Optimizing Power Split Controls in Hybrid-Electric Vehicles through Dynamic Programming

EEL4935 Electronics Project Internship Report

Asgard Kaleb Marroquin (U33253968)

Technische Universität Dresden, Germany

Summer 2016

Preface

My internship this summer took place in Dresden, a small city on the east side of Germany, where industry and STEM-related technical training have taken strong root. In particular, I interned with the Innovationzentrum Fahrzeugmechatronik, or the Department of Vehicle Mechatronics, at the Technische Universität Dresden. The department works directly with the transportation agency in Dresden – the Dresdener Verkehrsbetriebe (DVB) – in reducing fuel consumption and heat losses in the city's ever-growing hybrid-electric bus fleet. The group has also collaborated with private companies including BMW, Daimler, Honda, Volkswagen, Continental, Evobus, and MAN. The department investigates various vehicular aspects, such as energy optimization strategies for powertrain management, thermal losses from passenger exchange and smaller engine sizes, bus log data analysis, real-time component diagnostics, and conductor assistant systems for achieving more efficient driving cycles. This summer I worked on a project in collaboration with Continental evaluating online energy management systems, where I calculated the benchmark global minimum fuel consumption and corresponding optimal energy management control system for a hybrid electric vehicle through dynamic programming.

Introduction

Various control methods have been investigated for optimizing energy management systems, or EMSs, for hybrid electric vehicle (HEV) powertrains. Two main control strategies exist for developing optimal HEV powersplit strategies: numerical methods for known *a priori* speed trajectories demand profiles *(*noncausal) and analytical methods calculating without such knowledge (causal) (Jager, van Keulen, and Kessels 2013; Elbert 2013). Noncausal methods, requiring perfect future driving condition predictions, are able to compute across the entire given driving profile and find globally optimal solutions for EMSs (Tempelhahn et al 2016). Were noncausal optimization methods used in realtime, stochastic inputs deviating from the defined driving cycle would influence calculations; as a result, such methods are typically performed offline and are used as benchmarks for comparing and designing efficient online causal EMSs (Tempelhahn et al 2016). Causal methods performing online, on the other hand, can only optimize locally portions of the driving cycle problem and thus return suboptimal strategies (Tempelhahn et al 2016).

Dynamic programming (DP) is typically used for calculating globally optimal EMS solutions due to its flexibility and ability to ensure global optimality. DP can utilize both mixed-integer and continuous state variables with nonlinear and nonconvex models, conditions which other methods cannot directly make function (Elbert 2013; Murgovski et al 2013). However, dynamic programming can only practically optimize an EMS using a limited number of state variables because computational time increases exponentially as more state permutations for a given driving profile are analyzed – i.e. the "curse of dimensionality" (Bellman 1957). DP also requires predetermined travel profile data and is thus noncausal by nature. Nevertheless, due to the repetitive, predictable nature of city bus routes, speed profiles for hybrid-electric transit can be approximated based on past logged data records. Standard driving trajectories, such as the New European Driving Cycle, attempt to mimic urban driving speed profiles and are typically used for determining typical urban emissions (Global Fuel Economy Initiative); these trajectories can also be used as a baseline for testing HEV EMSs.

Alternative causal optimization methods such as convex optimization and methods utilizing Pontryagin's maximum principle (PMP) are also researched in order to implement online control systems in HEVs. Although far faster than DP, these methods naturally have their limitations. For example, convex optimization requires a linear convex model utilizing continuous state variables in order to function effectively. Should a problem be originally nonconvex, it must be manually made convex, possibly affecting optimality results due to model simplification (Tempelhahn et al 2016). And although mixed-integer boolean state variables – engine on-off control, for example – can be theoretically relaxed to become valid as continuous between zero and one (Murgovski et al 2013), it was agreed upon by the research group that smoothing out strict boolean controls may yield questionable results at best and the procedure may not yet be generally accepted as sound. PMP-based strategies are also limited to processing only continuous, unrestricted state variables for short time/space horizons, but can also find near-globally optimal results for such variables at an exponentially faster rate than dynamic programming (Tempelhahn et al 2016).

Due to the strengths and weaknesses of various different optimization control methods, they have been combined in order to compliment each other's characteristics. Faster procedures such as convex optimization and PMP can be integrated into DP algorithms and optimize the continuous variables in the model. This frees up the number of states DP must calculate and dedicates the algorithm to optimize just mixed-integer states, considerably reducing computation time as a result (Uebel et al 2016, Murgovski et al 2013). Should an accurate DP-PMP combined method become quick enough to be implementable online, near optimal EMSs would improve on-road HEV fuel efficiencies. However, as concluded by Uebel et al, this method is not yet fast enough to be implemented online and must be further investigated.

Two main projects were defined for this summer internship. In an attempt to speed up DP-PMP calculations for online testing, my first task involved further speeding up and converting the MATLAB/MEX DP-PMP algorithm developed by my supervisor into C/MEX so that it may be tested in an Autobox, a rapid control prototyping device. This algorithm uses the continuous variables vehicle kinetic energy, battery electric energy, and time, as well as discrete gears, as state variables to minimize fuel use for an articulated bus model making control decisions at a spatial resolution of every ten meters. The details of this control method can be found in (Uebel et al 2016). Although the port was successful, I found no noticeable improvement in calculation time when comparing the C/MEX port to the original MATLAB/MEX script to make the method viable for online testing. This is most likely as a result of the large amount of optimization that has already gone into running MEX functions so that their computation speed is essentially the same as an equivalent C script (not to mention the extraordinary convenience of also avoiding having to manually port code). Nevertheless, I was able to speed up the algorithm by approximately 33% by vectorizing and rearranging state permutation calculations within the code. The project aided in learning the fundamentals of dynamic programming and modern controls through “hands-on” experience with DP, and proved invaluable for completing the second project described in this report.

More relevant to working with controls, this report presents my second task in developing a new DP algorithm which would find the dynamically optimal power split control of a Ford Focus-like P2-HEV model based on continuous battery energy levels, discrete gears, and boolean engine on-off state variables. The goal of developing this dynamic programming method was to create a globally optimal benchmark with which future online strategies can be compared to. The temporally resolved offline model, requiring a predefined driving profile, is based on the New European Driving Cycle velocity trajectory. By developing a DP algorithm based on modern control theory which models and dynamically optimizes battery use in a hybrid-electric bus at the macroscopic level, this internship has exposed me to the field of controls, a fundamental field in electrical engineering.

In order to describe the project in detail, the vehicle model utilized this summer is first described. The state variables – battery energy, gear, and engine state –and the battery energy state dynamics alongside gear and engine control signals defining changes in the state variables are also defined. The vehicle model and state variables are combined in the description of the DP model minimizing the given cost function – that is, fuel use. Finally, results from the algorithm showing optimal trajectories for all possible final ending battery energy states are shown and discussed.

Problem Description

New European Driving Cycle

The New European Driving Cycle (NEDC), a speed profile mimicking a typical commute in a European urban environment, is used as a vehicular travel profile for the DP algorithm (Figure 1). The driving cycle consists of four Urban Driving Cycle (UDC) driving periods simulating sporadic low vehicle speed and engine load experienced in large cities, followed by an Extra Urban Driving Cycle (EUDC) simulating faster, more aggressive travel (Global Fuel Economy Initiative).

State Variables

The HEV is being optimized for minimal fuel use based on three state variables: battery buffer state of charge (SOC) energy level E_B, gear selection g, and engine on-off decision e (1).

$$\boldsymbol{x}=\langle E_B, g, e\rangle \quad (1)$$

Control signals for the discrete gear and engine states are also defined as an input vector (2).

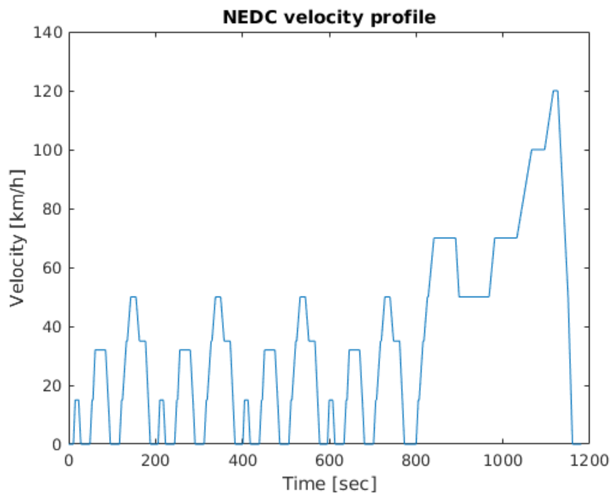
$$\boldsymbol{u}=\langle u_g, u_e\rangle \quad (2)$$

The battery energy level E_B, a continuous variable in reality, must be discretized so that the state variable can exist in the DP state space. The finer the resolution, less overshoot will occur and the algorithm would yield more accurate results. However, more finely resolved changes in E_B require exponentially higher computing times due to the increased number of possible E_B states and thus requires checking additional state variable permutations. Therefore, a battery energy level resolution ΔE_B of 1000 J is chosen in order to attempt to balance these two factors.

Vehicle Model

In order to optimize the power split control in the HEV, a vehicle model able to meet the wheel torque demand satisfying the NEDC must be defined. Vehicle model parameters (eg vehicle mass, battery power and energy limits, internal combustion engine (ICE) and electric motor (EM) speed and torque bounds, etc) in this model are based on a hypothetical Ford Focus HEV (Table 1).

Defining the power demand split for an HEV will depend on the vehicle's drivetrain architecture. The model works with a P2, or parallel drivetrain, in which the ICE and the EM are completely decoupled but are both connected to the gearbox. This permits both machines to independently contribute to the requested crankshaft power demand $P_{CRS}(g, u_g)$ (Jager, van Keulen, and



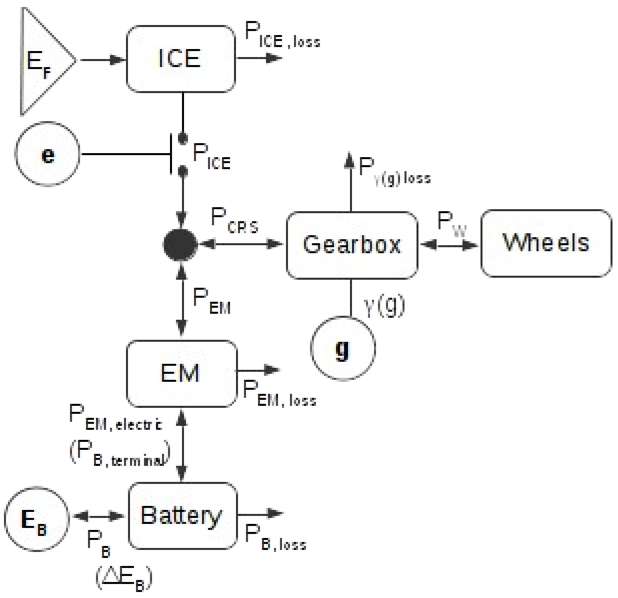


Figure 1: The New European Driving Cycle (NEDC), the velocity profile used in this project's DP optimization. (Global Fuel Economy Initiative)

Figure 2: Model parallel HEV drivetrain used. The three state variables (shown as circles) affect how the ICE and EM satisfy the power demanded by the wheels. DP analyzes possible state variables values and, by controlling their state dynamics at discrete points of time, optimizes cumulative fuel use (shown as a triangle). Figure based on (Jager, van Keulen, and Kessels 2013). P_{ICE} and P_{EM} losses are considered during interpolation.

Kessels 2013).As engine on-off control e is a state being optimized for in this EMS, the ICE can either contribute power $P_{ICE}(\boldsymbol{x}, \boldsymbol{u})$ to (e=1) or be disengaged from (e=0) the crankshaft, leaving the EM to supply enough power $P_{EM}(E_B)$ to completely satisfy P_{CRS} for the given time index decision moment (3a). Should a given P_{ICE} exceed the P_{CRS} demand when e=1, the excess power can go back into recharging the battery buffer (Figure 2).

$$P_{ICE}(\boldsymbol{x},\boldsymbol{u})=\begin{cases} P_{CRS}(g,u_g)-P_{EM}(E_B) if\ e=1 \\ 0\ if\ e=0 \end{cases} \quad (3a)$$

$$P_{CRS}(g,u_g)=P_{ICE}(\boldsymbol{x},\boldsymbol{u})*e+P_{EM}(E_B)-P_{BRAKE}*(1-e) \quad (3b)$$

To calculate the requested power demand, the crankshaft speed and the sum of forces at the wheel must first be determined. The vehicle's velocity v, wheel constant rolling radius r, and gear ratio $\gamma(g)$ define the crankshaft's rotational velocity $\omega_{CRS}(g)$ (4), where $\boldsymbol{v}$ is the NEDC speed profile discretized every second.

$$\omega_{CRS}(g)=\frac{v}{r}\gamma(g) \quad (4)$$

Drag Coefficient ($0.5c_wA\rho$)	kg/m	Scalar
Vehicle Mass (including rotational masses)	kg	Scalar
Rolling Resistance Coefficient		Scalar
Dynamic Rolling Radius of Wheels	m	Scalar
Battery Resistance (Charge & Discharge)	Ohm	Scalar
Open Circuit Voltage of Battery depending on Battery Energy	V	Lookup Table ($\mathbb{R}^1$)
Power Limits of Battery	W	Scalar
Maximal Battery Energy	J	Scalar
Gear Ratio for every Gear		Lookup Table ($\mathbb{R}^1$)
Gear Efficiency		Scalar
Fuel Density	kg/l	
Fuel's Lower Heating Value	J/kg	
Fuel Power depending on ICE speed and ICE torque	W	Lookup Table ($\mathbb{R}^2$)
ICE speed meshgrid (equidistant)	rad/s	Lookup Table ($\mathbb{R}^2$)
ICE torque meshgrid (equidistant)	Nm	Lookup Table ($\mathbb{R}^2$)
Boundaries of ICE torque depending on ICE speed (in rad/s)	Nm	Lookup Table ($\mathbb{R}^1$)
Electric Power of EM depending on EM speed and EM torque	W	Lookup Table ($\mathbb{R}^2$)
EM speed meshgrid (equidistant)	rad/s	Lookup Table ($\mathbb{R}^2$)
EM torque meshgrid (equidistant)	Nm	Lookup Table ($\mathbb{R}^2$)
Boundaries of EM torque depending on EM speed (in rad/s)	Nm	Lookup Table ($\mathbb{R}^1$)
Torque of EM depending on EM speed and EM power	Nm	Lookup Table ($\mathbb{R}^2$)
EM power meshgrid (equidistant)	W	Lookup Table ($\mathbb{R}^2$)
Boundaries of EM power	W	Lookup Table ($\mathbb{R}^1$)

Table 1: List of given input vehicle parameters for the P2-HEV Ford Focus

The forces at the wheel can be described though use of Newton's Second Law (5),

$$m\frac{dv}{dt}+c_\alpha+c_\beta+c_a v^2+F_B=(F_{WHEEL})\gamma(g) \quad (5)$$

where m is the vehicle mass, dv/dt is vehicular acceleration, $c_\alpha = m*g*\sin(\Theta_{gradient})$ is the vehicle's gradient force, $c_\beta = \mu_{drag}*m*g*\cos(\Theta_{gradient})$ is the vehicle's drag force, $\Theta_{gradient}$ is the speed trajectory's slope profile (defined here flat). F_B are braking forces, and F_{WHEEL} is the amount of force the wheel must deliver.

The wheel torque demand $T_{WHEEL} = F_{WHEEL}*r$ for the NEDC is found and subsequently the crankshaft torque T_{CRS} (6a) and power P_{CRS} (6b) are derived. The powertrain split satisfying P_{CRS} between the internal combustion engine's torque P_{ICE} and electric motor's torque P_{EM} is what is optimized for this HEV's EMS during DP calculations.

$$T_{CRS}(g)=\begin{cases}\frac{T_{WHEEL}}{\gamma(g)}*\eta_g \, for \, T_{CRS}(T_{WHEEL}<0)\\ \frac{T_{WHEEL}}{\gamma(g)}/\eta_g \, for \, T_{CRS}(T_{WHEEL}\geq 0)\end{cases} \quad (6a)$$

$$P_{CRS}(g)=T_{CRS}(g)*\omega_{CRS}(g) \quad (6b)$$

Battery Model

Describing the effects of E_B's state dynamics (7) on the crankshaft power demand requires a battery model illustrating the intermediate processes (Figure 3).

$$E'_B=f_t(E_B)=P_B*\Delta t \quad (7)$$

Rearranging (7), using a select possible present and predecessor E_B value yields a battery power $P_B(t)$ for the given time index (8).

$$P_B(E_B)=\frac{E_B(t)-E_B(t-1)}{\Delta t} \quad (8)$$

In this model, dissipation conversion losses $P_{EM, loss}$, input auxiliary power P_{AUX}, and internal battery losses $P_{B,loss}$ are taken into consideration when translating a change in the state E_B into P_{EM} (9).

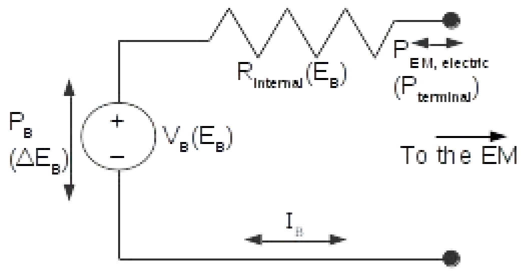


Figure 3: Battery model circuit.

$$P_{EM}(E_B)-P_{EM,loss}=P_{EM,electric}(E_B)=P_B(E_B)-P_{B,loss}-P_{AUX} \quad (9)$$

Electric battery power losses can be easily derived from Ohm's Law. The amount of electric power transferred to the EM is simply the difference between the battery's terminal power (10) and the battery's internal losses (11)..

$$P_B=V(E_B)I \quad (10)$$

$$P_{B,loss}=I^2R(E_B) \quad (11)$$

If a certain E_B level is being evaluated, as is the case when looping through various E_B values in DP, the battery current I can be easily calculated from (10), allowing for a straightforward $P_{B,loss}$ calculation in (11). Otherwise, if a sample $P_{EM,electric}$ is given instead (for example, during preprocessing when setting bounds on reachable E_B states), equation (9) must be rearranged so that the battery current

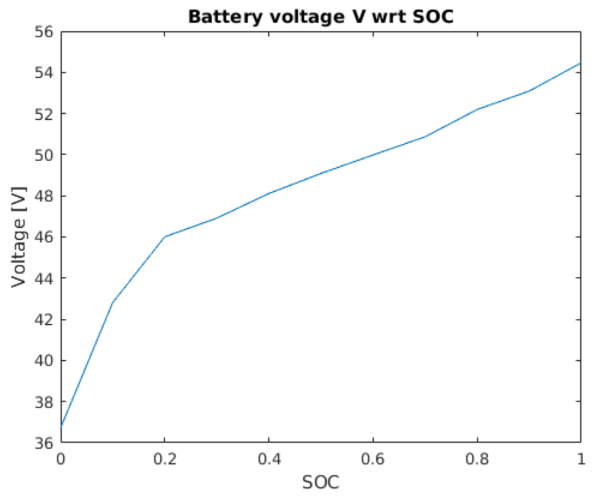


Figure 4: Model battery voltage with respect to its SOC(E_B) level.

I can be calculated quadratically and substituted back into (11), as shown in (12).

$$P_{B,loss} = \frac{V(E_B) - \sqrt{(V^2(E_B) - 4RP_{EM,electric})}}{4R} \qquad (12)$$

The battery's open circuit voltage *V* is nonlinearly dependent on E_B. *V* is derived from a model input lookup table (Figure 4). The internal resistance *R* will have a different sign (and possibly a different value) if either the input P_B or $P_{EM,\ electric}$ (depending on how *I* was calculated, during DP or preprocessing, respectively) is charging or discharging the battery (13).

$$R = \begin{cases} R_{CHARGE}\ for(P_B > 0 \vee P_{EM,electric} > 0) \\ R_{DISCHARGE}\ for(P_B \leq 0 \vee P_{EM,electric} \leq 0) \end{cases} \qquad (13)$$

State Variable Bounds

Bounds must also be placed on the state variables due to physical limitations. Because the entire speed demand profile is known ahead of time, various bounds for E_B can be calculated beforehand. Two bounds for E_B's state dynamics ($P_{B,MIN}$ and $P_{B,MAX}$) are preprocessed: power limitations based on the HEV model's EM speed limits $P_{EM,\ MIN}(\omega_{CRS}, t)$ and $P_{EM,\ MAX}(\omega_{CRS}, t)$, and power limitations based on the model battery's energy charge/discharge limits P_{BAT}. Gear changes u_g are also bounded so that the gear can only transition to adjacent states from one time interval to the next.

This preprocessing drastically reduced computational time while running DP, as various physically unreachable states were immediately ignored during calculations. In this DP algorithm, calculation time dropped drastically after implementing preprocessed bounds from 24 hours to just over 80 minutes. Defining these bounds beforehand for all possible crankshaft speeds and torques through time removes the rote calculation otherwise required in DP when bounds would have to be calculated for each time interval and nested previous and current state permutation loops.

Objective Function

The objective of this EMS is to find the path burning the least amount of fuel for the NEDC trajectory. Therefore, the objective function (14) is the integral of the ICE's power along the path with respect to time, along with any penalties defined for switching gear (β_g) or engine (β_e) states.

$$minimize\ E_{FUEL} = \int_{t_i}^{t_f} \left(\left(P_{ICE}(\boldsymbol{x}, \boldsymbol{u}) + P_{ICE,loss} \right) + |u_g| \beta_g + |u_e| \beta_e \right) dt = \int_{t_i}^{t_f} \Lambda(\boldsymbol{x}, \boldsymbol{u}) dt \quad (14)$$

Because this function must analyze both the change in E_{FUEL} as well as the cumulative sum of all previously selected E_{FUEL} values at every time interval *t*, at each change in E_{FUEL} for the current time index *t,* all previous reachable states ***x(t-1)*** (with their respective control signals ***u(t-1)*** and state dynamic *f(E_B(t-1))*) must be calculated for every given state ***x(t)***. Thus, in order to minimize the objective, the recursive feedback effects of selected predecessor states ***x*** for every optimally calculated E_{FUEL} must be taken into consideration; otherwise, the algorithm is greedy and will not ensure a globally optimal result (Bertsekas 2007).

Dynamic Programming Implementation

Now that the state variables, battery dynamics, gear and engine control signals, vehicle and battery models, and an objective cost function to minimize for have been defined, dynamic programming can be used to run through each realistic state variable permutation through the use of nested for-loops. DP is based on Bellman's Principle of Optimality, where, for a given problem which can be broken up into smaller subproblems (a characteristic the problem must have for DP to work), the globally optimum path in the problem must be by definition comprised of combined smaller optimal paths along its intermediate steps, as long as the cost function is additive over time (Bellman 1957).

Therefore, given initial and final state conditions, DP can be run forwards or backwards in time and calculate the optimal trajectory either way for a given set of initial or ending state variable **x** conditions, respectively (15e),. In this model, DP runs forward with intermediate steps *Δt* spaced one second apart in time (15a). The number of possible changes in the continuous state variables E_B for every time index *t* is limited by the discretized state dynamic bounds (15b) defined during preprocessing, The discrete states *g* and *e* are also subject to the control signals ***u*** triggering at every decision moment *t* (15c, 15d), where ***u*** is also limited due to physical restrictions (ie, gear can change only to adjacent gears) (15g 15h).

$$minimize\ \Lambda_{t,f}(\boldsymbol{x}) + \sum_{k=t_i+1}^{t_f} \Lambda_k(\boldsymbol{x}, \boldsymbol{u}) \Delta t \quad (15a)$$

$$f(E_{B,(t)}) \in [P_{B,(t),MIN}, P_{B,(t),MIN} + \Delta P_B, \ldots P_{B,(t),MAX}] \quad (15b)$$

$$g(t) = g(t-1) + u_g \quad (15c)$$

$$e(t) = u_e \quad (15d)$$

$$\boldsymbol{x}(t_i)=\boldsymbol{x}_i, \boldsymbol{x}(t_f)=\boldsymbol{x}_f \quad (15e)$$

$$\boldsymbol{x}\in[\boldsymbol{x}_{min}, \boldsymbol{x}_{max}] \quad (15f)$$

$$u_g=\{-1,0,1\} \quad (15g)$$

$$u_e=\{0,1\} \quad (15h)$$

States satisfying the criteria set in (15b), (15g), (15h) are looped through the DP algorithm, and each respective predecessor state variable's output fuel use E_{FUEL} is calculated and stored in a temporary three-dimensional tensor (one dimension per state variable). The optimal **x**(t) which most minimizes *Λ(t)* is then added into an ongoing cumulative cost tensor in a process called memoization. Memoization, or saving the optimal solutions of subproblems, is a part of what makes DP so efficient, as it eliminates the need to recursively calculate the values of all previous subproblems to reach the current problem by instead loading the additive cumulative sum of said previous subproblem costs into a tensor. (Figure 5). Here, the "curse of dimensionality" DP suffers from can be easily demonstrated. The original DP calculation, consisting of three state variables (each with 1731, 6, and 2 possible state values for E_B, g, and e, respectively), completes calculations at just over 80 minutes. Because of the introduction of predecessor state restrictions during a preprocessing phase, this speed is a vast

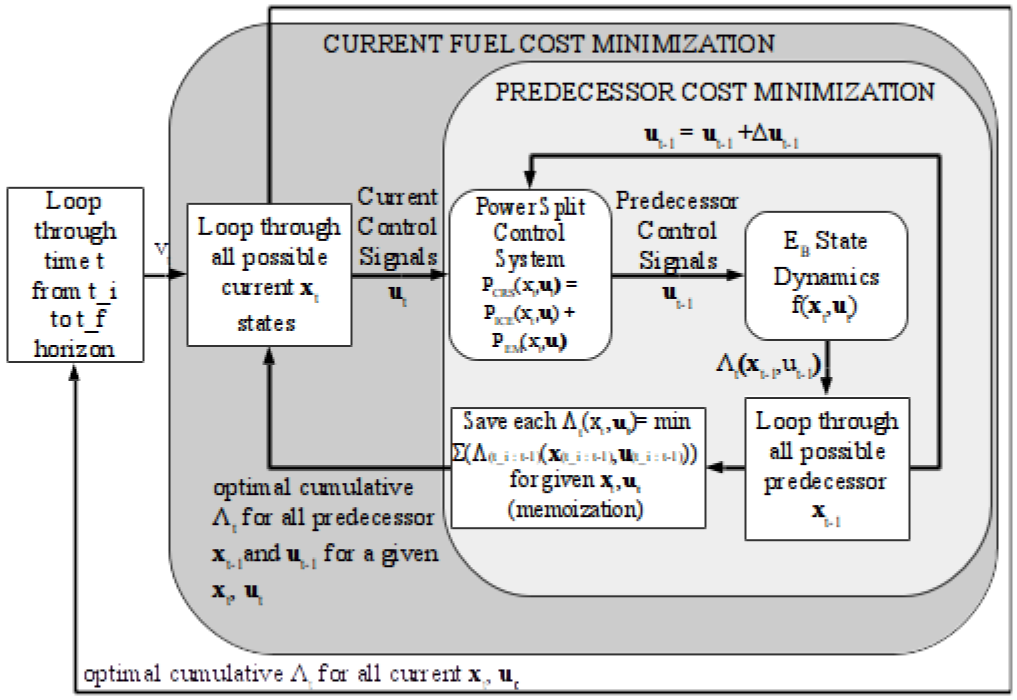

CURRENT FUEL COST MINIMIZATION
PREDECESSOR COST MINIMIZATION
$\mathbf{u}_{t-1} = \mathbf{u}_{t-1} + \Delta\mathbf{u}_{t-1}$
Loop through time t from t_i to t_f horizon
V
Loop through all possible current $\mathbf{x}_t$ states
Current Control Signals $\mathbf{u}_t$
Power Split Control System $P_{CHS}(\mathbf{x},\mathbf{u}) = P_{ICE}(\mathbf{x},\mathbf{u}) + P_{EM}(\mathbf{x},\mathbf{u})$
Predecessor Control Signals $\mathbf{u}_{t-1}$
E_B State Dynamics $f(\mathbf{x}_t,\mathbf{u}_t)$
$\Lambda_t(\mathbf{x}_{t-1},\mathbf{u}_{t-1})$
Loop through all possible predecessor $\mathbf{x}_{t-1}$
Save each $\Lambda_t(\mathbf{x}_t,\mathbf{u}_t)$= min $\Sigma(\Lambda_{(t\_i:t-1)}(\mathbf{x}_{(t\_i:t-1)},\mathbf{u}_{(t\_i:t-1)}))$ for given $\mathbf{x}_t,\mathbf{u}_t$ (memoization)
optimal cumulative Λ_t for all predecessor $\mathbf{x}_{t-1}$ and $\mathbf{u}_{t-1}$ for a given $\mathbf{x}_t$, $\mathbf{u}_t$
optimal cumulative Λ_t for all current $\mathbf{x}_t$, $\mathbf{u}_t$

Figure 5: Process flow for selecting and populating tensors filled with optimal fuel costs Λ and their corresponding optimal states and controls through memoization. Note that minimizing Λ relies on recursive feedback when analyzing all possible state **x** permutations due to cumulative costs; without memoization, the summation of all Λ_t would require a recalculation of the sum of all $\Lambda_{t\_i : t-1}$, requiring orders of magnitude more calculation time redoing previously calculated operations.

improvement to unregulated dynamic programming, which would have taken over 24 hours. Nevertheless, by removing just one state – an alternative DP algorithm was written which would take in optimal gear states as in input in lieu of calculating them – computation time dropped to about eleven minutes.

The effects of the three state variables x on the target E_{FUEL} and its DP implementation can be graphically summarized in Figure 2. The optimal control signal u_g, analyzing possible gear states g_t for every time interval t, will define $\omega_{CRS,(t)}(g_t)$ and $T_{CRS(t)}(g_t)$ values for the HEV. Working alongside selected crankshaft parameters, the control signal u_e determines whether the ICE should engage to contribute to $T_{CRS(t)}(g_t)$. Should e_t be 0 and therefore the ICE be off, $E_{B(t)}$ must change enough in order to meet or exceed $P_{CRS(t)}(g_t,,u_g)$, since $P_{ICE(t)}(\mathbf{x}, \boldsymbol{u})$ will be nonexistent (3a). Note that if $\omega_{CRS,(t)}(g_t)$ is not rotating fast enough to meet the ICE's idling speed $\omega_{ICE,MIN}$ (set at 89 rad/s, or 850 RPM, for this model), u_e will automatically decide an ICE state e_t value of 0 and the engine will fail to engage, leaving the EM to supply the small $P_{CRS\,(t)}(g_t)$. Otherwise, if e_t=1, discretized changes in $E_{B(t)}$ based on its state dynamics (7) and bounded by (15b) are processed by the battery model and consequently define possible $P_{EM\,(t)}(E_B)$ values that the EM can contribute to $P_{CRS\,(t)}(g_t)$; the ICE makes up for any left over required power with $P_{ICE(t)}$ (3a).

Running this process for every time time interval t from $[t_{i+1} : t_f]$ populates a map of nodes and edges, where each node represents a state variable permutation at a specified point in time and where weights connecting nodes across adjacent time intervals represent optimal E_{FUEL} values for each subproblem (Figure 5). Finally, after generating the map of all possible **x** nodes and E_{FUEL} weights, running through this map selecting the saved optimal **x** values (and therefore the optimal $f(E_B)$ and $\boldsymbol{u}$) through time for specified end conditions (15e) yields the trajectory's optimal minimal cumulative fuel use E_F through the generated map, effectively applying Bellman's Principle of Optimality.

Results

Figure 6 demonstrates the optimal state variables **x**, crankshaft parameters $\omega_{CRS}(g)$ and $T_{CRS}(g)$, and the target minimization variable's E_{FUEL} equivalent in volume (liters) required to satisfy the NEDC velocity profile **v** (Figure 1). Optimal state variable values are found across all optimal paths from a starting state of charge (SOC) E_B value of 60% to all possible E_B SOC values. Line trajectories that are more blue signal a lower terminating SOC, whereas more red lines indicating higher SOC end values.

As the crankshaft speed and torque are functions of the model's optimal gear state selection g (see (4) and (6a)), the crankshaft parameters $\omega_{CRS}(g)$ and $T_{CRS}(g)$ can be seen to change in accordance with different selected optimal g states; the parameters can also be seen to increase and decrease in

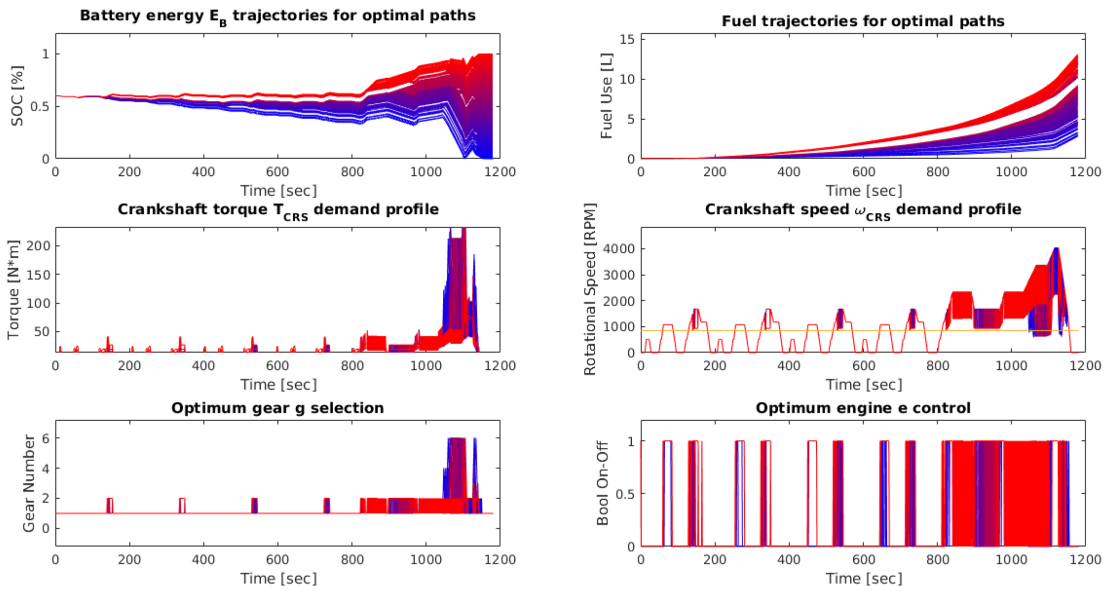


Figure 6: Results from dynamic programming implementation for optimal state variables **x**, crankshaft parameters $\omega_{CRS}(g)$ and $T_{CRS}(g)$, and target minimization variable E_{FUEL} (here converted to volume units for improved legibility) for all possible ending state of charge (SOC) trajectories – from completely drained (0%, blue line paths) to completely charged (100%, red line paths). Since no penalties were assigned to β_g and β, the model switches between discrete states far more often than can be realistically expected while finding globally optimal fuel use.

correspondence with the NEDC's velocity intervals. The HEV model predicts that, for the majority of the velocity profile, the vehicle can run just off of it's electric motor. Aided by the fact that the majority of the beginning of the speed profile is slow enough to not require the ICE to engage, the EM powers the vehicle for most of the first four slow velocity periods, regardless of terminal E_B SOC value. Periods do exist during the peaks of the low speed areas of the profile where the engine does contribute to the power demand, in particular for higher SOC targets. Once the vehicle reaches the high velocity period however, where $\omega_{CRS}(g)$ and $T_{CRS}(g)$ demands rise to a much higher level, the ICE must begin supplying power P_{ICE}, as demonstrated by a very noticeable "on" e signal at around 800 seconds. The ratio, or power split, for which P_{ICE} and P_{EM} contribute to the the overall P_{CRS} depends on the path's final E_B SOC value.

It can also be seen that trajectories with a lower SOC final E_B horizon value are noticeably more liberal with EM use than those with higher final SOC values. Since the battery does not need to be charged as much towards the end of the trajectory to realize a lower terminal SOC, the battery-EM can afford to deliver more power to the drivetrain as a result. This trend corresponds well with the corresponding target variable E_{FUEL}, where lower SOC trajectories require less fuel to make up for the

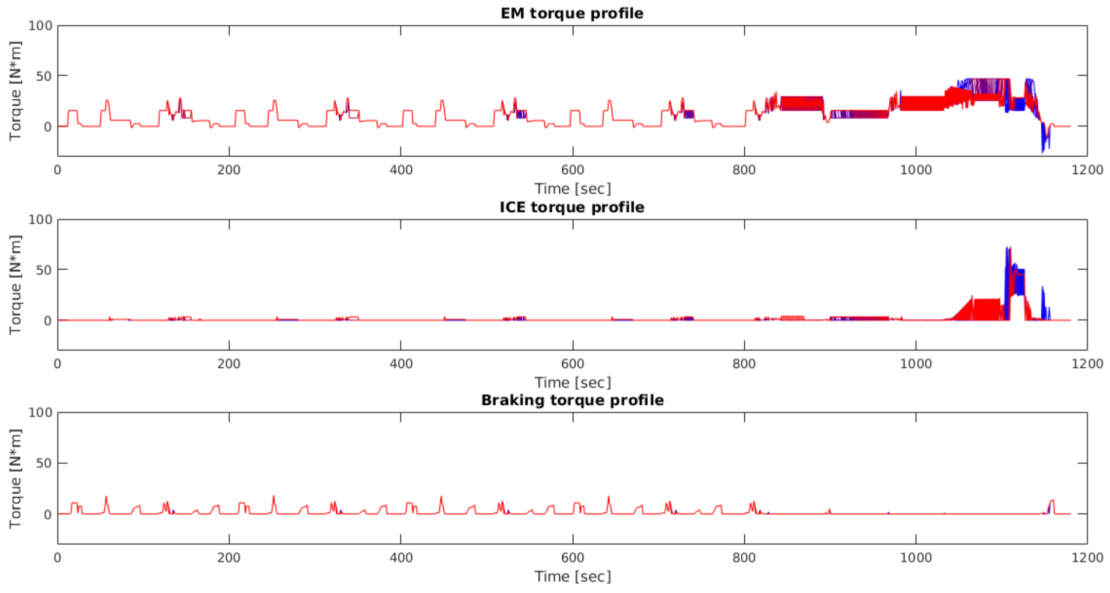


Figure 7: Results from dynamic programming implementation for optimal torque split between the EM and ICE

crankshaft power demand due to increased EM activity. The final volume of used fuel ranges from 3 to 13 liters, a promising sign when considering that my supervisor estimated a fuel use of about 10 L for optimal fuel use without any change in start and end SOC for the trajectory.

One peculiarity about the results is the HEV's tendency to stay in first gear for the majority of the calm driving periods with only an occasional spike into second gear at their peaks, although this is most likely due to the low driving speeds. Also, the torque values, such as for T_{CRS}, seem to be far too low when considering torque demands from a vehicle. This may be due to a dimensions error in the model's input torque boundaries and lookup tables (Table 1), as input torque unites scaled 10^3 higher (from N*m to kN*m) would make far more sense (a similar situation occurred with the fuel's lower heating value being in units MJ/kg instead of J/kg). This suspicion has not yet been confirmed with my supervisor and is therefore simply reported here.

Figure 7 plots the amount of torque delivered by the EM and ICE along the profile. Braking torques can be observed during periods when e=0, ie the ICE is off. Due to the discretization of the continuous variable E_B (7), the EM may provide too high a P_{EM} value and overshoot the required P_{CRS} value; in order for (3b) to balance, a brake P_{BRAKE} is included. P_{BRAKE} is not required when e=1, as P_{ICE} is not subject to discretization and can satisfy any power difference between P_{CRS} and P_{EM}. Also, the sum of torques T_{ICE} and T_{EM} also do not seem to add up to T_{CRS} during the high velocity period. Two possibilities may explain this. A dimensional units error in torque (Table 1) may partly explain this observation. Also, since P_{CRS} is the product of T_{CRS} and ω_{CRS}, when deriving T_{CRS} from P_{CRS}, the value is linearly related to the crankshaft's rotational velocity. In comparison, EM power P_{EM} is nonlinearly related to crankshaft rotational velocity. Because of this, although powers may balance (7), the torques may not. This issue must be further analyzed, as it may have a significant effect on optimal results.

Conclusion

Two main projects were undertaken at the Department of Vehicle Mechatronics in Germany this summer. The first involved attempting to speed up and port a previously developed DP-PMP method with near globally optimal results in an effort to prepare the algorithm for online testing and implementation. Although the method was sped up by about 33%, further changes to the code are still required before it is fast enough for online use. Working with the model firsthand set a solid foundation for developing a globally optimal dynamic programming algorithm which optimizes a hybrid electric vehicle's energy management system by recursively saving and checking all possible controls responsible for changes in the method's state variables (along with their associated transition weights) across the NEDC velocity profile. The process finds the optimal set of controls controlling battery energy, gear, and engine on-off states through discrete one second time steps with the end goal of minimizing fuel use. Further analysis and testing may be required before the method can guarantee optimal results, but the grand majority of the foundation was successfully developed. Working with dynamic programming, as well as learning about other optimization strategies, exposed me to the vast field of control theory, in particular the modern portion. Combined with the use of programming and work in a subject area heavily related to electrical engineering, this internship has given me valuable and practical insight, knowledge, and experience in my field of study.

Acknowledgements

I would like to especially thank Dipl.-Ing. Stephan Uebel, Dipl.-Ing. Conny Tempelhahn, Dipl.-Ing. Robert Fechert,, and Prof. Dr.-Ing. Bernard Bäker for the opportunity to work on such a fascinating subject this summer. Their mentoring, supervision, and advice was instrumental in understanding and developing optimal control theory this summer. I also thank all those at the Innovationzentrum Fahrzeugmechatronik, whose openness and inclusiveness helped me integrate into the mechatronics team professionally, academically, and culturally.

References

R. Bellman, *The Theory of Dynamic Programming* . Princeton: Princeton University Press, 1972.

D. P. Bertsekas, *Dynamic programming and optimal control* , 3rd ed. Belmont, Mass: Athena Scientific, 2005-2007.

de Jager, Bram, Thijs van Keulen, and John Kessels. *Optimal Control of Hybrid Vehicles*. Advances in Industrial Control. 1 ed. London: Springer-Verlag London, 2013. Print.

Elbert, Philipp. "Noncausal and Causal Optimization Strategies for Hybrid Electric Vehicles." Dissertation. ETH Zurich, 2014. Print.

Initiative, Global Fuel Economy. "Test Cycles." United Nations. Web.

Murgovski, N, LM Johannesson, and J Sjoberg. "Engine on/Off Control for Dimensioning Hybrid Electric Powertrains Via Convex Optimization." *Ieee Transactions on Vehicular Technology* 62.7 (2013): 2949-62. Print.

Tempelhahn, Conny, Stephan Uebel, and Bäker Bernard. "State of the Art in Optimal Control of Series Hybrid Electric Vehicles Taking Account
of Driveability." *EEHE - Electric & Electronic Systems in Hybrid and Electric Vehicles and Electrical Energy Management*. 2016. Print.

Uebel, Stephan, et al. "Optimal Velocity and Power Split Control of Hybrid Electric Vehicles." currently under review, 2016. Print.