Assignment #4: dynamic programming

Table of Contents

Group information

Group number: 33

Students:

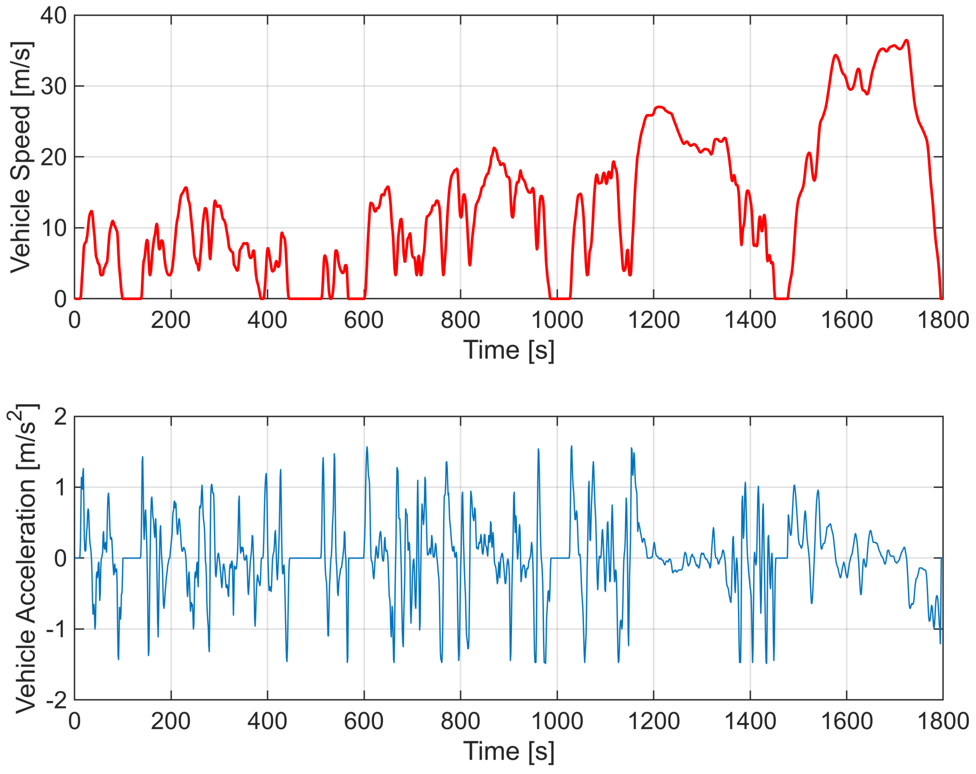
- Loris Fonseca, s342696
- Maurizio Pio Vergara, s346643
- Nikoloz Kapanadze, s342649

Load the cycle and vehicle data

The simulation conducted in this project focuses on implementing a control strategy based on Dynamic Programming evaluated on the WLTP driving cycle. This cycle is initially loaded and initialized in the following section, with its variables for mission, time, vehicle speed, and acceleration. To visually inspect the nature of the cycle, both the vehicle speed and acceleration are plotted over time. These plots help to understand the dynamic behavior of the test case and anticipate how challenging each driving pattern might be for the Dynamic Programming based controller.

```
clear
close all
clc

% Load folders containing the vehicle model, mission and functions
addpath("models");
addpath("data");
addpath("utilities");

%% LOAD CYCLE DATA
mission = load("data\WLTP.mat");
time = mission.time_s;                  % [s]
vehSpd = mission.speed_km_h ./ 3.6;     % [m/s]
vehAcc = mission.acceleration_m_s2;     % [m/s^2]

%% PLOT OF WLTP CYCLE (Speed and Acceleration)
```

```
% Vehicle Speed plot
nexttile(1);
plot(time, vehSpd, Color='r', LineWidth = 1);
xlabel('Time [s]');
ylabel('Vehicle Speed [m/s]');
grid on;

% Vehicle Acceleration plot
nexttile(2);
plot(time, vehAcc);
xlabel('Time [s]');
ylabel('Vehicle Acceleration [m/s^2]');
grid on;
```



```
% Load vehicle parameters
veh = load("data\vehData.mat");

% Scale of vehicle data according to group parameters through the given scale
function
veh = scaleVehData(veh, 82000, 55000, 1260);
```

Develop a fuel-optimal EMS with dynamic programming

Dynamic Programming (DP) is an optimization algorithm that identifies the global minimum of a cost function by breaking down a complex problem into simpler subproblems. The optimal solution to the overall problem is then constructed by combining the optimal solutions of these smaller components. To apply DP, the time span, the state variables, and the control variables must be discretized. A finer discretization, hence a more realistic representation of the real problem, leads to more accurate results, since DP guarantees the global minimum of the discretized problem. However, increasing the resolution also increases computational effort. Indeed, the more the number of state and control variables involved, the more the number of combinations that must be investigated, making the problem significantly more complex. This phenomenon is known as the curse of dimensionality.

The DP solution process consists of two main phases: the backward phase and the forward phase. In the backward phase, a map of all feasible options, according to the discretized state and control variables, is built by solving the problem iteratively in reverse time order, thus starting from the final stage and using previously stored solutions of simpler subproblems to compute the cost-to-go for each feasible state. Once this map is complete, the forward phase identifies the optimal control path that leads to the solution found in the backward phase.

When applied to Energy Management Strategies for Hybrid Electric Vehicles, Dynamic Programming can be used to minimize performance indicators such as fuel consumption, emissions, or a combination of multiple criteria, while respecting operational constraints such as battery state of charge and power demand. In this project, two optimization objectives are considered: the first aims to minimize fuel consumption, while the second researches a trade-off between fuel consumption and battery aging. To limit the effects of the curse of dimensionality, only one state variable and two control variables are considered. The state variable is the battery State of Charge, while the control variables are engine speed and torque. The discretization is defined as follows: the number of DP stages corresponds to the duration of the driving mission, with one iteration per second. At each iteration, multiple SoC values are possible depending on the combination of engine speed and torque applied. The feasible SoC values are defined on a grid ranging from 40% to 80%, constraints that are imposed to preserve battery health. The control variables, engine speed and torque, are also discretized into grids spanning from zero up to their respective maximum feasible values.

As previously mentioned, the first simulation focuses exclusively on the optimization of fuel consumption, without taking into account any phenomena related to battery aging. The expected outcome is a significant low fuel consumption, but at the cost of a shorter battery lifespan in terms of kilometers, since the detrimental effects of battery current are not considered in the optimization.

```
% DEFINITION OF THE STATE VARIBLE GRID
SOCgrid = {0.4:0.001:0.8};            % the feasible SOC values range from 40% to
80%
initialSOC = {0.6};                   % initial SOC is assumed to be 60%
finalSOCrange = {[0.59, 0.61]};       % final SOC imposed to be equal to the
initial one with an accepted error of +-1%

% DEFINITION OF THE CONTROL GRID
% Definition of engine speed and torque limits
% Speed
engSpeedMin = veh.eng.idleSpd;
engSpeedMax = veh.eng.maxSpd;
% Torque
engTorqueMin = 0;
engTorqueMax = max(veh.eng.maxTrq.Values);
```

```
% Definition of possible speed torque grid values
k = 10;                                                     % grid dimension
engSpeedSet = linspace(engSpeedMin, engSpeedMax, k-1);
engSpeedSet = [0, engSpeedSet];                             % adding zero speed value
(engine off)
engTorqueSet = linspace(engTorqueMin, engTorqueMax, k);

controlGrid = {engSpeedSet, engTorqueSet};

% DEFINITION OF THE STAGES NUMBER
Nstages = length(time);

% DEFINITION OF EXOGENEOUS INPUTS
w{1} = vehSpd;
w{2} = vehAcc;

% CREATION AND EXECUTION OF THE OPTIMIZATION PROCESS USING DYNAMIC PROGRAMMING
prob = DynaProg(SOCgrid, initialSOC, finalSOCrange, controlGrid, Nstages, @(x, u,
w) hev_cell_model(x, u, w, veh), 'ExogenousInput', w);
prob = run(prob);
```

```
DP backward progress:100 %
DP forward progress:100 %
```

```
fuelConsumptionInstant = [prob.AddOutputsProfile{1,1}.fuelFlwRate];
fuelConsumptionVariation = cumtrapz(time, fuelConsumptionInstant);
```

Estimate the battery's lifetime

In this section, the battery lifespan is estimated using a simplified semi-empirical model that evaluates capacity loss over a single cycle. This estimation relies on a degradation factor, which depends on the battery's nominal capacity, and the ampere-hour throughput, calculated as the integral of current absolute value over the cycle duration. The equation implementing this degradation model is reported below:

$$C_{\text{loss},\%,\text{cycle}} = \beta\, Q_{\text{cycle}}$$

Once the capacity loss per cycle is determined, the total number of cycles, and thus the total distance in kilometres required to reach the end of life threshold, is calculated. Typically, automotive traction batteries are considered due for replacement when their State of Health drops to 80%, meaning that their capacity has decreased to 80% of the nominal capacity of a new battery of the same type.

```
beta = 5e-6;
Ibat = [prob.AddOutputsProfile{1,1}.battCurr]';

Q = trapz(time, abs(Ibat))/3600;                % charge needed over a cycle    [Ah]
Closs = beta * Q;                               % capacity loss in a cycle
cyclesNumber = 0.2/Closs;                       % number of cycles to reach a SoH =
80%
cycleDistance = trapz(time, vehSpd) / 10^3;     % [km]
```

```
distance_eol = cycleDistance * cyclesNumber;        % distance on WLTP cycle to
reach SoH = 80%         [km]

finalFuelConsumption = fuelConsumptionVariation(end);        % [g]
fuelConsumption = finalFuelConsumption * 10^(-3);            % [kg]
fuelEconomy = fuelConsumption / veh.eng.fuelDensity * 100/cycleDistance;        %
[L/100km]
finalSOC = prob.StateProfile{1,1}(end);                      % [-]
```

Save results

```
% Store results
save("results_base.mat", "fuelConsumption", "fuelEconomy", "finalSOC",
"distance_eol")
```

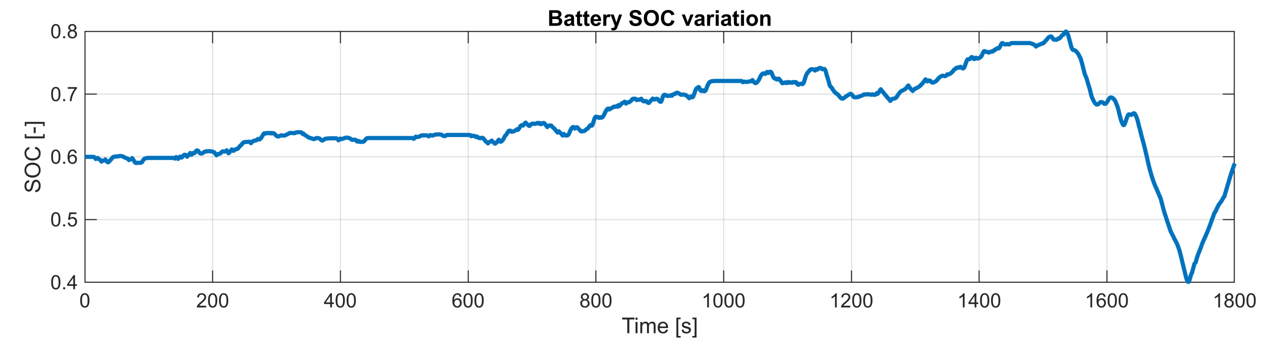
Analyze the fuel-optimal EMS

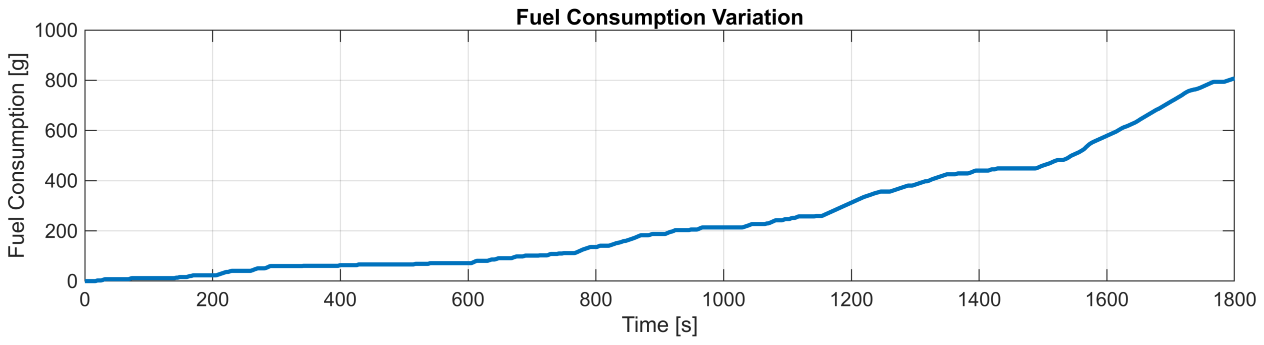
The current section aims to analyze the performance of the previously implemented Energy Management Strategy, with a primary focus on the evolution of the battery State of Charge, fuel consumption, battery lifespan, and engine operating points.

The first observation concerns the validation of the controller. To verify whether the management strategy has been correctly implemented, the SoC variation should be examined. For a proper control strategy, it is essential to ensure charge-sustaining operation, which is a typical requirement for HEVs. By analyzing the SoC variation plot, it can be observed that the battery SoC at the end of the driving mission is 59%. Since the initial SoC was 60%, and the final value remains within an acceptable tolerance of ±1%, the charge-sustaining condition is considered to be maintained, thus validating the control strategy.

Regarding fuel consumption, the trend over time reveals a piecewise linear behavior, with an increasing slope as the cycle progresses. This is attributed to the growing power demand throughout the driving cycle. The total fuel consumption for the WLTP driving mission is 4.42 L/100 km, which is highly efficient compared to the fuel consumption of contemporary hybrid electric vehicles. This demonstrates that the developed Energy Management Strategy effectively minimizes fuel consumption over the entire cycle. Furthermore, among the various types of energy management strategies, the one implemented in this simulation, based on Dynamic Programming, achieves the best fuel consumption results. It achieves better results than previous approaches, such as rule-based controller and Adaptive-ECMS. Indeed, fuel economy values for these two strategies at the end of the WLTP mission were 5.37 L/100 km for the rule-based controller and 4.50 L/100 km for the adaptive-ECMS. These results align with theoretical expectations, as dynamic programming is capable of finding the global minimum of the optimization problem, ensuring the most efficient fuel usage.

```
[fig, t] = SOCfuelConsumption(prob, time, fuelConsumptionVariation);
```





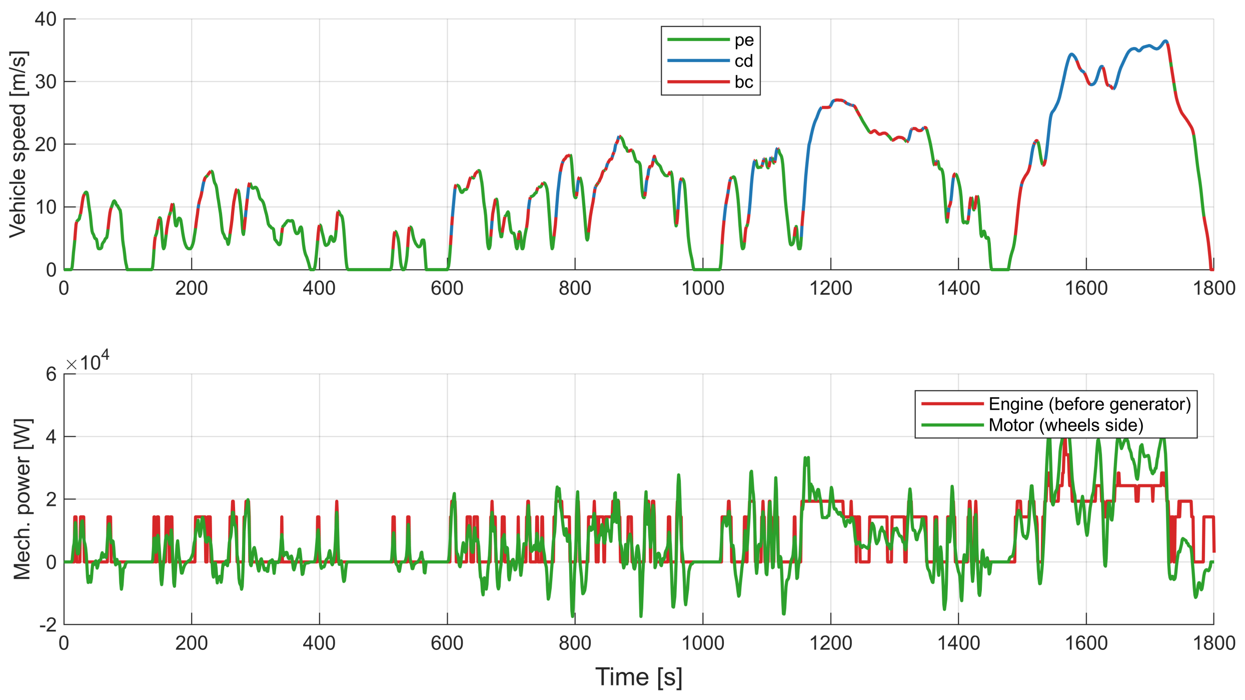
By analyzing the vehicle speed profile and its operating modes, it is possible to go deeper in the analysis of the EMS behavior. The aforementioned vehicle speed profile, along with powertrain operating modes, is reported below, as well as the engine and motor power evolution over the drive cycle. The first mode is the pure electric mode (pe), in which the engine is off. In this condition, the battery discharges when the required power at the wheels is positive and charges when the motor operates in regenerative braking. The second mode is the charge-depleting mode (cd), where the engine is running but the battery continues to discharge because the power provided by the engine is lower than the power delivered by the battery to the electric motor. The last mode is the battery charging mode (bc), in which the engine is on and actively charging the battery.

One of the first insights from the plot is the operation of the pure electric mode at low speeds. Nearly all segments of the speed profile below approximately 5 m/s are colored green, indicating PE mode activation. This behavior aligns with standard HEV control strategies, which favor electric-only operation under low power demands. An exception appears at the very end of the drive mission, during the final deceleration phase, where a red segment indicates a deviation from PE mode. The cause of this anomaly is addressed later. Moreover, it can also be noticed that throughout the drive cycle, the EMS consistently engages PE mode during deceleration phases to enable regenerative braking and maximize energy recovery. However, again this strategy is not applied in the final deceleration phase, where regenerative braking is either only partially activated or entirely absent. Conversely, during acceleration phases above 5 m/s, the engine is always active, indicating either charge depleting or battery charging mode.

This operating pattern is closely related to power demand to be provided by the powertrain. Given the limited battery capacity in hybrid electric vehicles, the EMS avoids pure electric mode during high-power events to prevent rapid battery depletion. As a result, PE mode is generally enabled only at low power demands, typically corresponding to low-speed operation, while the engine supports higher power demand phases such as acceleration at high speed. On the other hand, during deceleration, pure electric mode is strategically engaged to improve efficiency. Indeed, this strategy allows the vehicle to exploit all the energy coming from the wheels to recharge the battery, minimizing energy losses through mechanical braking that would occur if the engine were on. Given that the architecture under study is a series hybrid, all energy produced by the engine is directed to the battery via the generator. Therefore, if the engine remains on during braking, two energy sources, the regenerative braking system and the engine, would simultaneously supply power to the battery. Since the battery has a limited charging capability, this could result in exceeding its input power limits. To prevent this, the power contribution from one of the sources must be reduced. However, since there is no clutch between the engine and the battery, the engine's energy path cannot be decoupled. The only alternative would be to dissipate the regenerative energy from the wheels through the mechanical brakes, a clearly suboptimal solution, as it wastes electric energy that could otherwise be recovered "for free" in favor of engine generated energy that consumes fuel. This is why the controller engages pure electric mode during braking: it prioritizes energy efficiency by eliminating unnecessary fuel consumption and maximizing regenerative braking potential.

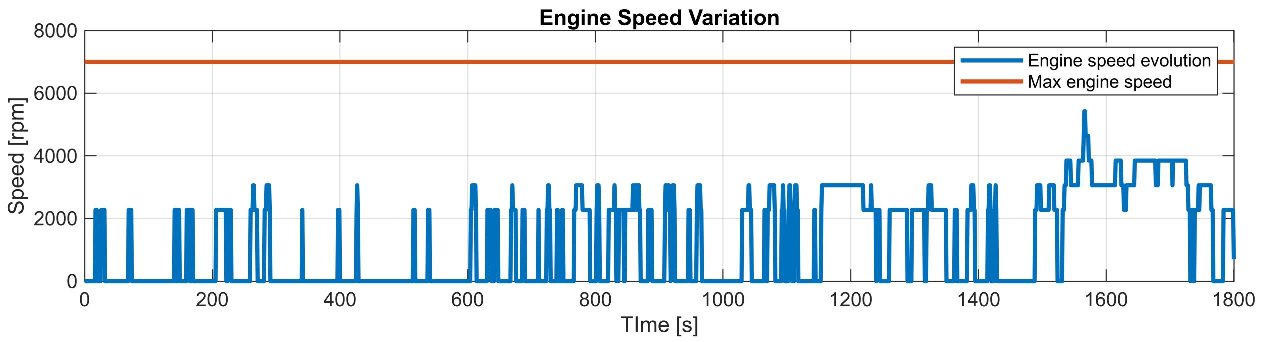
The different behavior observed during the final phase of the cycle, compared to the earlier phases, is primarily due to the nature of the WLTP cycle and the imposed State of Charge constraints. Specifically, the WLTP cycle used in the simulation is characterized by a progressive increase in average power demand. As a result, the final portion of the cycle exhibits the highest power requirements, as clearly shown in the plot of motor power. Due to this high demand, the battery must supply significant power during the last phase. Meanwhile, the control strategy ensures that the engine operates within its high efficiency zone to avoid inefficient conditions. However, this zone typically corresponds to high power levels, thus the engine alone cannot provide enough power to both meet the demand and recharge the battery during this phase. This leads to a sharp decline in SoC, which drops from the upper constraint of 80% to the lower limit of 40%, starting around *t = 1530s* and lasting approximately 200 seconds. The unusual operating behavior of the powertrain during this phase is therefore directly linked to these SoC constraints. Ideally, to avoid engine on operations also during the last deceleration phase, or at least to reduce them, the control strategy should aim to store more electrical energy earlier in the cycle. This would ensure that enough battery charge is available to handle the high power demands of the final phase, allowing the SoC to remain at a level that allows the energy recovery through regenerative braking of the last deceleration phase to be sufficient to restore the SoC value that guarantee charge sustaining. However, such a strategy is not recommended in practical applications, as it would involve increasing the upper SoC threshold beyond the typical 80%, or even removing it entirely, to meet the requirement in terms of increased energy storage. While this could reduce fuel consumption, it would come at the cost of increased battery degradation and reduced battery lifespan, an undesirable trade-off in real-world scenarios.

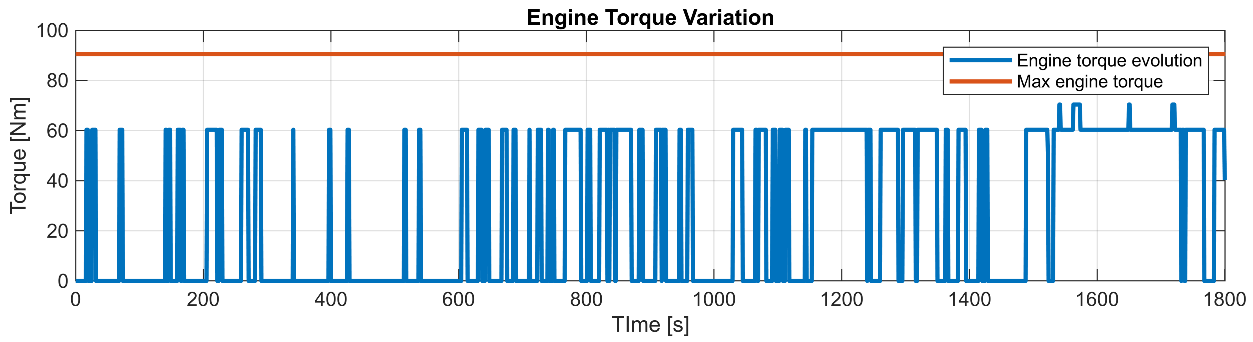
```
[fig1, t1] = engMotPwrComparison(prob.AddOutputsProfile{1,1});
```



From the engine speed and torque variation plots, it can also be observed that the engine operates predominantly at medium speeds and medium-to-high torque levels. This operating condition further confirms the proper implementation of the controller, as it ensures the engine functions within its most fuel-efficient region. Operating in this range, i.e. mid-speed and mid-high load, maximizes fuel efficiency, aligning with the objective of minimizing fuel consumption, which is the only optimization criterion considered by the EMS in this initial simulation. The engine speed and torque profiles imposed by the EMS result in an almost constant engine power level throughout the cycle whenever the engine is active. Notable power peaks occur only during the final phase of the mission, where a higher power demand is required from the powertrain. This trend is also evident in the previously discussed plot comparing engine and motor power.

```
[fig2, t2] = engSpdTrq(prob, veh, time);
```





Although this Energy Management Strategy allows for the identification of the optimal fuel consumption, it presents two major critical issues. The first is inherent to the structure of Dynamic Programming itself. This strategy requires prior knowledge of the entire driving mission to perform the characteristic backward and forward phases necessary to evaluate the global minimum of the cost function. As a result, it cannot be implemented in real-time vehicle controllers, since in real-world scenarios the future driving profile is not known in advance. The second critical issue is not related to the structure of Dynamic Programming, but rather to the way it has been implemented. In particular, as previously mentioned, the cost function adopted in this initial simulation considers only fuel consumption, completely neglecting the effects of battery degradation. This design choice significantly impacts battery lifespan, which drops to a State of Health of 80% after just 92,500 kilometers, an unacceptable value for real-world applications, especially considering that the average lifespan of a hybrid vehicle is approximately 200'000 kilometers. An energy management strategy that leads to battery replacement after so few kilometers, not only has a substantial environmental impact, but also negatively affects the user experience, as most users are not willing to tolerate the cost of a new battery only because of a suboptimal energy management. Such considerations clearly indicates the need to revise the cost function. Therefore, a second simulation is carried out in the following section. This time, the cost function also accounts for battery aging, and the corresponding results are analyzed accordingly.

Develop an aging-aware EMS with dynamic programming

The following section aims to develop an Energy Management Strategy that takes into account not only the minimization of fuel consumption but also battery aging. Indeed, the State of Charge is not the only factor affecting battery lifespan. Therefore, considering only SoC constraints is not sufficient to achieve a good result in terms of battery durability. Several factors contribute to the reduction in battery capacity, including high temperatures and large current values. Since accurately modeling thermal effects requires a complex thermal model, this second part of the project focuses specifically on the impact of battery current. SoC levels are also taken into account, as they were in the initial optimization focused only on fuel consumption. As a result, the differences observed between the two simulations are exclusively due to the inclusion or exclusion of current-related degradation in the Energy Management Strategy.

To account also for the influence of current on battery degradation in the evaluation of the optimal control strategy, the stage cost function between two consecutive iterations is modified by introducing an additional term that penalizes high current values. This penalization term allows to calibrate the controller through a trade-off parameter, α. By adjusting the value of this parameter, different weights can be assigned to the optimization objectives. Specifically, setting $\alpha = 1$ results in a strategy that minimizes only fuel consumption, while $\alpha = 0$ prioritizes minimizing battery degradation due to high current. Intermediate values of the parameter represent different trade-off combinations between the two objectives. The equation implementing the revised formulation of the cost function is reported below:

$$L_k = \alpha \frac{\dot{m}_{\mathrm{f},k}}{\dot{m}_{\mathrm{f,max}}} + (1-\alpha)\frac{|i_{\mathrm{b},k}|}{i_{\mathrm{b,max}}}$$

As shown in the formula, the fuel flow rate and battery current are normalized by the engine's maximum fuel flow rate and the battery's current limit, respectively, to enable a meaningful comparison and trade-off between the two quantities.

In this simulation, the trade-off parameter is initialized to the value that yields the best balance between fuel consumption and battery aging. The process used to determine this optimal value is discussed in greater detail in the dedicated paragraph.

```
alpha = 0.8;
fuelFlwRateMax = maxFuelFlwRate(veh.eng);

% CREATION AND EXECUTION OF THE OPTIMIZATION PROCESS USING DYNAMIC PROGRAMMING
prob_aging = DynaProg(SOCgrid, initialSOC, finalSOCrange, controlGrid, Nstages,
@(x, u, w) hev_cell_model_BatteryAging(x, u, w, veh, alpha, fuelFlwRateMax),
'ExogenousInput', w);
prob_aging = run(prob_aging);
```

```
DP backward progress:100 %
DP forward progress:100 %
```

```
Ibat_aging = [prob_aging.AddOutputsProfile{1,1}.battCurr]';

Q_aging = trapz(time, abs(Ibat_aging))/3600;             % charge needed over a
cycle    [Ah]
Closs_aging = beta * Q_aging;                            % capacity loss in a cycle
cyclesNumber_aging = 0.2/Closs_aging;                    % number of cycles to reach
a SoH = 80%
distance_eol = cycleDistance * cyclesNumber_aging;       % distance on WLTP cycle to
reach SoH = 80%        [km]

fuelConsumptionInstant_aging = [prob_aging.AddOutputsProfile{1,1}.fuelFlwRate];
fuelConsumptionVariation_aging = cumtrapz(time, fuelConsumptionInstant_aging);
```

```
finalFuelConsumption_aging = fuelConsumptionVariation_aging(end);        % [g]
fuelConsumption = finalFuelConsumption_aging * 10^(-3);                    % [kg]
fuelEconomy = fuelConsumption / veh.eng.fuelDensity * 100/cycleDistance;       %
[L/100km]
finalSOC = prob_aging.StateProfile{1,1}(end);                            % [-]
```

Save results

```
% Store results
save("results_ageing.mat", "fuelConsumption", "fuelEconomy", "finalSOC",
"distance_eol", "alpha");
```

Analyze the fuel economy-mileage trade-off

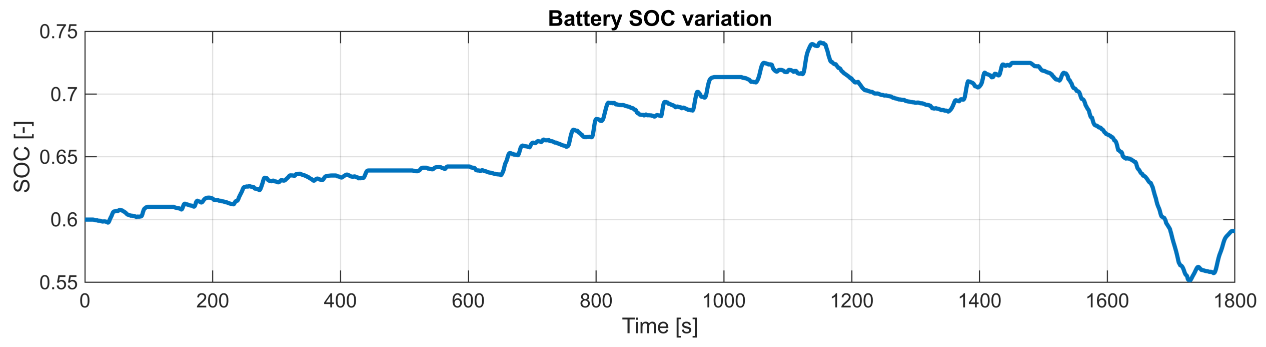
This section aims to analyze the behavior of the Energy Management Strategy when battery aging is taken into account. The analysis includes a comparison with the results obtained in the previous study, which focused only on minimizing fuel consumption. Furthermore, it explores the influence of the trade-off parameter on balancing fuel consumption and battery aging, specifically the battery's distance to its End of Life.

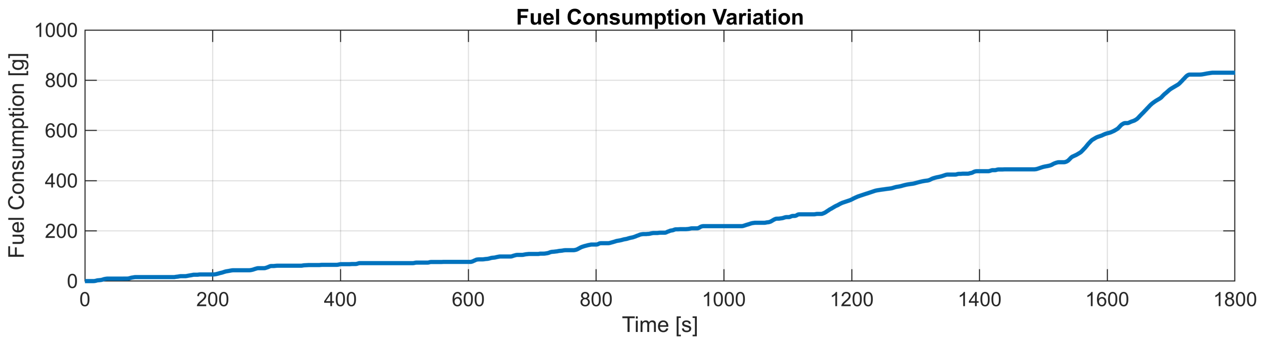
As done in the first simulation, the initial plots used in this analysis show the trends of the battery State of Charge and fuel consumption variation over the drive cycle. By analyzing the SoC variation, a behavior similar in shape to that observed in the previous simulation is noted. This similarity arises because the overall trend of the main results is more influenced by the characteristics of the drive cycle rather than the specific optimization parameters. Therefore, since both simulations are based on the WLTP cycle, the power demand at each time step is identical, leading to similar control response patterns in terms of trend.

However, a significant difference emerges due to the revised cost function. This difference lies in the upper and lower limits reached by the SoC. In the previous simulation, focused solely on fuel consumption minimization, the SoC ranged between 40% and 80%, consistent with the constraints set during controller initialization.
In contrast, in the current simulation, which also accounts for battery aging, the SoC remains within a narrower range, approximately 56% to 76%, despite the same SoC constraints being applied. Moreover, charge sustaining is still guaranteed. At first glance, this behavior may appear to result directly from the battery aging model integrated into the strategy. Indeed, maintaining the SoC closer to mid-range values tends to slow battery degradation, and the controller appears to implement this consideration. However, as previously mentioned, the only aging-related factor considered in this second simulation is the battery current, with no explicit mechanism introduced to limit the SoC range. Therefore, the observed narrowing of the SoC range is not directly enforced by the controller but is instead a beneficial indirect effect of the power levels demanded from the engine under the updated control strategy. This aspect will be further discussed in the following sections.

Regarding fuel consumption, the overall trend is similar to the one observed in the first simulation. However, a higher fuel consumption is obtained, resulting in a fuel economy of 4.55 L/100 km. This outcome aligns with both the predictions and the expected results. Since the optimization strategy is not exclusively focused on minimizing fuel consumption, the global minimum for this parameter cannot be achieved, as was the case in the first simulation, leading to a deterioration in fuel economy.

```
[fig3, t3] = SOCfuelConsumption(prob_aging, time, fuelConsumptionVariation_aging);
```



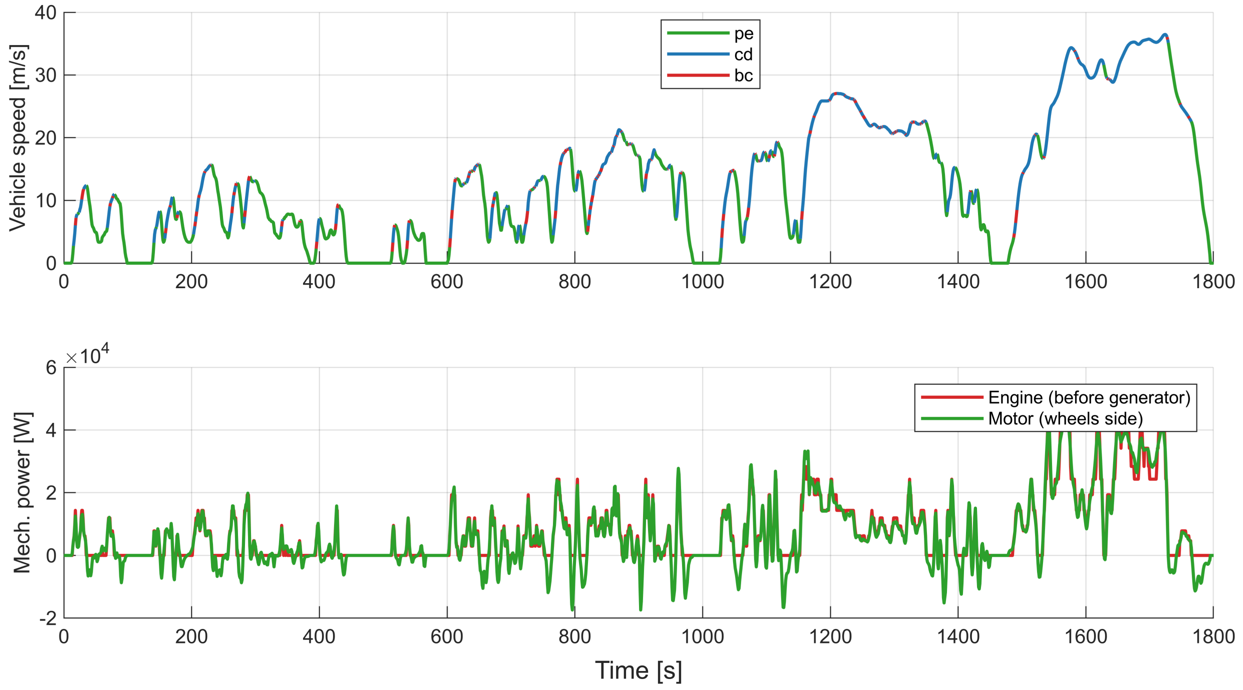


To better understand the controller's behavior and provide additional insights into the implemented Energy Management Strategy, it is useful to analyze the engine power in comparison with the motor power across the different time instants of the driving cycle. The charts below show that the engine power closely follows the profile of the motor power, i.e. the power demand at the wheels, throughout the entire mission. This represents a markedly different strategy compared to the previous simulation, where the engine power exhibited a more stair-step evolution. The speed profile also reveals that, for most of the cycle, the powertrain operates in battery depletion or pure electric mode, with significantly fewer instances of battery charging compared to the first simulation. However, although battery depletion is predominant during acceleration phases, the State of Charge increases steadily for most of the mission, until the high power demand phase begins. This behavior is still linked to the difference between the power demand and the engine power. When battery charging is active, the gap between engine and motor power is more pronounced than during battery depletion. Indeed, in the latter case, the engine power closely matches the power demand, indicating that battery discharging is very limited. This trend is also clearly reflected in the SoC evolution plot. In fact, during charging phases, the SoC increases more steeply but over shorter time intervals, while in depletion phases it remains nearly constant. Consequently, this pattern results in a gradual rise of the SoC, as previously mentioned. This slow increase continues until the onset of the high power demand phase, where the SoC drops, similarly to what was observed in the first simulation.

Moreover, as observed in the first simulation, the pure electric mode is activated during deceleration phases and during low-speed acceleration. In contrast, the other two operating modes are engaged primarily during acceleration at higher speeds.

This behavior of the engine power, which closely follows the power demand at each instant, explains the narrower variation range of the State of Charge. Since the power difference between the engine and the electric motor is small, the net power flowing into or out of the battery is correspondingly small. As a result, the battery experiences only minor charging or discharging events, leading to a very limited variation in SoC. Consequently, the SoC does not reach the limits imposed by the control strategy, unlike in the first simulation, where those limits were reached. This behavior confirms the effectiveness of the controller in implementing the battery aging optimization algorithm. The reduced power flow through the battery results in lower current levels, which directly impacts the battery's degradation rate. In this simulation, the C-rate is the key parameter to minimize in order to slow battery aging. Therefore, lower current levels contribute to lower C-rates, reducing battery wear and supporting the goal of extending battery life.

```
[fig4, t4] = engMotPwrComparison(prob_aging.AddOutputsProfile{1,1});
```

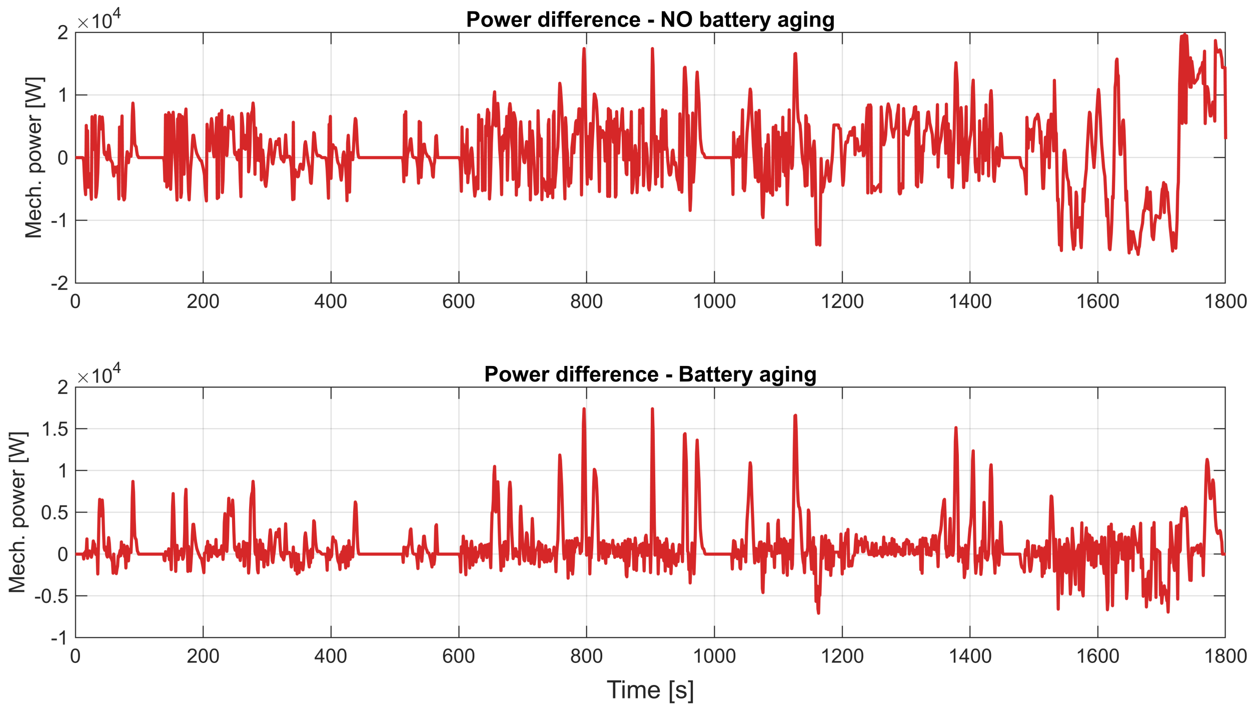


The difference in net battery power flow between the two simulations is also evident in the second set of red charts reported below. It is clear that the amplitude of the power variation is significantly greater in the simulation where battery aging was not included in the cost function. Therefore, it can be deduced that lower values of α leads to lower C-rates and battery usage.

This difference in power gap also affects the battery lifespan in the two simulations. Specifically, the simulation that accounts for battery degradation results in a significantly greater distance before reaching 80% State of Health, which marks the EoL condition. In fact, the EoL distance in this case is approximately 208,000 km, more than twice the value obtained in the first simulation, where battery degradation was not considered in the cost function. Moreover, this value aligns with the average lifespan of a Hybrid Electric Vehicle, indicating that the end user can operate the vehicle throughout its expected lifetime without the need to replace the battery.

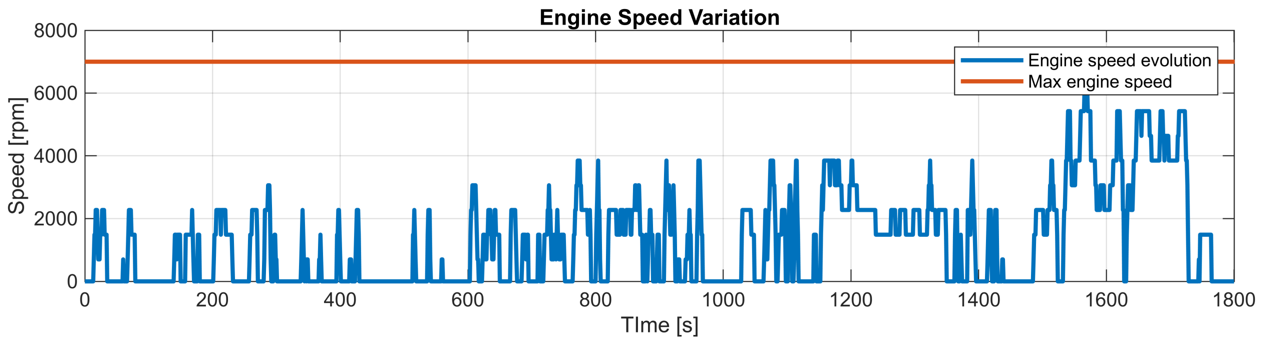
Although the net power, and consequently the current, flowing through the battery is significantly lower in the second simulation, high power peaks are still observed. However, these peaks, which contribute to elevated C-rates and thus negatively affect battery lifespan, are not the result of an incorrect implementation of the EMS. A closer analysis reveals that these peaks occur during regenerative braking phases. When this operating mode is present, the vehicle is undergoing deceleration phases and the powertrain is set on pure electric mode, with the internal combustion engine turned off. Therefore, the reason for the peaks is that during these phases, the engine power cannot match the motor power in order to avoid large power imbalances in the battery. This occurs because the thermal engine cannot provide negative power, unlike the electric motor. This leads to higher current rates in the battery, which may compromise its lifespan. It is also worth noting that these peaks appear in both simulations with the same amplitude. This is because, in both EMSs, the engine is turned off during regenerative braking, hence the amplitude of the peaks depends exclusively on the nature of the driving cycle, which is identical in both cases. A possible solution to mitigate the effects of large power peaks is to use the mechanical brakes to dissipate part of the kinetic energy. However, this strategy results in higher fuel consumption, as the energy wasted through mechanical braking, unlike regenerative braking, is not recovered and must be supplied by the thermal engine in subsequent time steps to maintain charge-sustaining operation.

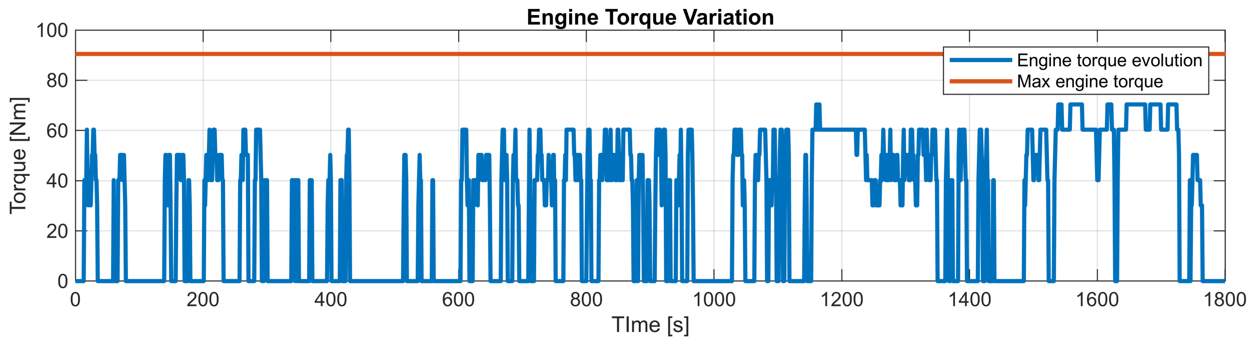
```
[fig5] = engMotPwrDifference(prob.AddOutputsProfile{1,1},
prob_aging.AddOutputsProfile{1,1});
```



The following charts illustrate the variations in engine speed and torque. Even in this second simulation, it is evident that the engine operates within the high-efficiency zone, as the engine speed is maintained at medium levels and the torque at mid-to-high levels, indicating optimized fuel consumption. However, a key difference emerges: since the engine power closely follows the motor power, which depends on the driving cycle, the engine operates across a wider range of power levels. As a result, the engine exhibits a broader range of both torque and speed, which contributes to the optimization of battery lifespan.

```
[fig6, t5] = engSpdTrq(prob_aging, veh, time);
```





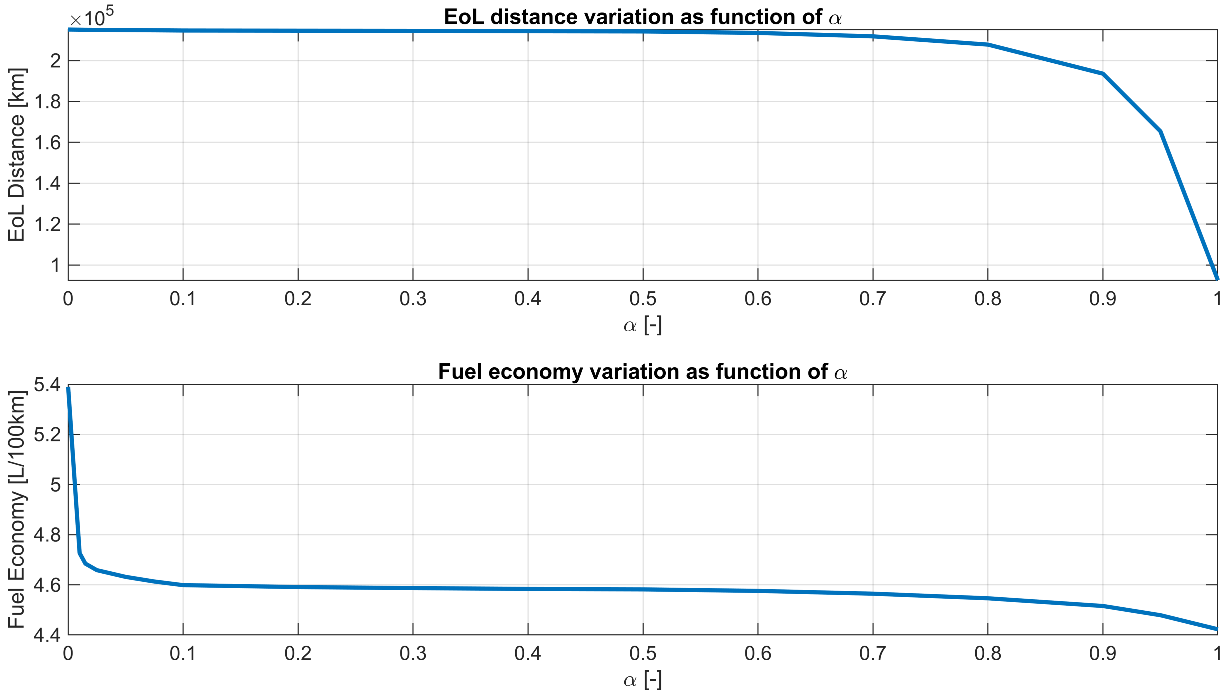
In conclusion, the analysis demonstrates that battery aging is effectively reduced, thereby confirming the correct implementation of the Energy Management Strategy. Moreover, fuel consumption does not significantly increase, remaining close to the value obtained in the first simulation where only fuel consumption was optimized. Moreover, another beneficial effect emerged from incorporating battery aging into the cost function: the State of Charge remains constrained within a narrower range. This tighter SoC regulation can further improve battery life, as smaller SoC variations, especially when maintained closer to mid-range values, reduce battery degradation. However, despite this advantage in the second simulation, it is not possible to quantify the extension in driving range until battery End of Life, since the current control strategy does not include a model to predict how SoC range tightening impacts battery lifespan. Implementing such a predictive model could be a valuable future development to enhance the accuracy and real-life relevance of the simulation.

The following final sections focus on analyzing how the trade-off parameter affects the EMS and the balance between fuel consumption and the battery's lifespan. Additionally, it examines how the parameter's value was selected based on a cost analysis.

The following charts present a sensitivity analysis conducted to show how the trade-off parameter influences both fuel economy and battery EoL distance. This analysis was performed asynchronously, with the results subsequently loaded and plotted in the current online simulation. The first observation concerns the results obtained at the boundary values of the trade-off parameter. As previously mentioned, a value of $\alpha = 1$ corresponds to minimizing fuel consumption only, meanwhile $\alpha = 0$ corresponds to optimizing only for battery degradation. The results align with these expectations: for $\alpha = 1$, the minimum fuel consumption is achieved, but the EoL distance is also at its minimum; conversely, for $\alpha = 0$ the opposite trend is observed. The validity of this analysis is further confirmed by the fact that when$\alpha = 1$, the fuel economy and battery EoL distance results match those of the first simulation, where only fuel consumption was minimized.

Another notable characteristic is that both fuel economy and EoL distance exhibit a plateau across most of the feasible range of the trade-off parameter, with sharp variations occurring only near the extreme values. Specifically, fuel economy rapidly rise as α approaches 0, while EoL distance decreases sharply when approaching 1. This highly non-linear behavior allows the selection of a trade-off parameter that significantly extends the battery's EoL distance without substantially compromising fuel economy, as also observed in the analysis of the second simulation results. This result highlights a key insight: incorporating an aging cost into the EMS substantially increases battery lifetime, achieving this benefit without causing a significant reduction in fuel economy.

```
EoLvariation = load('alpha_EOL.mat');

figure;
set(gcf, 'Position', [0 0 1600 800]);
tiledlayout(2,1);
nexttile;
plot(EoLvariation.alpha, EoLvariation.distance_eol, LineWidth=2);
title('EoL distance variation as function of \alpha');
xlabel('\alpha [-]');
ylabel('EoL Distance [km]');
grid on;
fuelEconomyVariation = load("alpha_FuelEconomy.mat");
nexttile;
plot(fuelEconomyVariation.alpha, fuelEconomyVariation.fuelEconomy, LineWidth=2);
title('Fuel economy variation as function of \alpha');
xlabel('\alpha [-]');
ylabel('Fuel Economy [L/100km]');
grid on;
```



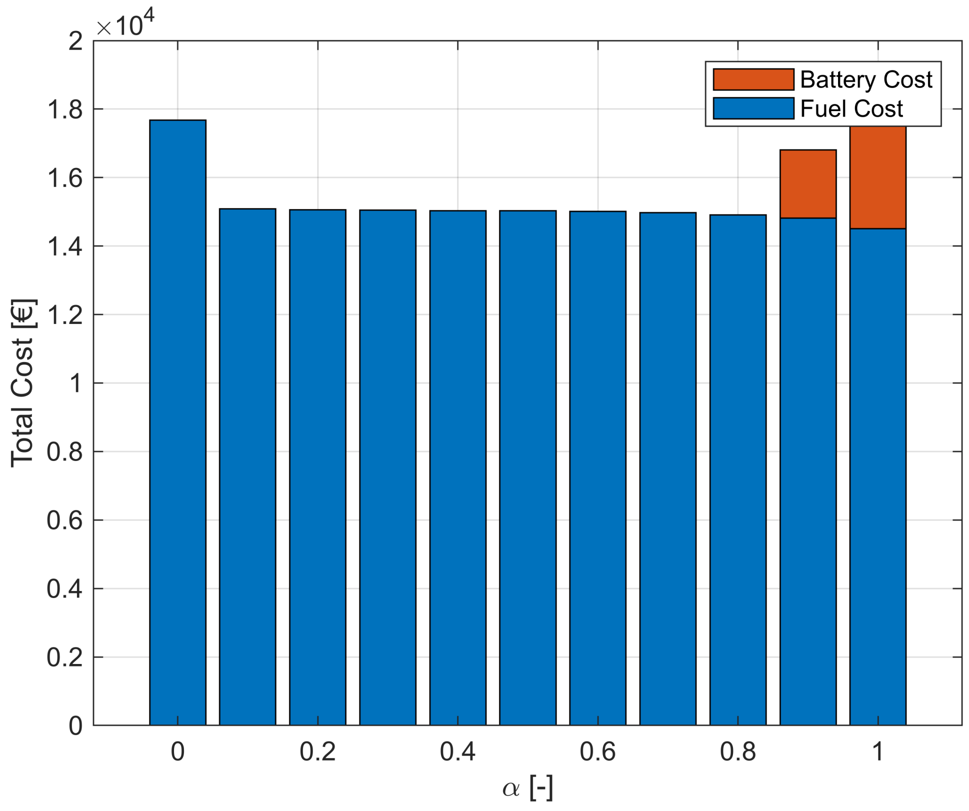
To determine the optimal trade-off coefficient value, a cost analysis was performed considering the total usage cost of the HEV for the end user over the vehicle's average life cycle. In particular, for different values of the trade-off coefficient, the total cost in euros, comprising the cost of the fuel and the one of a battery replacement, if needed, was calculated. To carry this simulation, different parameters based on Italian averages are considered. These parameters, along with their respective values, are summarized in the table below which includes the average fuel price, the average cost of a Li-ion battery replacement with the same capacity as the vehicle used in this project, and the average distance traveled by a HEV before being decommissioned.

Fuel Price	1.64 €/L
Battery Price	2'000 €
HEV EoL distance	200'000 km

The results of the ownership cost for different values of α are shown in the chart below. From this plot, it can be concluded that the value of the trade-off parameter that minimizes the total cost of ownership for the end user is $\alpha = 0.8$, the same value used in the second simulation which was chosen based on this cost analysis. However, due to the plateau trend discussed earlier, changes in the trade-off parameter within the mid-range have only a minor impact on the total ownership cost. Additionally, it is worth noting that battery replacement is required in only two cases, both occurring when the trade-off parameter is set to high values. This result aligns with expectations, as a larger value of the trade-off parameter places greater importance on fuel consumption optimization rather than on improving battery aging.

It should also be noted that the powertrain architecture modeled in this project is equipped with a small-capacity battery, which means its cost is relatively low. If the simulation were performed on a PHEV instead of an HEV, the battery cost would increase significantly, resulting in a much higher total cost of ownership for EMS configurations that require battery replacement due to a high trade-off parameter value. This last consideration highlights that the optimal value of α is not fixed for all operating conditions. On the contrary, it depends on various factors such as the type of vehicle and driving conditions. Indeed, keeping the same trade-off parameter used in this project but simulating the vehicle under different driving missions would yield different results. For example, a more aggressive driving cycle would lead to higher battery degradation and a shorter lifespan. Therefore, it is reasonable to conclude that a lower value of α is required. Conversely, a less aggressive cycle could extend battery life, allowing α to be set at higher values to further reduce fuel consumption.

```
totalCost = load('totalCost.mat');
fig7 = totalCostHistogram(totalCost);
```



References

This section lists the publications that provide the average values used in the trade-off analysis:

https://www.mimit.gov.it/it/prezzo-medio-carburanti/regioni

https://www.beechmonttoyota.com/service/service-tips-tricks/hybrid-battery-cost/

Daniele Candelaresi, Antonio Valente, Eleonora Bargiacchi, and Giuseppe Spazzafumo. *Life cycle assessment of hybrid passenger electric vehicle.*