.

This mapping set uses the motor speed and dc power as inputs to a 2-D lookup table, which outputs tractive torque. Electrical losses at negative torque are assumed symmetrical at positive torque; however, it is important to consider that directionally dependent losses (bearing friction, windage, and cogging losses), quantified in a zero-load test, are not also reversed. At low positive dc powers, the net shaft torque may still be negative (small drag); however, at low negative dc powers (regeneration), the torque produced must never be positive as this would imply an efficiency greater than 100%.

### D. Battery

Battery voltage is calculated based on open-circuit voltage and then accounting for the voltage drop/rise due to internal resistance

$$V_{\text{batt}} = V_{\text{OC}}(\text{SOC}) - I_{dc} \cdot R_{\text{int}}$$  (2)

where $V_{\text{batt}}$ is the battery terminal voltage, $V_{\text{OC}}$ is the battery open-circuit voltage, which itself is a function of SOC, $I_{dc}$ is the battery current, where positive current indicates discharge and negative current indicates charge, and $R_{\text{int}}$ is the total internal resistance of the battery.

Open-circuit battery voltage is determined as a function of SOC from manufacturer data sheets, and modeled as a third-order polynomial in the range 30%–90% SOC. Polynomial models do not represent $V_{\text{OC}}$ well beyond these limits; however, it is not desirable to use the battery outside of this range and the control strategy uses only a portion of this.
To track SOC, the majority of existing literature performs a numerical integration of battery current, assuming a 100% charge/discharge efficiency of the battery. It is important to acknowledge that the battery is not 100% efficient, and at higher discharge rates, the observed internal capacity is reduced. Johannesson et al. [3] included battery efficiency data in the SOC model. Since this was not available here, the discharge efficiency was modeled using the Peukert effect [26]–[28], with the Peukert constant determined from discharge data provided by the cell manufacturer. It has been suggested by others that the approach can be improved [29],[30] and provisional testing showed that the battery does not deviate significantly in this application from the temperature at which the manufacturer’s tests were performed. The application of Peukert’s law involves the calculation of an effective current, \( I_{eff} \), which includes battery losses and is used to calculate the SOC at the next timestep.

\[
SOC_{k+1} = \frac{SOC_k \times C_{bat} \times 3600}{I_{eff} \cdot dt} - \frac{SOC_k \times C_{bat} \times 3600}{dt} \tag{3}
\]

where

\[
I_{eff} = \begin{cases}
C_{bat} \times \left( \frac{I_{dc}}{C_{bat}} \right)^{K_p} & \text{for } I_{dc} \geq 0 \\
I_{dc} + \frac{I_{dc} \cdot R_{int}}{V_{bat}} & \text{for } I_{dc} < 0.
\end{cases} \tag{4}
\]

Equation (4) incorporates Peukert’s law to describe capacity reduction during discharge events, where \( K_p \) is the Peukert constant and \( C_{bat} \) is the 1 C discharge capacity of the battery in ampere hours. Since this is not applicable to charge events, losses and effective current are calculated assuming that ohmic heating is the dominant factor.

IV. PROBLEM FORMULATION USING STOCHASTIC DYNAMIC PROGRAMMING

Any decision at the present timestep will affect future costs because for a charge sustaining policy, any battery charge consumed now must be replaced in the future; present savings therefore need to be substantial enough to offset any additional cost incurred in the future. The discounted cost over an infinite horizon when starting from the present state \( x_0 \) and following control policy \( \pi \) is defined as

\[
J_{\pi}(x_0) = \lim_{N \to \infty} E \left[ \sum_{k=0}^{N-1} \lambda^k c(x_k, \pi(x_k)) \right] \tag{5}
\]

where \( c(x_k, \pi(x_k)) \) is the instantaneous cost at each timestep \( k \), which is a function of the vehicle state \( x_k \) and the control decision at that vehicle state \( \pi(x_k) \). Since the cost over an infinite horizon would be infinite (5) uses a discount factor, \( 0 < \lambda < 1 \), which reduces the contribution of the costs exponentially as they extend into the future. The cost at the present timestep \( (k = 0) \) is therefore represented with a weighting of 1, while costs that occur in the future are increasingly insignificant; the discount factor controls the rate at which this decay happens. \( E[\cdot\cdot\cdot] \) denotes the expectation of the future cost, since only the cost at the present timestep is known precisely, while all other costs are the result of a statistical distribution.

An appropriate choice of the vehicle state vector is important, as this must allow sufficiently accurate calculation of the instantaneous cost. This work uses the state vector

\[
x_k = [v_k, a_k, g_k, soc_k] \tag{6}
\]

with state variables of vehicle speed, acceleration, gear selection, and SOC, respectively. While the inclusion of gear selection is unusual, this is in our view essential for a proper view of fuel consumption. The resolution of state variables used is shown in Table II; note that the vector \( v \) is nonuniformly spaced.

The optimal policy minimizes \( J_{\pi} \) and is obtained by successive refinement of the policy and the resulting expected cost. For reasons identified in Section I-B, the instantaneous cost is defined as

\[
c(x_k, \pi(x_k)) = f(v_k, a_k, g_k, u_k) + \alpha \cdot C(v_k, u_k)^2 \tag{7}
\]

where the function \( f \) denotes the instantaneous fuel consumption and \( u_k = \pi(x_k) \) is the present control decision, which has the range \( \pm 255 \) and specifies a dc battery current demand as a proportion of the maximum possible at the present speed. Although the electrical energy delivered in one timestep depends on \( V_{bat} \), \( V_{oc} \) in (2) is assumed relatively constant over the usable SOC range and so the instantaneous cost is not a function of SOC. The second term in (7) represents the penalty on high-power system use, but has been implemented using the battery C-rate, \( C = I_{dc}/C_{bat} \), rather than directly using the dc current, in order to make the results more scalable and transferable. The relative importance given to this term is controlled using the weighting factor \( \alpha \).

The statistical model of the future is a Markov chain in which the transition probability of moving from any one speed–acceleration–gear point to any other is explicitly known

\[
\mathbb{P}([V, A, G]_{k+1} = [v, a, g]_{k+1} | [v, a, g]_k). \tag{8}
\]

Transitions between these three state variables \((v, a, \text{ and } g)\) are defined by the drive cycle, and so their probabilities are calculated directly from a library of recorded real-world data. Gear selection at any point in the recorded data is inferred from the ratio of vehicle to engine speed. It is necessary for the state space to be discretized, and the number of discrete states...
heavily impacts the computation time to solve the problem; there is therefore an incentive to use coarse discretization. The state space used here is finer than any other the authors are aware of, but not fine enough that, for example, given \([v, a]_k\), the subsequent \(v_{k+1}\) may be adequately approximated by rounding to the nearest value in the state vector. For this reason, interpolation is used to distribute each state transition between multiple transition probabilities, rather than rounding the subsequent state to the nearest grid node.

In order to complete the full-state transition probability matrix, this Markov chain is augmented with information about SOC transitions, as described in the following section.

V. SDP PRACTICAL IMPLEMENTATION

A. Information Required

Implementation of the SDP algorithm requires three pieces of information.

1) Instantaneous cost of any combination of vehicle state and control decision. This is not a function of SOC (though future cost is), so is 4-D

\[ c_k(v_k, a_k, g_k, u_k). \]  

2) Drive cycle transition probability matrix, as already described by (8).

3) SOC transition matrix, describing the probability of progressing to each SOC at the next timestep, given the SOC and control decision at the present. This transition here depends on vehicle speed, because the control decision specifies a proportion of dc current available at the present speed. Although for each transition \(SOC_{k+1}\) is a scalar, linear interpolation is used to assign this between two grid nodes in a probability distribution. This increases accuracy without the need for an extremely fine grid

\[ \mathbb{P}(SOC_{k+1} = soc_{k+1} | [soc, u, v])_k. \]  

Although the two transition probability matrices in (8) and (10) could be combined into a single matrix in practice, they are kept separate because \(\mathbb{P}(X_{k+1} = x_{k+1} | x_k)\) would far exceed the RAM capacity of modern desktop computers.

B. Computational Procedure

Computation of the optimal policy is achieved by repeating two steps many times over, iteratively approaching the optimal solution. The present iteration of the policy is \(\pi_i\), where the subscript refers to the iteration number (not a time index). An initial guess is required for the control policy \(\pi_0\) and the future expected cost for this policy \(J_{\pi_0}(x_{k+1})\); in this work, both were null (all zeroes). Two steps are then repeated until convergence [31].

1) Policy Evaluation: This step evaluates the future expected cost of following the control policy \(\pi_i\) from every \(x_k\)

\[ J_{\pi_i}(x_k) = c(x_k, u) + \lambda \sum_{x_{k+1} \in X} \mathbb{P}(X_{k+1} = x_{k+1} | x_k, u) J_{\pi_i}(x_{k+1}). \]  

Since (11) is recursive in \(J_{\pi_i}\), it is necessary to iterate within this step, replacing the old value of \(J_{\pi_i}(x_{k+1})\) with the newly calculated value \(J_{\pi_i}(x_k)\). Iterations were truncated when each element in \(J_{\pi_i}(x_k)\) was changing by less than 0.1% per iteration or after a minimum of 20 iterations. On the first execution of this step \((i = 0)\), the cost is constructed from scratch and so several thousand iterations are required; for subsequent executions, \(J_{\pi_{i+1}}(x_k)\) was used as an initial estimate of \(J_{\pi_i}(x_{k+1})\) and so considerably fewer iterations are required.

2) Policy Improvement: Having formed an estimate of the future expected cost, this step seeks to return the control policy that minimizes it

\[ \pi_{i+1}(x_k) = \arg \min_u \left\{ c(x_k, u) + \lambda \sum_{x_{k+1} \in X} \mathbb{P}(X_{k+1} = x_{k+1} | x_k, u) J_{\pi_i}(x_{k+1}) \right\}. \]  

These two steps are repeated until the shape of the control policy stops changing, though the absolute values in the cost function may still be changing.

C. Practical Recommendations

Practical implementation of SDP requires considerable effort to streamline calculations. The techniques used here are detailed in [32] as well as in [15] and [16]; in brief, these include the following.

1) Generating Cost and SOC Transition Matrices: Populating the instantaneous cost and SOC transition matrices (9), (10) is best accomplished using For Each subsystems in SIMULINK. This allows whole matrices to be calculated in one call using vectorized inputs to the vehicle model. If model components are built in a block library, the same models may be used for the main vehicle simulation, thereby avoiding the effort and potential for errors in writing MATLAB scripts, which are equivalent to a SIMULINK model.

2) Sparse Matrices and Cross Products: The probability matrices tend to be extremely sparse, and defining them as sparse matrices (where only nonzero elements are stored in memory) saves enormous amounts of memory and computation time for matrix multiplication. Although limited to two dimensions, this does not limit their usefulness, as multiple states can very easily be nested inside a single dimension for the purposes of matrix operations and then reshaped afterward.

When combining the probability matrices in (8) and (10) for a subset of states, it is very efficient to rearrange one as a sparse column vector and the other as a row vector and use the matrix cross product.

3) Interpolation: A considerable amount of interpolation is performed, and use of MATLAB’s quick version (interp1q) with reduced error checking is worthwhile.

D. Drive Cycle Analysis

A fundamental component in the SDP controller development is the drive cycle data used to create the Markov chain.
With a few exceptions [33], [34], previous works have tended to use a collection of legislative drive cycles to form this data set, and the potential of the resulting controller has not always been demonstrated using a drive cycle realistic of real-world usage.

An enormous collection of real-world data was available to the authors as a result of [35], and a subset of this data representing the vehicles with powertrains most similar to the test vehicle was isolated for creation of the Markov model. The subset of data used represents 2470 km and 101 h of driving.

To justify the potential of the SDP algorithm, it should be tested over a drive cycle representative of the data set with which it was designed. While it would have been possible to develop a bespoke driving cycle based on the recorded driving data and to evaluate the controllers using this, the results would be less meaningful and reproducible for other researchers. For this reason, it was preferred to select the most representative driving cycle from a range of common and publically available legislative drive cycles, using similarity between the speed–acceleration frequency distributions (SAFDs) as the selection criteria. The LA92 was found to be most representative of the vehicles’ real world usage; the details of the analysis are available elsewhere [32]. The SAFD plots for the recorded driving data and the LA92 are shown in Figs. 2 and 3 and the similarity between the two can be seen, though the surface derived from the LA92 is considerably more jagged as a result of the vastly smaller data set on which it is based. Having selected the LA92 speed trace, the gear shift speeds were calculated using the procedure for a shift speed survey set out in code of federal regulations Title 40 §86.128-00 [36] and A/C 72A [37], which the LA92 adheres to for manual vehicles.

It should be noted that in selecting a driving cycle, which is similar to the recorded driving data with which the controller was designed, the results will be optimistic as the energy management will be near optimal. Nevertheless, this remains a logical decision because it is presumed that if the actual driving patterns differed drastically from those with which the controller was designed, then the controller would be reoptimized, if possible.

VI. SELECTION OF THE SDP PARAMETERS

A. Effect of $\lambda$ on Charge Sustenance

Perhaps the most important parameter to select appropriately is the discount factor $\lambda$. Since this defines how quickly the future costs are discounted, it essentially determines how future looking the control strategy is; for example, when $\lambda = 0$, the strategy considers only the immediate cost of its decision, whereas at $\lambda = 1$, the strategy would attempt to look infinitely far into the future.

A strategy that does not look very far into the future will not account for the future costs required to replenish any battery energy consumed in the present; for this reason, strategies built with a low $\lambda$ tend to be charge depleting. Conversely, strategies with $\lambda \to 1$ are more aware of energy balance costs, and so are more charge sustaining. Higher values of $\lambda$ require more iterations before the solution converges, and so computation times escalate quickly. It is not necessary to place a cost on SOC deviation to ensure charge sustenance because with a high enough $\lambda$, the controller can only be charge depleting over an infinite horizon, though it may be desirable to enable the use of a smaller $\lambda$ or to reduce the likelihood of the SOC being very low when the vehicle is switched OFF [8]. Our ambition is to have a $\lambda$ sufficiently high to ensure that the control strategy is charge sustaining, but which does not increase computation time unnecessarily.

To understand the charge sustaining properties of a control strategy, the current ratio $Q_t$ is defined as the ratio of positive to negative dc current occurring at each SOC over the probabilistic drive cycle (assist/regeneration). The result is essentially the controller response as a function of SOC, normalized with respect to the charge sustaining...
condition. This is calculated by multiplying the probability of each \([v, a, g]\) with the controller’s response to that state and dividing the sum of positive values by the sum of negative values

\[
Q_l(soc) = \frac{\sum P(v, a, g) \cdot (M(\pi(x), v) > 0)}{\sum P(v, a, g) \cdot (M(\pi(x), v) < 0)}
\]

where \(M(\pi(x), v)\) is the linear function which maps the normalized control decision into the dc battery current. The ratio \(Q_l\) is greater than 1 where the strategy is generally charge depleting, whereas when less than 1, it is charge gaining. Fig. 4 shows the importance of \(\lambda\) in determining the equilibrium/nominal SOC; it can be seen that increasing \(\lambda\) above 0.999999 does not change the controller response as a function of SOC and so this value was selected.

Probabilistic charge sustenance plots, as presented in Fig. 4, may also prove useful where developers wish to make minor adjustments to the nominal SOC that a controller achieves. For example, if the \(Q_l = 1\) intercept is at 73% SOC, but it is desirable for other reasons that the nominal SOC is 78%, the SOC reported to the controller may simply be offset by 5% (provided that sufficient flexibility exists in the original definition of the usable SOC).

As identified in Section I-A, previous researchers have reported using the values of \(\lambda\) in the range 0.95–0.995. The considerably higher values required in this study are most likely a result of the mild hybridization, use of 10-Hz data where others have used only 1-Hz data, and absence of a cost on SOC deviation. Since driving data were available at 10-Hz, it was used at this frequency throughout and not down sampled, because reduced frequencies would result in considerable loss of information regarding vehicle energy [38].

B. Number of Iterations

Complete convergence of the future expected cost \(J_\pi\) would take an extremely long time; however, we may permit the truncation of iterations when the shape of the control strategy stops changing. Comparing the amount of change between successive control strategies does not give a clear picture of change as there is some amount of cyclical change. For this reason, it is more helpful to define the maximum percentage change of any element in \(J_\pi\) as \(\varepsilon\) and curtail iterations when this falls below a threshold, chosen here as 0.0008%. The difference in the number of grid positions between the final control strategy \(\pi_N\) and each \(\pi_i\) may then be counted \((n_{\Delta})\) to ensure that no cumulative change takes place. Note that each value in \(\pi\) may change by more than one grid position.

Fig. 5 shows by example that \(J_\pi\) is converging exponentially, the threshold is reached after 5719 iterations of \(\pi\), and there is no cumulative change in \(\pi\).

The implementation of SDP realized took about 30 s for each policy evaluation step (which includes 20 iterations) and a similar time for each policy improvement, running on a desktop computer with an Intel Core i7 CPU at 2.0 GHz. Therefore, the evaluation of 5719 iterations of \(\pi\) represents approximately four days of contemporary desktop computing.
Fig. 6. Control policy surface as a function of $\alpha$ and SOC at $v = 32$ km/h and in fourth gear, $\alpha = 0.001$ g·h$^2$. The control decision $u$ is a unitless number defining dc battery current (positive or negative) as a proportion of the maximum available at the present speed. (a) $u$ is in the range $\pm 255$. (b) $u$ is mapped to the range 1–31 to allow storage as an 8-bit integer in memory, and the number of states has been reduced while maintaining the salient shape of the control surface.

Since the rate of convergence is considerably slower with higher values of $\lambda$, it is easy to appreciate the desire to use a $\lambda$ no larger than necessary.

VII. EMBEDDED IMPLEMENTATION

The raw output of the SDP algorithm is a 4-D map, a cross section of which is shown in Fig. 6(a). Before implementation, some postprocessing was necessary.

First, wherever a particular driving state $[v_k, a_k, g_k]$ was not visited in historic real-world driving data, the state transition probability matrix $P(X_1 = x_1 | x_0, u)$ is not well defined, nor is the control policy. In our implementation, the control response was set to zero at these points. By the nature of the problem, these null values occur only in extremely rare driving conditions; however, it is important that these are not problematic in real-world driving. For this reason, any null responses were replaced with the nearest valid control value.

Second, in this implementation, the raw controller map includes 258,912 elements, occupying over 500 kB of memory in single precision. Since our hardware had around half of this memory for all functions, some significant reduction was necessary. This was achieved both by storing map values as indices in the range [1:31], allowing the values to be stored as 8-bit integers, and by reduction in the number of stored elements. For this purpose, a MATLAB script was written to cycle through and remove slices from the $v$, $a$, and SOC dimensions. The slice that minimized the absolute error when missing values were interpolated from the surrounding values was then permanently deleted. This procedure was looped until the resulting map would occupy less than 90 kB. Each dimension was not necessarily reduced in size by the same proportion; in fact, the $a$ and SOC dimensions were reduced most, while the $v$ dimension remained mostly intact. This process allowed the memory required to be reduced considerably without significant detriment to the fidelity of the policy, as can be observed in Fig. 6(b). Leroy et al. [8] proposed using neural networks to model the controller surface, and it is likely that this would also work well. Nevertheless, the process described was used for simplicity and robustness.

VIII. SIMULATION RESULTS

A set of control strategies was created with a range of $\alpha$ to evaluate the effect of this parameter on the control strategy and resulting fuel savings. The expectation is that increasing $\alpha$ should deter operation of the hybrid system at high power, and instead encourage more consistent low-power operation. Naturally, the multiobjective optimization will result in a tradeoff, so some sacrifice of fuel savings is expected.

Normalized fuel savings are plotted against the mean square battery C-rate in Fig. 7 for a range of $\alpha$. Since the controllers tended to be charge depleting over a LA92 simulation, an offset was applied to the controller’s SOC state vector, as suggested in Section VI-A; this was up to 9.5% SOC, depending on the controller and resulting in all of the simulations being charge sustaining to within $\pm 0.5\%$ SOC.

As previously described, the square of the battery C-rate is proposed as an indicator of the aggressivity at which the hybrid system is operated and the stresses to which components are exposed. The nonlinear relationship observed in Fig. 7 suggests that $\alpha$ may be used to reduce aggressive use of the electrical powertrain while maintaining the majority of fuel consumption savings; for example, in moving to 3.24 h$^{-2}$ at $\alpha = 0.002$ g·h$^2$, stress on the electrical powertrain is reduced by 51%, while sacrificing only 20% of the possible fuel savings. This tradeoff strongly resembles that observed by others [18], [19] when incorporating a more formally defined cost specifically on battery aging. It should of course be noted that the reduction in mean square C-rate will be more dramatic
than the reduction in mean C-rate (throughput) would be; however, if the intention is indeed to reduce the incidence of high power use, then the tradeoff curve does reflect this.

The SDP simulation results were compared against the widely documented and accepted equivalent consumption minimization strategy (ECMS) controller, which minimizes the instantaneous consumption of equivalent fuel, where factors other than fuel are represented in the cost function using weighting factors.

To ensure charge sustenance, a weighting factor on electrical energy use was required since a large proportion is regenerated during braking (not generated from burning fuel) and so has no associated fuel cost. Using a notation consistent with (7) and (12), the ECMS controller may be defined as

$$\pi_{\text{ECMS}}(x_k) = \arg \min_u \{c(x_k, u) + \gamma \cdot C(v_k, u)\}. \quad (14)$$

As in the SDP formulation, the first term in the cost function being minimized incorporates the instantaneous fuel consumption and cost for high-power operation. The second term adds a cost on electricity consumption. Values of $\gamma$ were selected with the aid of the simulation to achieve charge sustaining behavior of the policy over a LA92.

Simulation results for ECMS are presented alongside those for SDP in Fig. 8, showing how each is able to tradeoff fuel consumption savings with reduced high-power operation of
the electrical powertrain. In both cases, the value of $\alpha$ varies between 0 and 0.006 g · h$^2$. Both control strategies appear to be able to achieve similar fuel consumption savings exerting comparable stress on the electrical powertrain. However, it is significant that the curve for ECMS extends considerably further into the high-stress region than that for SDP, suggesting that when left unabated (with small values of $\alpha$), ECMS tends to put unnecessary stress on the electrical powertrain—in this case an additional 39%—while actually achieving suboptimal fuel savings. This observation supports that of Liu and Peng [5] who observed through simulation that ECMS can tend to oscillate between heavy assist and regeneration, because engines tend to be most efficient when heavily loaded. In the region of high electrical powertrain stress, $\alpha \to 0$ and the difference in behavior of the two control strategies is due solely to the fact that SDP employs a variable estimation of electrical energy value, whereas ECMS assumes a fixed value, as seen by comparing (12) and (14).

**IX. CHASSIS DYNAMOMETER TESTING**

Testing of the SDP control strategies was performed on a chassis dynamometer test facility at the University of Bath, with the installation as shown in Fig. 9.

Possibly the greatest source of variability in chassis dynamometer testing is the human driver, and it is essential to have an experienced individual in this role. The driver used throughout this work is well accustomed to the task, and the degree of fidelity achieved in following the speed trace is shown in Fig. 10. The role of the driver is yet more important when a highly transient drive cycle such as the LA92 is used, compared with a modal drive cycle such as the new European driving cycle (NEDC), which is comparatively easy to drive accurately. While the NEDC allows generous opportunities during which to change gear, the LA92 does not, and so the greatest deviations from the scheduled speed are due to the driver compensating for the absence of tractive power during a gear shift.

The use of a robot driver was considered for improved repeatability; however, for proper calibration and evaluation of the hybrid controllers, it was important that system inputs were representative of a human driver. Robot drivers may certainly achieve better repeatability; however, their PID control loops and relatively aggressive gear changes do not replicate human behavior, and therefore, it was decided not to use a robot.

**A. Test Procedure**

All tests were carried out at 25 °C ambient temperature. Over a NEDC, the facility is known to have a repeatability on the order of 1% coefficient of variance (CoV) when variables known to significantly affect fuel consumption are properly monitored, such as tire pressures and SOC of the vehicle 12 V battery. Tests were performed with a hot engine in order to allow multiple tests per day. A consistent start of test condition

<table>
<thead>
<tr>
<th>Number of repeats</th>
<th>Std.</th>
<th>SDP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proportion of max. saving (%)</td>
<td>-</td>
<td>97.4</td>
</tr>
<tr>
<td>CoV fuel consumption (%)</td>
<td>0.24</td>
<td>0.53</td>
</tr>
<tr>
<td>$\Delta SOC$ (kJ)</td>
<td>+43.2</td>
<td>+43.8</td>
</tr>
<tr>
<td>$\Delta SOC$ (%)</td>
<td>+2.0</td>
<td>+2.0</td>
</tr>
<tr>
<td>Mean square C-rate, $\overline{C^2}$ (h$^2$)</td>
<td>0</td>
<td>6.58</td>
</tr>
</tbody>
</table>

Fig. 11. Results of chassis dynamometer testing showing the tradeoff between fuel savings achieved and aggressivity of electric powertrain use.
Fig. 13. Excerpt from the LA92 drive cycle showing the instantaneous response of the SDP control algorithm with different values of $\alpha$.

for the engine was achieved by first driving it until the engine coolant reached the ECU target of 90 $^\circ$C; the vehicle was then driven through the extra-urban section of the NEDC (the EUDC), which fed directly into the LA92 test, the point at which logging of emission data commenced. This procedure ensured no delay between the conditioning cycle and the test cycle, resulting in extremely good repeatability. The emission measurement was via two Horiba MEXA 7000 series analyzers, and quoted fuel consumption figures were recorded via the industry standard bag analysis method.

Accurate battery SOC estimation is notoriously difficult, particularly in the normal SOC working range where the battery voltage profile is relatively flat; however, it is essential that the SOC reported to the hybrid controller be consistent between tests in order for the controller’s response to be at all repeatable. For this reason, the battery was charged or discharged prior to each test such that cell voltages were indicative of approximately 70% SOC, and the SOC stored in the battery management system was then manually updated to 70% to ensure consistency between tests. The strategies were generally very good at charge sustaining, and for all of the tests reported, the maximum change in SOC was 73 kJ (about 3%). This was achieved without any offsets applied to the SOC reported to the strategy.

B. Effect of $\alpha$

In total, four different configurations were tested, namely, a baseline configuration with the hybrid system inactive and three variations of the SDP controller with different values of $\alpha$. In each case, a minimum of three repeats were conducted in order to monitor repeatability, and the average results of each configuration are presented in Table III and Fig. 11.

Overall, the trends predicted by the model are upheld: the effect of $\alpha$ is to reduce the mean square C-rate, while this does not necessarily mean a proportional sacrifice of fuel savings. The use of $\alpha = 0.001$ g·h$^2$ enables a reduction in system stress of 13% without any sacrifice of fuel savings. The slight increase in fuel savings with $\alpha = 0.001$ g·h$^2$ with respect to $\alpha = 0$ is considered due to experimental variability, and the real difference in fuel consumption between these points is small.
C. Instantaneous Controller Behavior

To further explore the instantaneous behavior of the controllers with changing values of $\alpha$, a representative test from each configuration is selected. The time series of SOC and dc current are shown in Figs. 12 and 13, respectively. The three tests shown have $\alpha$ values of 0.003, 0.001, and 0 g · h$^{-2}$ and mean square C-rates of 1.84, 5.73, and 6.44 h$^{-2}$, and realized 55%, 111%, and 111% of the maximum achievable fuel savings, respectively (over 100% indicates exceeding the best fuel savings achieved by any controller after repeats are averaged). Fig. 12 suggests that all three strategies follow similar SOC trajectories over a LA92 test, absorbing and discharging energy in the same places, though the tests with a lower $\alpha$ tend to deviate further from the nominal SOC during the test due to more aggressive use of the electrical powertrain. In general, there is no major shift in the scheme that each controller uses. This is further supported by Fig. 13, in which assist and regeneration are generally seen to occur in the same places, though the higher value of $\alpha$ encourages more consistent assist events at around 10 A as well as less aggressive regeneration. The test with $\alpha = 0.001$ g · h$^{-2}$ recorded identical fuel consumption to that with $\alpha = 0$, (despite being more charge gaining) yet had a lower mean square C-rate. Finally, Fig. 13 reemphasizes the extremely transient nature of the LA92 and real world driving, with continual change of acceleration and frequent gear changes, in complete contrast to the NEDC.

X. ROAD TESTING

Following successful testing of the control strategies in a controlled environment, it was desired to test their behavior on road. For all purposes, so far, the controllers’ state vector has included vehicle acceleration as a measure of engine load. This was chosen in order to remain as faithful as possible to the original problem formulation in which the Markov chain uses acceleration. In the absence of any road gradient, this was a valid approximation in the calculation of fuel consumption and therefore the cost function. In the real world, however, the presence of road gradients may introduce a disconnect between acceleration, engine load, and fuel consumption, so it is no longer acceptable to base control decisions on vehicle acceleration; in the worst scenarios, this could be dangerous.

In order to safely test the SDP control strategy on road, it was felt necessary to depart from the acceleration state space and instead to transform this into the engine load state space. This signal was seen as preferable to the throttle pedal signal, which is typically highly calibrated and nonlinear, whereas for diesel engine vehicles, the OBD load parameter reports the present engine torque output as a proportion of the maximum torque available at that engine speed. Having already mapped the engine torque on the dynamometer, it was therefore relatively straightforward to establish a direct relationship between acceleration (at zero road gradient) and engine load and to make the substitution in state space. An example of the resulting transformed surface is shown in Fig. 14, which may be compared against the equivalent surface in Fig. 6.

The procedure used in this work to create a Markov chain based on acceleration, assuming no road gradient, and then transform the resulting control strategy to being load based was not ideal as this process will undoubtedly have incurred some loss of information and therefore loss of optimality. The reasons for having taken this approach are that the vehicles used to collect drive cycle data did not all have the same engine variant, and so the load signals would not have been comparable, nor was the fuel consumption map based on load available for these engines. Furthermore, since the engine load signal can be extremely volatile, it is possible that this may cause unexpected problems in its statistical modeling.
Nevertheless, this is an incremental advance on the work of others and may be reviewed in future research.

Despite deviation from the original, the resulting strategy appears to operate well and manage battery SOC easily. Fig. 15 shows an extract of data recorded on a journey leaving Bath in the direction of Exeter, U.K., following the A36, A39, and A368 roads. The initial SOC is slightly low and this is gradually recovered by reducing the magnitude of assist events as per the controller’s map. As the SOC increases, the assist events become larger, and as a result, the SOC approaches and fluctuates around a nominal SOC. The probabilistic nominal SOC predicted for this control strategy in the simulation was 73% and the behavior on road appears consistent with this; this demonstrates some inherent resilience of the SDP control algorithm to postprocessing and external influences.

XI. CONCLUSION

SDP has been demonstrated as a promising algorithm for optimal hybrid vehicle control by many researchers in a simulation environment. This paper has presented a functioning controller, evaluated on a retrofit hybrid vehicle using representative real-world driving data.

A robust method was set out for the design of the controller, including selection of a discount factor, and graphically representing the charge sustaining properties of the resulting control policy with a view to fine tuning. Those procedures have been applied here to create a SDP controller, which has been successfully demonstrated on a chassis dynamometer and on road.

A novel cost function was used to introduce a simple means by which aggressive use of the hybrid system can be discouraged in favor of more regular low-power operation. This cost function reduces unnecessary load on the electrical powertrain while maintaining the majority of the possible fuel savings; in dynamometer testing, this enabled a 13% reduction in the mean square battery C-rate without compromising any fuel savings. This is expected to yield dividends during real-world operation in thermal management of the electric powertrain (battery and motor) to avoid thermal cutback scenarios, battery management and cell balancing, and long-term battery life.

ACKNOWLEDGMENT

The authors would like to thank Ashwoods Automotive Limited for providing support in supplying performance data and facilitating the integration of the developed control strategies.

REFERENCES


Christopher Vagg received the M.Eng. degree in mechanical engineering from the University of Bristol, Bristol, U.K., in 2009, and the Ph.D. degree from the University of Bath, Bath, U.K., in 2014. He was with the motorsport industry for the Mercedes GP Petronas Formula One team, before undertaking post-graduate study. He was a part-time Development Engineer with Ashwoods Automotive Ltd., Exeter, U.K. He is currently with Renault Sport F1, Paris, France, where he is involved in the development of electric powertrains for the Formula E racing series. His current research interests include control of hybrid vehicles, analysis and modification of driver behavior, and the effect of driver behavior on hybrid vehicle control.

Dr. Vagg is a member of the Institution of Mechanical Engineers. He received the 2012 Fiona and Nicholas Hawley Award for Excellence in Environmental Engineering by the Worshipful Company of Engineers.

Lloyd Ash specialized and began trading in dual fuel LPG autogas conversions and his company Lloyd Ash Conversion Centre has converted over 700 engines, including standard/premium cars, boats, go-karts, and quad bikes in 1999. He saw a gap in the market for hybrid electric vehicle retrofit systems and started Ashwoods Automotive Ltd., Exeter, U.K. Since 2010, Ashwoods have supplied over 750 Ashwoods Hybrid Transits and 6000 driver behavior aids to some of the UK’s major fleets including Royal Mail, BT and the Environment Agency. Ashwoods pride themselves with the skills and knowledge learnt to date and offer custom solutions including electric and hybrid drivetrain development and integration, modeling/simulation and design/build of custom embedded systems. He is currently the Founding Owner and Technical Director of Ashwoods Automotive Ltd.

Sam Akehurst received the B.Eng. degree in mechanical engineering and the Ph.D. degree with a focus on the parasitic losses associated with a pushing metal V-belt continuously variable transmission from the University of Bath, Bath, U.K., in 1996 and 2001, respectively.

He was appointed as a Lecturer with the University of Bath in 2010, where he became a Senior Lecturer in 2014.

Dr. Akehurst is a member of the Institution of Mechanical Engineers and the Society of Automotive Engineers. He received an EPSRC Advanced Fellowship to investigate novel approaches to future vehicle powertrain optimization from 2005 to 2010.

Chris J. Brace received the B.Eng. degree in mechanical engineering and the Ph.D. degree with a focus on transient simulation of direct injection turbocharged and intercooled diesel engines from the University of Bath, Bath, U.K., in 1990 and 1996, respectively.

He was with Massey Ferguson Tractors, Coventry, U.K., as a Transmission Design Engineer. He returned to the University of Bath as a Research Officer in 1992, where he became a Lecturer in 2000, Senior Lecturer in 2006, Reader in 2012, and Professor in 2014.

Prof. Brace is a fellow of the Institution of Mechanical Engineers. He was Chair of the Institution of Mechanical Engineers Automobile Division.