POLITECNICO MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE E DELL'INFORMAZIONE

FUEL CONSUMPTION MINIMIZATION FOR A FULL HYBRID AUTONOMOUS VEHICLE - MPC VS DYNAMIC PROGRAMMING PERFORMANCES

Project Report - Course of Numerical Optimization for Control

Automation and Control Engineering

Francesco Berruti, 10777631;
Andrea Bonafini, 10753023;
Federico Bordini, 10766923;
Lorenzo Marzi, 11015873;

Contents

1. Introduction

Nowadays, the transition towards the vehicles' electrification is becoming a worldwide spread phenomenon. The main objective is to reduce the CO_2 emissions; such goal, in terms of automotive, can be reached by reducing the number of classical gasoline vehicles circulating, while increasing the presence of the electrified ones.
However, the available possibilities among which it is possible to choose are many: from a very general perspective, without considering any particular architecture, it is possible to find Mild Hybrid, Full Hybrid, Plug-In Hybrid and Full Electric vehicles. In the wide range of Hybrid vehicles, the fundamental aspect to point out is that the torque needed to move the vehicle is provided by both a classical internal combustion engine and an electric motor (named ICE and EM, respectively). This torque split allows an operating strategy that minimizes consumption by leveraging the strengths of its power sources:

- At high speeds, the vehicle demands significant power, which is better supplied by the ICE due to its ability to operate efficiently, and its use further preserves the battery.
- At low speeds and in stop-and-go traffic, the EM is highly efficient. Moreover, this approach avoids using the ICE, which may suffer from incomplete combustion, leading to greater energy losses.

In this report the focus is set on a full hybrid electric vehicle, which implies it is not possible to recharge it directly from any external energy source (like in the case of Plug-In Hybrid vehicles), so the battery has to be recharged by the ICE or via regenerative braking, whenever possible. This implies that the torque split has a dominant role in determining the fuel consumption of the vehicle, which has to be minimized.
In this study, real data from the 2013 Ford Fiesta hybrid are used, including ICE and EM efficiency maps.
Since the literature has already demonstrated that hybrid systems exhibit lower fuel consumption and reduced emissions compared to internal combustion engine (ICE)-only vehicles under nearly all driving conditions, the objective of this study is to develop a method that can be implemented in real-time (i.e., during driving) to efficiently determine the optimal torque split factor.
Two solutions are presented: Dynamic Programming (DP) and a Model Predictive Control (MPC)-based approach. DP provides a benchmark as it delivers optimal solutions offline, while MPC enables online causal solutions. To fairly compare these methods, it is assumed that the vehicle is autonomous, enabling trajectory planning. The presence of a planning layer, particularly a global one, enables the utilization of predictive capabilities, which in turn allows the implementation of the MPC strategy.
The two solutions are generated both over standard driving cycles (e.g. WLTC) and randomized ones, generated by us.

The complete hybrid vehicle model is introduced in Section 3, along with all its components. Next, in Section 4, the general optimization problem to be solved is outlined. In Section 5, it is explained how Dynamic Programming and MPC address the problem, and, in particular, how the problem is adapted for online implementation. Finally, the solver setup is presented in Section 6, along with the results of the MATLAB simulation.

2. Notation

Notation	Meaning	Value	Unit
Vehicle Model Parameters			
m_v	Mass of the vehicle	$1.2 \cdot 10^3$	Kg
r	Wheel radius	0.28	m
γ	Coasting down parameter 1	$3.199 \cdot 10^{-4}$	$m \cdot s^{-2}$
β	Coasting down parameter 2	$2.11 \cdot 10^{-3}$	s^{-1}
α	Coasting down parameter 3	$7.76 \cdot 10^{-2}$	m^{-1}
g	Gravitational acceleration	9.81	$m \cdot s^{-2}$
v	Vehicle speed		$m \cdot s^{-1}$
$\dot{v}$	Vehicle acceleration		$m \cdot s^{-2}$
ω_v	Angular velocity of the Vehicle's Wheels		$rad\ s^{-1}$
T_v	Torque applied to the Vehicle's Wheels		Nm
τ_0	Gear ratio from the axis of the gearbox to the wheels		$-$
C_0	Static Friction Coefficient		$-$
C_1	Dynamic Friction Coefficient		$s\ m^{-1}$
θ	Road angle		rad
ρ_a	Density of Air		$kg\ m^{-3}$
A_f	Front surface of the vehicle		m^2
C_d	Equivalent drag coefficient on the vehicle		$-$

Table 1: Vehicle Model Parameters

Notation	Meaning	Unit
PowerTrain Parameters - Electric Motor Side		
ω_{EM}	Angular Velocity of the Electric Motor	$rad\ s^{-1}$
T_{EM}	Torque of the Electric Motor	Nm
τ_{EM}	Gear Ratio of the Electric Motor	$-$
η_{EM}	Efficiency of the Electric Motor	$-$
P_{EM}	Power required/provided by the Electric Motor	W

Table 2: PowerTrain Parameters - Electric Motor Side

Notation	Meaning	Unit
PowerTrain Parameters - Internal Combustion Engine Side		
$GLHV$	Gasoline Lower Heating Value	$J\ kg^{-1}$
$\dot{m}_f$	Fuel Mass Flow	$kg\ s^-1$
ω_{ICE}	Angular Velocity of the Internal Combustion Engine	$rad\ s^{-1}$
T_{ICE}	Torque of the Internal Combustion Engine	Nm
τ_{ICE}	Gear Ratio of the Internal Combustion Engine	$-$
τ'_g	Gear ratio from the driver to the gearbox (depends on actual gear)	$-$
η_{ICE}	Efficiency of the Internal Combustion Engine	$-$
P_{ICE}	Power provided by the Internal Combustion Engine	W

Table 3: PowerTrain Parameters - Internal Combustion Engine Side

Notation	Meaning	Value	Unit
Battery Parameters			
P_b	Power required/provided by the Battery		W
ξ	State of Charge of the Battery		$-$
I_b	Current of the Battery		A
I_n	Maximum charging/discharging current of the Battery		A
Q_{nom}	Nominal capacity of the Battery	45	Ah
V_{oc}	Open loop voltage of the Battery	100	V
R_o	Internal resistance of the Battery	$3.9 \cdot 10^{-2}$	Ω

Table 4: Battery Parameters

Acronym	Meaning
DP	Dynamic Programming
MPC	Model Predictive Control
ICE	Internal Combustion Engine
EM	Electrical Motor
EPL	Electrical Power Link
MPL	Mechanical Power Link
DOH	Degree Of Hybridization
FHOCP	Finite Horizon Optimal Control Problems
SoC	State of Charge
NLP	Non Linear Problem
SQP	Sequential Quadratic Programming

Table 5: Acronym Table

3. Problem Modeling

The mathematical model for the parallel hybrid electric vehicle, whose logic is illustrated in Figure 1, is adopted from [3] and [4]. The model takes as input the reference velocity provided by the driving cycle and outputs the required fuel mass flow rate and the battery's state of charge. Therefore, the model of the vehicle and of its components aims to highlight the energy flow among the various elements, according to the final goal of this work. To derive the models according to this principles, the involved quantities shown in figure 1 represent the requested ones, e.g. P_b represents the power requested to the battery and not the power the battery provides. This modeling approach is called "Backward modeling". The main difference with respect to the classical "Forward modeling" is that the vehicle speed is imposed, regardless of driver actions, i.e. v coincides with v_{ref}, that is the reference speed given by the specific driving cycle. Thus, overall, the causality is being "reversed" by fixing the output and the state (effect) to compute the force (cause). The equations obtained in the two approaches are identical, however the total power is different, because in the backward approach the speed corresponds exactly to the reference one while in the forward approach the speed is influenced by the driver model.
In the following, the models of the various components along with their respective assumptions are briefly outlined and, at last, all of the equations used in the calculations will be written altogether. For the notation, please refer to Section 2.

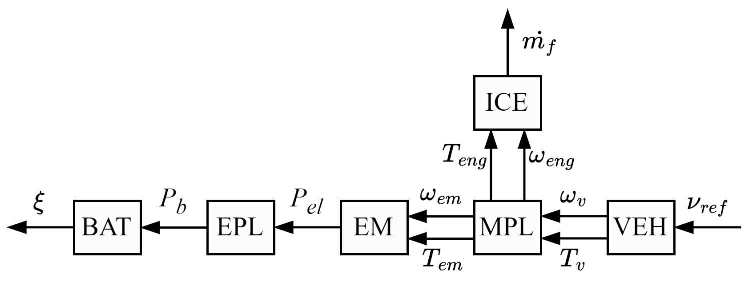
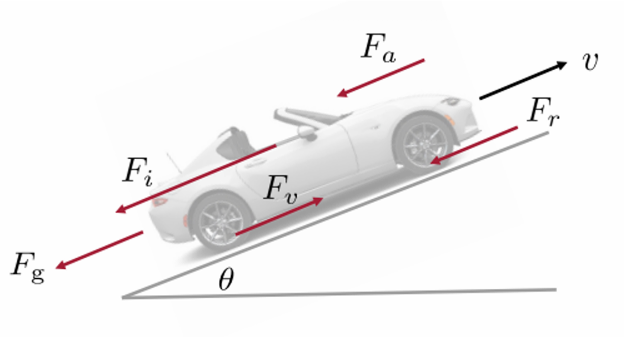


Figure 1: Full vehicle's model

3.1. VEH - Vehicle

It is assumed that the car follows a straight road with negligible irregularities, allowing us to consider only the vehicle longitudinal dynamics. Under the assumption of negligible wheels slip (reasonable, since the aim is to model the energy flow between the components), the equations that describe the longitudinal dynamics are:

- F_i = Inertia Force $\longrightarrow m_v\dot{v}$;
- F_r = Rolling resistance $\longrightarrow (C_0 + C_1 v)m_v g\cos\theta$;
- F_g = Gravitational Force $\longrightarrow m_v g\sin\theta$;
- F_a = Aerodynamics $\longrightarrow \frac{1}{2}\rho_a A_f C_d v^2$;
- F_v = Wheel Force $\longrightarrow \frac{T_v}{r}$.



By applying Newton's law and rearranging the terms, one can derive the previously introduced backward model of the vehicle (see Figure 2), which maps the reference velocity v_{ref} to the required wheel force F_v. Finally, by considering the wheel radius r, one obtains the required tire torque T_v and angular speed ω_v (eq 1), as utilized in the block VEH of Figure 1.

$$\begin{aligned} v &= v_{ref} \\ T_v &= r\cdot(m_v\dot{v} + m_v g\sin\theta + m_v\gamma\cos\theta + m_v\beta v\cos\theta + m_v\alpha v^2) \\ \omega_v &= \frac{v}{r} \end{aligned} \tag{1}$$

Since some parameters are very difficult to be measured directly, especially those related to friction and aerodynamics, they are experimentally estimated. Thus, the parameters α, β, γ are obtained by performing a proper experiment, known as coasting down test. Since it is not of interest for this work, the whole identification procedure is not explained and the parameters α, β, γ are treated as given vehicle's parameters.

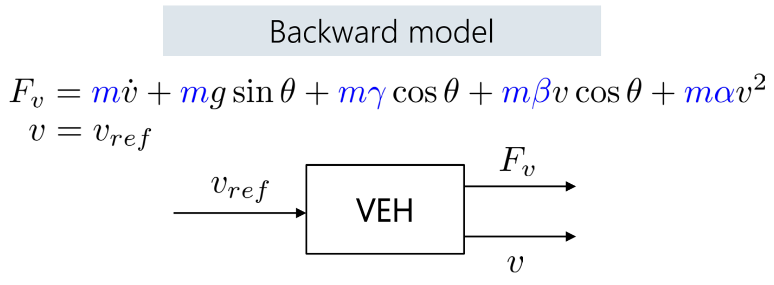

Backward model
$F_v = m\dot{v} + mg\sin\theta + m\gamma\cos\theta + m\beta v\cos\theta + m\alpha v^2$
$v = v_{ref}$
v_{ref}
VEH
F_v
v

Figure 2: Backward model

3.2. MPL - Mechanical Power Link

The MPL is a torque coupler: it relates an incoming power with two outgoing powers. In particular, there are 2 gearboxes, one for each input of the gearbox. The electric motor can be connected, through the gearbox, to the output shaft of the transmission, and the same can be done with the ICE to another gearbox, which is always coaxial with the vehicle shaft. In this way, it is possible to have a combination of electrical and mechanical power. The equations that model this component are:

$$\begin{aligned} &\omega_{EM} = \tau_{EM} \cdot \omega_v \\ &\omega_{ICE} = \tau_{ICE} \cdot \omega_v \\ &P_{EM} + P_{ICE} = P_v \implies T_{EM} \cdot \tau_{EM} + T_{ICE} \cdot \tau_{ICE} = T_v \end{aligned} \tag{2}$$

Note that the value of the gear of the ICE depends on two components: a constant ratio τ_0, derived from the axle to the wheel, and a variable ratio τ_g', derived from the gear selected by the driver, which depends on the speed of the vehicle.

3.3. ICE - Internal Combustion Engine

This component transforms the chemical energy of the fuel into mechanical power. A simple quasi-static model is considered where the efficiency curves, depicted in Figure 3, are obtained through experiments. The equation for the ICE-block is the following:

$$\dot{m}_f = \frac{T_{ICE} \cdot \omega_{ICE}}{GLHV \cdot \eta_{ICE}(\omega_{ICE}, T_{ICE})} \tag{3}$$

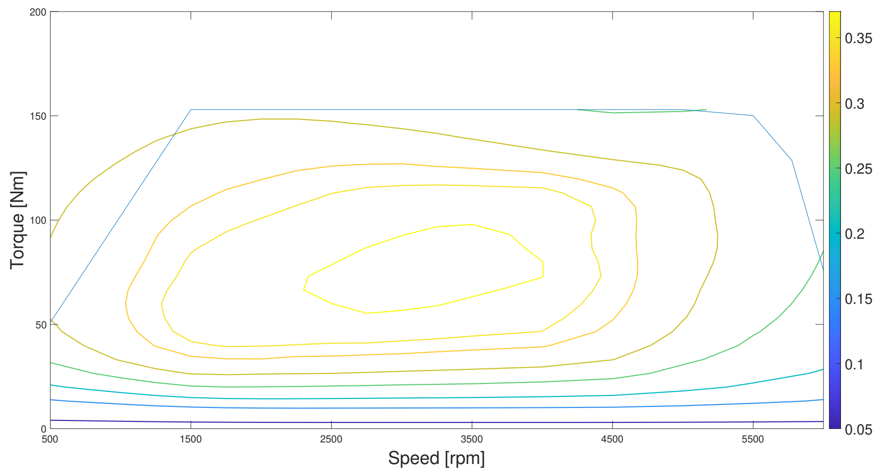


Figure 3: ICE efficiency map. The level curves represent constant efficiency values and each value can be retrieved from the colourmap on the right side. The light-blue thick line represents the maximum torque.

3.4. EM - Electrical Motor and EPL - Electrical Power Link

The electric motor is powered by the battery through the inverter and delivers mechanical power with high efficiency. A simple quasi-static model, combining the EM and EPL, is employed, with efficiency maps (Figure 4) derived from steady-state experiments. The model also accounts for both traction and regeneration scenarios, leading to the following formulation:

$$\begin{aligned} T_{EM}\omega_{EM} &= P_b \cdot \eta_{EM},\ P_{EM} > 0 \text{ (Traction)} \\ T_{EM}\omega_{EM} &= \frac{P_b}{\eta_{EM}},\ P_{EM} < 0 \text{ (Regeneration)} \end{aligned} \tag{4}$$

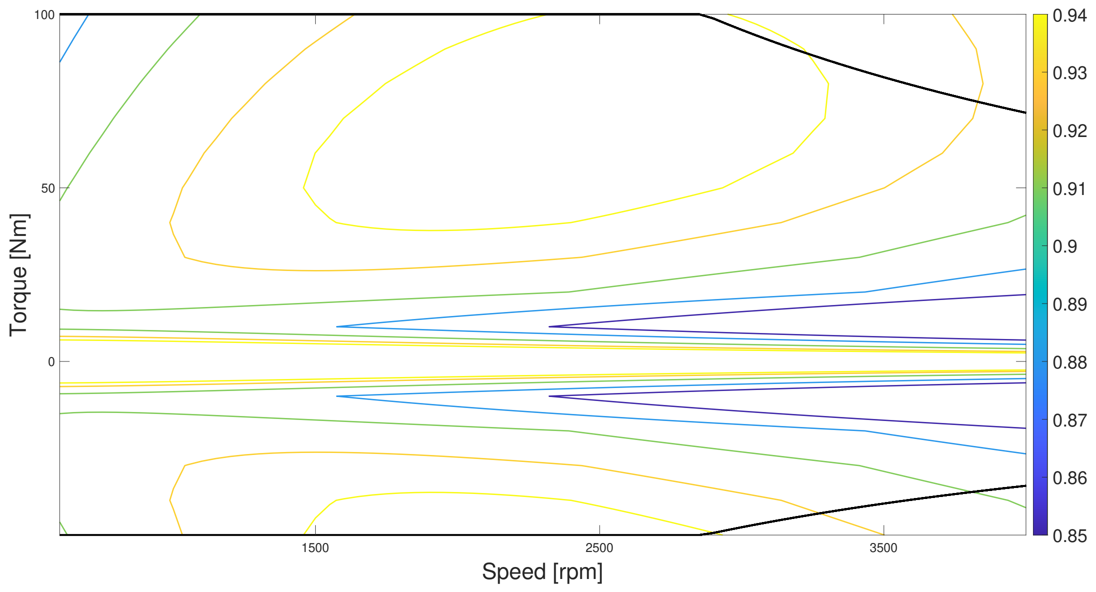


Figure 4: EM efficiency map. The level curves represent constant efficiency values and each value can be retrieved from the colourmap on the right side. The black thick lines are the maximum/minimum torque limits.

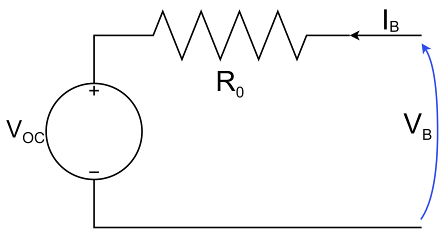
3.5. BAT - Battery

The battery can be represented using a simple static model consisting of an ideal voltage source in series with a resistor, as illustrated in Figure 5a, when the state of charge is maintained within the optimal range, see figure 5b. By using simple electric circuit theory, one can obtain the following equation:

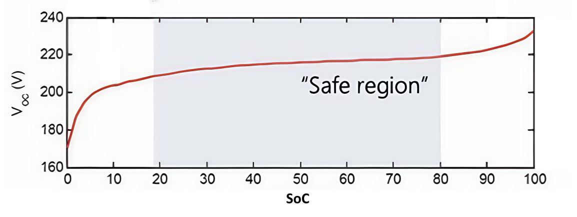
$$I_b = \frac{V_{oc}}{2R_0} - \sqrt{\left(\frac{V_{oc}}{2R_0}\right)^2 - \frac{P_b(t)}{R_0}} \tag{5}$$

To ensure that the SoC stays within ranges, it is necessary to implement a model to track the battery's state of charge. In this study, the Coulomb counting method is used, which leads to the following equation:

$$\dot{\xi}(t) = \frac{1}{Q_{nom}} \cdot \frac{P_b(t)}{V_{oc}} \tag{6}$$



(a) Battery circuital model



(b) Open loop voltage V_{oc} as function of the SoC

3.6. Full model of the parallel hybrid vehicle

By combining all the equations, one obtains the full backward model of the parallel hybrid vehicles used, hereafter reported for clarity.

$$
\begin{aligned}
v &= v_{ref} \\
T_v &= r \cdot (m_v \dot{v} + m_v g \sin\theta + m_v \gamma \cos\theta + m_v \beta v \cos\theta + m_v \alpha v^2) \\
\omega_v &= \frac{v}{r} \\
\omega_{EM} &= \tau_{EM} \cdot \omega_v \\
\omega_{ICE} &= \tau_{ICE} \cdot \omega_v \\
T_v &= T_{EM} \cdot \tau_{EM} + T_{ICE} \cdot \tau_{ICE} \\
\dot{m}_f &= \frac{T_{ICE} \cdot \omega_{ICE}}{GLHV \cdot \eta_{ICE}(\omega_{ICE}, T_{ICE})} \\
T_{EM}\omega_{EM} &= P_b \cdot \eta_{EM},\ P_{EM} > 0 \text{ (Traction)} \\
T_{EM}\omega_{EM} &= \frac{P_b}{\eta_{EM}},\ P_{EM} < 0 \text{ (Regeneration)} \\
I_b &= \frac{V_{oc}}{2R_0} - \sqrt{\left(\frac{V_{oc}}{2R_0}\right)^2 - \frac{P_b}{R_0}} \\
\dot{\xi} &= \frac{1}{Q_{nom}} \cdot \frac{P_b}{V_{oc}}
\end{aligned}
$$

Finally, the model has been discretized through Forward Euler method with sampling time $T_s = 1s$, which, for the application at hand, is a sensible choice.

4. Problem Abstraction

As stated in the Introduction, the goal of this work is to design an optimization problem whose solution is the optimal split of the total torque requested by the vehicle between the EM and the ICE in a full hybrid electric vehicle. The optimization problem at hand can be written as a NLP[2], whose formulation is described in this Section. Then, since the aim is to solve it online through MPC, it can be rewritten as a FHOCP, whose formulation is presented in Section 5.1 for clarity.

4.1. Formulation

A detailed analysis (presented in Section 4.4) reveals that the problem is nonlinear. This nonlinearity arises from the fact that nearly all constraints and the cost function are influenced by motor efficiencies which depend on the optimization variables. Thus, the optimization problem can be written as follows:

$$\begin{aligned} \min_{U} \sum_{i=1}^{T} & c(x(i), u(i)) \\ & \text{subject to} \\ & Ax = b, \\ & h(x(i), u(i)) \geq 0, \quad i = 1, ..., T \end{aligned}$$

where T is the total duration of the driving cycle, $U = [u(1), ..., u(T)] \in \mathbb{R}^T$ is the vector of control variables, x represents the state variable, $c(x(i), u(i))$ represents an economic stage cost, $Ax = b$ represent the linear equality constraints and $h(x, u) \geq 0$ is related to the linear and nonlinear inequality constraints. The cost function term is explained in Section 4.3 and the two expressions related to the equality and inequality constraints are explained in Section 4.4. Finally, please note that the index i starts from 1 due to the simulation environment used (in MATLAB is not possible to start from $i = 0$).

4.2. Optimization Variable

The optimization variable $u(t)$ is defined as: $u(t) = \frac{P_{EM}}{P_{TOT}}$. The values taken by this variable can be interpreted as follows:

- $u(t) = 1$ means that the total power requested by the vehicle is given only by the EM;
- $u(t) \in (0, 1)$ means that the total power requested is given by both the EM and the ICE (known as "torque assist mode");
- $u(t) = 0$ means that the total power requested by the vehicle is given only by the ICE;
- $u(t) < 0$ means that the EM acts as a generator, recharging the battery.

4.3. Cost Function

As anticipated in 4.1, the cost function can be written as:

$$c(x(i), u(i)) = \frac{P_{ICE}(i)}{GLHV} = \frac{T_{ICE} \cdot \omega_{ICE}}{\eta_{ICE} \cdot GLHV} = \dot{m}_f(i) \cdot T_s \tag{7}$$

This quantity describes the mass of fuel that is being provided to the ICE at each time instant. This quantity is non-linear, since the efficiency depends on the optimization variables. Finally, T_s represents the sampling time with which the model is discretized, as explained in 3.6.

4.4. Constraints

The problem features linear equality constraints and both linear and nonlinear inequality constraints:

- $\xi(1) = \xi_0$ enforces the initial condition on the state variable, i.e. it sets the initial SoC to a given value. In this case, $\xi(1) = 55\%$;
- $\xi(T) = \xi_0$ enforces the global constraint on the state variable, i.e. it sets the final SoC to the initial value. This is needed for the charge-sustaining goal. This constraint, however, can be enforced only if the total duration of the driving cycle is known, i.e. T is available a priori. For this reason, two versions of MPC are proposed in section 5.1: the first does not take into account this constraint, while the second is designed to satisfy it (thus under the assumption that the duration of the route is known);
- $\xi \in (\underline{\xi}, \overline{\xi})$, which represent the upper and lower bounds on the state variable. These constraints are important for two main reasons: first of all, it is fundamental to stay in the linear section of the SoC profile (as shown in figure 5b), which in this case means $\xi \in (40\%, 70\%)$; second, it avoids under/over-charging, which causes both an acceleration in the aging of the battery and an increase the risk of thermal runaway incidents, jeopardizing passenger safety [1];
- $T_{ICE} \in (0, \overline{T_{ICE}})$, where $\overline{T_{ICE}}$ represents the upper torque limit that the ICE can provide. Moreover, the non-negativity of the ICE torque is enforced, since the ICE cannot provide negative torques. Finally, the constraint depends on the speed at which the axis of the ICE is rotating, i.e. $\overline{T_{ICE}}(\omega_{ICE})$;
- $T_{EM} \in (\underline{T_{EM}}, \overline{T_{EM}})$, which represent the upper and lower boundaries on the torque provided by the EM. Both boundaries are dictated by the physical limitations of the motor and, as for the ICE, they depend on the speed at which the axis is rotating, i.e. $T_{em} \in (\underline{T_{EM}}(\omega_{EM}), \overline{T_{EM}}(\omega_{EM}))$. Please note that for structural reasons, usually $\|\underline{T_{EM}}\| \leq \overline{T_{EM}}$;
- $P_{EM} \leq \frac{V_{oc}^2}{4R_0}$, which represents an upper bound to the power given by the electric motor. It is needed to limit the maximum current circulating;
- $|I_b| \leq I_n$, whose aim is to guarantee that the current flowing through the battery is less than the nominal one;
- $u \in [-1, 1]$, which represent the upper and lower bounds on the control variable.

Additionally, these constraints have been normalized with respect to their respective maximum values to maintain a consistent order of magnitude when solving the problem.

4.5. Complete formulation

The complete formulation of the problem is thus:

$$\begin{aligned}
\min_{U} \quad & \sum_{i=1}^{T} \dot{m}_f(i) \\
\text{subject to} \quad & \xi(i+1) \;=\; f_z\big(\xi(i),\, u(i)\big), \quad i = 1, \dots, T-1, \\
& \xi(1) \;=\; 0.55, \\
& \xi(T) \;=\; 0.55, \\
& \underline{\xi} \;\leq\; \xi(i) \;\leq\; \overline{\xi}, \quad i = 1, \dots, T, \\
& 0 \;\leq\; T_{\text{ICE}}(i) \;\leq\; \overline{T_{\text{ICE}}}, \quad i = 1, \dots, T, \\
& \underline{T_{\text{EM}}} \;\leq\; T_{\text{EM}}(i) \;\leq\; \overline{T_{\text{EM}}}, \quad i = 1, \dots, T, \\
& P_{\text{EM}}(i) \;\leq\; \frac{V_{\text{oc}}^2}{4R_0}, \quad i = 1, \dots, T, \\
& -I_n \;\leq\; I_b(i) \;\leq\; I_n, \quad i = 1, \dots, T, \\
& -1 \;\leq\; u(i) \;\leq\; 1, \quad i = 1, \dots, T.
\end{aligned} \tag{8}$$

The constraint on the system dynamics is written for completeness, but it is implicitly enforced in the simulation of the system.

5. Dynamic Programming and Model Predictive Control setup

This section presents the formulation of the MPC problem, based on 4.5. It is highlighted the necessary modifications to the cost function to address the challenge of enforcing the final SoC constraint, which would otherwise excessively restrict the solution space due to the prediction horizon, that is much shorter than the total cycle duration. Finally, we provide a brief overview of the Dynamic Programming approach, which is used to solve the problem in a one-shot, offline manner, serving as a benchmark for comparison.

5.1. Model Predictive Control - MPC

In MPC, the strategy followed to find the solution is different from DP. In particular, the problem is not addressed all at once, but at each time instant the following optimization problem is solved[1]:

$$
\begin{aligned}
\min_{U} \quad & \sum_{i=1}^{N} l\big(x(i \mid t),\, u(i \mid t)\big) \\
\text{subject to} \quad & \xi(i+1 \mid t) \;=\; f_z\big(\xi(i \mid t),\, u(i \mid t)\big), \quad i = 1,\dots,N-1, \\
& \xi(1 \mid t) \;=\; \xi_{\text{meas}}, \\
& \underline{\xi} \;\le\; \xi(i \mid t) \;\le\; \overline{\xi}, \quad i = 1,\dots,N, \\
& 0 \;\le\; T_{\text{ICE}}(i \mid t) \;\le\; \overline{T_{\text{ICE}}}, \quad i = 1,\dots,N, \\
& \underline{T_{\text{EM}}} \;\le\; T_{\text{EM}}(i \mid t) \;\le\; \overline{T_{\text{EM}}}, \quad i = 1,\dots,N, \\
& P_{\text{EM}}(i \mid t) \;\le\; \frac{V_{\text{oc}}^2}{4\,R_0}, \quad i = 1,\dots,N, \\
& -I_n \;\le\; I_b(i \mid t) \;\le\; I_n, \quad i = 1,\dots,N, \\
& -1 \;\le\; u(i \mid t) \;\le\; 1, \quad i = 1,\dots,N.
\end{aligned}
\tag{9}
$$

where N is the prediction horizon, t is the current time instant, U is the vector of control variables ($U = [u(1|t), ...u(N|t)] \in \mathbb{R}^N$), $l(x(i|t), u(i|t))$ is the stage cost described in (10) (different from $c(x(i), u(i))$ of 4.3), $\xi(1|t) = \xi_{meas}$ sets the initial condition and the other terms are related to the linear and nonlinear inequality constraints. The constraints are equal to the ones of 4.4 with an exception, regarding the equality constraint $\xi(1|t) = \overline{\xi}$. In the framework of MPC, this holds true only at first iteration, where $\overline{\xi} = 0.55$ is the initial condition. For the following iterations, $\xi(1|t) = \xi_{meas}$, where ξ_{meas} is the measured SoC. In a real case either a measurement is available or a suitable observer is setup. In this work, the next initial state is instead computed using the model and applying the MPC input. In other words, ξ_{meas} is taken as the first element of the optimal vector $\xi^* \in \mathbb{R}^N$ returned by MPC at the previous iteration $t-1$, i.e., $\xi^*(1|t-1)$. This statement implies that the model is considered ideal and no measurement error is present. The MPC at each time instant solves a FHOCP. In this case, if the optimal solution to the FHOCP is $U^* \in \mathbb{R}^N$, the control input applied to the system at t is $U^*(1|t)$.

The parameter that influences MPC the most is the prediction horizon N. Its choice is a trade-off between the quality of the solution and computational load; for this reason, the proper selection of N is described in detail in section 6.
It's important to notice that in MPC, due to the short horizon the final constraint considered in the problem 4.5 cannot be enforced, as it would restrict too much the variation of the SoC, which implies a low usage of the battery and, hence, of the electric motor.
The removal of the final constraint results in suboptimal battery management if the cost function in 4.3 remains unchanged. In this case, the optimization algorithm would prioritize fuel minimization by excessively depleting the battery, eventually reaching the lower operational limit of 40% and potentially violating system constraints. Therefore, a different way of approaching the problem is needed; in particular, to overcome this limitation, a new cost function has been defined:

$$
l((x(i|t), u(i|t)) = \dot{m}_f(i|t) + s_{eq}(SoC(i|t)) \cdot \dot{m}_b(i|t) \tag{10}
$$

where $\dot{m}_f$ is the fuel flow as in (7), $\dot{m}_b$ is the virtual fuel provided the battery and s_{eq} is an equivalence factor which depends on the state variable. The idea is that the employed energy battery, sooner or later, must be restored (charge sustaining) at the cost of some actual fuel.
The equivalence factor acts as a weight that is strongly dependent on the current state of charge, decreasing

[1]Please note that as for Section 4.5 the index i starts from $i = 1$ due to the simulation environment used.

the cost of battery use when the SoC, ξ, is high and increasing when ξ is low, as reported in fig.6. Using this incentive-disincentive technique, it is possible to never actually approach the bounds imposed on SoC, but go arbitrarily close to them, depending on the weights given to s_{eq}, which are free design choices. For all the experiments performed in a purely simulation-based environment, it was possible to notice that the explicit constraints on the SoC were not necessary, since the actual SoC boundaries were never reached.
Nevertheless, since there is no mathematical guarantee that they will never be reached or even surpassed in a real-world scenario and since their removal does not bring any significant computational advantage, these constraints are kept as explicit.

$$s_{eq}(T_{EM}) = \begin{cases} \eta_1 \cdot W(SoC) & \text{if } T_{EM} \geq 0 \ (i.e.discharge \\ \eta_2 \cdot W(SoC) & \text{if } T_{EM} < 0 \ (i.e.recharge) \end{cases} \tag{11}$$

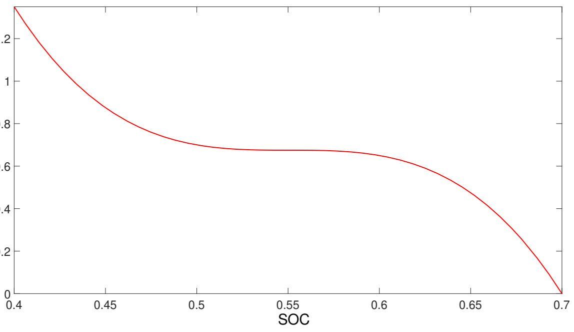


Figure 6: W(SoC)

The values for η_1 and η_2 can be chosen according to different criteria. In this particular case, firstly the average efficiency of both motors have been computed, $\overline{\eta_{ICE}}$ and $\overline{\eta_{EM}}$; then η_1 and η_2 are computed as $\eta_1 = \frac{1}{\eta_{ICE} \cdot \eta_{EM}}$ and $\eta_2 = \frac{\eta_{EM}}{\eta_{ICE}}$. Since $\overline{\eta_{ICE}} = 0.2757$ and $\overline{\eta_{EM}} = 0.8879$, then $\eta_1 = 4.0851$ and $\eta_2 = 3.2205$.
The idea behind this coefficients is, in a nutshell:

- In case of battery discharge, the amount of fuel power needed to restore the employed battery power in the future is given by P_b (battery power to be recharged)$\rightarrow \frac{P_b}{\eta_{EM}}$(necessary electric power)$\rightarrow \frac{P_b}{\eta_{EM} \cdot \eta_{ICE}}$ (necessary fuel power). From this, η_1 is easily retrieved;

- In case of battery charge, the stored battery energy will save some fuel in the future. Thus, P_b (delivered battery power)$\rightarrow P_b \cdot \eta_{EM}$(electric power to the wheel)$\rightarrow \frac{P_b \cdot \eta_{EM}}{\eta_{ICE}}$ (amount of fuel needed to deliver the same electric power to the wheel, i.e. saved fuel). From this, η_2 is easily retrieved.

Until now, however, there is no way to determine or predict the SoC value at the end of whole the driving cycle. To address this issue, an assumption must be made: the duration of the driving cycle is known *a priori*. In the context of an autonomous vehicle, which involves multiple levels of planning, including a global planning layer, this assumption remains realistic.

The proposed solution is based on a modification of the equivalence factor by introducing a time-varying term. It is defined as:

$$s_{eq}(T_{EM}, \tau) = \begin{cases} s_{eq}(T_{EM}) & \text{if } \tau \geq 10^3; \\ s_{eq}(T_{EM}) + 10^3 \cdot (SoC_{initial} - SoC_{new}) \cdot e^{-\frac{\tau}{150}} & \text{if } \tau < 10^3; \end{cases} \tag{12}$$

where τ indicates the remaining time until the end of the cycle and SOC_{new} indicates the latest value of the SoC returned by MPC.
The equivalence factor is kept to its nominal value until 10^3 seconds are left from the end. After that, an additive exponential term is added, whose role is to weigh the difference between the actual SoC and the target one which, in this case, is the initial one. Therefore, it can be interpreted as a terminal stage cost added to the cost function. Finally, please note that this term is actually independent from the prediction horizon employed, which allows to achieve a more general result.

5.2. Dynamic Programming - DP

Given the complexity of the algorithm and given that the sole purpose of using DP is to set an offline benchmark to evaluate the quality of the online results returned by MPC, in this subsection it will be presented strictly how it has been used to perform the validation procedure.
DP directly solves the optimization problem formulated in 4.5, thus it considers the whole driving cycle. In particular, since it is a non-causal method, it needs the full driving cycle in advance and then, starting from the end of the cycle and going backwards, it reconstructs the optimal control sequence according to Bellman's optimality principle.
In the implementation of DP, it is necessary to impose a global constraint on the final value of the control variable. This constraint will be set in two distinct ways:

- In the first case, $SoC_{DP}(t_{final}) = SoC_{MPC}(t_{final})$, where t_{final} is the time at which the chosen driving cycle ends. This is done to directly compare the solution found through MPC with the "absolute" optimal returned by the DP under the same conditions (the same path, the same initial conditions and the same final value of the state variable). Therefore in this case it's MPC that decides the final state, and it is then imposed onto DP.

- In the second case, $SoC_{DP}(t_{final}) = 0.55 = SoC(t_0)$, where t_{final} is the time at which the chosen driving cycle ends and t_0 is the initial time instant. This is used in a second version of MPC, where the cost function is modified to enforce the global constraint $SoC_{init} = SoC_{final} = 0.55$. Again, the goal is to use the solution provided by DP to evaluate the performance of MPC.

The performance of DP depends on several factors; the ones that have more influence are the quantization of the state and input space. However, since this solution is generated offline only for comparison purposes, the detailed description of its performance is not of interest for this work.

6. Problem Solution

In this section, firstly the approach employed to solve the optimization problem is outlined, specifically through Sequential Quadratic Programming. The selection of its parameters and the computation of derivatives is detailed in 6.1.
Then, the final setup of MPC is outlined in 6.2. The initialization strategy for the first iteration and the warm-starting technique for subsequent iterations is reported. Furthermore, a brief discussion is included on the preprocessing of efficiency maps, which were originally available as tabulated data. This is followed by an analysis aimed at determining the optimal prediction horizon.
Finally, in Sections 6.3 and 6.4 the simulation results for both MPC formulations are presented, comparing their performance against the benchmark solution obtained via Dynamic Programming. The analysis focuses on fuel consumption and battery management efficiency. The evaluations are conducted over both standard driving cycles and randomized ones.

6.1. SQP parameter definition

SQP parameters tuning

In both cases (whether using the nominal s_{eq} in (11) or the modified version in (12)), the chosen strategy to solve the Finite Horizon Optimal Control Problem is Sequential Quadratic Programming. This choice is justified by the nonlinearity of both the cost function and the constraints, categorizing the problem as a general Nonlinear Programming problem. SQP is well-suited for efficiently handling such problem classes.
To apply SQP, several key elements must be defined. Among these, the method used to compute the Hessian is of paramount importance. In this framework, the BFGS method is employed, primarily due to the presence of an economic stage cost in the objective function. Within the BFGS approach, the gamma factor used to apply Powell's trick plays a crucial role in ensuring that the computed Hessian matrix remains positive definite. In this case, the gamma factor is set to its nominal value of 0.2.
For gradient computation, the Central Finite Difference method is selected, as it provides a good balance between computational complexity and accuracy. The perturbation step is set to 2^{-17}, which is a reasonable step size when using double-precision floating-point arithmetic.
Other tuning parameters include tolerances on the cost function, constraints, and variations in optimization variables, as well as settings for the Backtracking Line Search subroutine. Most of these parameters are left at their nominal values, which can be referenced the appendix.

6.2. MPC implementation setup

Initial guess and warm starting

As with any iterative method, an initial guess is required. Since the vehicle starts from a rest position and the EM operates with high efficiency at low speeds, the best initial guess is to set $U = 1$. The vector dimensions depend on the chosen prediction horizon; specifically, $U \in \mathbb{R}^N$, where N represents the prediction horizon. Consequently, "1" denotes a vector of ones with appropriate dimensions. This strategy has to be applied only at the first iteration of the MPC algorithms as in the following warm-starting of the solver is employed. Specifically, if $U^* = [u^*(1|t-1), ..., u^*(N|t-1)]$ is the optimal control vector obtained in the previous MPC iteration at time $t-1$, then the vector $[u^*(2|t-1), ..., u^*(N|t-1), u^*(N|t-1)]$ is used to initialize the solver at time t, improving convergence and reducing computational overhead.

Precomputed offline grids

While solving the problem, it was observed that performing online interpolation of efficiency values directly from tabulated efficiency maps was computationally inefficient. To address this, a MATLAB function was employed to generate a structured grid using griddedInterpolant, allowing for efficient interpolation during simulations. This preprocessing step eliminates the need for real-time interpolation, significantly reducing computational overhead while preserving accuracy.

Choice of the time horizon

Within this framework, extensive testing was conducted on the *basic* MPC implementation (i.e., using the nominal s_{eq} defined in (11)), evaluating different prediction horizon choices to determine the optimal balance between computational efficiency and solution quality. The objective was to identify the best trade-off between computational time and the optimality of the solution.
To achieve this, the total fuel consumption m_f was computed across various driving cycles, considering prediction windows ranging from $N = 1$ to $N = 5$. The tests included scenarios with sudden accelerations, which led to battery discharge, as well as downhill paths that facilitated battery recharging. Interestingly, in some cases, increasing the prediction window resulted in higher fuel consumption. This is due to m_f not being the direct cost function of MPC but rather one of its contributing factors. Instead, as expected, the MPC cost function value decreases as the prediction horizon increases, as illustrated in Figure 7.

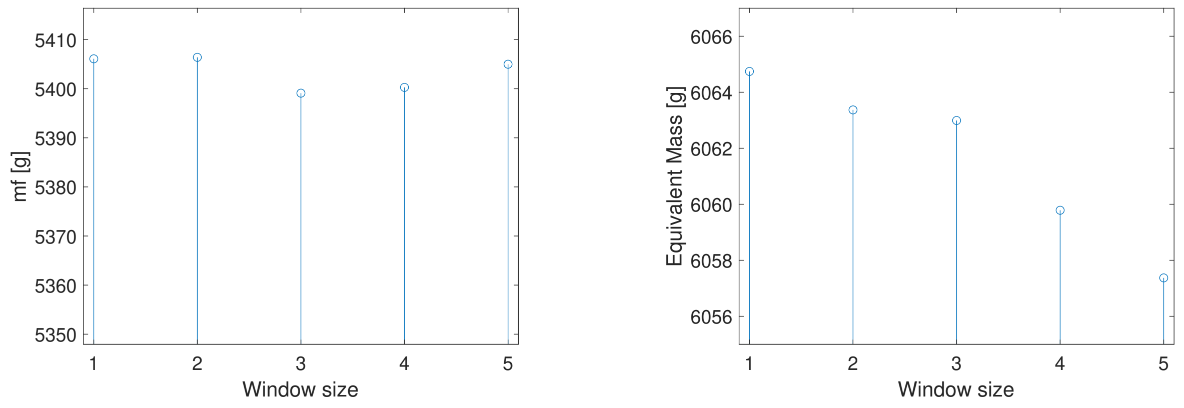


Figure 7: Comparison between m_f obtained through MPC with different prediction horizons on an 8 hour driving cycle (on the left) and the corresponding cost function values over the same cycle (on the right).

As can be observed, the variation in cost function values among the different solutions is marginal. Thus, provided that the size of the window does not have much relevance in the results, but it heavily affects the computational load, the prediction horizon has been set to N=1 [2] . By considering such a short horizon, the assumption of a deterministic route has been removed, given that the optimization is only performed on current speed and torque requests, available to the controller of an autonomous car, so no knowledge of the future is required. Actually, this solution could be implemented also in non-autonomous vehicle, taking as reference the driver's torque request.

[2]Note that setting $N = 1$ implies having a single optimization variable, i.e., $u(1|t)$.

6.3. Results with basic MPC

The results are compared to those obtained using the DP algorithm. Performance evaluation and comparison are conducted by normalizing the fuel consumption m_f of MPC with respect to that of DP, defining the ratio as $r = \frac{m_{f,mpc}}{m_{f,dp}}$.
After testing over 40 different driving paths with varying lengths, speed profiles, and downhill patterns[3], the average ratio r, obtained by setting the final SoC in DP equal to that found by MPC, was approximately 1.05, an excellent result[4]. Figure 8 presents a comparison of fuel consumption in one of the test scenarios.

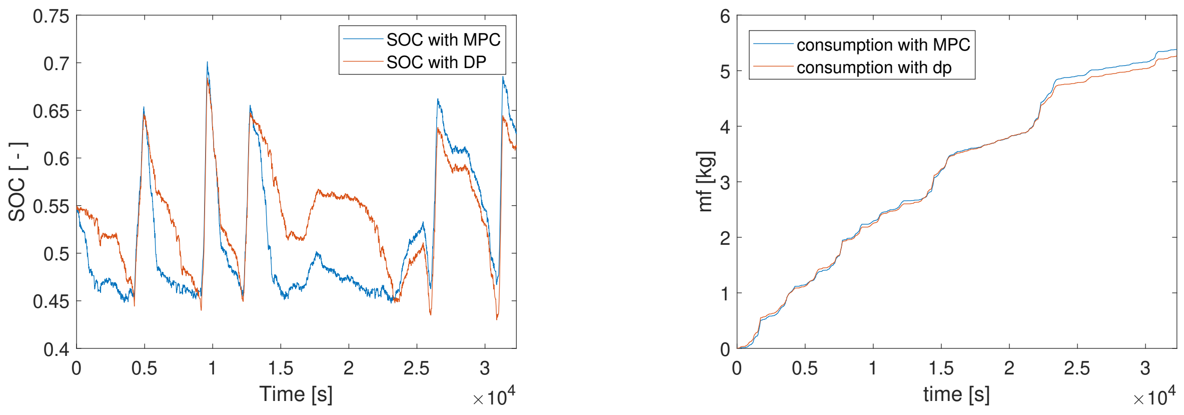


Figure 8: On the right, comparison in time between the fuel consumption achieved by applying the input obtained through MPC with N=1 and through DP on the 8h path, r=1.0218

This further highlights that having extensive foresight into future conditions rarely leads to significant improvements in the optimal control variable. This outcome suggests that the solution obtained using MPC with a prediction horizon of $N = 1$ is already very close to the optimal one provided by DP.
The results of the MPC performance over standard driving cycles are presented in Figure 9. The performance is evaluated in comparison to DP, where the final SoC constraint in DP is set to match the final SoC obtained from the MPC simulation.

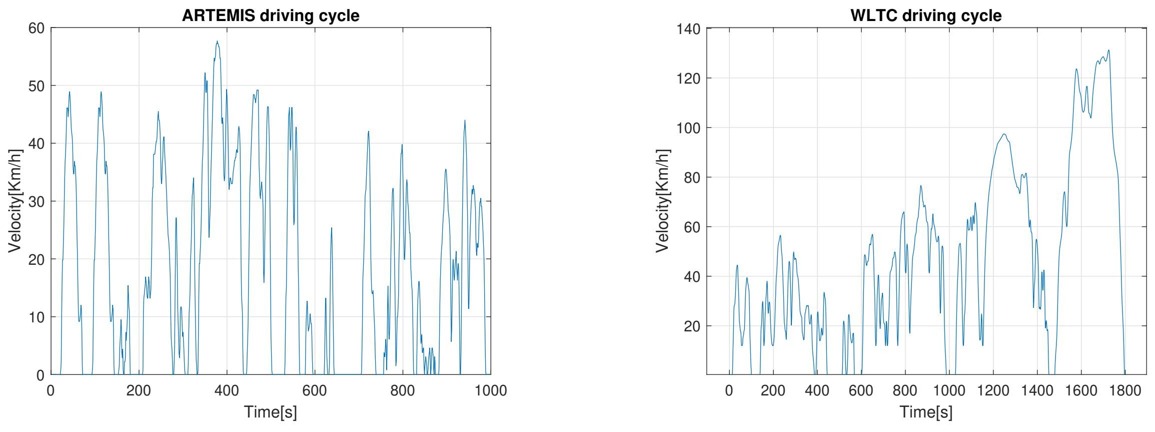


Figure 9: ARTEMIS driving cycle on the left, WLTC driving cycle on the right.

[3]Average length: 70 km (standard deviation: 54 km), average duration: 94 min (standard deviation: 61 min), downhill gradients between -10% and 0%;

[4]Standard deviation: $4.4 \cdot 10^{-2}$, maximum: 1.1759;

Figures 10 and 11 show the results for the ARTEMIS and WLTC driving cycles, respectively. The comparison indicates that the proposed MPC approach achieves satisfactory performance, closely aligning with the benchmark DP solution.

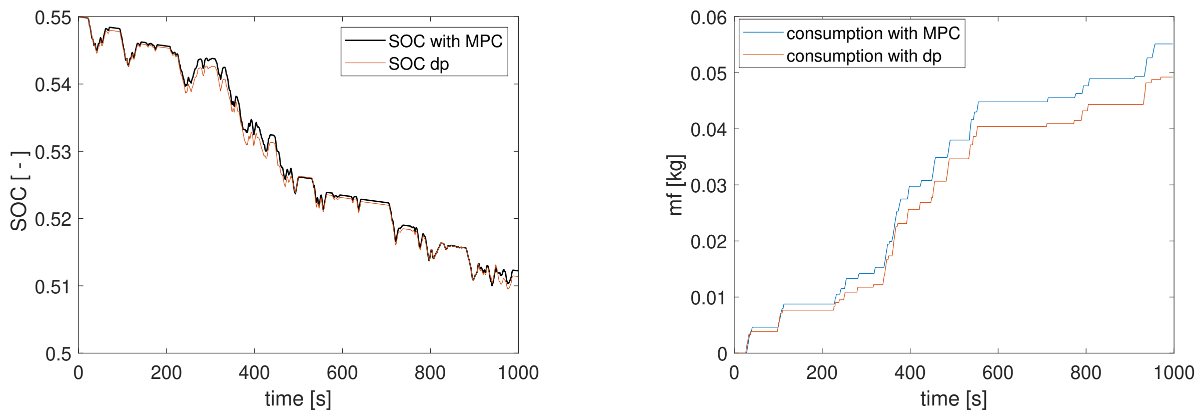


Figure 10: Comparison between SoC obtained through MPC with N=1 and DP (on the left) and the corresponding m_f (on the right) on ARTEMIS driving cycle. The m_f ratio is 1.11.

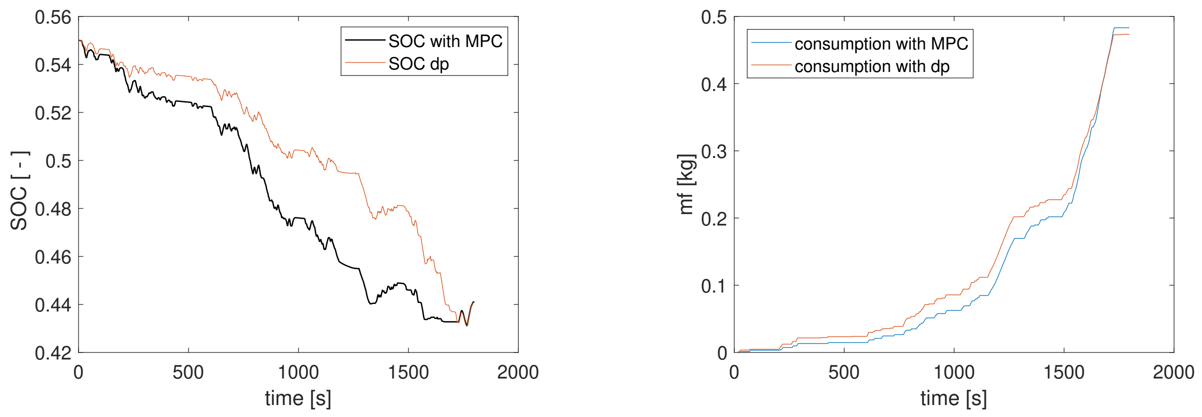


Figure 11: Comparison between SoC obtained through MPC with N=1 and DP (on the left) and the corresponding m_f (on the right) on WLTC driving cycle. The m_f ratio is 1.02.

6.4. Results with modified MPC

The objective of the modified MPC (the one with s_{eq} presented in 12) is to ensure that the desired final SoC is achieved by the end of the driving cycle. Through these simulations, it is shown that the adjusted s_{eq} formulation successfully meets the intended performance criteria.
By evaluating this approach across the same set of 40 driving paths used in 6.3, the average ratio r, computed by setting the final SoC in DP equal to the initial SoC, was found to be approximately 1.0873, which remains a highly satisfactory result[5]. The slightly higher average compared to the previous case is expected, as the original MPC formulation did not enforce a final SoC constraint. Moreover, the maximum observed r is significantly higher than in the unconstrained scenario, highlighting the impact of imposing this additional constraint. This discrepancy is particularly evident in scenarios where DP has a strategic advantage, such as when a downhill occurs near the end of the driving cycle. Since DP has full foresight of the cycle, it can deplete the battery earlier, knowing it will be recharged while descending. In contrast, MPC has to waste a significant amount of power at the end to make sure to meet the final constraint.

In fig.12 and 13, results over standard driving cycle are reported.

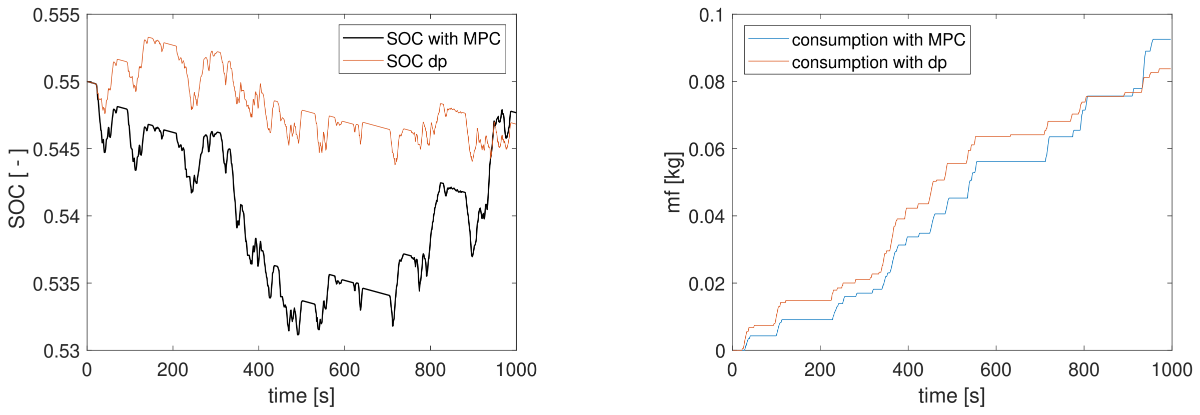


Figure 12: Comparison between SoC obtained through MPC with N=1 and DP (on the left) and the corresponding m_f (on the right) on ARTEMIS driving cycle. The m_f ratio is 1.104.

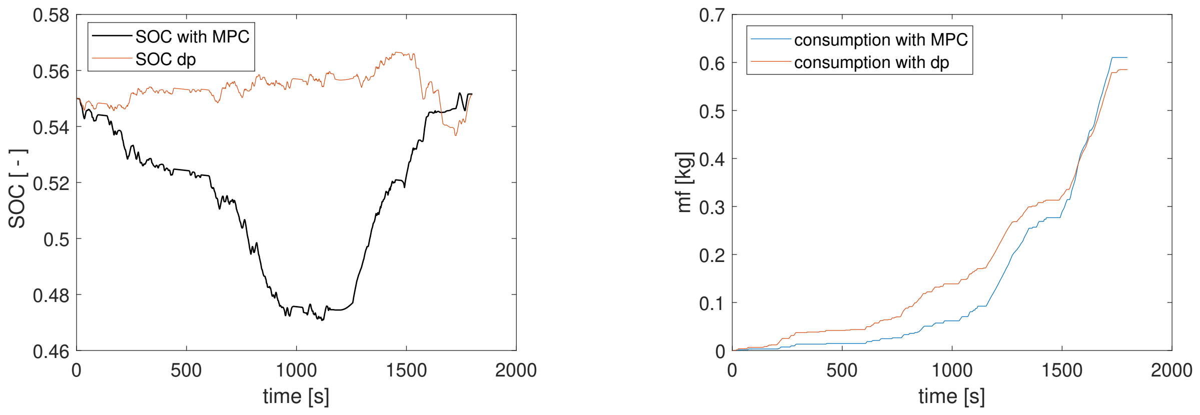


Figure 13: Comparison between SoC obtained through MPC with N=1 and DP (on the left) and the corresponding m_f (on the right) on WLTC driving cycle. The m_f ratio is 1.043.

[5]Standard deviation: $1.294 \cdot 10^{-3}$, maximum: 1.85;

7. Conclusion

In conclusion, this study focused on the implementation and evaluation of Model Predictive Control for fuel consumption minimization in a full hybrid autonomous vehicle. The experimental framework involved modeling the hybrid powertrain, formulating an optimization problem, and developing a real-time control strategy based on MPC. Dynamic Programming was used solely as a benchmark for comparison, while the study primarily explored the performance, feasibility, and efficiency of MPC in real-time applications.

The results demonstrate that MPC, even with a short prediction horizon (N=1), achieves fuel consumption values very close to the optimal benchmark provided by DP. Across multiple test scenarios, the ratio of fuel consumption between MPC and DP averaged around 1.05 in the unconstrained case and 1.087 when enforcing the final state-of-charge constraint.
This work could be further improved by accounting for noise in the state-of-charge measurements. Additionally, testing the proposed approach in real-world driving scenarios would provide further validation of its applicability and effectiveness.

Appendix - Matlab Script

The solution found is structured as follows:

- The script "*my_hev*" contains all the vehicle data, regarding the physical parameters, the powertrain parameters and the battery parameters. This script exploits the vehicle model and it is used for simulation purposes. It returns the SoC predicted value and the torque requests;
- The script "*my_fullhorizon*" is used as handle for the main script to run the simulation, i.e. internally it calls "*my_hev*". The role of this handle is to define the cost function and the constraints of the problem;
- The script "*dp_comp*" is used to compute the DP solution;
- The script "*main*" effectively implements MPC and runs the SQP iterative method for its resolution. It then runs the dp algorithm and compares the two solutions.

The SQP method, overall, is defined by means of three scripts:

- "*my_optimset*", which allows to define the parameters for the SQP method such as the method to compute the gradient, the number of maximum iterations etc.;
- "*my_fmincon*", which actually implements the SQP iterative method based on the parameters defined in "*my_optimset*". This script has an advantage w.r.t. the "MATLAB fmincon" since it takes as input immediately both the cost function and the constraints at once, so it is not needed to run the simulation twice over the same parameters, wasting computational resources. The solver selected to solve the resulting Quadratic Program is "Matlab quadprog", whose options are set in "*my_optimset*". In particular, the chosen algorithm is the "interior-point-convex";
- "*my_gradient*", which implements several methods to compute numerically the derivatives; among them, there is the possibility to opt for the Imaginary-Part Trick, the Forward or Central Finite Differences or for a User-Provided method.

Table 6: Optimization Settings in `my_fmincon`

Parameter	**Value**
Hessmethod	`BFGS`
gradmethod	`CD`
graddx	2^{-17}
tolgrad	1×10^{-6}
tolfun	1×10^{-12}
tolx	1×10^{-3}
ls_beta	0.2
ls_c	0.01
ls_nitermax	1×10^{2}
nitermax	1×10^{2}
tolconstr	1×10^{-3}

References

[1] Shun Chen, Guodong Fan, Yansong Wang, Boru Zhou, Siyi Ye, Yisheng Liu, Bangjun Guo, Chong Zhu, and Xi Zhang. The impact of intermittent overcharging on battery capacity and reliability: Electrochemical performance analysis and failure prediction.

[2] Lorenzo M. Fagiano. Numerical optimization for control. Lecture Notes, 9 2024.

[3] L. Guzzella. *Vehicle Propulsion Systems: Introduction to Modeling and Optimization.* Springer, 2 edition, 6 2007.

[4] Sergio Savaresi. Automation and control of electrical and hybrid vehicles. Lecture Notes, 2 2024.