

This is the author's copy of the publication as archived with the DLR's electronic library at <http://elib.dlr.de>. Please consult the original publication for citation, see e.g. <https://www.springer.com/series/7393>.

Learning-based Control for Hybrid Battery Management Systems

J. Mirwald, R. de Castro, J. Brembeck, J. Ultsch and R. E. Araujo

Battery packs of electric vehicles are prone to capacity, thermal, and aging imbalances in their cells, which limit power delivery to the vehicle. In this chapter, a hybrid battery management system (HBMS), capable of simultaneously equalizing battery capacity and temperature while enabling hybridization with supercapacitors, is investigated. We use model-free reinforcement learning to control the HBMS, where the control policy is obtained through direct interaction with the system's model. Our approach exploits the soft actor-critic algorithm to handle continuous control actions and feedback states, and deep neural networks as function approximators. The validation of the proposed control method is performed through numerical simulations, making use of numerically efficient models of the energy storage and power converters developed in Modelica language.

Copyright Notice

©2020 Springer Nature Switzerland AG. Personal use of this material is permitted. Permission from Springer Nature Switzerland AG must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

This is a pre-print of the following chapter:

J. Mirwald, R. de Castro, J. Brembeck, J. Ultsch and R. E. Araujo, "Learning-based Control for Hybrid Battery Management Systems", published in *Intelligent Control and Smart Energy Management: Renewable Resources and Transportation*, Springer Optimization and Its Applications, edited by P. M. Pardalos, M. T. Thai, D. Du. Cham, Switzerland: Springer, 2021.

Reproduced with permission of Springer Nature Switzerland AG. The final authenticated version is available online at: tbd.

Learning-based Control for Hybrid Battery Management Systems

Jonas Mirwald*, Ricardo de Castro*, Jonathan Brembeck*, Johannes Ultsch*, Rui Esteves Araujo **

*Institute of System Dynamics and Control, Robotics and Mechatronics Center, German Aerospace Center (DLR), 82234 Weßling, Germany; firstname.lastname@dlr.de

**INESC TEC and Faculty of Engineering, University of Porto, Porto, 4200-465, Portugal

1. Abstract

Battery packs of electric vehicles are prone to capacity, thermal, and aging imbalances in their cells, which limit power delivery to the vehicle. In this chapter, a hybrid battery management system (HBMS), capable of simultaneously equalizing battery capacity and temperature while enabling hybridization with supercapacitors, is investigated. We use model-free reinforcement learning to control the HBMS, where the control policy is obtained through direct interaction with the system's model. Our approach exploits the soft actor-critic algorithm to handle continuous control actions and feedback states, and deep neural networks as function approximators. The validation of the proposed control method is performed through numerical simulations, making use of numerically efficient models of the energy storage and power converters developed in Modelica language.

2. Introduction

Electric vehicles (EVs) are currently seen as a key technology for sustainable transportation. They enable the integration of renewable energies with transportation systems and provide a promising avenue to reduce environmental impact [1]. However, to fulfill these goals, several challenges need to be addressed. One challenge is the parameter variations of large battery packs, which occurs due to manufacturing tolerances and non-uniform aging of battery cells. These variations introduce the so-called weakest-cell problem, i.e., the performance of the battery pack is limited by the cell with the largest thermal and capacity degradation [2]. To overcome these issues, active balancing systems, capable of equalizing charge and temperature, have been developed [3]. Another challenge in the design of battery packs lies in the selection of battery chemistries which offer simultaneously high energy density, high power density and long life. To attenuate these issues, hybrid and modular energy storage systems, composed of heterogeneous units, have been investigated [4]. One promising research avenue deals with battery-supercapacitor hybridization. Supercapacitors with high power density and durability are particularly suited to handle rapid power bursts, while battery packs with high energy density can provide average power during vehicle cruising. Numerous works have been utilizing these properties to reduce peak power loads, weight and stress of the battery (see, e.g., [5] [6] and references therein).

Spurred by these balancing and hybridization challenges, a new class of battery balancing architecture, called hybrid battery management system, was recently proposed by our group [7] (see Figure 1). The HBMS is capable of simultaneously equalizing battery capacity and temperature while enabling hybridization with additional storage systems, such as supercapacitors. Despite these attractive features, the HBMS poses numerous control issues, such as coordinating a large number of power converters, enforcing actuation and safety constraints and making trade-offs between multiple technical and economic objectives. Model-based controllers, such as [7] [8], are one possible approach to tackle these issues. These controllers rely on mathematical models that approximate and predict the behavior of the plant (energy storage and power conversion in the HBMS). However, these models, usually obtained from first principles, are not always easy to derive or parameterize. For example, batteries depend on complex chemical reactions, requiring involved partial differential equations or complicated approximations via electric-equivalent circuits [9], while power converters have switching and nonlinear behavior [10]. Constructing and parameterizing these representations is time consuming, subject to uncertainties and modelling mismatches, and often needs engineering insights to find a good

balance between complexity and accuracy. Additionally, the deployment of model-based controllers needs, in some cases, significant computational effort, especially if optimization-based approaches are used, which presents hurdles for execution in embedded systems.

Reinforcement learning (RL) offers an alternative design route to tackle some of these hurdles. Similarly to optimal control, the RL algorithm tries to optimize a policy with regard to a predefined reward function, encoding the control goal. In model-free RL this optimization is done solely based on observed states, actions and rewards during the repeated interaction with the environment, called training. One major advantage of using RL is the fact that simulation models (i.e., synthetic data) or even the real plant can be used for the training process, avoiding the need for a controller synthesis model. By using offline training against a simulation model, the computational effort is shifted to the design stage, where significantly higher computational power is available. The obtained control policy, usually a rather small multilayer neural network, can then be evaluated efficiently after deployment on the target platform.

RL has been applied to a wide variety of problems, ranging from self-driving cars, to finance and healthcare [11]. More recently, it has been considered for energy storage systems. For example, [12] used RL to search for battery electrolyte compositions that can reduce electric conductivity. Energy management of multiple energy storage systems (e.g. combining batteries, fuel-cells and/or supercapacitors), is another task that plays to RL strengths. In this case, RL is well suited to handle the stochastic uncertainties associated with future driving cycle information and to reduce computational effort when compared to optimal receding control approaches [13] [14]. Energy storage arbitrage in smart grids is another emerging problem tackled by RL. For example, [15] developed a Q-learning based algorithms to decide when to buy energy from the grid, accumulate it in the battery and re-sell it to the grid, while maximizing profit. In comparison with model-based solutions, these RL algorithms were able to increase profit margins of grid operators by more than 50%. RL is also being applied to derive control strategies for fast battery charging. For instance, [16] investigated the use of deep deterministic policy gradient (DDPG) algorithms to reduce the charging times of batteries in the presence of voltage and temperature constraints and parameter variations. One common element in these previous works is the treatment of the battery at a pack-level, i.e., where capacity and temperature variations of the modules/cells are neglected, and the battery operation is approximated by a single virtual cell. To the best of the authors' knowledge, the application of RL to control batteries at module- or cell-level has received less attention in the literature to date (particularly in HBMS), which represents a research gap that this work addresses.

The main contribution of this work consists in evaluating the potential of RL algorithms to control HBMS. To support this investigation, we develop a numerical efficient simulation model of the HBMS in Modelica language. Particular attention is dedicated to reducing numerical complexity of the HBMS model, which is instrumental to decrease the training times of the RL algorithm - one of the main hurdles when applying this method in practice. This efficient simulation model is then used to train a control policy for the HBMS based on a soft actor-critic (SAC) algorithm [17] [18]. The application of SAC algorithms brings two advantages: i) SAC is able to handle continuous control actions and feedback states; ii) SAC offers a sample-efficient learning, thanks to its ability to automatically adapt exploration control policies during training, often requiring less training time than other RL algorithms previously employed in the control of energy storage systems, such as Q-learning [15] and DDPG [16].

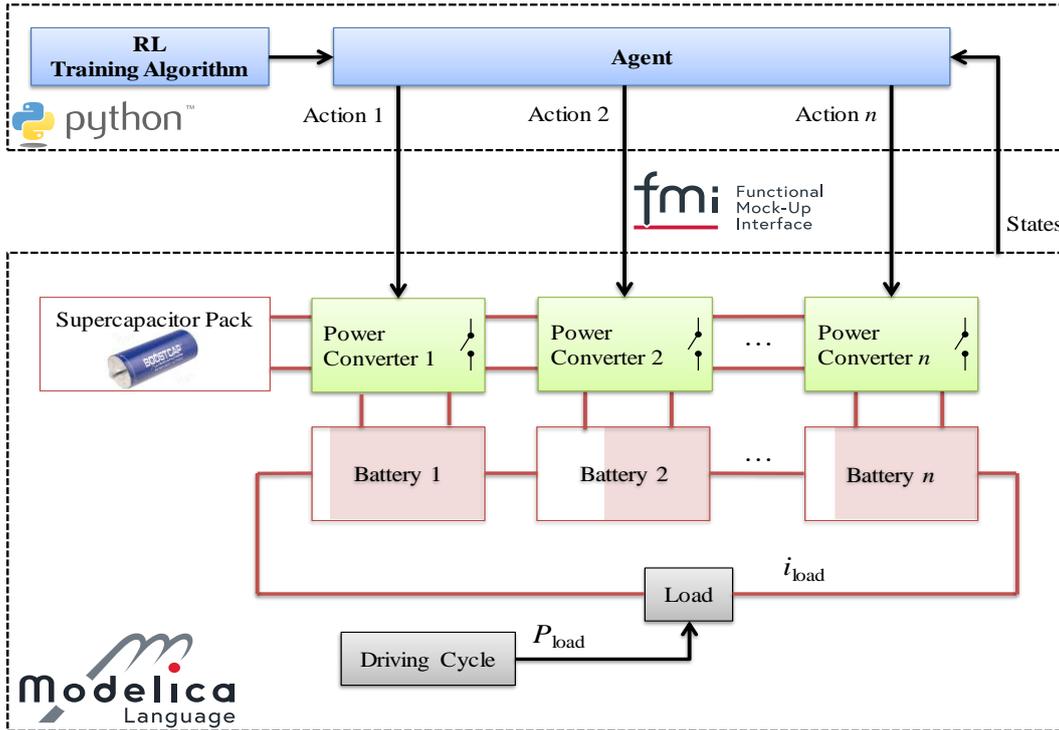


Figure 1: Block Diagram of the hybrid balancing system and the RL-based controller.

3. HBMS Modeling

3.1. Overview

As depicted in Figure 1, the HBMS is composed of a battery pack, supercapacitors (SCs), power conversion and the load, which emulates the power consumption of the electric vehicle. The battery pack contains n single-cell modules, connected in series. Each cell is linked with the primary side of a bi-directional DC/DC power converter, while the converter's secondary side is connected with SCs. The power converters enable the cell-to-cell and the cell-to-SC transfer of energy. Our goal is to design a RL-based control algorithm that uses the power converters to equalize charge and temperature, while reducing current stress in the cells.

To support the design of the RL algorithm, we present in this section the modeling environment for the HBMS. This modeling is carried out in Modelica, an object orientated, acausal and open source language [19]. Modelica allows us to integrate different physical domains, such as electrical and thermal, in the HBMS model. Its object-oriented features also enable us to automatize the creation of multiple instances of battery cells and power converters. As a result, HBMS models with different dimensions (ranging from a few cells to hundreds of cells) can be created with reduced coding effort. Additionally, Modelica's HBMS model can be exported through the functional mock-up interface (FMI) standard [19] and combined with other simulation environments, such as Python [18], for training RL agents.

3.2. Power Conversion

This sub-chapter presents the principle of operation and modelling of the power converter, with particular emphasis on the efficient computation of energy losses.

3.2.1. Dual Half Bridge Converter

The power balancing hardware relies on a dual half bridge (DHB) configuration. In addition to galvanic isolation, DHB offers zero-voltage switching, bidirectional power flow, and a reduced number of

switches when compared to dual full bridge configurations [20]. As depicted in Figure 2 the DHB interfaces with the battery module (v_b) on the primary side and with the SCs pack (v_{sc}) on the secondary side. The DHB is composed of four main components: two input inductors (L_b, L_{sc}), two half bridges (S_1, S_2 and S_3, S_4), auxiliary capacitors (C_1, \dots, C_4) and a high-frequency transformer. The two half bridges, and their four switches, regulate the power flow between primary and secondary sides of the converter. This power is transferred via the high-frequency transformer and its leakage inductor (L_s). Through pulse-width modulation of the switches, two square voltage waveforms, shifted by ϕ [rad] and with switching frequency f_{sw} [1/s], are generated by the two half bridges and applied to the terminals (v_{ab}, v_{cd} in Figure 2) transformer. The resulting switching scheme is discussed in more detail at the end of this section. The power extracted from the battery module is given by [21]:

$$P(\phi) = v_b i_b = \frac{\phi(\pi - \phi)v_b^2}{\pi 2\pi f_{sw} L_s} \quad (1)$$

which reveals that the maximum power flow is achieved when the phase shifts ϕ reaches $\pi/2$. To further facilitate the control of power, we included an inner current loop in the DHB. This loop is based on a proportional plus integrational control law and regulates the input current (i_b) via manipulation of the phase shift (ϕ). The interested reader is referred to [21] for additional details.

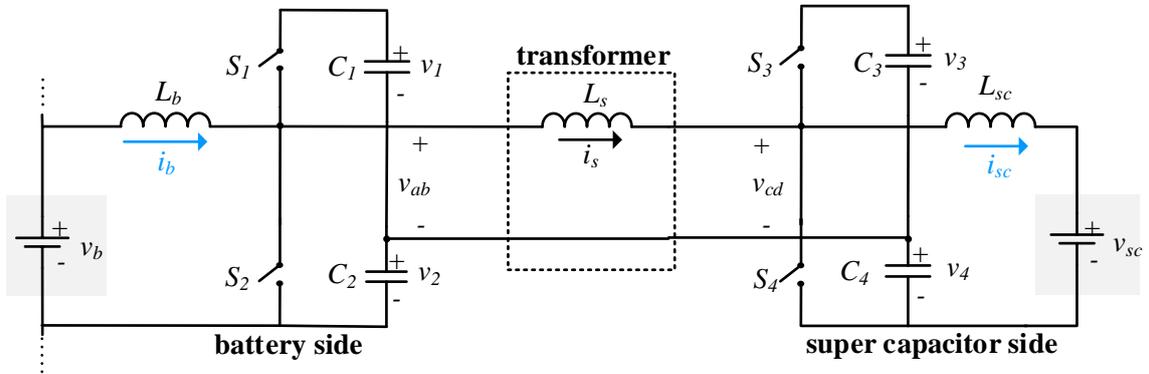


Figure 2: Schematic configuration of the dual half bridge.

We developed a detailed Modelica model of the DHB, incorporating energy losses. In Figure 3 (top) the dual half bridge converter with a high frequency transformer is shown in a typical Modelica diagram layer. All components are described in an a-causal representation with flow and potential variables (i.e. voltage and current in case of electric systems – blue components). The red interfaces capture the thermal flow between the components, and the magenta indicate Boolean control signals of the switches. In Figure 3 (bottom) the sub-model of the half bridge, which is identical on both sides of the DHB, is shown. The half-bridge model implements a DC-to-AC voltage conversion, and contains copper and switching losses, as well parasitic elements, such as inductors' resistances. The switches are implemented through MOSFET transistors and an anti-parallel freewheeling diode, based on the Infineon IPB009N03LG. The control strategy must prevent that the Boolean signals $fire_p$ and $fire_n$ are true at the same time to avoid short circuits. Thanks to Modelica's object-oriented design paradigm, the DHB can be constructed using two instances of the half-bridges- see Figure 3 (top). The value of the DHB parameters ($L_b, L_{sc}, C_1, \dots, C_4$, etc) can be found in [21]- Table 5.3.

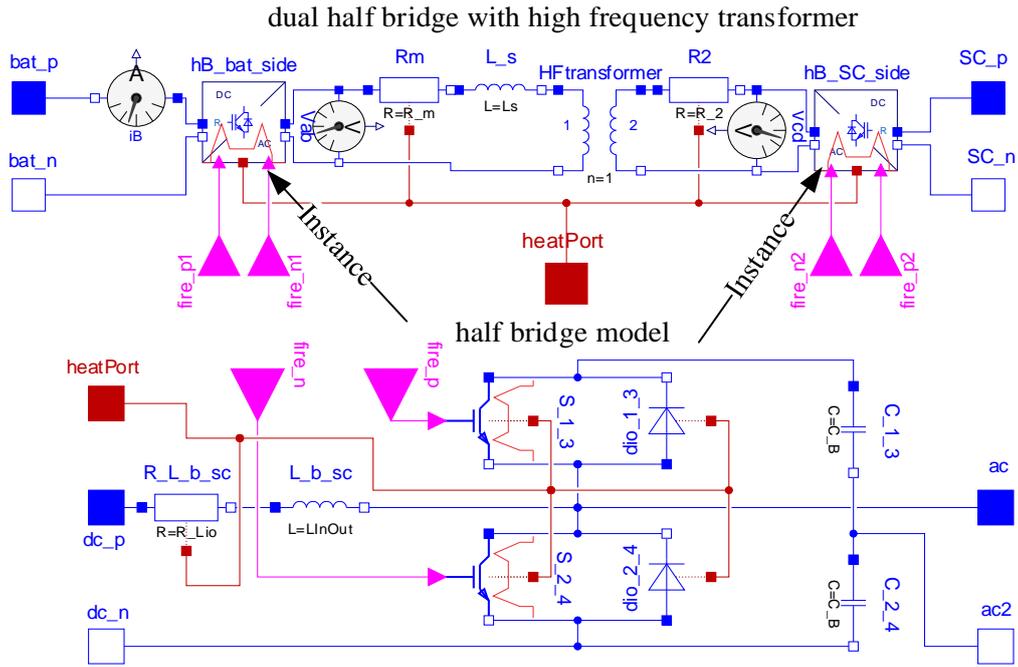


Figure 3: Modelica Model of the Dual Half Bridge Converter.

Figure 4 shows the simulation results of the DHB when operating with constant phase shift of $\phi = 0.3 \cdot \pi/2$ and voltages $v_b = 5.4 \text{ V}$, $v_{SC} = 4.5 \text{ V}$. The first plot shows the two MOSFET *fire* signals. These signals are shifted by ϕ and generate square voltages (v_{ab} , v_{cd}) in the half-bridges (second plot). The square voltages are applied to the transformer, leading to a large variation in the transformer's leakage inductor (third plot). The fourth plot shows the currents in the battery and SCs, which are filtered by the DHB's inductors. Finally, the last plot illustrates the power losses, which are discussed in the detail in the next sub-section.

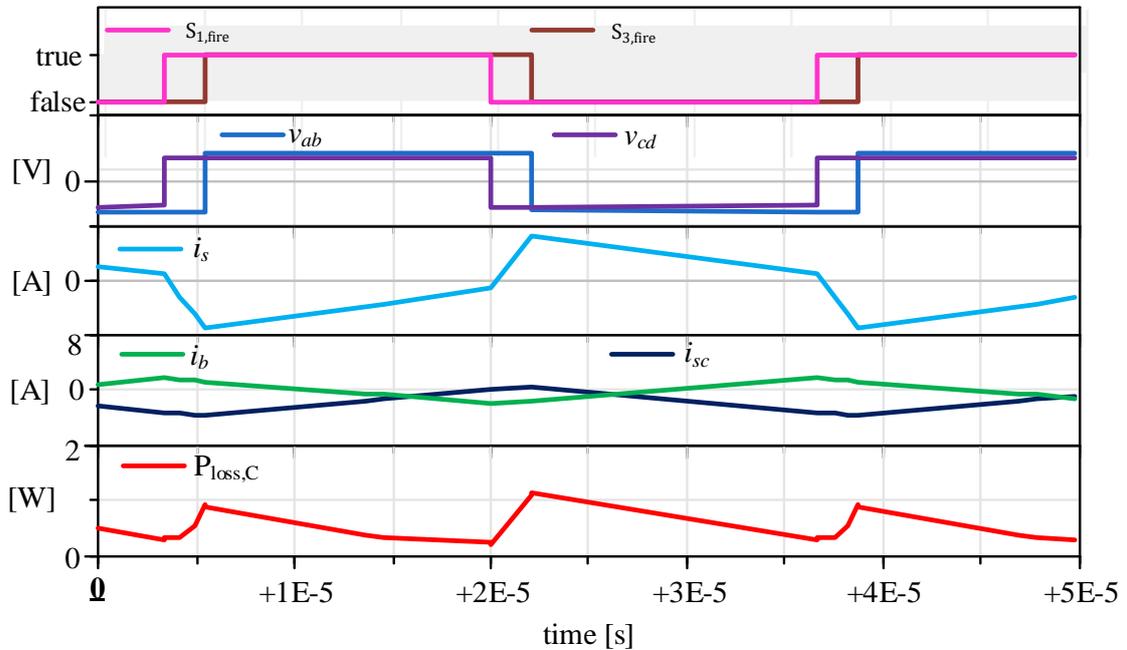


Figure 4: Switching signals, voltages, currents and power losses of the DHB, obtained with phase shift of $\phi = 0.3 \cdot \pi/2$.

3.2.2. Energy Losses in the MOSFETs

This section models the switching (P_{on}, P_{off}) and conduction (ohmic) losses (P_{Ω}) of the MOSFETs. As a starting point, we considered the ideal switch¹ with closed resistance (R_D) and opened conductance losses from the Modelica Standard Library [19]. This model was extended with additional elements to capture the switch-on and switch-off losses. To better understand how these losses are modelled, let us consider a single MOSFET (Figure 5).

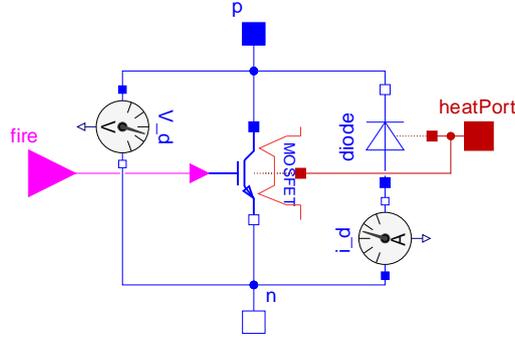


Figure 5: Single MOSFET with diode and measurements of V_D and I_D .

During the switch-on phase, the current I_D of the MOSFET/diode rises linearly, while the blocking voltage V_D falls close to zero, leading to a triangular shaped power loss (see Figure 6 and [22] for details). A similar pattern also appears during the switch-off phase. Since the raise (t_{on}) and fall (t_{off}) times are usually very small (in order of nanoseconds), the numerical simulator needs small integration steps to accurately represent the “triangular” power losses. This causes a significant slowdown of the numerical simulation, which is further aggravated by the high number of events generated by the switching frequency of the MOSFETs (usually in the order of kHz). To improve numerical efficiency, we decided to develop a more pragmatic approach for modelling the switching losses. We assume that the energy lost during the switching is continuously dissipated in the “on” and “off” state, respectively (see Figure 6). This equivalent power loss depends on the switching frequency f_{sw} , and has as a rectangular shape with duty cycle D . It is defined as:

$$P_{on} = \begin{cases} \left(\frac{V_{D_{pre}} \cdot I_{D_{post}} \cdot t_{on} \cdot \frac{f_{sw}}{2}}{D} \right) & S_{j,fire} = \text{true} \\ 0 & S_{j,fire} = \text{false} \end{cases} \quad (2)$$

$$P_{off} = \begin{cases} \left(\frac{V_{D_{post}} \cdot I_{D_{pre}} \cdot t_{off} \cdot \frac{f_{sw}}{2}}{1 - D} \right) & S_{j,fire} = \text{true} \\ 0 & S_{j,fire} = \text{false} \end{cases} \quad (3)$$

The lower scripts $(\cdot)_{pre}$ and $(\cdot)_{post}$ denote the point in time before (pre) or after (post) the switching event when the value is recorded (see Figure 6). Although the time-domain representation of the “triangular” and the “rectangular” power losses differ, the accumulated energy losses of these two models is the same. In other words, the area under the “rectangular” power losses is equal to the area under the “triangular” power losses (see the green and red areas represented in Figure 6).

¹ Modelica.Electrical.Analog.Interfaces.IdealSemiconductor

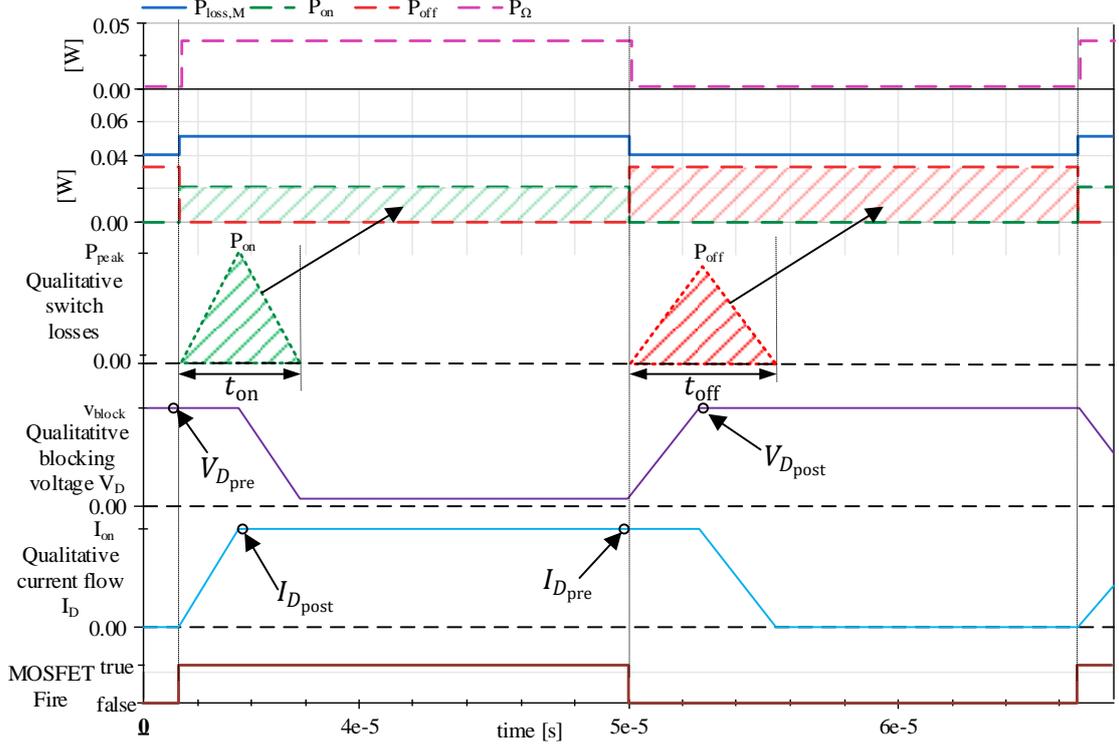


Figure 6: Approximated MOSFET with switching and conduction (ohmic) losses.

Thanks to the “rectangular” approximation, we are able to avoid very small integration time steps that would be necessary if the “triangular” power losses were modeled in detail. This brings an important practical benefit: the simulation time of the HBSM can be reduced. The overall losses in the MOSFET simulation model ($P_{\text{loss},M}$) are therefore the sum of switching and conduction losses:

$$P_{\text{loss},M} = P_{\Omega} + P_{\text{on}} + P_{\text{off}} \quad (4)$$

where P_{Ω} captures the Joule losses dissipated in the MOSFET resistance ($R_D I_D^2$).

3.2.3. Quasi-Stationary Model of the DHB

Despite the above simplifications, the simulation of the DHB model still generates a large number of events. For example, for a test simulation run with constant voltage on the battery and SC sides and a total simulation time of 5.7 s, the Modelica model of the DHB still needs a calculation time of about 421 s and generates $\sim 1.3 \cdot 10^6$ events. This yields a poor real-time factor of 0.014 (DASSL² solver, Intel i7-8665U, 16 GB RAM, NVMe SSD).

To further reduce the simulation time, we decided to develop a quasi-stationary DHB model. This model captures the main time constants of the DHB and calculates the energy losses and actuation limits through multi-dimensional look-up tables. To parameterize these tables, we defined the following operational range: i) voltage in the battery side, $v_b \in [2.7 \text{ V}, 4.5 \text{ V}]$, ii) voltage in the SC side $v_{\text{sc}} \in [2.7 \text{ V}, 5.4 \text{ V}]$ and iii) phase shift $\phi \in [0, \pi/2]$. By means of parallelized parameter sweeping simulations with an equidistant grid, all relevant data at the operations point could be generated. In Figure 7 the resulting power losses for different operating points are shown; this allow us to build a smooth multi-dimensional look-up table for the DHB power losses, $P_{\text{loss},C} = f(i_b, v_b, v_{\text{sc}})$, which depends on the battery/SC voltage and input current (i_b). Note that $P_{\text{loss},C}$ includes not only the

² DASSL is a variable-order, variable-step numerical integration method

MOSFETs losses, but also power losses in the other components of the DHB (such as inductor and transformer).

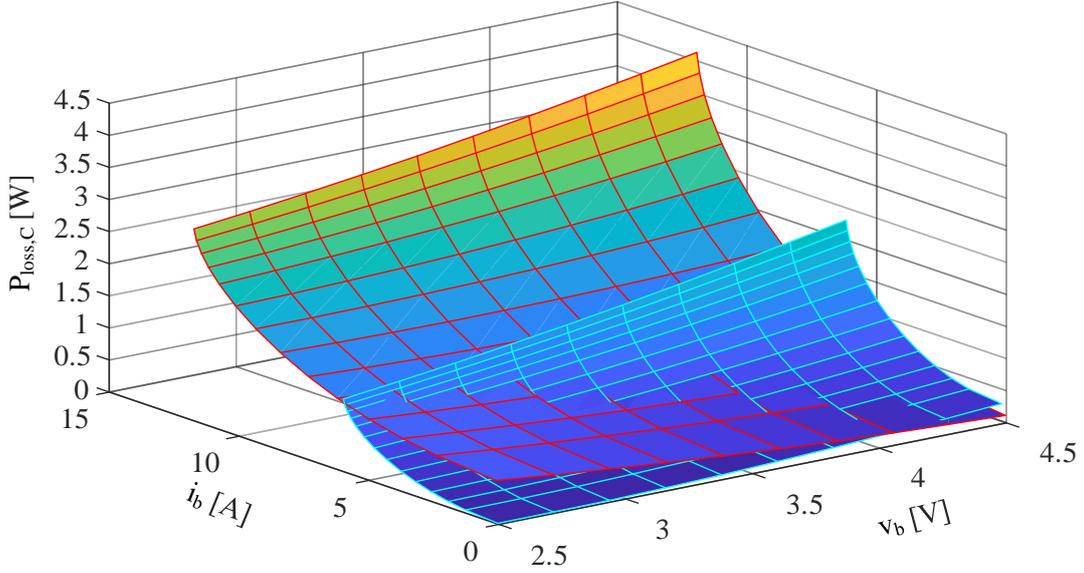


Figure 7: DHB losses at $v_{sc} = 2.7$ V (cyan edges) and 5.4 V (red edges). Note: the power losses for negative balancing currents ($i_b < 0$) are symmetrical and are not shown in this plot.

The DHB maximum power and input current (i_b) are subject to physical limits. For example, the (theoretical) maximum power that the DHB can transfer is obtained when the phase shift reaches $\phi_{\max} = \pi/2$, as expressed in (1). In practice, the maximum power is also affected by the DHB power losses. Since these losses are difficult to define analytically, we constructed a look-up table for determining the maximum transferable power and maximum input current i_b^{\max} , using the same approach as described in the previous paragraph.

The dynamic response of the DHB is dominated by the input current controller. The goal of this controller is to manipulate the phase shift ϕ such that the input current i_b accurately follows the reference i_b^* (generated by the RL agent). The closed-loop response of this controller – which is designed using linear methods [23] – is approximated here by a second-order filter ($G_i(s)$) with critical damping and cut-off frequency $f_{\text{cut}} = 65$ Hz. The resulting quasi-stationary Modelica model of the DHB is shown in Figure 8. Both sides of the DHB converter are modeled through current sources. The current source on the battery side is controlled through the second-order filter, i.e., $i_b = G_i(s)i_b^*$. The current source on the SC side is controlled through a power-balance constraint: $i_{sc} = (v_b i_b - P_{\text{loss},C})/v_{sc}$. Finally, the power losses $P_{\text{loss},C}$ are computed from a look-up table and transferred to a Modelica heat port.

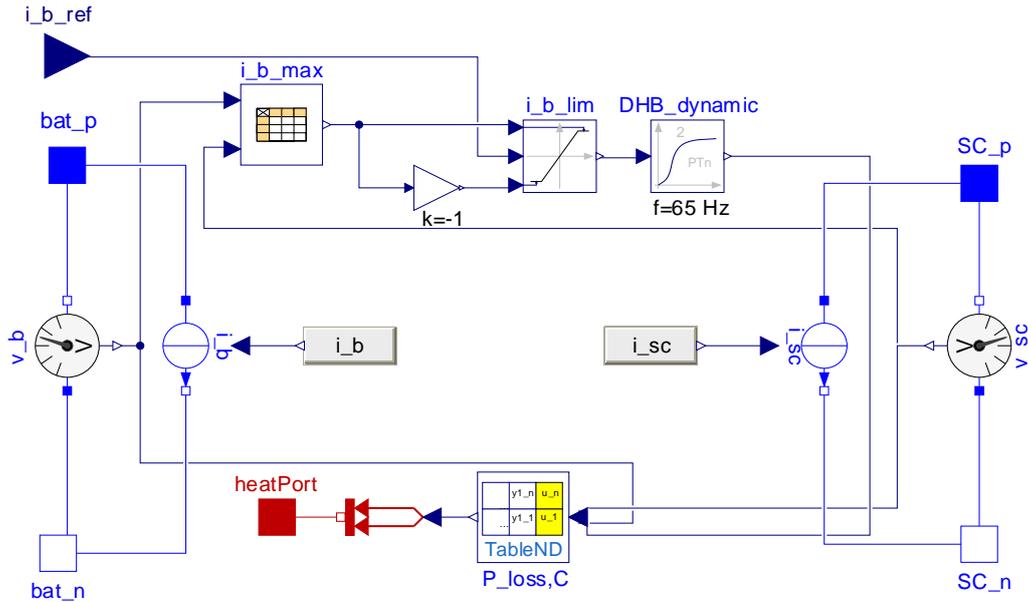


Figure 8: Quasi-stationary dual half bridge model in a mixed diagram and equation representation – Note: $i_{b_ref} = i_b^*$.

The quasi-stationary DHB model provides a real-time factor of ~ 44.5 , which is an improvement of more than $3.2 \cdot 10^5\%$ when compared to the switching DHB model presented in the previous section. Figure 9 depicts the power losses obtained with the two modelling approaches using balancing currents determined in [10]. The percentage goodness of fit (cf. [24] – C.5) of the power losses is about 84%, which is an acceptable value considering the substantial improvement of the real-time simulation factor obtained with the quasi-stationary DHB model. Because of these features, the quasi-stationary DHB model was used in the training of the RL agent.

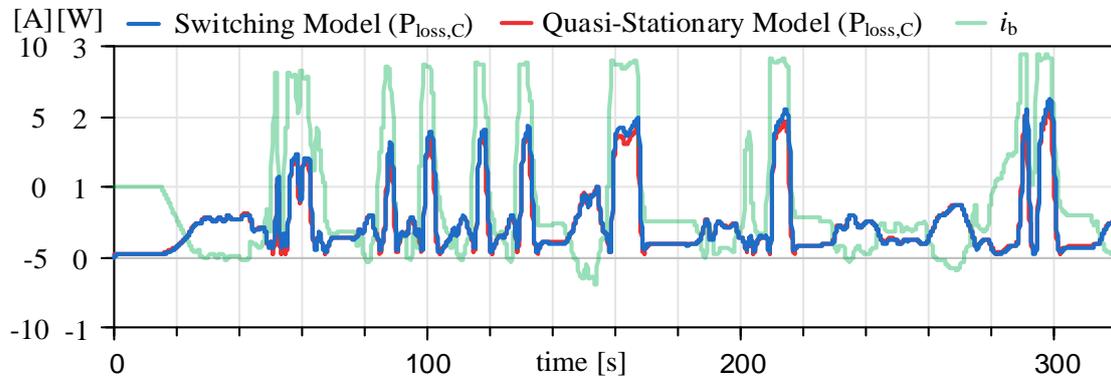


Figure 9: Validation of the quasi-stationary DHB model.

3.3. Battery

This chapter discusses the battery model and its parameterization.

3.3.1. Battery Modeling

The battery model relies on the representation proposed in [25] and [26]. This model is modified to better match the lithium nickel manganese cobalt oxide (NMC) chemistry (Li-Tec HEI40 40 Ah [27]) cell that is used in our group's experimental vehicle, ROboMObil [28].

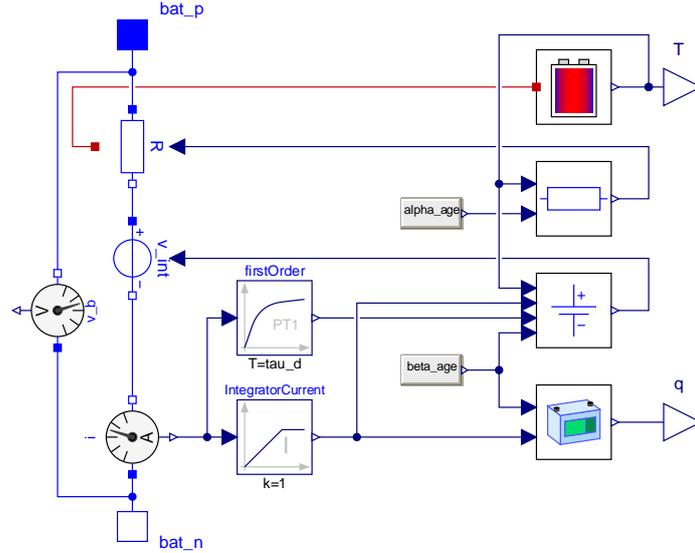


Figure 10: Implementation of the modified temperature-dependent generic battery model for NMC li-ion cells (simplification) with age-defining parameters α_{age} and β_{age} .

Figure 10 shows the Modelica implementation of the battery cell. Each cell is composed of an internal voltage source (v_{int}), equivalent series resistance (R), and a Coulomb counter for the state-of-charge (SoC). The internal cell's voltage is calculated based on a combination of the following terms (see Table 1):

- v_0 captures linear temperature effects in the open-circuit voltage.
- ($v_{p,fs}$, $v_{p,vs}$) are fixed/variable structure polarization voltages. The variable structure voltage ($v_{p,vs}$) has switching dynamics, which is dependent on the filtered current i_f and its sign. The filtered current i_f is calculated by applying a first-order filter to the cell current i with a time constant τ_d .
- v_e captures the exponential voltage variations when the cell approaches full charging conditions
- v_{11} is a linear voltage term, proportional to the battery charge $I(t) = I_0 + \int i(t)dt$, where I_0 is the initial charge.
- v_{12} is an additional term that improves voltage course fitting to the NMC type cells used in this work. This term provides a smooth transition to a second linear voltage term, which is activated when the charge reaches the value I_0 . This transition is implemented through a logistic sigmoid function $f(x) = \text{logsig}(x) = 1/(1 + \exp(-x))$ and its smoothness is adjusted via the parameter I_{norm} .

The equivalent series resistance R (see Equation (14) in Table 1) has a nominal value $R|_{T_{a,ref,1}}$, which is obtained at a reference temperature $T_{a,ref,1}$. This nominal value can be modified due to battery aging factors (α_{age}) and Arrhenius-based temperature effects [9]. To compute the cell's temperature, a heat exchange model ([24], Appendix C.2) is employed, which assumes that the heat transfer is dominated by Joule losses in the series resistance, $P_{loss,B} = R(T(t)) \cdot i^2(t)$. Finally, the cell's capacity is described by the variable Q (see (15) in Table 1), whose value can be reduced by the aging factor β_{age} . The description of all the parameters employed in the battery model is presented in Table 2.

Table 1: Battery model equations

Description	Equation	Number
Internal cell's voltage	$v_{\text{int}} = v_0 + v_{p,fs} + v_{p,vs} + v_e + v_{l1} + v_{l2}$	(5)
Thermodynamics voltage	$v_0 = v_0 _{T_{a,\text{ref},1}} + \frac{\partial v}{\partial T} \cdot (T(t) - T_{a,\text{ref},1})$	(6)
Fixed-structure polarization voltage	$v_{p,fs} = -K(T(t))I(t) \cdot \frac{Q}{Q - I(t)}$	(7)
Variable-structure polarization voltage	$v_{p,vs} = \left(-\frac{1}{3600}\right) \cdot K(T(t)) i_f(t) \cdot p_r(Q, I(t), i_f(t))$	(8)
Exponential zone voltage	$v_e = A \cdot \exp(-B \cdot I(t))$	(9)
Linear zone 1 voltage	$v_{l1} = (-1) \cdot C_{l,1} \cdot I(t)$	(10)
Linear zone 2 voltage	$v_{l2} = -C_{l,2}(I(t) - I_1) \cdot \text{logsig}\left(\frac{I(t) - I_1}{I_{\text{norm}}}\right)$	(11)
Polarization constant	$K(T(t)) = K _{T_{a,\text{ref},1}} \cdot \exp\left(\alpha_K \cdot \left(\frac{1}{T(t)} - \frac{1}{T_{a,\text{ref},1}}\right)\right)$	(12)
Variable-structure polarization ratio	$p_r = \begin{cases} \frac{Q}{\xi_p Q - I(t)}, & \text{if } i_f(t) \geq 0 \\ \frac{Q}{\zeta_p Q + I(t)}, & \text{else} \end{cases}$	(13)
Resistance	$R = (1 + \alpha_{\text{age}}) \cdot R _{T_{a,\text{ref},1}} \cdot \exp\left(\beta_R \cdot \left(\frac{1}{T(t)} - \frac{1}{T_{a,\text{ref},1}}\right)\right)$	(14)
Capacity	$Q = (1 - \beta_{\text{age}}) \cdot Q _{T_{a,\text{ref},1}}$	(15)
State-of-charge	$q = 1 - \frac{I(t)}{Q}$	(16)

The computational efficiency provided by this battery model is high. For example, a dynamic single-cell discharge simulation with a length of 1280 s, performed with a load as used during validation (cf. Section 5.3), only takes about 0.5 s, yielding a real-time factor of 2560. Because of this fast simulation time, no simplifications were performed in the battery model.

3.3.2. Battery Model Parametrization

The battery model was parameterized for NMC chemistry and used information from the cell's datasheet and experimental measurements [27]. The datasheet information allowed us to identify parameters such as the nominal cell voltage. The experimental data was used to identify the remaining parameters of the battery model through the following three-step optimization approach:

1. Experimental measurements from dynamic discharges (0°C and 25°C) were used to estimate the temperature-related parameters ($R|_{T_{a,\text{ref},1}}$, β_R). The squared temperature error was used as cost function and minimized through the Simplex optimization algorithm [29].
2. Experimental measurements from the constant-current charge/discharge references (0°C and 25°C) were used to estimate the parameters that affect the internal voltage (such as $\frac{\partial v}{\partial T}$, $K|_{T_{a,\text{ref},1}}$, α_K , $C_{l,1}$, $Q|_{T_{a,\text{ref},1}}$, $C_{l,2}$, I_1 and I_{norm}). The squared voltage error was used as cost function and minimized through a genetic algorithm [29].
3. The parameters obtained in step 2 were then used as initial values for a second optimization run using the Simplex algorithm, allowing us to further refine the parameter estimate.

Note that, during the parametrization, the cell is assumed to be at the beginning-of-life (BOL): $\alpha_{\text{age}} = \beta_{\text{age}} = 0$.

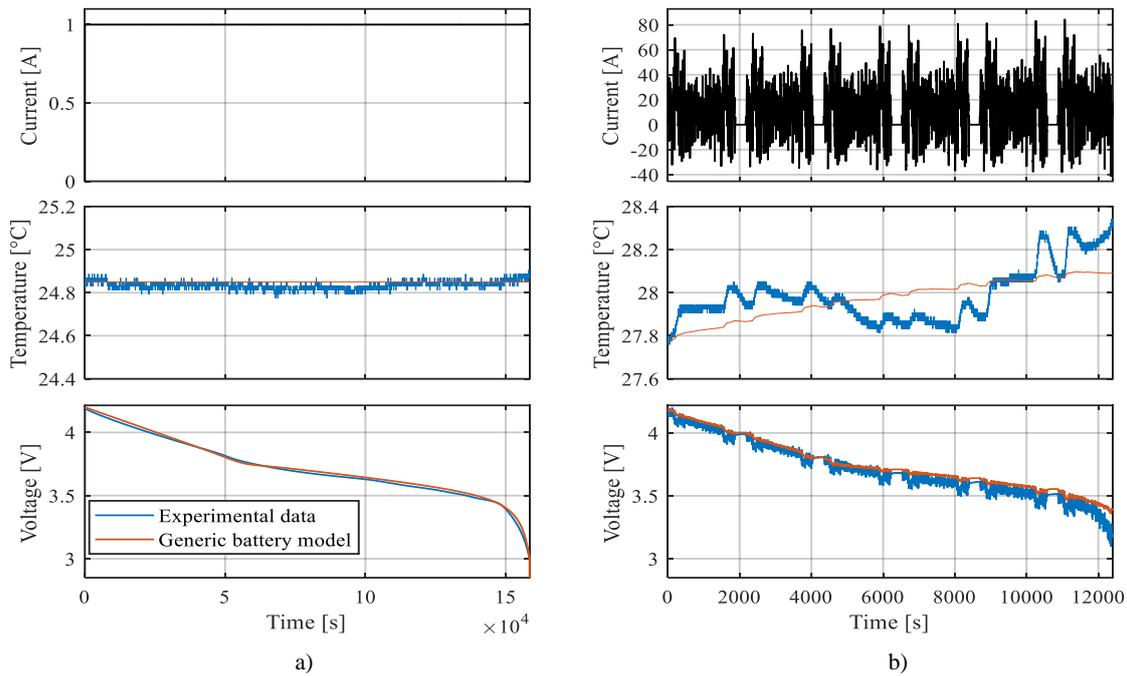


Figure 11: Optimization results of the temperature-dependent battery model for NMC li-ion cells for continuous discharge at 24.85°C ambient temperature (a) and dynamic discharge at 27.78°C ambient temperature (b).

Figure 11 compares the battery model output and experimental measurements. Overall, one can find a small error between the measured terminal voltage and the battery model, for both constant and dynamic current excitation. The dynamic excitation test presents a less smooth voltage behavior than the constant current test, which can be explained by the faster and higher current excitation employed in this test (cf. Figure 11 b). It is also worth noticing that the battery temperature did not change much during the tests. This can be explained by the large heat capacity of the cells. All optimized parameters of the battery model can be found in Table 2.

Table 2: Optimized battery parameters

Parameter	Value	Parameter	Value
$R _{T_{a,ref,1}}$	$2.14 \cdot 10^{-4} \Omega$	$Q _{T_{a,ref,1}}$	44.99 Ah
β_R	$3.24 \cdot 10^3 \text{ K}$	$(\xi_P, \zeta_P)^*$	(1,0.1)
$\frac{\partial v}{\partial T}$	$7.25 \cdot 10^{-4} \text{ V/K}$	A^*	0
$K _{T_{a,ref,1}}$	$2.50 \cdot 10^{-4} \text{ V/Ah}$	B^*	0
α_K	0 K	$v_0 _{T_{a,ref,1}}^*$	4.2 V
$C_{1,1}$	$7.74 \cdot 10^{-6} \text{ V/As}$	$T_{a,ref,1}^*$	24.85°C
$C_{1,2}$	$-5.44 \cdot 10^{-6} \text{ V/As}$	$T_{a,ref,2}^*$	-1.32°C
I_1	15.66 Ah	τ_d^*	30 s
I_{norm}	0.86 Ah		

* Parameter value extracted from datasheet or pre-fixed

3.4. Supercapacitors (SCs)

As the last component of the HBMS we discuss the modelling of the SCs. Two types of models were considered. The first (Figure 12, left) is based on a simple RC equivalent circuit composed of an ideal capacitance, equivalent series resistance and a self-discharge parallel resistance. The second (Figure 12 right) considers a more complex “two branches” equivalent circuit proposed in [30]; it is composed of: i) one RC branch, where the capacitance has a linear voltage dependence, ii) one RC branch with constant parameters, and iii) a parallel self-discharge resistance. In both models, the SCs energy loss $P_{loss,SC}$ is captured by heat dissipated in the resistances.

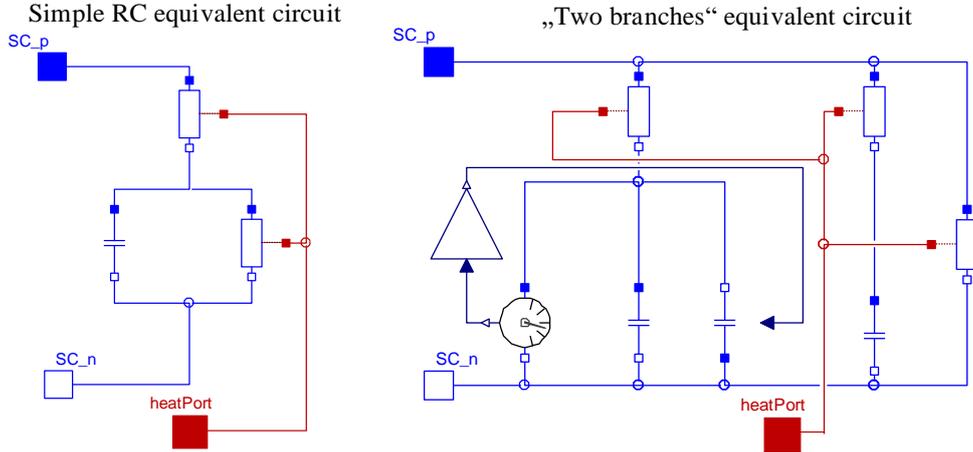


Figure 12: Modeling of the SCs with one (left) and two branches (right).

Figure 13 provides a comparison of the SC terminal voltage for both models during a charge/discharge test. The “two branches” model exhibits a non-linear behavior during the charging and discharging phases. Nevertheless, the voltage mismatch between both models is relatively small. Additionally, the simple RC model is about ten times faster to simulate than the more complex “two-branches” model. Since the simple RC model provide good accuracy and reduced simulation times, we decided to employ this model during the training of the RL agent.

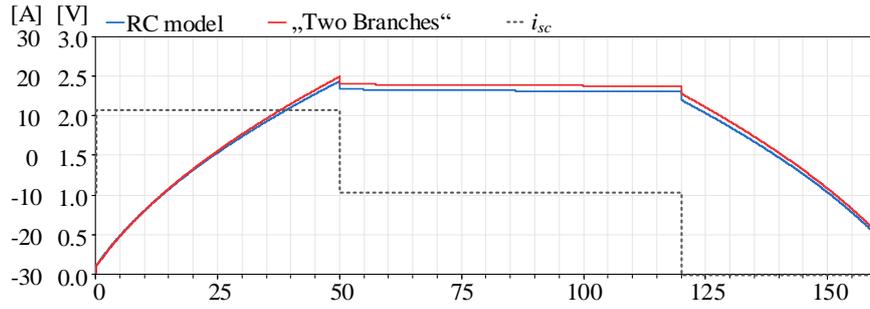


Figure 13: Comparison of the modeling with one or two branches (parameters extracted from [30]).

3.5. Load and Integration of Components

In what follows, we will consider a small-scale HBMS, composed of:

- a small battery pack with three NMC cells [27] in series (3s1p); this represents a 1-to-30 reduction factor with respect to the ROboMObil [28], the reference vehicle employed in this study and which contains a 90s1p full-scale battery pack
- a SC pack (2s4p), with a hybridization ratio γ of 2.3 % (i.e. ratio of energy accumulated in the SCs and the battery pack); this is in line with the sizing pattern adopted in [7]
- DHB converters with a maximum current of 11.8 A (cf. Figure 8); the parameters of the converter were extracted from [21].

The HBMS is also connected to a load that emulates the power consumption of the vehicle. This load power is computed using a Modelica drivetrain model of the ROboMObil [28] [31]. To ensure that the current requested from the battery cells (of the small-scale HBMS) reaches average values close to 1 C³, a 1-to-15 power re-scaling factor is used. Figure 14 illustrates the velocity and the load power (P_{load}) requested from the small-scale HBMS for the two driving cycles considered in this work.

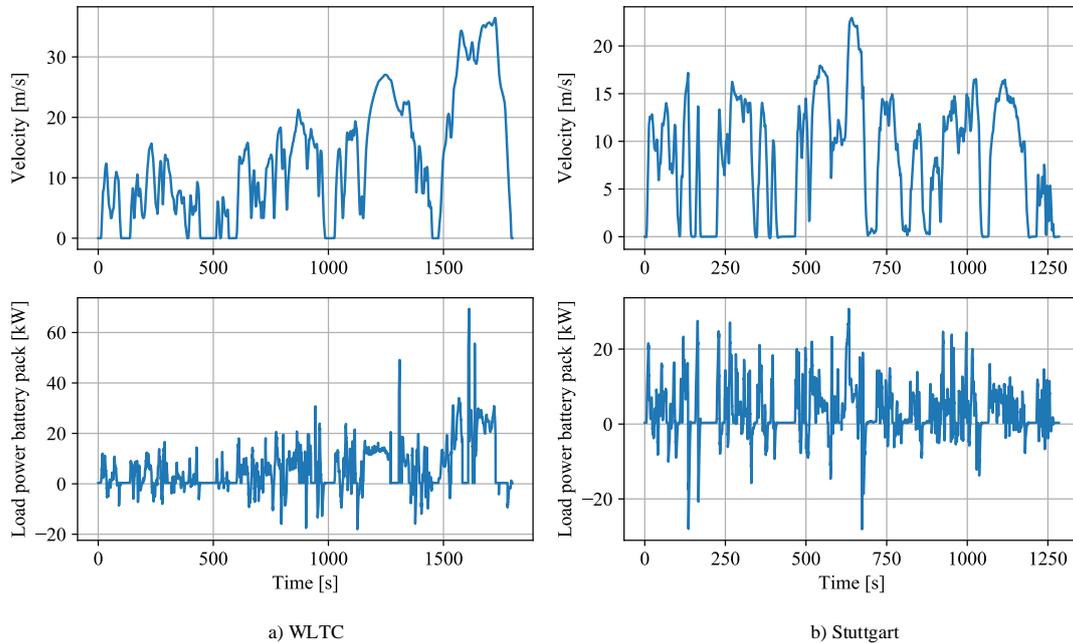


Figure 14: Velocity and load power (P_{load}) of the driving cycles employed in this work.

³ The C-rate of 1 C describes the current necessary to discharge the battery's nominal capacity during one hour.

4. Reinforcement Learning

This section presents the principle of operation of the selected reinforcement learning algorithm of this work as well as its application to the HBMS.

4.1. Principle of Operation

RL algorithms are developed assuming a stochastic environment, usually formulated as a Markov decision process (MDP). At time step k , the MDP receives an action $\mathbf{a}_k \in \mathcal{A} \subset \mathbb{R}^m$ and produces a state vector $\mathbf{s}_k \in \mathcal{S} \subset \mathbb{R}^n$, where \mathcal{A} and \mathcal{S} are the action and state spaces, respectively. The state evolves according to state-transition probability $p(\mathbf{s}_{k+1}|\mathbf{s}_k, \mathbf{a}_k)$, which denotes the probability of transitioning from state \mathbf{s}_k to state \mathbf{s}_{k+1} by taking the action \mathbf{a}_k [32]. After the transition to state \mathbf{s}_{k+1} , the reward r_{k+1} is assigned. This interaction, as depicted in Figure 15, leads to a trajectory $\mathbf{s}_k, \mathbf{a}_k, r_{k+1}, \mathbf{s}_{k+1}, \mathbf{a}_{k+1}, r_{k+2}, \dots$. A trajectory is called an episode, if a terminal state is reached and a reset to initial conditions occurs. The sum of rewards of an episode is referred to as the return.

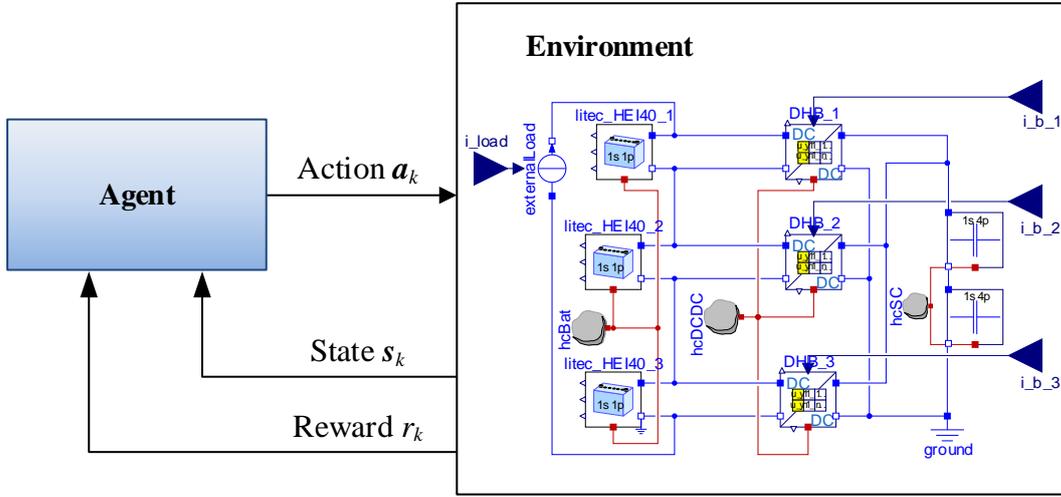


Figure 15: Interaction of reinforcement learning agent with environment [32].

The RL goal is to find an optimal policy $\pi^*(\mathbf{a}_k|\mathbf{s}_k)$ that maximizes the expected return J

$$J = \sum_k \mathbb{E}_{(\mathbf{s}_k, \mathbf{a}_k) \sim \rho_\pi} [r(\mathbf{s}_k, \mathbf{a}_k)] \quad (17)$$

with the expectation denoted by \mathbb{E} and $\rho_\pi(\mathbf{s}_k, \mathbf{a}_k)$ being the state-action marginal of the trajectory distribution induced by the stochastic policy $\pi(\mathbf{a}_k|\mathbf{s}_k)$ (cf. [17]). While trying to optimize the policy (exploitation) based on the observed transition trajectories, a certain randomness during the training is necessary to discover the full state and action space (exploration). Several types of RL algorithms can be used to minimize J . In the Soft Actor-Critic algorithm [17], which is adopted in this work, the standard RL objective (17) is augmented with an information-theoretical entropy term $\mathcal{H}(X)$ (cf. [33]):

$$\tilde{J} = \sum_k \mathbb{E}_{(\mathbf{s}_k, \mathbf{a}_k) \sim \rho_\pi} [r(\mathbf{s}_k, \mathbf{a}_k) + \alpha \mathcal{H}(\pi(\cdot|\mathbf{s}_k))] \quad (18)$$

where α represents a weight (known as temperature parameter), which allows us to balance exploration and exploitation. By starting the training with a high value of α , higher levels of exploration and discovery can be promoted. Afterwards, as the agent learns the impact of its actions in the environment and rewards, lower values of α can be deployed, shifting the focus toward exploitation [34]. Previous research has shown that the SAC algorithm offers a sample-efficient learning and outperforms other widely-used RL algorithms [34].

The SAC algorithm [34] is an off-policy actor-critic algorithm incorporating a replay buffer and neural networks to approximate the policy (actor) as well as two *action-value functions* (critic). The action-

value functions are used to account for positive bias in the policy improvement step which can deteriorate performance (cf. [35], [36]). Moreover, two additional target neural networks are used to stabilize training. In the SAC algorithm, the stochastic policy is usually parametrized as a gaussian with mean and variance given by the actor neural network. During training, the SAC alternates between performing environment steps and gradient steps. To perform an environment step, an action is sampled out of the gaussian policy and applied to the environment. Together with the returned reward and next state, the sampled action is then stored inside the replay buffer. During the gradient step phase the parameters of the policy, the two action-value functions and the temperature parameter are updated with stochastic gradient descent steps, which are computed based on uniformly sampled batches of the replay buffer. As a last step during the gradient step phase, the target networks' parameters are computed as an exponentially moving average of the parameters of the two action-value functions. The environment step phase and the gradient step phase are repeated until a certain amount of iterations is reached [35], [36]. After training the deterministic part of the policy is used for deployment.

4.2. Application of Deep Reinforcement Learning to the HBMS

To apply the RL to the HBMS, we need to specify: i) the actions (\mathbf{a}_k); ii) the state vector (\mathbf{s}_k) and iii) the reward function (r_k) that contains the control objectives.

4.2.1. Actions

The RL actions \mathbf{a}_k consist of the reference currents $i_{b,k,j}^*$ for each j -th power converter:

$$\mathbf{a}_k = [i_{b,k,1}^*, \dots, i_{b,k,n}^*]^T. \quad (19)$$

The actions are subject to saturation and rate limit constraints. As discussed in Section 3.2.3, the DHB saturates the maximum current $i_{b,k,j}^{\max}(q_{SC,k}, q_{k,j})$ that can be extracted from each battery cell. This saturation is modelled here as:

$$a_{k,j}^{\text{sat}} = \begin{cases} a_{k,j}, & \text{if } |a_{k,j}| \leq \eta_a \cdot a_{\max,k} \\ \eta_a \cdot a_{\max,k}, & \text{if } a_{k,j} \geq \eta_a \cdot a_{\max,k} \\ -\eta_a \cdot a_{\max,k}, & \text{if } a_{k,j} \leq -\eta_a \cdot a_{\max,k} \end{cases}, \quad a_{\max,k} = \min_j i_{b,k,j}^{\max}(q_{SC,k}, q_{k,j}) \quad (20)$$

where $a_{k,j}^{\text{sat}}$ is the saturated action for the j -th power converter, $a_{\max,k}$ the worst-case current limit among all power converters, and $\eta_a = 0.9$ a safety margin.

The action is rate limited as a means to enforce smoothness of the RL actions. It is defined as:

$$\Delta a_{k,j}^{\text{sat}} = |a_{k,j} - a_{k-1,j}^{\text{sat}}| \leq \Delta a_{\max} \cdot \Delta t. \quad (21)$$

where $a_{k-1,j}^{\text{sat}}$ is the action applied in the previous time step, $\Delta a_{\max} = 2.5$ A/s the current rate limit (defined by the control engineer) and $\Delta t = 1$ s the sample time.

4.2.2. State Vector

The state vector \mathbf{s}_k is defined as follows:

$$\mathbf{s}_k = [q_k^T, q_{SC,k}, \Delta q_k^T, \Delta T_k^T, \mathbf{a}_{k-1}^{\text{sat}T}, a_{\max,k}, \mathbf{a}_{\text{age}}^T, \mathbf{\beta}_{\text{age}}^T, i_{\text{load},k}]^T. \quad (22)$$

where

- $\mathbf{q}_k = [q_{k,1}, \dots, q_{k,n}]^T$ and $q_{SC,k}$ represent the SoC of the battery cells and SC pack
- $\Delta q_{k,j}$ and $\Delta T_{k,j}$ represent the SoC and temperature deviations, computed as:

$$\Delta \mathbf{q}_k = [\Delta q_{k,1}, \dots, \Delta q_{k,n}]^T, \quad \Delta q_{k,j} = q_{k,j} - \bar{q}_k \quad (23)$$

$$\Delta \mathbf{T}_k = [\Delta T_{k,1}, \dots, \Delta T_{k,n}]^T, \quad \Delta T_{k,j} = T_{k,j} - \bar{T}_k \quad (24)$$

where \bar{q} and \bar{T} correspond to the mean SoC and mean temperature of the battery pack, respectively.

- $\mathbf{a}_{k-1}^{\text{sat}}$ and $\mathbf{a}_{\max,k}$ are the actions applied in the previous time step and the saturation limits, respectively. This information helps the RL agent to enforce rate and range limits.
- $\boldsymbol{\alpha}_{\text{age}} = [\alpha_{\text{age},1}, \dots, \alpha_{\text{age},n}]^T$ and $\boldsymbol{\beta}_{\text{age}} = [\beta_{\text{age},1}, \dots, \beta_{\text{age},n}]^T$ contain information about the variability of the cell's inner resistance and capacity. This information can be obtained by battery diagnosis algorithms (such as [9]) and it helps the RL agent to understand the cell-to-cell variability in the battery pack.
- $i_{\text{load},k}$ represents the load current requested from the battery pack. It is computed based on the battery voltage and load power (P_{load}) requested from the battery pack (see Section 3.5)

It is worth noticing that the above states are normalized with respect to their maximum expected values before passing them to the RL agent.

4.2.3. Reward Function

The goal of the RL is to minimize: i) battery current stress, ii) SoC deviations $\Delta \mathbf{q}$, iii) temperature deviations $\Delta \mathbf{T}$, and iv) power losses, v) while maintaining smooth control values \mathbf{a} . For this, the following nominal reward function is used:

$$r_{\text{no abort},k} = -\frac{1}{w_i^2} \cdot \sum_{j=1}^n (1 + \alpha_{\text{age},j}) \cdot i_{k,j}^2 - \frac{1}{w_{\Delta T}^2} \cdot \Delta \mathbf{T}_k^T \Delta \mathbf{T}_k - \frac{1}{w_{\Delta q}^2} \cdot \Delta \mathbf{q}_k^T \Delta \mathbf{q}_k - \frac{1}{w_l} \cdot P_{\text{losses},k} - \frac{1}{w_{\Delta a}} \cdot \sum_{j=1}^n |\Delta a_{k,j}^{\text{sat}}| \quad (25)$$

where $w_i, w_{\Delta T}, w_{\Delta q}, w_{\Delta a}$ are weights that the designer can use to prioritize different control goals and normalize the different reward terms.

The first term of the reward function aims at the reduction of the module current $i_{k,j} = a_{k,j}^{\text{sat}} + i_{\text{load},k}$; it encourages the use of battery modules with less degradation, i.e. with lower aging factor $\alpha_{\text{age},j}$. The following two terms aim at the reduction of temperature and SoC deviations, respectively. The fourth term enables the learning of smooth actions \mathbf{a} by penalizing large differences between the current (\mathbf{a}_k) and the previously action ($\mathbf{a}_{k-1}^{\text{sat}}$). Finally, the last term aims at the reduction of the power losses in the battery modules, converters and SCs:

$$P_{\text{losses}} = \sum_{j=1}^n (P_{\text{loss},B,j} + P_{\text{loss},C,j} + P_{\text{loss},SC,j}) \quad (26)$$

There are several conditions that might prematurely end an episode, in which case the (highly) negative reward $r_{\text{abort}} < 0$ is assigned at that time step k instead of $r_{\text{no abort},k}$. This abort mechanism speeds up the training by showing the agent that specific state-action pairs are less desirable. The final reward function is therefore described as:

$$r_k = \begin{cases} r_{\text{no abort},k}, & \text{if } \text{abort}_k = \text{false} \\ r_{\text{abort}}, & \text{else} \end{cases} \quad (27)$$

where abort_k is a Boolean condition that becomes true when the following state constraints are violated:

$$50 \% \leq q_{\text{SC},k} \leq 100 \%, \quad 5 \% \leq q_{k,j} \leq 95 \% \quad (28)$$

These constraints prevent deep discharge and overcharge of the battery cells and SCs and must be enforced by the RL agent.

5. Results and Discussion

This section presents the training setup, training results and validation results obtained when controlling the HBMS with the RL algorithm.

5.1. Training Setup

To train the RL agent, the control engineer needs to specify several parameters, including the training length, the initial values of the HBMS model, the SAC parameters and the weights of the reward function.

Regarding the training length: each episode has a maximum length of 500 s (representing 500 time steps), which is sufficiently large to cover the dominant time constants of the HBMS. The overall training process consists of $2 \cdot 10^6$ time steps, which is also adequately long to achieve convergence for the reward functions.

The initial values of the parameters of the environment (the HBMS model) are randomized in order to increase the robustness of the RL agent; this also decreases overfitting to a specific system initialization. The initial values are sampled from a continuous uniform distribution with intervals summarized in Table 3. These parameters include, for example, the initial time of the driving cycle ($t_{dc,0}$), which allows the RL agent to be exposed to a different sequence of load current $i_{load,k}$ during the episodes. Miscellaneous values for the initial SoC aging are also considered. For the aging factors, we assume that $\alpha_{age,j}$ and $\beta_{age,j}$ are fully correlated for a single module (e.g., when $\alpha_{age,j} = 0$ we also have $\beta_{age,j} = 0$), while there is no correlation assumed between modules.

Table 3: Random sampling for training initialization

Description	Variable	Sampling interval
Beginning of driving cycle*	$t_{dc,0}$	[0 s, 1299 s]
Module's initial temperature**	$T_{0,j}$	$[T_a - 1^\circ C, T_a + 1^\circ C]$
Initial mean of the modules' SoC	\bar{q}_0	[45 %, 85 %]
Initial module's deviation from the sampled mean	$\Delta q_{0,j}$	[-5 %, 5 %]
Fixed resistance increase***	$\alpha_{age,j}$	$[0, \alpha_{age,EOL}]$
Fixed capacity decrease***	$\beta_{age,j}$	$[0, \beta_{age,EOL}]$
Supercapacitor pack's SoC	$q_{SC,0}$	[75 %, 95%]

* the WLTC driving cycle is used during training of the RL agent

** ambient temperature: $T_a = 25^\circ C$

*** end-of-life (EOL) values: $\alpha_{age,EOL} = 240\%$ and $\beta_{age,EOL} = 12\%$ [23] [37] [38].

Regarding the SAC hyperparameters: most of them are kept on the default settings provided by [18]. The only major modification was the size of the replay buffer, which was increased to $2 \cdot 10^6$ time steps; this gives the RL agent an additional possibility to learn from the whole training process. The mapping between states and actions is performed by the RL agent using a feed-forward neural network with 256 neurons, two hidden layers and the rectified linear unit $f(x) = \text{relu}(x) = \max(0, x)$ [39] as activation function.

Four types of reward functions, with different weights, were considered in this study (see Table 4). This yields the following family of RL agents:

- Agent a) and b) focus on minimization of SoC and temperature deviations, respectively

- Agent c) aims primarily at current stress reduction, with a small incentive for reducing temperature deviations.
- Agent d) attempts to achieve a balance between all goals (SoC balancing, temperature balancing, current stress and losses).

Table 4: Reward function parametrizations

Agent	Description	w_i	$w_{\Delta T}$	$w_{\Delta q}$	$w_{\Delta a}$	w_l	r_{abort}
a	SoC deviation	NA*	NA*	0.01	2	NA*	-3000
b	Temperature deviation	NA*	0.25	NA*	2	NA*	-3000
c	Module current and temperature deviation	60	1	NA*	10	NA*	-3000
d	Module current, temperature deviation, SoC deviation and losses	60	1	0.01	2	1	-10000

*not available (NA): corresponding term is removed from the reward function

5.2. Training Results

Figure 16 shows the development of episode returns and episode lengths for a RL agent a). For the episode returns, a high variance can be observed, which appears due to difficult initial conditions caused by the randomization. The mean of the episode returns increases during the training from about -9000 to about -5000 . The low changes at the end demonstrate the convergence of the training. The episode lengths initially show many aborts, but quickly increase towards the maximum episode length. This shows that the agent learns to consider the abort conditions and to adjust its actions \mathbf{a}_k accordingly. The later decrease of the mean episode length can also be explained by the agent’s exploration.

The reward functions for the other RL agents (b,c,d) have a similar training behavior, and are omitted here for the sake of brevity.

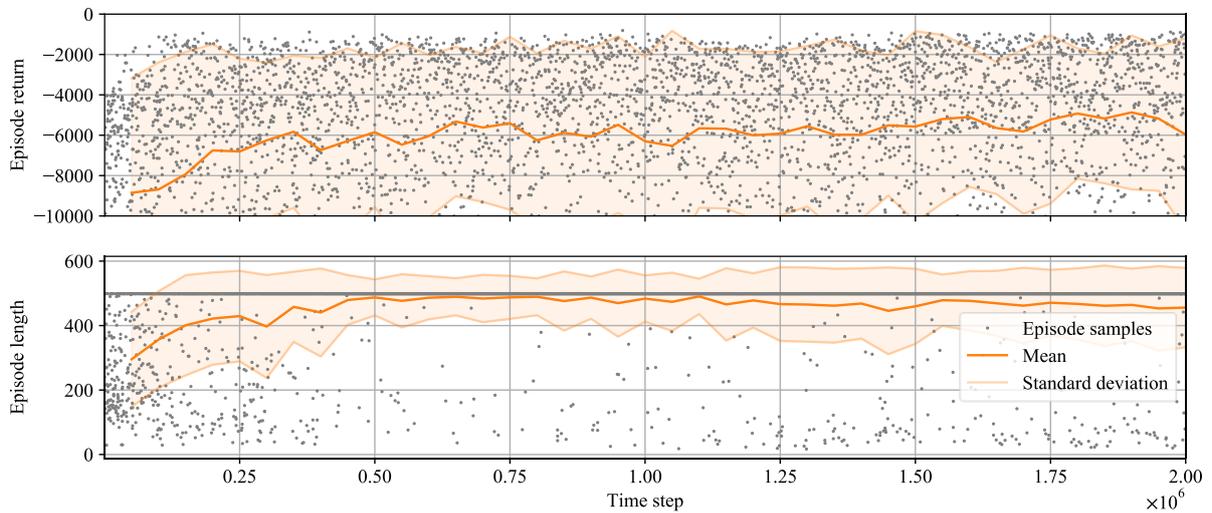


Figure 16: Episode returns and lengths during training RL agent a). Outlier returns below -10000 not shown. Mean with added and subtracted standard deviation shown (note that there are no samples with an episode length larger than 500 s).

5.3. Validation Results

This section presents the validation results of the RL agents. In contrast with the previous section, the custom Stuttgart driving cycle in its full length (cf. Figure 14) is employed in the validation. This allows us to assess the performance of the RL agents on a driving cycle that is entirely unknown to them. Additionally, the initial values of the environment are fixed in order to make the results comparable (see Table 5).

Table 5: Validation initialization

Description	Variable	Initialization
Beginning of driving cycle	$t_{dc,0}$	0 s
Modules' initial temperature*	\mathbf{T}_0	$[T_a, T_a, T_a]^T$
Initial modules' SoC	\mathbf{q}_0	$[90\%, 87.5\%, 85\%]^T$
Fixed resistance increase	$\boldsymbol{\alpha}_{age}$	$[0.25 \cdot \alpha_{age,EOL}, 0.35 \cdot \alpha_{age,EOL}, \alpha_{age,EOL}]^T$
Fixed capacity decrease	$\boldsymbol{\beta}_{age}$	$[0.25 \cdot \beta_{age,EOL}, 0.35 \cdot \beta_{age,EOL}, \beta_{age,EOL}]^T$
Supercapacitor pack's SoC	$q_{SC,0}$	95%

* ambient temperature: $T_a = 25\text{ }^\circ\text{C}$

Table 6 summarizes the validation results of the different RL agents. The following performance metrics are considered:

- mean (absolute) SoC deviations, $\mu_{|\Delta q|} = \frac{1}{K} \sum_{k=1}^K \sum_{j=1}^n |\Delta q_{k,j}|$, with K denoting the number of validation time steps
- mean (absolute) temperature deviations, $\mu_{|\Delta T|} = \frac{1}{K} \sum_{k=1}^K \sum_{j=1}^n |\Delta T_{k,j}|$
- smoothness of control actions, measured as the mean differences in control actions: $\mu_{|\Delta a|} = \frac{1}{K} \sum_k \sum_j |\Delta a_{k,j}^{sat}|$
- root-mean-squared (RMS) current for each module (RMS_j)
- average RMS current in the battery pack ($\frac{1}{n} \sum_j RMS_j$)
- overall energy losses in the battery modules, power converters and SCs.

Table 6: Summary of evaluation metrics' results for Stuttgart driving cycle with no balancing actions and different trained agents. Best results are marked in green.

Balancing actions	$\mu_{ \Delta q }$ [%/100]	$\mu_{ \Delta T }$ [$^\circ\text{C}$]	$\mu_{ \Delta a }$ [A]	RMS current [A]			Integrated losses [Wh]	
				Aver.	$RMS_{j=1}$	$RMS_{j=2}$		$RMS_{j=3}$
No balancing	0.0603	0.422	0.0	41.60	41.60	41.60	41.60	0.98
Agent a)	0.0171	0.315	0.883	41.77	43.35	42.01	39.94	1.46
Agent b)	0.0463	0.291	2.678	41.82	42.24	44.27	38.95	1.78
Agent c)	0.0407	0.337	8.840	41.19	42.81	41.12	39.63	1.56
Agent d)	0.0245	0.330	1.731	41.22	43.15	40.76	39.76	1.29

5.3.1. No Balancing

Figure 17 shows the SoC and temperature behavior with no control ($\mathbf{a}_k = 0$). Due to different aging of the modules, variations in the SoC and temperature emerge. The most-aged module 3 is the fastest to discharge, while the least-aged module 1 is the slowest to do so. Module 3 also presents higher temperature than the other modules, which is expected given its higher internal resistance. Furthermore, the temperature increase during the driving cycle is relatively moderate: 1°C increase with respect to

ambient temperature. This can be explained by the high heat capacity of the battery cells (cf. Section 3.3.2).

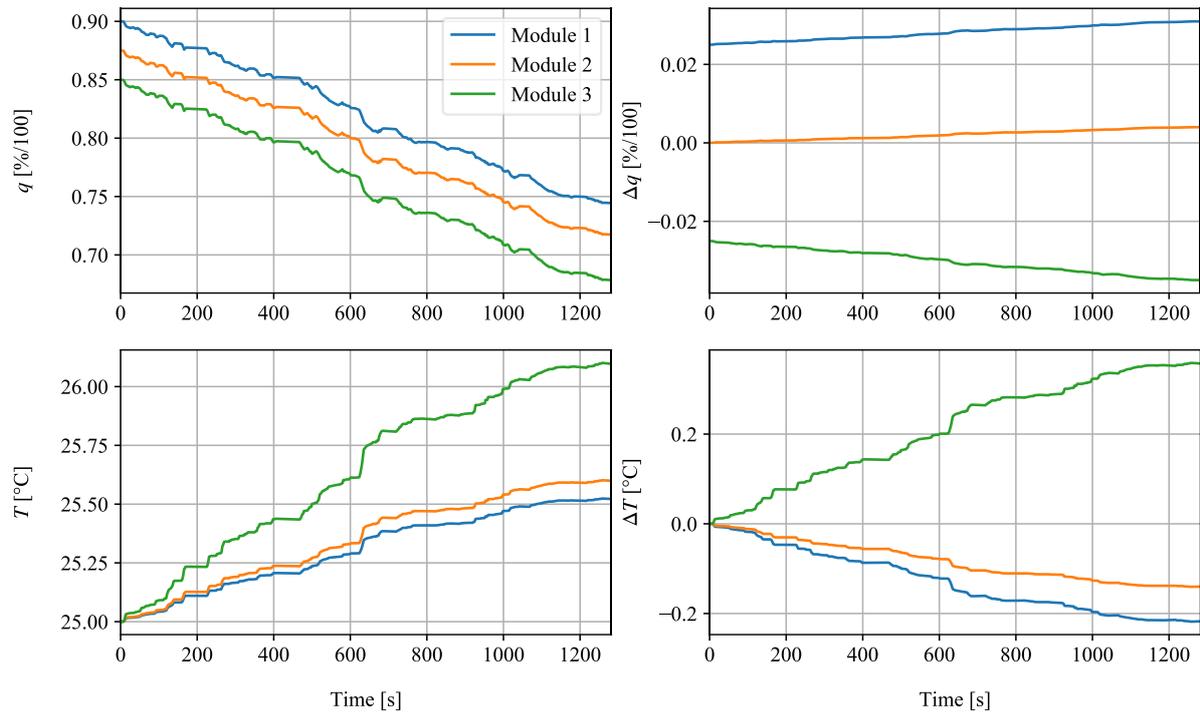


Figure 17: SoCs, temperatures, SoC deviations and temperature deviations of modules and SoC of SC pack during validation on Stuttgart driving cycle with no balancing actions.

5.3.2. Agent a)

Figure 18 shows the validation results of RL agent a). This agent reduces the SoC deviations Δq_j within the first 600 s of the driving cycle. During this initial period, the agent charges the most-aged module 3 with the maximum balancing current and discharges (most of the time) the stronger modules (1 and 2). Afterwards, the magnitude of the control actions is reduced, but the agent is still able to keep the SoC differences (Δq_j) low. In comparison to the “no balancing” scenario, the RL agent (cf. Table 6)

- reduces $\mu_{|\Delta q|}$ by 72 % and $\mu_{|\Delta T|}$ by 25 %. This suggests that the control objectives of SoC balancing and temperature balancing are not independent from each other;
- reduces current stress in the most-aged module 3 (-3.9%) and increases stress in the least-aged module 1 (+4.2%). The average RMS current does decrease, because it is not an explicit part of the reward function;
- increases the integrated losses (+48%). This mainly stems from the higher losses generated by the power converters.

It is also interesting to note that the SC is mainly used during the first 600s to support the SoC balancing task.

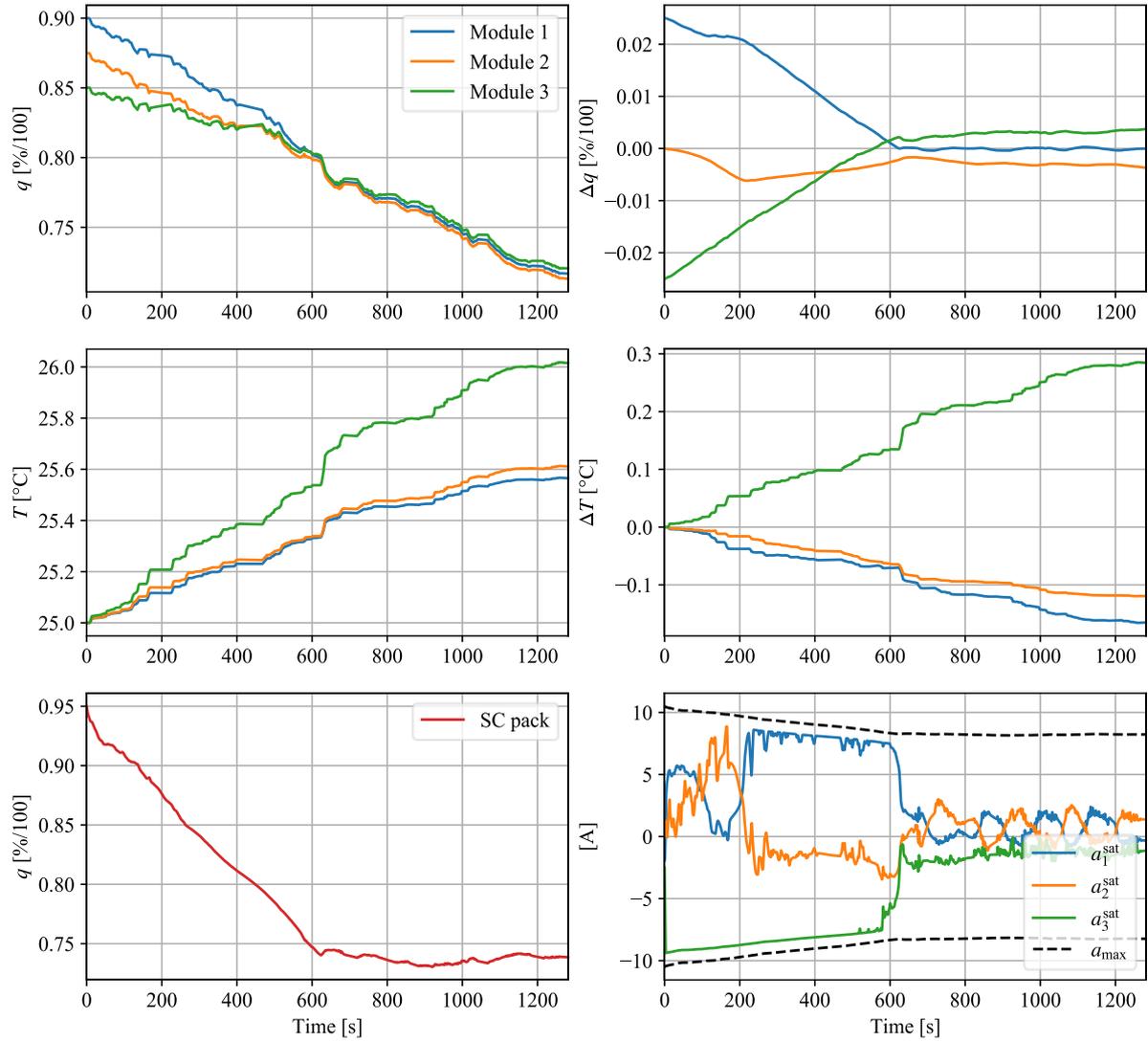


Figure 18: SoCs, temperatures, SoC deviations and temperature deviations of modules and SoC of SC as well as balancing actions by SoC deviation minimizing agent pack during validation on Stuttgart driving cycle.

5.3.3. Agent b)

Figure 19 shows the results of the RL agent b), which focuses primarily on reducing temperature variations. It can be observed that this agent charges the most-aged module 3 with a value that is always very close to saturation; the agent's goal is to shift thermal load from the most-aged model 3 to the least-aged modules (1 and 2). Despite these efforts, only modest improvements in temperature variation are obtained: $\mu_{|\Delta T|}$ is decreased by 8% with respect to RL agent a). This result is due to the high heat capacity of the battery cells, which leads to small temperature increases during the driving cycles (with a load current of 1 C on average).

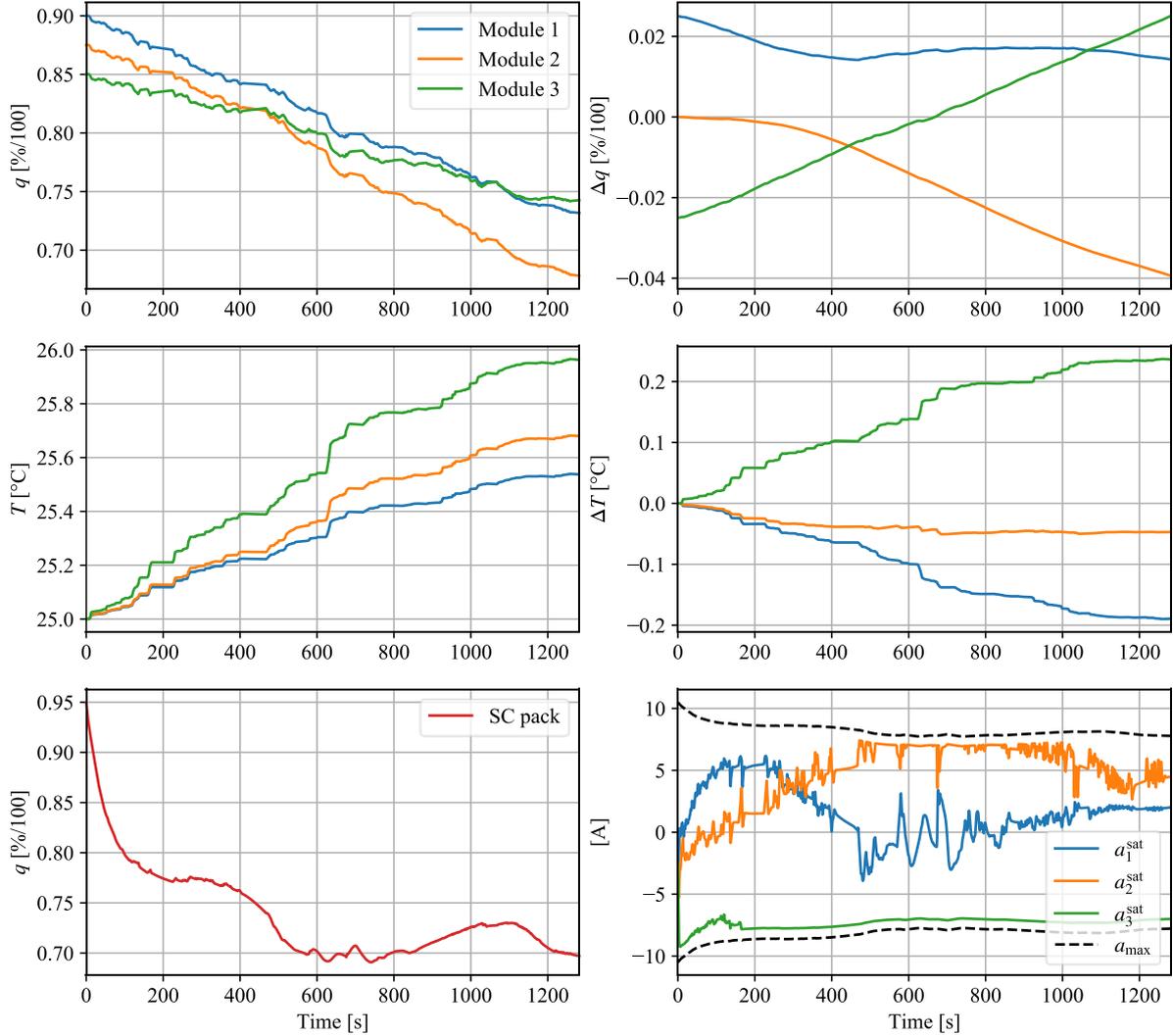


Figure 19: SoCs, temperatures, SoC deviations and temperature deviations of modules and SoC of SC as well as balancing actions by temperature deviation minimizing agent pack during validation on Stuttgart driving cycle.

5.3.4. Agent c)

Figure 20 shows the results of the RL agent c), which focuses on minimization of the current stress. During the first 200 s, the RL agent extracts energy from the SCs to support the battery cells. Interestingly, this support is aging-aware: the agent generates actions that decrease the current in the most-aged modules (2 and 3) and increase the current in the least-aged module (1). After 200 s, the SCs provide minimum support to the battery pack.

The RL agent c) features the lowest average RMS of all agents, 41.19 A, which represents a reduction of 1.0 % when compared to the no balancing scenario. At cell level, the RL agent c) reduces the current of the most-aged module (3) in 4.7 %, while increasing the current in the least-aged module in 2.9 %.

In contrast with the previous test cases, RL agent c) presents more chattering in the control action, which is due to the reduction of the weight ($1/w_{\Delta a}$). It had therefore less incentive to learn a smooth action behavior.

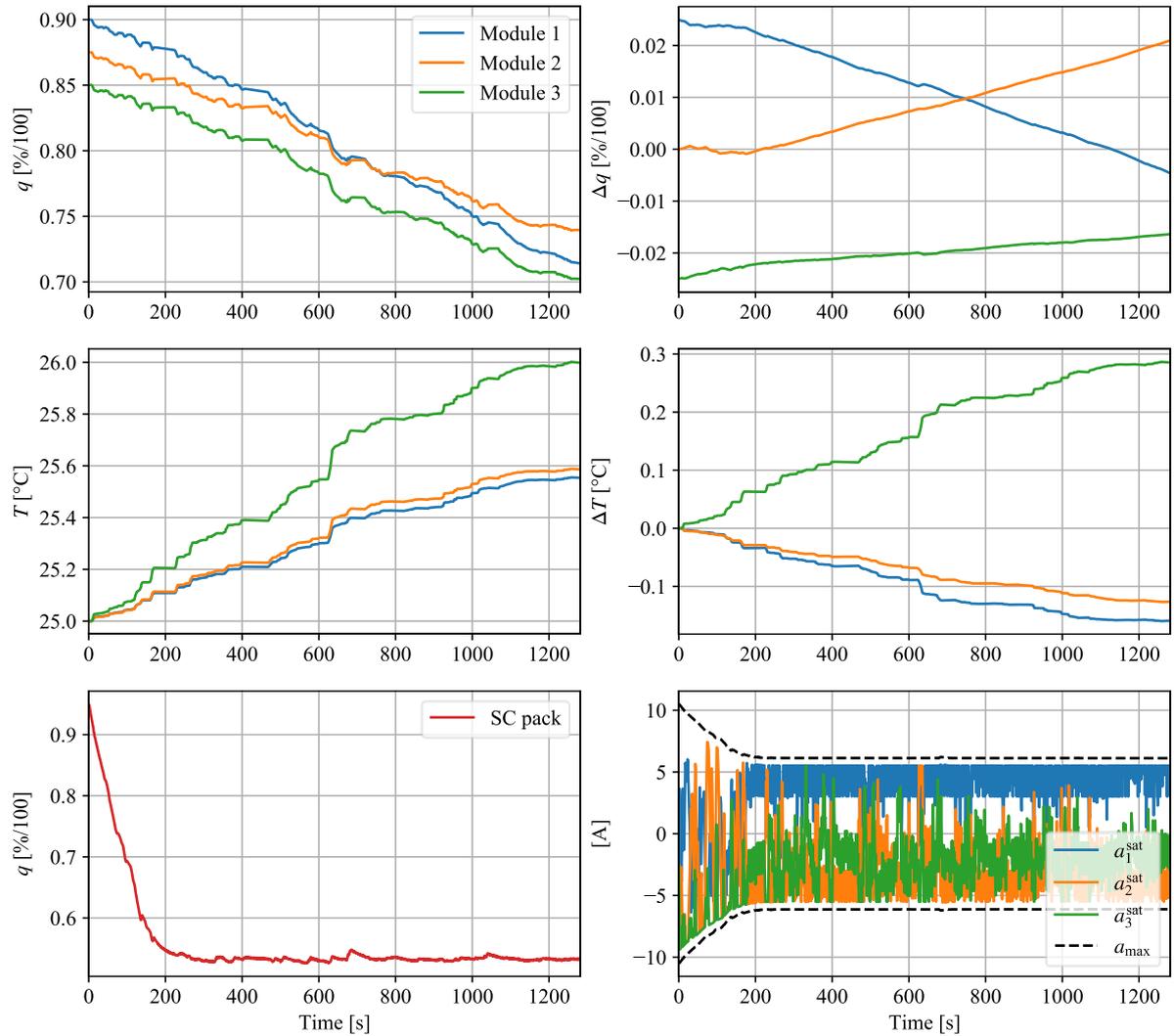


Figure 20: SoCs, temperatures, SoC deviations and temperature deviations of modules and SoC of SC pack as well as balancing actions by module current and temperature deviation minimizing agent during validation on Stuttgart driving cycle.

5.3.5. Agent d)

This final agent d) aims at fulfilling all control objectives simultaneously. Table 6 shows that this agent achieves

- second place in SoC variation (just behind agent a)
- third place in temperature variation (but very close to agents a) and b)
- second place in average current stress (very close to agent c)
- second place in integrated losses (behind no balancing, but ahead of all RL agents)

The time-domain results of agent d) (Figure 21) show a reduction in SoC imbalances during the first 400 s. After this initial transient, the SoC deviations are approximately kept at a constant value. By doing so, the agent's actions can be kept at a lower (absolute) amplitude, providing a good compromise between power losses and SoC balance. The corresponding actions are less smooth than for RL agent a), but smoother than for agent c). This can be explained by the less weighted penalization of action and control value differences in reward function c).

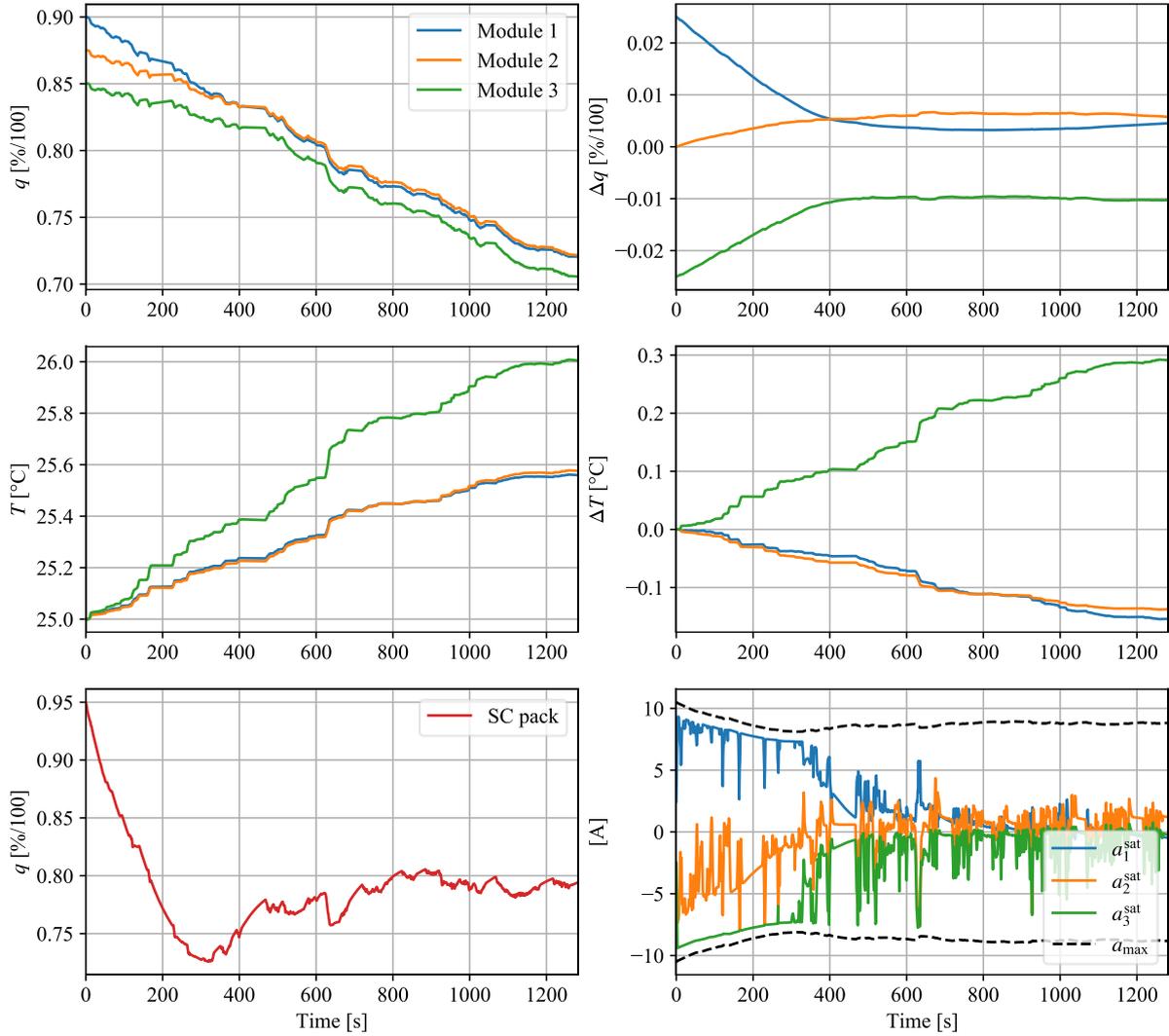


Figure 21: SoCs, temperatures, SoC deviations and temperature deviations of modules and SoC of SC pack as well as balancing actions by module current, temperature deviation, SoC deviation and losses minimizing agent during validation on Stuttgart driving cycle.

6. Conclusion and Outlook

This work investigated the use of RL to control HBMS. First, a multi-physical model of the HBMS' components—including power converters, battery and supercapacitors—was implemented in Modelica. To improve numerical efficiency, a quasi-static model of the power converter was developed. This allowed us to improve by more than 3000-fold the real-time simulation factor of the HBMS model and accelerate the RL training. Additionally, the battery model was optimized and validated with experimental data from NMC cells.

The model was incorporated into a RL toolchain featuring the SAC algorithm. Multiple control objectives were assessed, yielding RL agents that focus on: a) SoC equalization, b) temperature equalization, c) age-weighted module current reduction, and d) trade-off between these three objectives and power losses. To increase the robustness of the obtained RL agents, several randomizations were incorporated in the training process.

The trained agents were then validated with an unknown driving cycle. In comparison to the scenario without control, RL agents that prioritize a single objective (a,b,c) were able to reduce SoC deviations by 72%, temperature deviation by 31%, and the RMS current in the most-aged cell by 4.7%. When all objectives are traded off simultaneously, the RL agent d) still performed well, reducing the SoC deviations by 59%, temperature deviations by 22% and the RMS current in the most-aged cell by

4.4%. All RL agents increased energy losses due to balancing actions, which is a drawback of the HBMS.

Additionally, the maximum temperatures in the battery cells were only marginally reduced by the RL agents due to the large heat capacity of the battery cells. Future work should consider cells with smaller heat capacity to better assess the potential of RL for thermal management. We also plan to compare the RL agents' performance with model-based controllers, incorporate preview information in the RL's state vector and perform experimental validation.

Author Contributions: Conceptualization, R.C., J.M., J.B., R.A.; methodology, R.C., J.M., J.B., R.A.; RL and FMI framework, J.U., J.M.; Modelica models, J.B., J.M.; validation, J.M., R.C.; resources, J.B., J.U., R.C.; writing – original draft preparation, J.M., J.B., R.C., J.U., R.A.; writing – review and editing, J.M., J.B., R.C., J.U., R.A.; visualization, J.M., J.B., R.C., J.U.; supervision, R.C. All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded by the DLR internal project NGC-A&E.

Acknowledgments: The authors' thanks go to Tobias Posielek for his feedback and help in scientific writing.

7. References

- [1] International Energy Agency (IEA), "Global EV Outlook 2020," Paris, France, 2020. [Online]. Available: <https://www.iea.org/reports/global-ev-outlook-2020>.
- [2] J. Barreras, D. Frost and D. Howey, "Smart Balancing Systems: An Ultimate Solution to the Weakest Cell Problem?," in *IEEE Vehicular Technology Society Newsletter*, 2018.
- [3] J. Barreras, C. Pinto, R. de Castro, E. Schaltz, S. Andreasen and R. Araujo, "Multi-Objective Control of Balancing Systems for Li-Ion Battery Packs: A Paradigm Shift?," in *IEEE Vehicle Power and Propulsion Conference*, Coimbra, Portugal, 2014.
- [4] E. Chemali, M. Preindl, P. Malysz and A. Emadi, "Electrochemical and Electrostatic Energy Storage and Management Systems for Electric Drive Vehicles: State-of-the-Art Review and Future Trends.," *IEEE Journal of Emerging and Selected Topics in Power Electronics*, vol. 4, no. 3, pp. 1117-1134, 2016.
- [5] R. Araujo, R. de Castro, C. Pinto, P. Melo and D. Freitas, "Combined Sizing and Energy Management in EVs With Batteries and Supercapacitors," *IEEE Transactions on Vehicular Technology*, vol. 63, no. 7, pp. 3062-3076, 2014.
- [6] Q. Zhang, W. Deng and G. Li, "Stochastic Control of Predictive Power Management for Battery/Supercapacitor Hybrid Energy Storage Systems of Electric Vehicles," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 7, pp. 3023-3030, 2018.
- [7] R. de Castro, C. Pinto, J. Barreras, R. Araújo and D. Howey, "Smart and Hybrid Balancing System: Design, Modeling, and Experimental Demonstration," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 12, pp. 11449-11461, 2019.
- [8] F. Altaf, B. Egardt and L. Johannesson Mardh, "Load Management of Modular Battery Using Model Predictive Control: Thermal and State-of-Charge Balancing," *IEEE Transactions on Control Systems Technology*, vol. 25, no. 1, pp. 47-62, 2017.
- [9] G. L. Plett, *Battery Management Systems, Vol. 2: Equivalent-Circuit Methods*, Norwood, USA: Artech House, 2016.
- [10] R. de Castro, J. Brembeck and R. E. Araujo, "Nonlinear Control of Dual Half Bridge Converters in Hybrid Energy Storage Systems," in *IEEE Vehicular Power and Propulsion Conference*, Gijon, Spain, 2020.
- [11] Y. Li, "Reinforcement Learning Applications," 2019. [Online]. Available: <https://arxiv.org/abs/1908.06973>.
- [12] D. Liu, "Application of Deep Reinforcement Learning for Battery Design," Master's Thesis, University of Missouri, USA, 2020.
- [13] H. Sun, Z. Fu, F. Tao, L. Zhu and P. Si, "Data-Driven Reinforcement-Learning-Based Hierarchical Energy Management Strategy for Fuel Cell/Battery/Ultracapacitor Hybrid Electric Vehicles," *Journal of Power Sources*, vol. 455, 2020.
- [14] B. Xu, J. Shi, S. Li, H. Li and Z. Wang, "Energy Consumption and Battery Aging Minimization Using a Q-learning Strategy for a Battery/Ultracapacitor Electric Vehicle," 2020. [Online]. Available: <https://arxiv.org/abs/2010.14115>.

- [15] J. Cao, D. Harrold, Z. Fan, T. Morstyn, D. Healey and K. Li, "Deep Reinforcement Learning-Based Energy Storage Arbitrage With Accurate Lithium-Ion Battery Degradation Model," *IEEE Transactions on Smart Grid*, vol. 11, no. 5, pp. 4513-4521, 2020.
- [16] S. Park, A. Pozzi, M. Whitmeyer, W. T. Joe, D. M. Raimondo and S. Moura, "Reinforcement Learning-based Fast Charging Control Strategy for Li-ion Batteries," 2020. [Online]. Available: <https://arxiv.org/abs/2002.02060>.
- [17] T. Haarnoja, A. Zhou, P. Abbeel and S. Levine, "Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor," in *International Conference on Machine Learning*, Stockholm, Sweden, 2018.
- [18] A. Hill, A. Raffin, M. Ernestus, A. Gleave, R. Traore, P. Dhariwal, C. Hesse, O. Klimov, A. Nichol, M. Plappert, A. Radford, J. Schulman, S. Sidor and Y. Wu, *Stable Baselines*, <https://github.com/hill-a/stable-baselines>, 2018.
- [19] Modelica Association, "Modelica," 2020. [Online]. Available: <http://www.modelica.org>. [Accessed 10 2020].
- [20] F. Gao, N. Mugwisi and D. Rogers, "Three Degrees of Freedom Operation of a Dual Half Bridge," in *European Conference on Power Electronics and Applications*, Genova, Italy, 2019.
- [21] C. F. A. Pinto, "Sizing and Energy Management of a Distributed Hybrid Energy Storage System for Electric Vehicles," Ph.D. Thesis, University of Porto, Porto, Portugal, 2018.
- [22] D. Graovac, M. Pürschel and A. Kiep, "MOSFET Power Losses Calculation Using the Data-Sheet Parameters," Infineon Technologies AG, Neubiberg, Germany, 2006.
- [23] C. Pinto, J. V. Barreras, E. Schartz and R. E. Araujo, "Evaluation of Advanced Control for Li-ion Battery Balancing Systems Using Convex Optimization," *IEEE Transactions on Sustainable Energy*, vol. 7, pp. 1703-1717, 2016.
- [24] J. Brembeck, "Model Based Energy Management and State Estimation for the Robotic Electric Vehicle ROboMObil," Ph.D. Thesis, Technical University of Munich, Munich, 2018.
- [25] O. Tremblay and L.-A. Dessaint, "Experimental Validation of a Battery Dynamic Model for EV Applications," *World Electric Vehicle Journal*, vol. 3, pp. 289-298, 2009.
- [26] S. N. Motapon, A. Lupien-Bedard, L.-A. Dessaint, H. Fortin-Blanchette and K. Al-Haddad, "A Generic Electrothermal Li-ion Battery Model for Rapid Evaluation of Cell Temperature Temporal Evolution," *IEEE Transactions on Industrial Electronics*, vol. 64, pp. 998-1008, 2017.
- [27] J. Brembeck, "A Physical Model-Based Observer Framework for Nonlinear Constrained State Estimation Applied to Battery State Estimation," *Sensors*, vol. 19, no. 20, 4402, 2019.
- [28] J. Brembeck, L. M. Ho, A. Schaub, C. Satzger, J. Tobolar, J. Bals and G. Hirzinger, "ROMO – The Robotic Electric Vehicle," in *IAVSD International Symposium on Dynamics of Vehicle on Roads and Tracks*, Manchester, UK, 2011.
- [29] A. Pfeiffer, "Optimization Library for Interactive Multi-Criteria Optimization Tasks," in *International MODELICA Conference*, Munich, Germany, 2012.
- [30] R. Faranda, M. Gallina and D. T. Son, "A New Simplified Model of Double-Layer Capacitors," in *International Conference on Clean Electrical Power*, Capri, Italy, 2007.
- [31] J. Tobolar, M. Otter and T. Bünthe, "Modelling of Vehicle Powertrains with the Modelica PowerTrain Library," in *Systemanalyse in der Kfz-Antriebstechnik*, Augsburg, Germany, 2007.

- [32] R. S. Sutton and A. G. Barto, Reinforcement Learning: An Introduction, Second edition ed., Cambridge, USA: The MIT Press, 2018.
- [33] B. D. Ziebart, "Modeling Purposeful Adaptive Behavior with the Principle of Maximum Causal Entropy," Ph.D. Thesis, Carnegie Mellon University, Pittsburgh, USA, 2010.
- [34] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel and S. Levine, "Soft Actor-Critic Algorithms and Applications," 2018. [Online]. Available: <https://arxiv.org/abs/1812.05905>.
- [35] H. v. Hasselt, "Double Q-learning," in *International Conference on Neural Information Processing Systems*, Vancouver, Canada, 2010.
- [36] S. Fujimoto, H. van Hoof and D. Meger, "Addressing Function Approximation Error in Actor-Critic Methods," in *International Conference on Machine Learning*, Stockholm, Sweden, 2018.
- [37] S. F. Schuster, M. J. Brand, P. Berg, M. Gleissenberger and A. Jossen, "Lithium-ion Cell-to-Cell Variation During Battery Electric Vehicle Operation," *Journal of Power Sources*, vol. 297, pp. 242-251, 2015.
- [38] W. Waag, S. Käbitz and D. U. Sauer, "Experimental Investigation of the Lithium-ion Battery Impedance Characteristic at Various Conditions and Aging States and Its Influence on the Application," *Applied Energy*, vol. 102, pp. 885-897, 2013.
- [39] X. Glorot, A. Bordes and Y. Bengio, "Deep Sparse Rectifier Neural Networks," in *International Conference on Artificial Intelligence and Statistics*, Fort Lauderdale, USA, 2011.