Comprehensive Energy Footprint Benchmarking Algorithm for Electrified Powertrains

Hamza Anwar, Qadeer Ahmed, Senior Member, IEEE, Aashrith Vishwanath, and Apurva Chunodkar

Abstract—Electrification in automotive control systems has made powertrains a complex cyber-physical system. This article presents a benchmark algorithm to quantify the performance of complex automotive systems exhibiting mechanical, electrical, and thermal interactions at various timescales. Traditionally, dynamic programming (DP) has been used for benchmarking performance; however, it fails to deliver results for a system with a higher number of states and control levers due to the curse of dimensionality. We propose “PS3,” a three-step algorithm for mixed-integer nonlinear optimal control problems (OCP) with application to powertrain energy management. PS3 uses pseudospectral collocation (PSC) theory for accurate modeling of dynamics with fast and slow dynamic states, discontinuous behaviors, nondifferentiable and linearly interpolated 1-D and 2-D maps, as well as combinatorial constraints. We give a feasibility analysis along with a discussion on the convergence and optimality gap of the proposed algorithm. Based on the validated powertrain component models, we have addressed simultaneous optimization of electrical [state of charge (SOC)], vehicular (eco-driving), and thermal (after-treatment and battery temperatures) dynamics along with an integer (gear and engine on/off) control and its corresponding (dwell time) constraints. Six powertrain control problems are given to benchmark the accuracy and computational effort against DP. Our analysis shows that this algorithm does not scale computational burden as DP does and can handle highly complex interactions that occur in modern-day powertrains, without compromising nonlinear and complex plant modeling.

Index Terms—Mixed-integer nonlinear programming (MINLP), optimal powertrain control, pseudospectral collocation (PSC).

I. INTRODUCTION

As opposed to fully electric battery electric vehicles (BEVs), modern plug-in hybrid electric vehicles (PHEVs) and hybrid electric vehicles (HEVs) have more complex system structures bringing research challenges in optimal energy management and control design.

Manuscript received 19 August 2022; revised 11 September 2023 and 11 January 2024; accepted 4 February 2024. Date of publication 4 March 2024; date of current version 25 June 2024. Recommended by Associate Editor S. Evangelou. (Corresponding author: Qadeer Ahmed.)

Hamza Anwar and Aashrith Vishwanath were with the Center for Automotive Research, Ohio State University, Columbus, OH 43212 USA. They are now with Cummins Inc., Columbus, IN 47201 USA (e-mail: anwar.24@buckeyemail.osu.edu; vishwanath.16@osu.edu).

Qadeer Ahmed is with the Department of Mechanical and Aerospace Engineering, Ohio State University, Columbus, OH 43210 USA (e-mail: ahmed.358@osu.edu).

Apurva Chunodkar was with the Controls and Diagnostics Research, Cummins Inc., Columbus, IN 47201 USA (e-mail: apurva.chunodkar@gmail.com).

Digital Object Identifier 10.1109/TCST.2024.3367154

Market sales of HEV and PHEV steadily increased over the years 2018–2022 [1] and is predicted to reach 5.8 million global PHEV sales by 2030 [2] (from 1.9 million in 2021 [3]). In contrast to PHEVs, BEV adoption requires huge capital investment in charging infrastructure, which is an unaffordable luxury for most developing countries.

Optimal energy management strategies (EMSs) consider the various subsystems of a powertrain, with their interactions to achieve targets of fuel economy along with emissions of air pollutants and greenhouse gases. Many times, the objectives of an EMS will conflict with each other, such as minimizing the fuel and improving drivability performance. On the other hand, the subsystems of a powertrain may exhibit very different behaviors in their time dynamics and control. For example, an electric battery may have rapid charging and discharging, while, on the other hand, an after-treatment catalyst may have a slow change in its temperature. Likewise, the transmission and clutch subsystems will have discontinuous shifts and switches, while the ratio of power split between the internal combustion engine and the electric machine could be any real number within its bounds. Complicating it further can be a scenario when, in the same energy management problem, “eco-driving” is allowed where one is not restricted to operate on a given drive cycle but is allowed to modulate the speed profile around a target profile. As will be shown later, we consider all aforementioned complexities in this work.

For energy management, these dynamic and discontinuous interactions restrict engineers to only model incomplete and approximated relationships or to model the state and control variables in disjoint subproblems. And so, traditional control strategies focus on individual subcomponent optimality instead of a joint holistic system-level optimization. For example, the engine may be programed to solely operate at its optimal operating line, while the after-treatment system is independently tuned to maintain certain temperatures of its catalysts. But, whether or not all these systems will jointly meet the overall objectives of the powertrain operation remains a question. Hence, there is a need to develop energy management approaches that can handle a high degree of complex system-level interactions, tackle stiffness caused by fast and slow dynamics, and exhibit discontinuous and combinatorial interactions of the optimal control problem (OCP), while meeting the conflicting objectives of hybrid electric powertrains. An example of a powertrain with various subsystems, states,
Among the widely used offline EMSs is discrete dynamic programming (DP), a direct approach, which is popular for its globally optimal solution guarantee serving as a benchmark—originally dating to the year 2000 [12]. Its variants [13], [14], [15] have found space in online EMSs as well. But, its use is limited due to it requiring a priori route information and having an inherent curse of dimensionality, which restricts it from being scaled up for problems involving a handful of real-valued state variables [16]. If integer-valued variables (also called discrete variables) are optimized using DP, such as gear choices, DP may remain computationally tractable (up to a limit) due to state space not growing exponentially. As for DP’s variants, the critical step is to approximate the cost-to-go, to a limit) due to state space not growing exponentially. As for (also called discrete variables) are optimized using DP, such effects having an inherent curse of dimensionality, which restricts it from being used. As for (also called discrete variables) are optimized using DP, such hybrid piecewise methods, such as fast sequential quadratic programming- and semi-definite programming-based integer convex minimization [22], [23], [24]. However, the difficulty in representing highly nonlinear powertrain models in convex forms, as well as the discontinuities caused by integer-valued variables (such as gear shifts and engine on/off switches) and efficiency maps (such as those of combustion engines) restricts the application of convex optimization. Nonlinear programming (NLP) does not assume convex or linear models and is generally difficult to handle. Although there are powertrain energy management works that use NLP using traditional uniform discretization methods, such as [25], a recent trend is to discretize models and differential equations using the pseudospectral collocation (PSC) theory. PSC allows for highly accurate modeling of state dynamics at the expense of NLP size but preserving sparsity, which is exploited by state-of-the-art NLP solvers, such as interior point optimizer (IPOPT) [26] and we optimize really huge problems (WORHP) [27]. With great success in chemical industrial and aerospace research, PSC’s use in automotive research is quite recent. Relevant PSC-based NLP works solving various HEV energy management control problems include [28], [29], [30], [31], and [32].

Very recently, few works have surfaced that combine integer optimization along with PSC-based NLP to solve mixed-integer NLP (MINLP) problems. Closely related to our paper, Robuschi et al. [33] use combinatorial integral approximation theory to solve an MINLP problem involving gear shifts and torque split controls. Among many similarities, our proposed algorithm distinguishes from [33] in separating consistent and inconsistent real-valued variables and solving them in steps, which significantly reduces the computational effort, as explained later. Finally, we refer the reader to introductory tutorials and material on numerical optimization [34], mixed-integer programming [35], NLP, and its sparse solvers [36], [37].

B. Paper Overview

This article presents and validates PS3, an algorithm for solving optimal HEV energy management problems having a large (more than 5) number of state and control variables that cannot be handled by the benchmark DP algorithm due to its curse of dimensionality. PS3 employs PSC theory for direct transcription of the continuous-time optimal control problem into sparse numerical programs solved using numerical solvers. The main contributions are summarized as follows.
The state and control variables being optimized are separated based on modeled dynamics into three clear types (consistent, inconsistent, and discrete) and are solved in distinct steps of the algorithm.

2) PS3 algorithm’s properties of feasibility and convergence to near-optimal solutions are detailed by providing a theorem with supporting assumptions, propositions, and accompanying proofs.

3) Six case-study problems are showcased having an increasing number of variables on a pickup and delivery truck drive cycle with frequent starts/stops and a wide spectrum of power demands.

4) Various elements of complexity are exploited in the case studies: nonlinear dynamics, mixed-integer optimization variables, different timescales of dynamics, eco-driving, dwell-time combinatorial constraints, and so on.

5) Benchmarking of results is done in comparison with DP showing computational benefits with comparable solution quality.

Note that comprehensive implementation of the presented algorithm is in our other paper [38] that focuses not on the algorithm but on detailed energy analysis and commercial application scenarios for specific powertrain problems where the objective function has conflicting terms of fuel consumption and system-out NO₃ emissions.

Section II describes the general formulation of powertrain control problems that we aim to solve. Section III is about the problem formulation structure that we consider powertrain energy management problems we aim to solve. Through this section, we also explain the difficulties associated with solving modern-day HEV energy management problems when considering large-scale joint optimization.

II. PROBLEM DESCRIPTION

As described earlier, we aim to solve such optimal powertrain control problems, which exhibit a large number of dynamical states, a combination of real- and integer-valued controls, and constraints from various interacting powertrain components through complex relationships. This section gives details about the problem formulation structure that we consider for powertrain energy management problems we aim to solve. Through this section, we also explain the difficulties associated with solving modern-day HEV energy management problems when considering large-scale joint optimization.

A. Dynamic States and Controls

The generic optimal control problem can have continuous (real-valued) and discrete (integer-valued) state variables, denoted as \( \mathbf{x}(t) \in \mathcal{X} \subset \mathbb{R}^{|\mathcal{X}|} \) and \( \mathbf{z}_d(t) \in \mathcal{Z}_d \subset \mathbb{Z}^{|\mathcal{Z}_d|} \), respectively, at time \( t \in [0, T] \subset \mathbb{R} \), where \( |\cdot| \) is the set cardinality. Likewise, there can be continuous control variables \( \mathbf{u}(t) \in \mathcal{U} \subset \mathbb{R}^{|\mathcal{U}|} \) and discrete control variables \( \mathbf{u}_d(t) \in \mathcal{U}_d \subset \mathbb{Z}^{|\mathcal{U}_d|} \).

The dynamics for the continuous state variables are specified by ordinary differential equations (ODEs), whose right-hand sides (RHSs) are multiphase nonlinear (and possibly discontinuous) functions of the state and control variables, as well as other dependent signals. As an example, we consider fuel consumption to be a state variable whose time derivative is given by linear interpolation of a 2-D engine map (shaft speed versus net torque), hence having a discontinuous RHS in its differential equation. Similarly, when the vehicle is decelerating (as opposed to accelerating), the torque split control variable becomes free, net available energy can be used for electricity regeneration, and torque split does not affect state dynamics, making the RHS phase-dependent. On the other hand, discrete state variables, such as gear choice, are assumed to be a consequence of only discrete controls having dynamics dependent solely on their respective discrete controls. For example, the engine on/off switch is a discrete control taking values \( 0 \) (no change), \( 1 \) (turn on), and \( -1 \) (turn off). The engine status is a state variable with dynamics dependent only on the engine on/off switch. Thus, its differential equation can be written as a linear combination of shifted and scaled Dirac delta functions, having impulses at the engine on/off switch events. Consequently, the engine status variable will only take binary, i.e., discrete, values.

The continuous state or control variables \( [\mathbf{x}(t), \mathbf{u}(t)] \) are classified into two types: 1) consistent variables, \( \mathbf{x}_{\text{con}}(t) \in \mathcal{X}_c \subset \mathbb{R}^{|\mathcal{X}_c|} \) and \( \mathbf{u}_{\text{con}}(t) \in \mathcal{U}_c \subset \mathbb{R}^{|\mathcal{U}_c|} \), and 2) inconsistent variables, \( \mathbf{x}_{\text{inc}}(t) \in \mathcal{X}_i \subset \mathbb{R}^{|\mathcal{X}_i|} \) and \( \mathbf{u}_{\text{inc}}(t) \in \mathcal{U}_i \subset \mathbb{R}^{|\mathcal{U}_i|} \), where \( |\mathcal{X}_c| + |\mathcal{X}_i| = |\mathcal{X}| \) and \( |\mathcal{U}_c| + |\mathcal{U}_i| = |\mathcal{U}| \). Consistent state variables are defined as those continuous states whose dynamics do not depend on discrete states or controls. On the other hand, the inconsistent states have direct or indirect dependence on discrete states or controls. Furthermore, consistent control variables are all those controls that influence consistent state dynamics. All continuous controls that do not influence consistent dynamics are classified as inconsistent controls. Examples are given later. Thus, we have three types of state ODEs

\[
\begin{bmatrix}
\dot{x}_{\text{con}}(t) \\
\dot{x}_{\text{inc}}(t) \\
\dot{z}_d(t)
\end{bmatrix}
= \begin{bmatrix}
f_{\text{con}}(x_{\text{con}}(t), u_{\text{con}}(t), t) \\
f_{\text{inc}}(x(t), u(t), x_{\text{inc}}(t), u_{\text{inc}}(t), t) \\
f_d(x_d(t), u_d(t), t)
\end{bmatrix}
\]

where \( x(t) = [x_{\text{con}}(t) \ x_{\text{inc}}(t)]^\top \), \( u(t) = [u_{\text{con}}(t) \ u_{\text{inc}}(t)]^\top \), the vector-valued functions, \( f_{\text{con}} : \mathbb{R}^{|\mathcal{X}_c|} \times \mathbb{R}^{|\mathcal{U}_c|} \mapsto \mathbb{R}^{|\mathcal{X}_c|} \), \( f_{\text{inc}} : \mathbb{R}^{|\mathcal{X}_c|} \times \mathbb{R}^{|\mathcal{U}_c|} \times \mathbb{R}^{|\mathcal{X}_i|} \times \mathbb{Z}^{|\mathcal{U}_i|} \times \mathbb{Z}^{|\mathcal{U}_d|} \mapsto \mathbb{R}^{|\mathcal{X}|} \), and \( f_d : \mathbb{Z}^{|\mathcal{X}_d|} \times \mathbb{Z}^{|\mathcal{U}_d|} \mapsto \mathbb{Z}^{|\mathcal{X}_d|} \) are general forms of the RHSs of respective ODEs, which depend on system model dynamics.

The distinction of continuous state variables into consistent and inconsistent helps us bifurcate the solution methodology into three steps as explained later, wherein consistent variables are optimized in the first step, discrete in the second, and inconsistent in the third. Moreover, such distinction naturally arises due to the separability in powertrain problems caused by discrete variables. For example, in a backward powertrain model when gear status is considered to be a discrete state and the vehicle speed a continuous state, then the vehicle speed, distance, and acceleration are all consistent variables, since system causality dictates that these vehicle-level variables come from a known drive cycle, and gear selection depends on those but will not be the other way around. In that example, powertrain variables after transmission will be classified as inconsistent variables because of having dependence on gear...
selection. In general, wherever there is a discrete variable, we try to split the system into its two continuous variable types: consistent and inconsistent. In our case studies, this categorizes vehicle-level variables (speed, distance, and acceleration) as consistent and powertrain-level variables (state of charge (SOC), torque split, after-treatment temperatures, and so on) as inconsistent.

B. Boundary Constraints

All state variables require initial conditions to be defined. These are known constants defining values of every state variable at the initial time. Along with initial conditions for the state ODEs, some state variables also have constraints on the final values they take. For the problems presented in the results section, we impose the charge-sustaining constraint on the battery’s SOC. For problems considering eco-driving, the total distance covered by the eco-driving vehicle driven with speed \( v(t) \) must be the same as the total distance covered with the reference speed profile \( v_{\text{org}}(t) \)

\[
\begin{bmatrix}
  x(t) \\
  x_d(t)
\end{bmatrix}
\bigg|_{t=0} =
\begin{bmatrix}
  x_0 \\
  x_{d,0}
\end{bmatrix},
\begin{bmatrix}
  x(t) \\
  x_d(t)
\end{bmatrix}
\bigg|_{t=T} =
\begin{bmatrix}
  x_T \\
  x_{d,T}
\end{bmatrix}.
\] (2)

C. Algebraic Relationships

To characterize the plant behavior completely, we define another set of variables, which are functions of each other as well as of the states and controls. These are coupled in the optimal control problem through algebraic relationships. We term them signals. For example, in a parallel HEV during traction, the electric machine torque \( \tau_m(t) \) and total demand torque after transmission \( \tau_{\text{total}}(t) \) are related to torque split \( \mu(t) \) through \( \tau_m = \mu \tau_{\text{total}} \). The total demand torque is, likewise, algebraically related to the vehicle speed \( v(t) \) and acceleration \( a(t) \) through transmission efficiency, gear number \( g(t) \), and various road load signals. Another example is of the fuel consumption and engine-out NO\textsubscript{X} emissions, which are based on 2-D lookup tables of engine shaft speed \( \omega(t) \) and engine torque \( \tau_e(t) \). If any of these signals is constrained, then it is effectively a type of path constraint on states or controls. These relationships are numerous ranging from kinematic equations at vehicle and driveline levels to energy conservation and efficiency losses among propulsion (engine and traction motor), after-treatment, and driveline blocks, as well as thermal heat transfer and electric current dynamics.

D. Path and Box Constraints

Box constraints refer to the upper and lower bounds set on each state and control variable by reasonable constants. All box constraints are specified as vectors with subscripts \((\cdot)_\text{lb}\) and \((\cdot)_\text{ub}\) for lower and upper bounds, respectively,

\[
\begin{align*}
  u_{\text{lb}} &\leq u(t) \leq u_{\text{ub}} \\
  x_{\text{lb}} &\leq x(t) \leq x_{\text{ub}} \\
  u_{\text{d,lb}} &\leq u_d(t) \leq u_{\text{d,ub}} \\
  x_{d,\text{lb}} &\leq x_d(t) \leq x_{d,\text{ub}}.
\end{align*}
\] (3)

Along with box constraints, the state and control variables can have explicit or implicit constraints that are time-varying. These are jointly termed path constraints. The examples of explicit path constraints on a state variable are the eco-driving speed constraint and stop-at-stop constraint—vehicle speed \( v(t) \) is constrained to be within a constant envelope of \( V_{\text{margin}} = 5 \text{ km/h} \) above and below the reference speed profile \( v_{\text{org}}(t) \) and should stop when there is a stop in the reference

\[
\begin{cases}
  |v_{\text{org}}(t) - v(t)| \leq V_{\text{margin}}, & \text{if } v_{\text{org}}(t) \neq 0 \\
  0, & \text{if } v_{\text{org}}(t) = 0.
\end{cases}
\] (4)

Likewise, an example of implicit path constraints can be the time-varying min/max limits on signals, such as engine or motor torques

\[
\begin{align*}
  \tau_{e,\text{min}}(t) &\leq \tau_e(t) \leq \tau_{e,\text{max}}(t) \\
  \tau_{m,\text{min}}(t) &\leq \tau_m(t) \leq \tau_{m,\text{max}}(t).
\end{align*}
\] (5)

One important path constraint is the dwell-time constraint of discrete variables. As a motivation, if the controller optimizes the gear selection trajectory to minimize fuel consumption, one may observe a gear-chattering phenomenon. But, it is undesirable for gears to rapidly switch here and there as that causes immense drivability discomfort. Hence, an explicit path constraint is needed on gear switching that limits the number of gear shifts for a certain dwell-time period \( t_{\text{dwell}} \)—this is referred to as \( h_2 \) in (7). Path constraints can be grouped as follows:

\[
h_2(x(t), u(t), x_d(t), u_d(t), t) \leq 0
\] (6)

and are divided into four types

\[
\begin{align*}
  h_1(x_{\text{con}}(t), u_{\text{con}}(t), t) &\leq 0 \\
  h_2(x_d(t), u_d(t), t) &\leq 0 \\
  h_3(x_{\text{con}}(t), u_{\text{con}}(t), x_d(t), u_d(t), t) &\leq 0 \\
  h_4(x_{\text{con}}(t), u_{\text{con}}(t), x_{\text{inc}}(t), u_{\text{inc}}(t), x_d(t), u_d(t), t) &\leq 0.
\end{align*}
\] (7)

The path constraints that solely depend on consistent variables are denoted with \( h_1 \leq 0 \), an example of which is (4). Constraints covered by \( h_2 \) solely depend on discrete variables, an example of which is the dwell-time combinatorial constraint given in Problem 4. Constraints applying to combinations of consistent and discrete variables are captured under \( h_3 \), an example of which is constraining engine shaft speed in its realistic limits \( 0 \leq \omega(t) \leq \omega_{\text{max}} \) (here, \( \omega(t) \) is algebraically related to vehicle speed and gear ratio). Finally, the path constraints comprising \( h_4 \) are composed of all remaining combinations of the variables, examples of which are (5). This distinction within path constraints helps us split the problem and solve it in multiple steps, as explained in Section III.

E. Optimal Control Problem

Finally, using (1)–(3) and (6), we arrive at the complete mixed-integer optimal control problem, Problem 1, where \( x(t) = [x_{\text{con}}(t) \ x_{\text{inc}}(t)]^\top \) and \( u(t) = [u_{\text{con}}(t) \ u_{\text{inc}}(t)]^\top \). The cost function comprises of a running cost \( \psi \) and a terminal cost \( \psi \). Note that, in the following definition, we have particularly identified the vectors in boldface, time-varying signals with “(t)” and constants without “(t)”.
Problem 1 (OCP₁: Optimal Powertrain Control Problem):

\[
\begin{align*}
\min_{u(t), u_d(t)} & \quad \psi(x(T), x_d(T), T) + \int_0^T L(x(t), u(t), x_d(t), u_d(t), t) \, dt \\
\text{s.t.} & \quad \dot{x}(t) = f(x(t), u(t), t) \\
& \quad x(0) = x_0, \quad x_d(0) = x_d, \quad x_d(T) = x_d(T) \\
\end{align*}
\]

\text{Box Constr.:} \quad u_{lb} \leq u(t) \leq u_{ub}, \quad x_{lb} \leq x(t) \leq x_{ub}, \quad u_{d,lb} \leq u_d(t) \leq u_{d,ub}, \quad x_{d,lb} \leq x_d(t) \leq x_{d,ub}

\text{Path Constr.:} \quad h(x(t), u(t), x_d(t), u_d(t), t) \leq 0

\text{Boundary Constr.:} \quad x(0) = x_0, \quad x_d(0) = x_d, \quad x_d(T) = x_d(T)

III. PS3: A Three-Step Algorithm for HEV EMS

A direct (first-discretize-then-optimize) method of numerical optimization is used to solve the optimal control problem. Although an off-the-shelf general-purpose MINLP solver (such as basic open-source nonlinear mixed integer programming (BONMIN) [39] and nonlinear interior point trust region optimization (KNITRO [40]) can be used to solve the control problem directly, it is generally understood that a customized methodology can prove faster and more efficient exploiting problem structure. This is particularly more true for MINLP solvers due to them having many algorithmic parameter choices (e.g., branch-and-bound versus interval methods, strong versus weak branching, McCormick versus alphaBB relaxations, and so on). We propose a three-step approach “PS3,” which does so by solving NLP₁, MIQP₂, and NLP₃ in each step, respectively.

A. Discretization Using PSC

Discretization of the optimal control problem is the first crucial step in solving it using a numerical optimization technique. Furthermore, some of the constraints in our problem definition, that are related to gear dwell time, can be better described only after an equivalent discrete-time problem is defined. Hence, before we attempt to solve the optimal control problem to determine a solution, we discretize it in time. Once an equivalent discrete-time optimization problem is defined, we can then move on to formulating our three-step approach to solve the resultant MINLP. For this discretization of the continuous-time optimal control problem into a discrete-time numerical optimization problem, we make use of the PSC theory.

The pseudospectral method is essentially a high-order implicit Runge–Kutta (IRK)-based collocation scheme in which the time axis is discretized at nonuniform locations, which are determined based on roots of a certain family of orthogonal polynomials. These polynomials are employed to accurately approximate the state trajectories originating from the differential equations that govern the plant dynamics in optimal control problems. Due to high accuracy of derivatives and integrals that comes via such an approximation, PSC has gained a lot of popularity.

We use a discretization step size of \( \Delta t := \Delta t_k = 1 \) s \( \forall k \in \{1, \ldots, N\} \) for the \( k \)th time interval indicated by the time \( t \in [t_{k-1}, t_k) \), and having a total of \( N \) such intervals, called “control intervals,” spanning the complete time horizon \([0, T]\). For notational convenience, when dealing with discrete-time signals, we use \( k \) in parentheses instead of \( t \). The control signals are assumed to be piecewise constant within each of the \( N \) intervals. On the other hand, the state trajectories smoothly vary due to high-order IRK discretization at collocation points. For highly accurate state dynamics modeling, we use five Legendre–Gauss–Radau (LGR) collocation points within each control interval—see Fig. 2. However, for some of our experiments when accuracy is not expected to be compromised or when DP benchmark solution needs to be compared, we use one collocation point per interval.

One specialty of the LGR collocation scheme, unlike other choices of Legendre–Gauss–Lobatto (LGL) or Legendre–Gauss (LG) schemes, is that it includes the decay property, which helps to handle stiffness associated with the corresponding ODEs. We omit details of how the PSC scheme operates to achieve discretization at nonuniform points inside a control interval and refer the reader to exclusive works on the subject [41], [42]. Discretization step size of 1 s for control intervals is chosen, because it does the following: 1) it is the most common choice in the powertrain energy management literature that spans works of the last two decades; see references within [4] and 2) it offers sufficient granularity to be able to appropriately capture dynamics of the multiscale state variables.

Fig. 2. Discretization of continuous-time OCP with five Radau collocation points \( \xi_i \in \{0.06, 0.28, 0.58, 0.86, 1\} \) on an interval \((0, 1]\) for \( i = 1–5\) in each control interval to ensure state continuity and smoothness. Control signal is piecewise constant in each interval of 1-s duration, and corresponding state signal is determined by the fifth-order polynomial using Radau scheme.
Finally, we find that researchers have used PSC with hp-adaptive mesh refinement strategy [43]. Essentially, this phenomenon allows automatically varying the discretization step size as well as the polynomial degree in collocation based on problem dynamics. In other words, at the time of formulating the MINLP from the mixed-integer optimal control problem, it is possible to have different-sized control intervals for each dynamic state or control variable depending on the corresponding timescale. This add-on can reduce computation and improve convergence. Although our developed framework can capture such variable time discretizations, we go with fixed control interval size and collocation polynomial degree, because one of the primary purposes of our paper was to establish a comparison with deterministic DP solutions.

B. Constructing the Relaxed Nonlinear Program

Since the original optimal control problem (Problem 1) involves discrete state and control variables, discretization of the same using PSC into N control intervals, as mentioned above, will transcribe it into a mixed-integer nonlinear program. Instead of numerically solving the MINLP directly, we first apply relaxation to its discrete variables that allows the solver to assume continuous values for the otherwise discrete-valued variables, \([x_k(t), u_k(t)] \forall k \in \{1, \ldots, N\} \]. Thus, if we only have one discrete state, the gear number \(g(t)\) and then the relaxed gear number state \(\tilde{g}(t)\) discretized into N intervals \(\tilde{g}(k)\) \(\forall k \in \{1, \ldots, N\}\) can be any real number within \(1 =: g_{lb} \leq \tilde{g}(k) \leq g_{ub} := n_b\), where \(n_b\) is the total number of gear choices here. For a six-speed transmission, \(n_b = 6\). When \(\tilde{g}(k)\) takes whole number values, the gear ratios correspond to them; however, for fractional gear numbers, the gear ratios are linearly interpolated. For any other discrete state or control variable, an analogous relaxation will be applied to convert the MINLP into an NLP. Furthermore, since imposing combinatorial constraints (such as minimum dwell time) on discrete variables will not make sense for relaxed variables, so we do not impose those in the first step of our three-step algorithm, and take care of it in the second step. Finally, we arrive at the relaxed NLP (Problem 2), which is solved in step 1 of the proposed algorithm by an off-the-shelf gradient-based sparse NLP solver.

\textbf{Problem 2 (NLP\(_1\))}: It is defined as the discretized equivalent of Problem 1 with objective value \(J_1\), a total of \(N\) control intervals of step size \(\Delta t = 1\) s each, using PSC scheme at LGR points for the state trajectories, where the following hold.

1. The combinatorial constraints of discrete variables \(h_2(x_0(t), u_d(t), t) \leq 0\) are ignored.
2. Discrete states \(x_0(t) \in \mathbb{Z}^{[\mathbb{K}_0]}\) and discrete controls \(u_d(t) \in \mathbb{Z}^{[\mathbb{K}_d]}\) are, respectively, replaced by relaxed states \(\tilde{x}_0(t) \in \mathbb{R}^{[\mathbb{K}_0]}\) and relaxed controls \(\tilde{u}_d(t) \in \mathbb{R}^{[\mathbb{K}_d]}\) using piecewise linear interpolation.

\textbf{Note}: Although we have used inner convexification by having piecewise linear interpolation when relaxing the discrete variables above, this does not restrict us to use other reformulations for the same NLP that are better at guaranteeing feasibility of subsequent problems. As stated in [35], the use of inner convexification may make the optimality gap between the relaxed problem and the original arbitrarily large. This implies that the closest integer trajectories to the relaxed trajectories computed in Problem 2 can be far away from true optimal solution. However, for most practical real-world examples of integer controls as stated in [44, Ch. 3], especially with the use of piecewise linear interpolation in gears or decoupling in engine on/off, it will retain proximity to the true optimal solutions. Based on discussions in [44], partial outer convexification of discrete variables can better guarantee feasibility for our steps 2 and 3, by providing tighter relaxation and almost integer feasible relaxed solution of NLP\(_1\). Nonetheless, this relaxation in step 1 ensures that discrete variables after rounding retain the feasibility of \(h_3\) and \(h_4\).

C. Constructing the Mixed-Integer Quadratic Program

Assuming that we can obtain a solution \([x^\ast(k), u^\ast(k), \tilde{x}_0^\ast(k), \tilde{u}_d^\ast(k)] \forall k \in \{1, \ldots, N\}\) to the nonlinear program NLP\(_1\), we now describe step 2 of our algorithm that handles integer optimization. From the obtained solution, the consistent variables are assigned their fixed trajectories, which do not alter after this step.

\[ x^\ast_{\text{con}}(k) \xrightarrow{\text{assign}} x'_\text{con}(k), \quad u^\ast_{\text{con}}(k) \xrightarrow{\text{assign}} u'_\text{con}(k). \]

Step 1 resulted in relaxed trajectories for the discrete states and controls, which are not integer valued nor do they meet dwell-time constraints. Now, the primary focus in step 2 is to obtain the integer states and controls \([x^\ast_d(k), u^\ast_d(k)]\), which are closest to their relaxed counterparts from step 1, \([\tilde{x}_0^\ast(k), \tilde{u}_d^\ast(k)]\). In doing so, the solution should also satisfy any combinatorial constraints on discrete variables. This is achieved by defining a mixed-integer quadratic program (MIQP).

\textbf{Problem 3 (MIQP\(_2\))}: Given optimized relaxed trajectories of discrete variables \(\tilde{x}_0^\ast(k)\) and \(\tilde{u}_d^\ast(k)\) and fixed trajectories of consistent variables \(x^\ast_{\text{con}}(k)\) and \(u^\ast_{\text{con}}(k)\) for \(k \in \{1, \ldots, N\}\) from the solution of Problem 2, solve the following mixed integer quadratic program to obtain \(x^\ast_d(k)\) and \(u^\ast_d(k)\):

\[
\min_{x_d, u_d} J_2 := \sum_{k=1}^{N} \left\| x_d(k) - x'_d(k) \right\|^2
\]

\text{s.t.} \quad x_d(k + 1) = f_d(x_d(k), u_d(k))

\[
\begin{align*}
& u_{d,lb} \leq u_d(k) \leq u_{d,ub} \\
& x_{d,lb} \leq x_d(k) \leq x_{d,ub} \\
& x_d(1) = x_{d,0} \\
& x_d(N) = x_{d,T} \\
& h_2(x_0(k), u_d(k)) \leq 0 \\
& h_3(x^\ast_{\text{con}}(k), u^\ast_{\text{con}}(k), x_d(k), u_d(k)) \leq 0 \\
& h_4(x^\ast_{\text{con}}(k), u^\ast_{\text{con}}(k), x_{\text{inc}}(k), u_{\text{inc}}(k), x_d(k), u_d(k)) \leq 0
\end{align*}
\]

\[ \begin{cases} x_{\text{inc}}(k), u_{\text{inc}}(k) \text{ satisfying (1)--(3)} \end{cases} \]

Note that, since this problem optimizes only the discrete variables, direct shooting discretization scheme is adopted instead of direct PSC.

Typically, Problem 3 can be easily framed as an MIQP due to the following.
1) The objective function is quadratic in discrete variables.
2) The discrete state dynamics, \( f_b \), are typically linear combinations of shifted and scaled Dirac delta functions dependent on the integer-valued control—which makes them linear equality constraints.
3) Path constraints \( h_1(x_{\text{con}}(k), u_{\text{con}}(k)) \) are not included, as they are already met due to fixing consistent variables from step 1.
4) Combinatorial path constraints \( h_2(x_{\text{inc}}(k), u_{\text{inc}}(k)) \) can usually be formulated linearly in terms of discrete variables, such as the minimum dwell-time constraint.
5) In most practical applications, discrete variables usually decouple or enter linearly into the other path constraints \( h_3(x_{\text{con}}(k), u_{\text{con}}(k), x_{d}(k), u_{d}(k)) \) and \( h_4(x'_{\text{con}}(k), u'_{\text{con}}(k), x_{\text{inc}}(k), u_{\text{inc}}(k), x_{d}(k), u_{d}(k)) \) with fixed consistent variables \( [x_{\text{con}}(k), u^*_{\text{con}}(k)] \), avoiding nonlinearity with respect to those.
6) As for unknown inconsistent variables in \( h_1(\cdot, \cdot, x_{\text{inc}}(k), u_{\text{inc}}(k), x_{d}(k), u_{d}(k)) \), we make a critical assumption that there exist some trajectories \( [x_{\text{inc}}(k), u_{\text{inc}}(k)] \), which retain feasibility in step 3. In practice, many times it is possible to separate the dependence of \( [x_{\text{inc}}(k), u_{\text{inc}}(k)] \) with a convex hull—as is the case with min/max limits of motor and engine torque in imposing feasible gear choices in Problem 4—hence, this assumption is not difficult to satisfy. However, in general, this may not be guaranteed, making step 3 potentially infeasible. In such a situation, iteratively prefixing inconsistent variables (possibly by leveraging solution of step 1 and engineering knowledge about rules adopted in real scenarios) and solving step 2 may allow step 3 to retain feasibility. Of course, this will come at the expense of loss in optimality. For the scope of this article, we limit ourselves with Assumption 2 avoiding this possibility altogether. Nonetheless, for all problems encountered, any such heuristic was never needed.

1) Gear Example of MIQP: To better explain Problem 3, we give an example MIQP problem that has gear number as a discrete control variable and is imposed with dwell-time constraints. Let us denote the optimal relaxed gear number obtained from step 1 as \( \hat{g}(k) \). First, we transform the scalar relaxed gear trajectory \( \hat{g}(k) \in [1, n_g] \subset \mathbb{R} \) for \( k = 1, \ldots, N \) into a vectorized binary equivalent \( \mathbf{r}(k) = [r'_1(k), r'_2(k) \ldots r'_n(k)]^T \in [0, 1]^{n_g} \subset \mathbb{R} \) where each element of the vector represents one of the \( n_g \) gear choices. Representing dwell-time constraint using binary variables that take values 0 or 1 is simpler than representing it using integer variables. To arrive at \( \mathbf{r}(k) \) from \( \hat{g}(k) \), we distribute the percentage difference in between floor \( \lfloor \hat{g}(k) \rfloor \) and ceil \( \lceil \hat{g}(k) \rceil \) integers indicating likelihood of belonging to one of the two nearest integer gear numbers. For example, if the relaxed gear took a value of \( \hat{g}(k) = 3.38 \) at the \( k \)th control interval, then its best integer value has 38% likelihood of being in the fourth gear and 62% of being in the third gear. The equivalent binary vector for six-speed transmission will be \( \mathbf{r}(k) = [0 0 0.62 0.38 0 0]^T \). Given the relaxed vectorized gear trajectory as an input, the following MIQP aims to determine the binary-valued vectorized gear trajectory \( \mathbf{b}(k) \in [0, 1]^{n_g} \) for \( k = 1, \ldots, N \). The minimum gear dwell-time constraint needs to be properly defined in discrete-time at this point, and by the use of binary variables, this task is simplified to two sets of inequalities for each possible gear choice at each time step, as given in Problem 4.

Problem 4: For every \( k \in [1, \ldots, N] \) grid interval and \( n_b \) gear choices at each step, obtain the binary gear trajectory, \( \mathbf{b}(k) \in [0, 1]^{n_b} \) that minimizes sum of its squared differences from the input relaxed gear trajectory, \( \mathbf{r}(k) \in [0, 1]^{n_b} \)
\[
\min_{\mathbf{b}(k)} \sum_{k=1}^{N} \sum_{j=1}^{n_b} (b_j(k) - r'_j(k))^2
\]
s.t. one-gear-at-a-time constraint \( \forall k \)
\[
1 = \sum_{j=1}^{n_b} b_j(k)
\]
feasible gear selection constraint \( \forall k \forall j \)
\[
0 \leq b_j(k) \leq B_j(k) := \begin{cases} 1, & \text{if } j \text{th gear is feasible at } k \\ 0, & \text{if } j \text{th gear is infeasible at } k \end{cases}
\]
minimum dwell-time constraints \( \forall k \forall j \)
\[
\forall i \in \{k, k + 1, \ldots, k + \frac{t_{\text{dwell}}}{\Delta t} \} : \\
b_j(k) - b_j(k - 1) \leq b_j(i) \\
b_j(k - 1) - b_j(k) \leq 1 - b_j(i)
\]
where \( t_{\text{dwell}} \) is the minimum dwell-time duration in seconds that gear has to remain unchanged before next gear shift. Here, gear feasibility limit \( B_j(k) \forall k \forall j \) is precalculated using min/max shaft speed limits and torque limits of internal combustion engine and traction motor, based on Problem 2's solution \( [x_{\text{inc}}(k), u_{\text{inc}}(k)] \) of consistent variables.

As a result of solving Problem 4, we obtain the optimal binary-valued discrete variable trajectories \( \mathbf{b}(k) \) for \( k = 1, \ldots, N \). This is transformed back into optimal integer trajectories of the discrete variables \( x^*_d(k) \) and \( u^*_d(k) \), which completes integer optimization.

Problem 5 (\( \text{NLP}_3 \)): This is the same as Problem 2 where the only difference is that consistent variables \( (x_{\text{con}}^*, u_{\text{con}}^*) \) and discrete variables \( (x^*_d, u^*_d) \) are fixed and known beforehand from steps 1 and 2. It seeks to minimize the same objective function, denoted as \( J_3 \) using only the inconsistent variables \( [x_{\text{inc}}(k), u_{\text{inc}}(k)] \forall k \in [1, \ldots, N] \) subject to relevant set of dynamical, box, path, and boundary constraints from Problem 1. Here also, there are \( N \) control intervals of step size \( \Delta t = 1 \) s each, employing PSC using LGR points.

D. Overall Algorithm

Referring to the algorithm diagram of Fig. 3, once \( \text{NLP}_1 \) and \( \text{MIQP}_2 \) are solved, we have the optimal consistent variables \( (x_{\text{con}}^*, u_{\text{con}}^*) \) (see step 1) and the optimal integer variables \( (x^*_d, u^*_d) \) (see step 2). These are used to obtain overall optimal solution for the remaining variables, i.e., inconsistent variables \( (x_{\text{inc}}^*, u_{\text{inc}}^*) \) using the nonlinear program defined by Problem 5. This complete process is explained in an algorithmic form in
Algorithm 1 (PS3) Mixed-Integer Powertrain Control Using NLP and PSC

1: Load drive cycle information, model parameters, and maps
2: Obtain naïve initial guess for all the state and control variables \((\bar{x}_d, \bar{u}_d, \bar{x}_{\text{inc}}, \bar{u}_{\text{inc}}, \bar{x}_d, \bar{u}_d)\) which can simply be rule-based
3: **Step 1:** Assume relaxed values \((\tilde{x}_d, \tilde{u}_d)\) for the integer-valued variables, \((x_d, u_d)\) and then solve the large non-linear program (Prob. 2) to obtain optimal trajectories of all states \((\bar{x}'_{\text{con}}, \bar{x}'_{\text{inc}}, \tilde{x}_d)\), and controls \((\bar{u}'_{\text{con}}, \bar{u}'_{\text{inc}}, \tilde{u}_d)\):

\[
\begin{align*}
(\bar{x}'_{\text{con}}, \bar{x}'_{\text{inc}}, \tilde{x}_d) & \leftarrow \text{solve NLP}}_1(\bar{x}'_{\text{con}}, \bar{x}'_{\text{inc}}, \tilde{x}_d) \\
(\bar{u}'_{\text{con}}, \bar{u}'_{\text{inc}}, \tilde{u}_d) & \leftarrow \text{solve NLP}}_1(\bar{u}'_{\text{con}}, \bar{u}'_{\text{inc}}, \tilde{u}_d)
\end{align*}
\]

4: Fix the optimal trajectories of consistent variables from the obtained solution of step 1:

\[
(\bar{x}^*_{\text{con}}, \bar{u}^*_{\text{con}}) \leftarrow \text{assign } (\bar{x}'_{\text{con}}, \bar{u}'_{\text{con}})
\]

5: **Step 2:** Using optimal trajectories of consistent variables \((\bar{x}^*_{\text{con}}, \bar{u}^*_{\text{con}})\) and the relaxed variables \((\tilde{x}_d, \tilde{u}_d)\) solve mixed-integer quadratic program (Prob. 3) to obtain integer solutions \((\bar{x}'_{\text{inc}}, \bar{u}'_{\text{inc}})\) respecting all relevant constraints including the combinatorial constraints. This can be done by transforming integer variables into vectorized binary equivalents:

\[
(\bar{x}'_{\text{inc}}, \bar{u}'_{\text{inc}}) \leftarrow \text{solve MIQP}}_2(\bar{x}^*_{\text{con}}, \bar{u}^*_{\text{con}}, \tilde{x}_d, \tilde{u}_d)
\]

6: **Step 3:** By fixing the optimal trajectories of discrete variables from step 2 \((\tilde{x}_d, \tilde{u}_d)\) and consistent variables from step 1 \((\bar{x}^*_{\text{con}}, \bar{u}^*_{\text{con}})\), solve the second nonlinear program (Prob. 5) with step 1’s solution as an initial guess. Here, all the inconsistent state and control variables will be re-optimized.

\[
(\bar{x}'_{\text{inc}}, \bar{u}'_{\text{inc}}) \leftarrow \text{solve NLP}}_3(\bar{x}'_{\text{inc}}, \bar{u}'_{\text{inc}})
\]

7: The overall mixed-integer solution is finally thus obtained \((\tilde{x}, u^*, \tilde{x}'_{\text{inc}}, \tilde{u}'_{\text{inc}})\), where the continuous variables are \(u^* = [u^*_{\text{inc}} u^*_{\text{con}}]^T\) and \(x^* = [x'_{\text{inc}} x'^*_{\text{con}}]^T\).

Algorithm 1, where writing the parenthesized “(k)” is avoided for brevity.

### E. Convergence and Optimality

With the described problem structure in Section II, we assume that all constraints and dynamics conform to real-world scenarios and physics. In particular, path constraints \(h_1\) are only considered in step 1 of our algorithm, and hence, we make it a necessary assumption that any solution obtained for consistent variables satisfying \(h_1\), its ODEs, and corresponding box and bound constraints should have realistic feasible solutions for inconsistent and discrete variables. As an example, (4) is based on Environmental Protection Agency (EPA) guidelines of maintaining vehicle speed with \(\pm 5\) km/h of target, and such deviation from real-world driving cycles will not violate our assumption. Likewise, the combinatorial constraints of discrete variables \(B_2\) are assumed to be linear (or be able to be reformulated as linear), which is usually the practical case, such as for dwell-time and maximum switching time constraints. To avoid numerical issues, we relax bounds on final value constraints, such as total distance covered and final SOC value, as well as on some path constraints. These assumptions, together with comments on step 1’s relaxation approach discussed after Problem 2, help in avoiding infeasibility issues and guaranteeing convergence. Here, we point out that if we assume that solutions to NLP1 and MIQP2 exist, then it can be very well guaranteed that a solution to NLP3 will also exist. This follows from the fact that steps 1 and 2 already take care of all possible constraints that can affect step 3.

As for guaranteeing convergence to an optimal solution, this is possible to argue for because of the three steps being individually locally or globally optimal. Sparse off-the-shelf NLP and MIQP solvers that we use in each of the three steps ensure convergence to locally or globally optimal solutions. We outline this in Proposition 1, Assumption 2, and Theorem 3. Moreover, we present empirical evidence to argue for near-global optimality of PS3 algorithm. As will be shown in Section IV, PS3 is able to attain as good solutions as globally optimal DP can, at considerably fast speed.

**Proposition 1:** If the MIQP in Problem 3 has a feasible integer solution \(x_k(k) \in X_d \subset Z^{[d]}, u_k(k) \in U_d \subset Z^{[d]}\), for given trajectories of consistent variables \(x^*_{\text{con}}(k) \in X_c \subset R^{[c]}, u^*_{\text{con}}(k) \in U_c \subset R^{[c]}\) and of relaxed counterparts of the discrete variables \(\tilde{x}_d(k) \in X_d \subset Z^{[d]}, \tilde{u}_d(k) \in U_d \subset R^{[d]}\) from Problem 2, on a fixed equidistant discretization grid \(\forall k \in \{1, \ldots, N\}\), then the optimal objective value attained in solving it

\[
J_* \leftarrow \text{MIQP}_2(x^*_{\text{con}}(k), u^*_{\text{con}}(k), \tilde{x}_d(k), \tilde{u}_d(k))
\]

is bounded above by a constant (proportional to horizon length \(N\))

\[
J_* \leq \epsilon(N).
\]

**Proof:** See Appendix B.

**Assumption 2:** If feasible solutions exist for Problems 2 and 5, the NLP solver, IPOPT [26], used in steps 1 and 3 of PS3 algorithm converges to locally optimal solutions

\[
\begin{align*}
(J_1^*, x^*_{\text{con}}, u^*_{\text{con}}, \tilde{x}_d, \tilde{u}_d) & \leftarrow \text{NLP}}_1, \\
(J_3^*, x^*_{\text{inc}}, u^*_{\text{inc}}) & \leftarrow \text{NLP}}_3(x^*_{\text{con}}, u^*_{\text{con}}, \tilde{x}_d, \tilde{u}_d)
\end{align*}
\]

**Theorem 3:** Let \(J\) be the optimal objective value of Problem 1. For given trajectories of \(x^*_{\text{inc}}(t), u^*_{\text{inc}}(t), x^*_{\text{con}}(t),\) and \(u^*_{\text{con}}(t)\) with \(t \in [0, T] \subset R\), if there exist feasible trajectories of consistent variables \(x^*_{\text{con}}(t), u^*_{\text{con}}(t)\) satisfying all constraints of Problem 1, which keep \(|J|, |\nabla x_{\text{con}} J|,\) and \(\nabla u_{\text{con}} J|\) bounded, then from Proposition 1 and Assumption 2, it follows that

1) \(J_1^* \leq J^*\)

2) \(|J_3^* - J^*| \leq \delta,\) for some \(\delta \in R\).

**Proof:** Assertion 1), which provides a lower bound on the objective, is straightforward, because Problem 2 is essentially a relaxed version of the discretized Problem 1 where combinatorial constraint was ignored. Assertion 2) follows from the assumptions stated in the theorem statement and in Section III-E and the structure of the problem wherein consistent state dynamics are independent of discrete and independent variables. \(\square\)
TABLE I

| Variable Name [units] | Used in | Space Disc. | Lower Bound | Upper Bound |
|-----------------------|---------|-------------|--------------|-------------|
| Battery State of Charge [-] | Case 1-6 | 61 | 0.3 | 0.8 |
| Torque Split [-] | Case 1-6 | 21 | 1 | 1 |
| Battery Temperature [°C] | Case 2, 4, 6 | 8 | 23 | 30 |
| Gear Number [-] | Case 3, 4, 6 | 6 | 1 | 6 |
| Gear Dwell-time Count [s] | Case 3, 4, 6 | 5 | 0 | 4 |
| Gear Shift [-] | Case 3, 4, 6 | 3 | −1 | 1 |
| Vehicle Speed [m/s] | Case 5, 6 | 26 | 0 | 25 |
| Vehicle Position [m] | Case 5, 6 | 65 | 0 | 6463 |
| Vehicle Acceleration [m/s²] | Case 5, 6 | 15 | −2 | 1.5 |
| Engine Status [-] | Case 6 | 0 | 1 |
| Engine Dwell-time Count [s] | Case 6 | 0 | 3 |
| Engine On/Off Switch [-] | Case 6 | 0 | −1 | 1 |
| Pre-DOC Temperature [°C] | Case 6 | 0 | 600 |
| DOC Temperature [°C] | Case 6 | 0 | 600 |
| DPF Temperature [°C] | Case 6 | 0 | 600 |
| SCR Temperature [°C] | Case 6 | 0 | 600 |

**IV. PERFORMANCE ANALYSIS: PS3 VERSUS DP**

In this section, we present the usefulness of our proposed algorithm in comparison with the standard benchmark of DP for HEV energy management problems. We ran experiments for a variety of problems using PS3 and DP, having different combinations of real- and integer-valued state and control variables in the OCP. The cost function considered for all six case study problems in this article is the total fuel consumed over the drive cycle. The real-valued variables are all inconsistent variables in Cases 1–4, are all consistent variables in Case 5, and are a combination of consistent and inconsistent variables in Case 6. A discussion subsection summarizes conclusions from the results.

For uniform comparison, a discretization step size of one second is chosen for both algorithms (PS3 and DP), and a first-order polynomial degree is used for collocation in PS3 (with LGR points). The initial guess for the optimization variables used by PS3 was based on naive rule-based estimate of control trajectories and corresponding state trajectories. The DP solver we use is based on the well-known “dpm” function method by Sundstrom and Guzzella [16]. Space discretizations used in DP for the state and control variables are provided in Table I.

The hardware used for all simulation results presented in this article was a Lenovo ThinkPad X1 Carbon Laptop PC with an Intel® Core® i5-8250U CPU and 8-GB RAM running Windows 10. To model and solve the nonlinear programs, we use CasADi 3.4.5 [45], within MATLAB. A MATLAB-based toolbox that interfaces CasADi, yet another optimal control problem parser (YOP) [46], is used to parse the optimal control problems into nonlinear programs. Off-the-shelf solvers IPOPT [26] and Gurobi [47] are used as part of the algorithm.

**A. Architecture, Models, and Drive Cycle**

For all our case studies in this article, we have considered a parallel P2 HEV architecture, such as the one shown in Fig. 1. This is for a medium-duty truck with a diesel engine, a 90-kW-rated electric machine, 11-kWh lithium-iron-phosphate (LFP) battery pack with 31-Ah capacity, and six-speed automatic transmission. Data maps for the LFP battery model used in this article are given in Appendix A. All other modeling details about the internal combustion engine, electric machine, vehicle dynamics, and driveline are given in our other paper [38]. A short segment of 10-min duration from the NREL drive cycle for parcel delivery, Fig. 4, with a speed-dependent gear profile is used as the reference for the following experiments, as shown in Fig. 5.

**B. Case 1: Basic Hybrid (1S1C)**

This problem involves a single real-valued state, battery SOC (ζ), and a single control variable, torque split between the internal combustion engine and the electric motor (µ). Since no integer variables are involved in this problem, hence, only step 1 of Algorithm 1 is relevant and used, which gives the final optimal solution.

The control and state-space discretization required for DP is set to take 61 values for SOC (0.3 ≤ ζ ≤ 0.8) and 21 values...
for the control variable, torque split ($-1 \leq \mu \leq 1$). This discretization is chosen to keep minimum computation time and memory load, without a significant drop in the optimality of the solutions. A point to note is that, unlike DP, PS3 can take all real values up-to-machine precision for the state and control variables, i.e., its search space is not discretized the way it is for DP.

The obtained results are plotted in Fig. 6. Although the state (SOC) and control (torque split) trajectories appear different at many places in the plot, we observe that both the algorithms have comparable overall cost, i.e., total fuel consumed—1.920 kg (DP) and 1.921 kg (PS3), and have low computational times—4 s (DP) and 19 s (PS3).

C. Case 2: Thermal Hybrid (2S1C)

The second problem builds on top of the basic hybrid problem by involving two real-valued state variables, battery SOC and battery temperature, and one control variable, torque split. With the additional state variable of battery temperature having a first-order thermal dynamics model, we make use of a temperature-dependent (and SOC-dependent) 2-D lookup table for cell internal resistance (see Appendix A for its modeling details). Our LFP battery model has very low ohmic heat loss for a 10-min drive cycle. In fact, the overall change in battery temperature is within 1 °C of the ambient temperature (25 °C). For this reason, we discretize the battery temperature values in DP to take any of eight uniformly spaced values within 23 °C and 30 °C. Results are plotted in Fig. 7. Again, we observe that performance is comparable—1.93 kg (DP) and 1.921 kg (PS3), and computational times are still low—8.21 s (DP) and 23.27 s (PS3). Furthermore, the trajectories are different, yet the overall effect on the cost is similar. An observation is that DP has higher battery utilization instances causing more current to be drawn in and out, and hence, the battery temperature rises more in the DP solution.

D. Case 3: Gear Hybrid (1S1C and 2D1D1C)

This case involves a mixed-integer optimal control problem. It considers one real-valued state, battery SOC, and one real-valued control variable, torque split (1S1C), but also has two integer-valued states, gear number and gear dwell-time counter, and one integer-valued control, gear shift command (2D1D1C). As for DP, the space discretization for real-valued variables is the same as before, and the integer-valued variables have search space at only their respective feasible integer values (e.g., gear number can be an integer from 1 to 6, gear command can be an integer from −5 to 5, and so on). Being the first mixed-integer case of the six case studies, all three steps of the PS3 algorithm can be seen in action. In its first step, we obtain a relaxed gear profile (shown along with the results plot). The second step solves an MIQP to find an integer gear profile near the relaxed profile while meeting the 3-s dwell-time constraint. The third step obtains the optimal real-valued signals with the input of the known gear profile obtained from the second step. Results of this case study are shown in the plots of Fig. 8. Key observations for this problem from the plots are as follows.

1) Gear profiles of PS3 (blue) and DP (orange) are quite different, and so are the torque split profiles. But, the total fuel consumed by the end is almost identical—1.819 kg (DP) and 1.815 kg (PS3).
2) Computational times are 502.8 s (8.4 min) (DP) and [733.8+11.2+16.5=] 761.5 s (12.7 min) (PS3).
computational load is still reasonable, because most of the state or control variables involved are integer-valued in this problem. DP handles integer-valued variables quite well, because space discretization is simplified for integers. Computation time for PS3 is slightly higher than that for DP, as it requires the running of three sequential computational programs to be solved over its three steps.

3) Gear profile from DP solution tends to take higher values, which is generally better for fuel reduction, but it comes at the expense of steeper drops in the SOC. To meet the charge-sustaining constraint, the DP solution then uses larger magnitudes of engine torque values costing higher fuel. The net result is that DP's fuel trajectory is lower in the first half of the cycle, but by the end, it meets up with that of PS3 resulting in identical overall fuel consumed.

4) Since we use the interior-point solver, IPOPT, we see that the relaxed gear trajectory (magenta dashed line in the first subplot) of step 1 is close to the given rule-based initial guess (green solid line). This confirms that having a good initial guess, especially when integer-valued variables are involved, is critical for good solution quality.

We point out that if finer space discretization is set, DP can give a better solution than the one presented at the expense of a higher computational load. As a test, we solved the same gear hybrid problem with torque split discretization levels of 201 (instead of 21) and the battery SOC discretization levels of 101 (instead of 61), keeping the other control and state variables at default space discretization of Table I. With this, the DP solver achieved the fuel consumption of 1.76 kg taking 7144 s (119 min).

**E. Case 4: Thermal Gear Hybrid (2S1C and 2D\_S1D\_C)**

To demonstrate how DP starts to become intractable for more complicated problems, we combine Cases 2 and 3 to make Case 4. Essentially, on top of the state and control variables of the gear hybrid problem, now, we have a fourth state variable, that of battery temperature, which is real-valued like SOC. So, this problem involves two real-valued and two integer-valued state variables, and similarly one real-valued and one integer-valued control variable.

Some plots for the obtained results are shown in Fig. 9. Overall, fuel consumed is 1.818 kg (DP) and 1.817 kg (PS3), and computational times are 5046.6 s (84 min) (DP) and 385.02 + 9.15 + 71.19 = 465.4 s (8 min) (PS3). In Fig. 10, the corresponding engine operating points are compared between DP and PS3. Optimized gear trajectory by PS3 prefers overall lower gear selection, which is generally an indication of higher fuel consumption (which is because when upshifting to a higher gear on an isopower line of a typical engine map, engine shaft speed reduces, and fuel consumption lowers). However, the engine maps show a cluster of points at a high fuel range (top right) for DP, whereas that cluster is moved to a slightly lower fuel range for PS3. Thus, net fuel consumption is approximately equal between DP and PS3.

The key takeaway from this experiment is that PS3 remains computationally reasonable despite an added real-valued variable, however, due to its curse of dimensionality, DP starts to require large memory and computational resources. As for why this case takes less CPU time using PS3 compared with the previous one: the answer is ambiguous mainly because the error definition which is checked against set tolerances of the stopping criteria is not the same for all cases, and that is an
artifact of the NLP solver, IPOPT. This would happen due to various factors, such as differences in initial guess, nonuniform scaling of the optimization variables, and/or cost function.

**F. Case 5: Eco-Hybrid (3S2C)**

Finally, we consider a case in which we have three state variables (SOC, vehicle speed, and vehicle position) and two control variables (torque split and vehicle acceleration). The idea of eco-driving is to allow the vehicle to maneuver within a 5-km/h threshold of the reference target speed profile (shown in Fig. 11), such that the total distance covered and the total travel time on the whole route are the same. Simultaneously, torque split control is also optimized.

As opposed to DP, PS3 is able to arrive at a solution. The computation time running PS3 for this experiment was 585.62 s, and the total fuel consumed was 1.90 kg. Since all these variables are real-valued, DP exceeds the memory resources and fails to give a solution. When we tried a coarse space discretization that does not exceed available memory, it was unable to find a feasible solution that met total travel time and total distance covered constraints due to the coarseness. In Fig. 11, the benefit of eco-driving versus non-eco-driving (reference) scenario is highlighted, showing a reduction of 2.21% in cumulative net energy demand at the wheels. This cumulative net energy demand at wheels, shown in the third subplot, is a direct consequence of the vehicle speed related through the road load equation and has nothing to do with torque split optimization or battery SOC.

When analyzing the plots, we observe that the effect of eco-driving optimization is that the eco-driven vehicle operates at slightly lower speeds when the reference is at high speeds and operates at slightly higher speeds when the reference is at very low speed—this behavior allows the eco-driven vehicle to demand less energy from the powertrain, and thus, later on, the torque split optimization results in lower cumulative fuel of 1.90 kg compared with the basic hybrid problem’s 1.92 kg.

**G. Case 6: Large Case Study (8S2C and 4D21C)**

This problem involves after-treatment system dynamics along with all features of the above cases, thereby having two continuous controls, two discrete controls, eight continuous states, and four discrete states. A new discrete control variable is the engine on/off key switch. Engine switching is also constrained with a minimum dwell time to avoid chattering. When the engine is turned off, the traction force is solely provided by the battery. The diesel after-treatment system is modeled using four temperature states, of Pre-diesel oxidation catalyst (DOC), DOC, diesel particulate filter (DPF), and selective catalytic reduction (SCR) catalysts. Having these state variables allows us to additionally consider NOx emissions in the objective function if needed. Modeling details are given in [38] wherein various cost function cases and a Pareto-front study are analyzed and presented. As for problems in this article, the only objective function is to minimize fuel consumption.

In the results, we can see a drop in net fuel consumption from gear hybrid’s 1.82–1.40 kg. Computationally, this problem takes $551.3 + 40.2 + 761.8 = 1353.3$ s (22.5 min). Due to the large problem size and memory requirement, DP is unable to solve this problem. For comparison, we compare the PS3 solution (1.40 kg) with another solution obtained using PS3 having predetermined integer variables (gear and engine on/off) using a rule-based logic (1.44 kg). This rule-based approach has no discrete variable involved, and hence, its use of the PS3 algorithm is only a single-step solution. In other words, this comparison shows the benefit of the three-step approach in comparison with the single-step infused with the rule-based approach. The rule-based logic for choosing gear profile comes from a speed-dependent lookup table and may violate the dwell-time constraint, while that for switching the engine on/off is to turn the engine off whenever the vehicle decelerates.
The most significant tuning factors that PS3 requires by the user are (a) scaling of the cost function, which is also considered in other numerical solvers, including the following: 1) DP; 2) NLP solver parameters that define the stopping criteria, such as error tolerances (for IPOPT, “tol,” and “acceptable_tol”); and 3) initial guess of the solution. A good initial guess can tremendously improve performance. In the presented results, we provide naïve initial guesses for trajectories of control variables and use them to simulate open-loop model dynamics to get initial guesses for state variables. Naïve infeasible initial guesses were used in our experiments despite having better options, to show the robustness of the solver, which can be seen in the figures of results. All these parameters impact computation time, convergence, and optimality of the solution.

In Table II, we have summarized the total fuel consumed for the various problems we present to benchmark the performance of PS3 against DP. For Cases 5 and 6, DP cannot give a solution, and we compare their PS3 solution with another PS3 solution, “Other,” as discussed earlier. First, we observe that for problems that DP can solve, PS3’s solutions match DP’s globally optimal solutions, establishing the general acceptability of PS3 as an alternative benchmark with near-global optimality against DP. Sometimes, PS3 gives lower fuel values compared with those given by DP. This does not mean that DP’s solution is not globally optimal. Rather, such artifacts exist only because the DP solver has numerical limitations stemming from limited discretization in the space domain. Second, despite PS3 utilizing a gradient-based optimization approach, the integer-valued variables (Cases 3, 4, and 6) are well optimized, giving benchmark performance with the help of state-of-art MIQP solver, Gurobi. Third, it is well known that as the number of state and control variables in an optimal control problem increases, it becomes tedious for DP to give tractable solutions without compromising sensible space discretization levels. This is due to its curse of dimensionality and is particularly apparent when real-valued variables are involved. For this reason, DP is unable to work for Cases 5 and 6, which involve many control and state variables of diverse dynamics, while PS3 solves these in considerably little time.

Based on the presented results and remarks made, we estimate trends of computation time and cost function with increasing problem size in Figs. 12 and 13, respectively.
V. CONCLUSION

This article presents “PS3,” a benchmarking algorithm for solving mixed-integer optimal control problems arising in the energy management of electrified powertrain applications. It is appraised against the well-known DP, which suffers from the curse of dimensionality when handling a high number of states and controls. It employs direct PSC for accurate state dynamics estimation and relies on state-of-the-art numerical optimization solvers for NLPs and MIQPs. Built upon the modeling language CasADi and interfaced with the easy-to-use MATLAB-based NLP parser for optimal control problems, YOP, it runs IPOPT and Gurobi solvers in its three steps optimizing consistent continuous variables in the first step, discrete variables in the second, and inconsistent continuous variables in the third. The proposed algorithm is examined for handling feasibility at each of its steps and is supported mathematically by a proposition and a theorem along with underlying assumptions for its convergence to near-optimal solutions.

PS3 is applied for powertrain problems that involve simultaneous eco-driving and integer optimization along with nondifferentiable lookup tables, thermal states, and combinatorial path constraints in the models. The included results show six optimal powertrain control problems with increasing levels of complexity solved to minimize fuel consumption in offline simulations comparing performance with DP solutions. Various combinations of continuous and discrete states and controls were chosen in setting up the problems, with results providing empirical justification for PS3’s potential of being considered an alternative to DP for large-sized comprehensive problems. A pickup and delivery truck drive cycle was chosen with frequent starts and stops, steep acceleration and deceleration events, and a wide range of power demands. Our analysis shows that this algorithm does not scale in computational load as much as DP does and can handle highly complex interactions that occur in modern-day powertrains. It can be applied to difficult real-world powertrain control problems for offline benchmarking.

APPENDIX A

BATTERY MODEL

In the appendix, the battery model is presented that is used for the shown experiments and numerical results for the hybrid electric powertrain. The battery pack model used is of 11-kWh LFP having 350-V nominal voltage. Charge sustaining operation is assumed for the drive cycle, and so, the initial condition and final condition for SOC are set equal to 55%. For the electrical dynamics, we assume a zeroth-order equivalent circuit model, and for the thermal dynamics, a first-order temperature model with heat addition due to ohmic losses. The equations with further details are given in [38]. However, the only difference between the battery model presented in [38] and this is that here we have temperature-dependent internal resistance 2-D maps, which, along with the open-circuit voltage (OCV) plot, is shown in Fig. 14. OCV is modeled using the following expression:

$$V_{oc} = N_s(V_0 + \alpha_b(1 - e^{-\beta_b \zeta}) + \gamma_b \zeta + \zeta_b(1 - e^{-\epsilon_b/(1-\zeta)}))$$

where \(\zeta, N_s, V_0\) are battery SOC, number of cells in series, and nominal voltage, respectively, while the remaining constants \(\alpha_b, \beta_b, \gamma_b, \zeta_b, \) and \(\epsilon_b\) are obtained by curve fitting the OCV with respect to SOC using real-world empirical data.

APPENDIX B

PROOF FOR PROPOSITION 1

The proof is split into two cases, followed by a summary.
1) For the case, where no combinatorial constraints exist in Problem 3, i.e., the constraint $h_2(x_d(k), u_d(k)) \leq 0$ is not present, the problem reduces to standard rounding, i.e.,

$$x_d(k) = \left\lfloor \bar{x}_d(k) + \frac{1}{2} \right\rfloor.$$  

Thus, we have

$$J^*_2 = \sum_{k=1}^{N} \| x_d(k) - \bar{x}_d(k) \|^2 \leq N \left( \frac{1}{2} \right)^2 = \frac{N}{4}.  $$

2) If the combinatorial constraint of minimum dwell time, $h_2(x_d(k), u_d(k)) \leq 0$, is imposed for one discrete state, $x_d(k) \in \{1, \ldots, p\} \subset \mathbb{Z}$, where its minimum dwell time is denoted by $D \leq N$, we contend that $J^*_2$ remains bounded by a constant determined by $N$ and $D$

$$J^*_2 \leq \left( \frac{1}{2} \right)^2 \left\lfloor \frac{N}{D + 1} \right\rfloor + \left( p - \frac{1}{2} \right)^2 \left\lfloor \frac{ND}{D + 1} \right\rfloor.  \quad (8)$$

Inequality (8) is the worst-case bound for a suboptimal rounding strategy as described as follows: let the rounded integer value at step $k$ be $\bar{x}_d(k) :\{ \bar{x}_d(k) + (1/2) \}$; then, apply following rounding to obtain $x_d(k)$ that satisfies minimum dwell time constraint of $D$:

$$x_d(k) = \begin{cases} x_d(k-1), & \text{if } \sum_{i=1}^{k-1} |x_d(k-i) - \bar{x}_d(k)| > 0 \\ \bar{x}_d(k), & \text{otherwise} \end{cases}$$

where, for values of $k \leq D$, the summation is stopped before $k-i$ reaches 0. The worst-case difference at any given $m$ between the relaxed value and the integer value can be when the integer value is forced to be at one extreme due to recent switch just before $m$, $x_d(m) = 1$, and the relaxed value is at the other extreme, such that $\bar{x}_d(m) = (p + 0.499 \ldots)$, and this will cause their difference to be $\approx |p - 1 + 0.5| = |p - 0.5|$. With this described rounding scheme, we can see by induction that

$$D = 1 \implies J^*_2 \leq \left( \frac{1}{2} \right)^2 \left\lfloor \frac{N}{2} \right\rfloor + \left( p - \frac{1}{2} \right)^2 \left\lfloor \frac{N}{2} \right\rfloor$$

$$D = 2 \implies J^*_2 \leq \left( \frac{1}{2} \right)^2 \left\lfloor \frac{N}{3} \right\rfloor + \left( p - \frac{1}{2} \right)^2 \left\lfloor \frac{2N}{3} \right\rfloor$$

$$\vdots$$

$$J^*_2 \leq \left( \frac{1}{2} \right)^2 \left\lfloor \frac{N}{D + 1} \right\rfloor + \left( p - \frac{1}{2} \right)^2 \left\lfloor \frac{DN}{D + 1} \right\rfloor.$$  

3) At $D = 0$, which means no dwell-time constraint exists, the bound in 2) reduces to $N/4$ also shown in 1). Thus, we have a bound $\epsilon$ given in terms of $D, N$, and $p$ for the one variable case in inequality (8). This case for one discrete variable is easily extendable to more than one discrete variable having respective minimum dwell time constraints imposed, and we suffice it to say that $J^*_2 \leq \epsilon$. Because of assuming the existence of an integer feasible solution (see proposition statement) and the fact that relaxed counterparts, $\bar{x}_d(k)$ and $\bar{u}_d(k)$, were obtained using Problem 2, the path constraints $h_1 \leq 0$ and $h_2 \leq 0$ in (7) were ignored in the proof.

REFERENCES

[1] Transportation Energy Data Book, 40th ed., Table 6.2, U.S. Dept. Energy, Energy Technology Office, Oak Ridge Nat. Lab., Oak Ridge, TN, USA, Jun. 2022. [Online]. Available: https://tedb.ornl.gov/data/

[2] B. Walton, J. Hamilton, G. Alberts, S. Fullerton-Smith, E. Day, and J. Ringrow. (Jul. 2020). Electric Vehicles. [Online]. Available: https://www2.deloitee.com/us/en/insights/focus/future-of-mobility/electric-vehicle-trends-2030.html

[3] Global EV Outlook 2022, IEA, Paris, France. 2022. [Online]. Available: https://www.iea.org/reports/global-ev-outlook-2022

[4] D.-D. Tran, M. Vafaiepour, M. El Baggadi, R. Barrero, J. Van Mierlo, and O. Hegazy, “Thorough state-of-the-art analysis of electric and hybrid vehicle powertrains: Topologies and integrated energy management strategies,” Renew. Sustain. Energy Rev., vol. 119, Mar. 2020, Art. no. 109596, doi: 10.1016/j.rser.2019.109596.

[5] Y. Huang, H. Wang, A. Khajepour, H. He, and J. Ji, “Model predictive control power management strategies for HEVs: A review,” J. Power Sources, vol. 341, pp. 91–106, Feb. 2017.

[6] A. Bürger, C. Zeile, A. Altmann-Dieses, S. Sager, and M. Diehl, “Design, implementation and simulation of an MPC algorithm for switched nonlinear systems under combinatorial constraints,” J. Process Control, vol. 81, pp. 15–30, Sep. 2019.

[7] J. Wu, J. Ruan, N. Zhang, and P. D. Walker, “An optimized real-time energy management strategy for the power-split hybrid electric vehicles,” IEEE Trans. Control Syst. Technol., vol. 27, no. 3, pp. 1194–1202, May 2019.

[8] S. Onori, L. Serrao, and G. Rizzoni, Hybrid Electric Vehicles: Energy Management Strategies, vol. 13. London, U.K.: Springer, 2016.

[9] B. Egardt, N. Murgovski, M. Pourabdollah, and L. J. Mardh, “Electromobility studies based on convex optimization: Design and control issues regarding vehicle electrification,” IEEE Control Syst. Mag., vol. 34, no. 2, pp. 32–49, Apr. 2014.

[10] L. Serrao, S. Onori, and G. Rizzoni, “A comparative analysis of energy management strategies for hybrid electric vehicles,” J. Dyn. Syst., Meas., Control, vol. 133, no. 3, May 2011, Art. no. 031012.

[11] M. Ghasemi and X. Song, “Powertrain energy management for autonomous hybrid electric vehicles with flexible driveline power demand,” IEEE Trans. Control Syst. Technol., vol. 27, no. 5, pp. 2229–2236, Sep. 2019, doi: 10.1109/TCST.2018.2838555.

[12] A. Brahma, Y. Guzennec, and G. Rizzoni, “Optimal energy management in series hybrid electric vehicles,” in Proc. Amer. Control Conf. (ACC), Jun. 2000, pp. 60–64, doi: 10.1109/ACC.2000.878772.

[13] Y. Pan and E. A. Theodorou, “Probabilistic differential dynamic programming,” in Proc. Int. Conf. Adv. Neural Inf. Process. Syst., vol. 27, 2014, pp. 1907–1915.

[14] W. Li, G. Xu, Z. Wang, and Y. Xu, “Dynamic energy management for hybrid electric vehicle based on approximate dynamic programming,” in Proc. 7th World Congr. Intell. Control Autom., Jun. 2008, pp. 7864–7869.

[15] L. Johannesson, M. Ashbogard, and B. Egardt, “Assessing the potential of predictive control for hybrid vehicle powertrains using stochastic dynamic programming,” IEEE Trans. Intell. Transp. Syst., vol. 8, no. 1, pp. 71–83, Mar. 2007.

[16] O. Sundstrom and L. Guzzella, “A generic dynamic programming MATLAB function,” in Proc. IEEE Int. Conf. Control Appl., Jul. 2009, pp. 1625–1630.

[17] V. Larsson, L. Johannesson, and B. Egardt, “Cubic spline approximations of the dynamic programming cost-to-go in HEV energy management problems,” in Proc. Eur. Control Conf. (ECC), Jun. 2014, pp. 1699–1704.

[18] W. B. Powell, Approximate Dynamic Programming: Solving the Curses of Dimensionality, vol. 703, Hoboken, NJ, USA: Wiley, 2007.

[19] C.-C. Lin, H. Peng, and J. W. Grizzle, “A stochastic control strategy for hybrid electric vehicles,” in Proc. Amer. Control Conf., vol. 5, Jun. 2004, pp. 4710–4715.

[20] P. Psu, E. Silani, G. Rizzoni, and S. M. Savaresi, “A LMI-based supervisory robust control for hybrid vehicles,” in Proc. Amer. Control Conf., vol. 6, Jun. 2003, pp. 4681–4686.

[21] N. Robuschi, M. Salazar, N. Viscera, F. Braghin, and C. H. Onder, “Minimum-fuel energy management of a hybrid electric vehicle via iterative linear programming,” IEEE Trans. Veh. Technol., vol. 69, no. 12, pp. 14575–14587, Dec. 2020.
