Suitability of physicochemical models for embedded systems regarding a nickel-rich, silicon-graphite lithium-ion battery

Sturm, J.[a,*], Ludwig, S.[a], Zwirner, J.[a], Ramirez-Garcia, C.[a], Heinrich, B.[a], Horsche, M.F.[a], Jossen, A.[a]

[a] *Technical University of Munich (TUM), Institute for Electrical Energy Storage Technology (EES), Arcisstrasse 21, 80333 Munich, Germany*

Abstract

Local inhomogeneous electrode utilization in recent lithium-ion batteries tends to increase due to larger sizes and/or higher densification, which poses a challenge for accurate, model-based monitoring. Pseudo-two dimensional (p2D) physicochemical models (PCM) can offer such locality via calculating local potentials and concentrations through the thickness of the electrode stack and are numerically reduced for implementation in a microcontroller in this work. Finite difference method combined with solid-diffusion approximations and orthogonal collocation reformulation are applied to generate three MATLAB- and three microcontroller-suitable *C*-code p2D-PCMs, which are experimentally validated towards constant current charge/discharge and driving cycle loads on a high-energy NMC-811/SiC-18650 lithium-ion battery. Benchmarking to an equivalent circuit model reveals similar mean cell voltage errors below 20 mV for the driving cycle. Reducing spatial elements reveals errors below 1 % for local (*i.e.* concentrations/potentials) and global

*Corresponding author: johannes.sturm@tum.de

states (*i.e.* cell voltage/temperature) and is applied to speed-up the *C*-code p2D-PCMs in the microcontroller (max. 168 MHz with 192 kB RAM) to calculate at least 37 % faster than real-time. Real-time computability is investigated via varying processor frequencies and using hardware acceleration schemes. The memory allocation to solve and store the p2D-PCMs on the microcontroller require 115 kB and 213 kB at a maximum, respectively.
Keywords: Lithium-ion battery, Model reduction, Pseudo-two dimensional model, Microcontroller, Nickel-rich, Graphite-silicon composite

1. Introduction

Recent achievements in higher energy density of lithium-ion batteries (LIBs) promote inhomogeneous usage [1] either along the electrodes or through the thickness of the cell stack [2]. Therefore, monitoring and controlling of the battery's states on local scale are necessary to guarantee efficient utilization and safety during both dynamic (*i.e.* driving cycle) or rather static loads (*i.e.* fast charging).

Beside larger electrodes, thicker composite coatings, higher densification (*i.e.* porosity < 20 %) and high capacitive active materials such as nickel-rich cathodes (*e.g.* NMC-811) and graphite-silicon composite anodes (SiC) are applied to increase the energy density. The resulting increase of capacity can lead to local current densities along the electrodes exemplarily up to $4.91\,\mathrm{mA\,cm^{-2}}$ at 1C for a 3.35 Ah 18560 LIB (INR18650-MJ1, LGChem) incorporating low electrode porosities of 21.6 % and 17.1 % for the SiC anode and NMC-811 cathode [2]. As a result, inhomogeneous utilization through the cell stack and along the electrodes appears [1].

Beside the global states such as cell voltage, applied current and surface temperature, proper LIB monitoring should estimate also the local states such as potentials and concentrations in the electrolyte and the active material to ease harmful side-reactions such as lithium plating [3, 4] or solid-electrolyte-interphase (SEI) growth and cracking [5] or to avoid critical hot spots [1]. State-of-the-art model-based monitoring incorporate equivalent circuit models (ECMs) as it offers fast calculation and easy parameterization. However, only global states can be simulated. Physicochemical models (PCMs) such as the newman-type [6] pseudo-two dimensional (p2D) model offer simulated local states based on porous electrode, concentrated solution theory and electrode kinetics through the thickness of the cell stack. However, this model comes with computational complexity due to solving its differential algebraic equation (DAE) system, which significantly slows down the calculation. Together with the complex parameterization, application in battery management system (BMS) outside the research field is hindered.

In this matter, we want to investigate the suitability of the p2D model in embedded systems (*i.e.* microcontroller) via evaluating the computational perfomance and simulation accuracy of p2D-PCMs using different spatial and time discretizations, approximation schemes for the particle domain and solvers. Three different p2D-PCMs are parameterized for a 18650 NMC-811/SiC LIB (INR18650-MJ1 [2]) and implemented first in MATLAB® and second transferred into a stand-alone *C*-code for microcontroller implementation. Errors of parameterization, model reduction, transfer into the microcontroller and validation via measurements are outlined for constant current (CC) charge and discharge and a driving cycle scenario to evaluate the suit-

ability of the p2D-PCMs for real-time simulation in embedded systems.

2. Model reduction of the p2D physicochemical model

To ease the computational inefficiency, model reduction can be applied to the p2D-PCM [6], which is summarized in fundamental reviews [7, 8]. In this work, its actual implementation in the STM32F407VGT6 microcontroller (STM32, STMicroelectronics [9]) is evaluated towards computation speed and simulation accuracy, coming with crucial limitations in computation power (max. 168 MHz in a 32-bit ARM® Cortex®-M4 core) and memory ressources offering only a maximum of 1024 kB flash memory to store and a maximum of 192 kB static random accessible memory (RAM) to solve the model. These limitations are often not considered in research as reductions are investigated on desktop computers.

In this matter, low spatial discretizations with sufficient accuracy are favored as the total number of spatial elements defines the size of the DAE, the related memory requirements and thus the computational effort. Spatial discretization of the particle domain (*i.e.* 'pseudo'-domain, r-coordinate) can cause a large DAE system via discretizing the solid-diffusion partial differential equation (PDE). At every node in the electrolyte domain (x-coordinate), this PDE is solved for the concentration c_s of lithium-ions, which tremendously raises the allocated memory. As only the particle-surface concentration $c_{s,s}$ is needed for the kinetics, its numerical reduction is feasible. Approximation methods for the concentration profile in the particle were implemented in literature via volume averaging together with a parabolic profile (PP) [10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21] or diffusion length ap-

proaches [22, 23, 24], which suggests linearity between surface- and average-concentration $c_{s,ave}$. Sufficient accuracy and computation efficiency appeared via using an eigenfunction method (EM) [25, 26, 27, 28], where the solution is derived from a truncated, analytical solution of an infinite series of eigenfunctions. Maintaining a spatial discretization of the PDE, reformulations to an ordinary differential equation (ODE) in time via spectral methods (*e.g.* orthogonal collocation (OC)) [29, 30, 31, 32, 33, 34, 35] showed an enormous calculation acceleration while guaranteeing sufficient accuracy. Also standard discretization schemes such as finite volume methods (FVM) (*e.g* finite difference method (FDM) or finite element method (FEM)) [36, 30, 37, 38] were investgated next to integral methods such as the duhamel superposition integral (DSI) [6, 39, 40, 41, 42]. Reducing the entire solid phase to a single particle (SPM, [43, 16, 44, 45]) revealed promising computational efficicency, but is not regarded in the following as the original p2D-PCM [6] is focussed in this work.

Referring to real-time computability and sufficient simulation accuracy as seen in our previous work [46], we focus on a FDM discretization for the electrolyte domain accompanied with two different approximation schemes (*i.e.* PP- [11] and EM [25]) for the particle domain. For comparison, the third p2D-PCM uses orthogonal collocation and thus maintains a spatial discretization of the particle domain.

As the PP- [47] and EM-approximation are adopted from the corresponding original work, which have already shown its validity, accuracy and computational efficiency, the reader is referred to these works [13, 14, 48, 15, 17, 26, 27] for more information. The OC-method was used as well in literature be-

fore [49, 31] and a single work [34] investigated the performance on a microcontroller (ATMEL 32UC3A1512 at 16MHz and 512 kB RAM [34]) including 21 DAE which could be solved in at least 190 ms under 1C CC discharge. Unfortunately, no description of the actual implementation on this microcontroller is shown [34] and the work misses implementation recommendations, detailed computational performance analysis and application-near load scenarios.

According to literature, the PP-, EM- and OC-PCM offer significant computation speed, reduction of DAE size and maintain sufficient accuracy to be implemented in the STM32 [9] used in this work. Far to little work [34] had focussed on an actual microcontroller implementation of the p2D-PCM in the past, which is one of the main objectives of this work together with evaluating the most suitable discretization/approximation scheme to gain real-time computation and low simulation error in embedded systems.

3. Modeling of a 18650 NMC-811/SiC lithium-ion battery

As shown in Table 1, three different p2D-PCMs are investigated incorporating different spatial discretizations/approximations and two benchmark models (COMSOL-PCM and ECM) are used to simulate a 18650 NMC-811/SiC LIB [2]. The three p2D-PCMs are used as stand-alone MATLAB®- and *C*-code to simulate on a desktop computer and the STM32 microcontroller, respectively. The MATLAB®-codes are used for model parameterization and validation, determining the simulation error via reducing spatial discretization and evaluate the implementation error for the *C*-code equivalents in the microcontroller. As benchmarks, the COMSOL-PCM uses COM-

Table 1: Model overview

Model	Spatial discretization		Framework	Thermal model
	$x-$domain	$r-$domain		
PP-PCM	FDM	Parabolic [I]	MATLAB [III] & C-code [IV]	√
EM-PCM	FDM	Eigenfunction [II]		√
OC-PCM	Orthogonal collocation			√
COMSOL-PCM	FDM		COMSOL [V]	√
ECM	n.a.		MATLAB/Simulink [III]	√

[I] Ref.[11] [II] Ref.[25] [III] Ref.[50] [IV] for STM32 microcontroller [V] Ref. [51]

SOL Multiphysics® and the ECM is implemented in MATLAB/Simulink to simulate on a desktop computer. The PP- and EM-PCM revealed different suitability in terms of constant and dynamic loads [46] and are chosen in this work for evaluating a standard, equidistant spatial FDM discretization paired with different solid-approximations on a microcontroller instead of hardware and software oversized desktop PCs, which are not suitable to evaluate embedded system applicability. The OC-PCM uses no solid-approximation and no standard FDM-dicsretization, but a reformulation of the p2D-PCM equations to exclusively ODEs in time via Chebyshev orthogonal collocation, which revealed distinct speed-up on desktop PCs compared to models like the PP- and EM-PCM. However, this reformulation must be evaluated in a microcontroller to evaluate its suitability for embedded systems, which is investigated here.

The DAE system of the p2D-PCM is shown in Table A.14 and the parameterization [2] is shown in Table A.12 and A.13.

3.1. Equivalent circuit model

The ECM consists of a single capacitor/resistor network R_1 and C_1 ('RC'), an ohmic resistance R_i and an open-circuit voltage V_{OCV}. As the temperature has significant influence on the cell behavior [52], the parameterization tests of the ECM were proceeded at 25 and 40 °C beside the state of charge (SoC) dependency. This first-order model [26] offers the best compromise of accuracy and compexity [53] incorporating the fundamental equations as

$$I_{cell} = C_1 \cdot \frac{dV_1}{dt} + \frac{V_1}{R_1}$$

$$V_{cell} = V_{OCV} + V_1 + I_{cell} \cdot R_i$$

$$m\, c_p \frac{dT_{cell}}{dt} = (V_{OCV} - V_{cell} - \frac{dV_{OCV}}{dT} \cdot T) \cdot I_{cell} - I_{cell}^2\, R_i - \alpha_\infty A_{surf}(T_{cell} - T_\infty)$$

$$\text{with } I_{cell} > 0 \text{ for charge and } I_{cell} < 0 \text{ for discharge}$$

The resistance R_i represents the ohmic resistance on the current collector foils, the RC network accounts for any transient dynamics referring to electrochemical processes [54] and the voltage source V_{OCV} represents the equilibrium state. To parameterize the ECM variables (*i.e.* V_{OCV}, R_i, R_1, C_1, $\frac{dV_{OCV}}{dT}$), three different INR18650-MJ1 cells were tested and the generated data was averaged and interpolated in 1 % SoC steps. The tests included CC, constant voltage (CV), pulse current (PC) and electrochemical impedance spectroscopy (EIS) periods as summarized in Table 2. In terms of V_{OCV}, the charge and discharge measurement were averaged to compensate cell polarization effects. The entropic coefficient $\frac{dV_{OCV}}{dT}$ was derived from accelerated rate calorimetry (ARC) [2] and validated via the potentiometric method according to Zilberman et al. [55]. The passive components R_i, R_1 and C_1 were parameterized by pulse fitting as depicted in Table 2. 88 pulses for each temperature were used and a graphical illustration of the ECM and its

Table 2: Measurements for parameterizing the ECM

Parameter	Measurement	Voltage	Current	Temperature
V_{OCV}	CC charge/discharge	2.5 - 4.2 V	0.033C	25 °C
	CV	2.5 / 4.2 V	+0.01C/-0.01C [I]	
$\frac{dV_{OCV}}{dT}$	CC charge/discharge	2.5 - 4.2 V	0.2C	ARC[Ref. [55]]
	Potentiometric method according to Zilberman et al. [55]			
R_1, C_1	PC	2.5 - 4.2 V	±0.5/1C for 10/20 s [II]	25 °C
				40 °C
R_i	PC/EIS [III]	2.5 - 4.2 V	±0.5/1C for 10/20 s [II]	25 °C
				40 °C

[I] Measurement equipement (BaSyTec CTS) defines charge > 0 and discharge < 0 [II] 1 h rest before PC, applied in 10 % SoC steps from 2.5 to 4.2 V and vice versa [III] EIS at 0.042C before PC and zero-crossing at $\mathrm{Re}\{Z\} = 0$ as initial point for the fitting algorithm of R_i

parameters is shown in our supplementary part. The input variable is the applied current I_{cell} from which the SoC variable is integrated over time. The ECM is implemented in MATLAB®/Simulink and solved via the *ode*14*x* [56] solver at a step-size of 1 s.

The solving process and the necessary parameterization files are expected neither to overload the RAM and flash memory nor to exceed the computation power of the STM32 and other works [57, 58, 59] have already shown the actual implementation in microcontrollers. As this work focusses on the implementation and solving of the p2D-PCM, the ECM is not transferred into the microcontroller but used as a benchmark for state-of-the-art model-based monitoring of LIBs in real-time operating systems and is referenced for computation speed and simulation accuracy of the MATLAB®-code PCMs.

3.2. PP- and EM-PCM using FDM and solid-diffusion approximation

The PP- and EM-PCM were already presented in our previous work [46] in terms of steady-state representation together with a non-linear Kalman Filter [60, 61, 62] for state estimation of a $LiCoO_2/LiC_6$ LIB. In this work, the MATLAB®-code PP- and EM-PCM are parameterized for a NMC-811/SiC LIB and transferred into stand-alone *C*-codes for the microcontroller.
Figure 1 shows the flow chart of the PP- and EM-PCM, which differ in the approximation of the solid-diffusion PDE ('Mass Balance (solid)'). The MATLAB® codes start with the parameterization (see Table A.12 and A.13) and calculate the initial states by assuming an equilibrium state [42]. In the main part, a new time period $\triangle t$ is added until a stop condition as t_{max}, V_{min} or V_{max} is met. The initial state vector for the iterative time step k is set to the previous, consistent solution $k-1$ and the model is run to compute a new consistent solution of the state variables $\boldsymbol{x}_k$. The iterative approximation i of the model equations refers to every node j in the electrolyte domain and calculates the model equations $\boldsymbol{g}$ (see Table A.14) and the corresponding jacobian $\boldsymbol{J}$ [46]. The Crank-Nicolson method [40] is used for first order time derivatives [46] and the time step is set to 1 s. Next, the matrix inversion [63] (A/b, MATLAB®) is used to generate the state update $d\boldsymbol{x}$. Note, that the temperature is calculated afterwards [46] according to heat generation q and heat loss to the ambience q_∞. The heat calculation proposed from COMSOL Multiphysics® and other works [49] revealed similar results with small deviations up to 0.2 /0.7 % on average for 1C CC charge/discharge as the computational less expensive calculation [52, 64, 65] used in this work (see Table A.14).

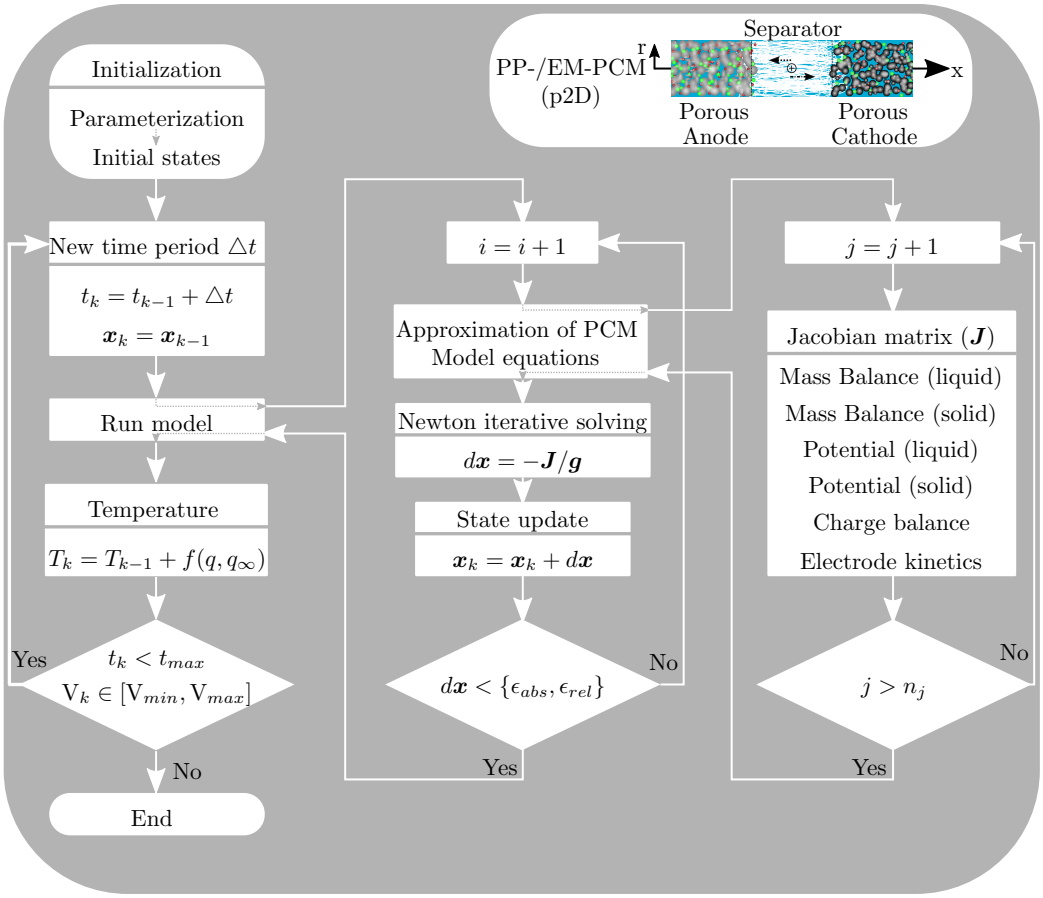

Figure 1: Simplified flow chart of the stand-alone-codes of PP- and EM-PCM implemented in MATLAB®2017b. The approximation of the solid-diffusion PDE ('Mass balance (solid)') is either implemented via the polynomial profile (PP) [11] or an eigenfunction method (EM) [25] for the PP- and EM-PCM, respectively.

To conclude, the PP- and EM-PCM use FDM together with solid-diffusion approximation and in this work we want to evaluate, if such standard techniques are sufficient to gain real-time computability of the p2D-PCM in the STM32 microcontroller.

3.3. OC-PCM using orthogonal collocation on Chebyshev nodes

The OC-PCM uses a spectral method to reformulate the spatial discretization on Chebyshev collocation nodes [49] of the DAEs in both the x and r domain. The resulting ODEs in time and algebraic equations (AEs) form a DAE system as

$$\boldsymbol{M}\boldsymbol{x'} = \boldsymbol{f}(t_k, \boldsymbol{x})$$

which is solved via the *ode15s* [66, 67, 68] solver of MATLAB®. In terms of the reformulation, an unknown continuous function is approximated by a polynomial, which is determined by its values at the so called Chebyshev nodes (*i.e.* $x_j = cos\,(\frac{\pi\, j}{n_j})$) for a given number of nodes j after rescaling each domain into $[-1, 1]$ [49]. In contrast to other work [49], the electrolyte potential Φ_l accounts for activity formulation $f_\pm$ [6] as

$$\frac{\partial \Phi_l(x,t)}{\partial x} = -\frac{i_l(x,t)}{\kappa_l^{eff}} + \frac{2\,RT}{F}(1 - t_+^0) \cdot \left[1 + \frac{d\,ln\,f_\pm}{d\,ln\,c_l(x,t)}\right] \cdot \frac{\partial\, ln\, c_l(x,t)}{\partial x}$$

and the temperature calculation is identical to the PP- and EM-PCM [52, 64, 65]. Figure 2 shows the flow chart of the OC-PCM. It starts with the calculation of the required Chebyshev differentiation matrix $\boldsymbol{D}$, which is calculated once together with the mass matrix $\boldsymbol{M}$ and the Clenshaw quadrature weights $\boldsymbol{\omega}$ to evaluate the sum of finite integrals to determine the jacobian matrix $\boldsymbol{J}$. Next, the initialization of the ODE solver (*ode15s* [66, 67, 68]) is defined via calculating an initial jacobian (*daeic12* [69]) and a first time step is estimated. The following Newton iterative solving uses the same thresholds for the tolerances ϵ_{abs} and ϵ_{rel} within the cell voltage range and the time span as given for the PP- and EM-PCM. If four iterations offer no convergence, the solver updates the jacobian and respectively the iteration

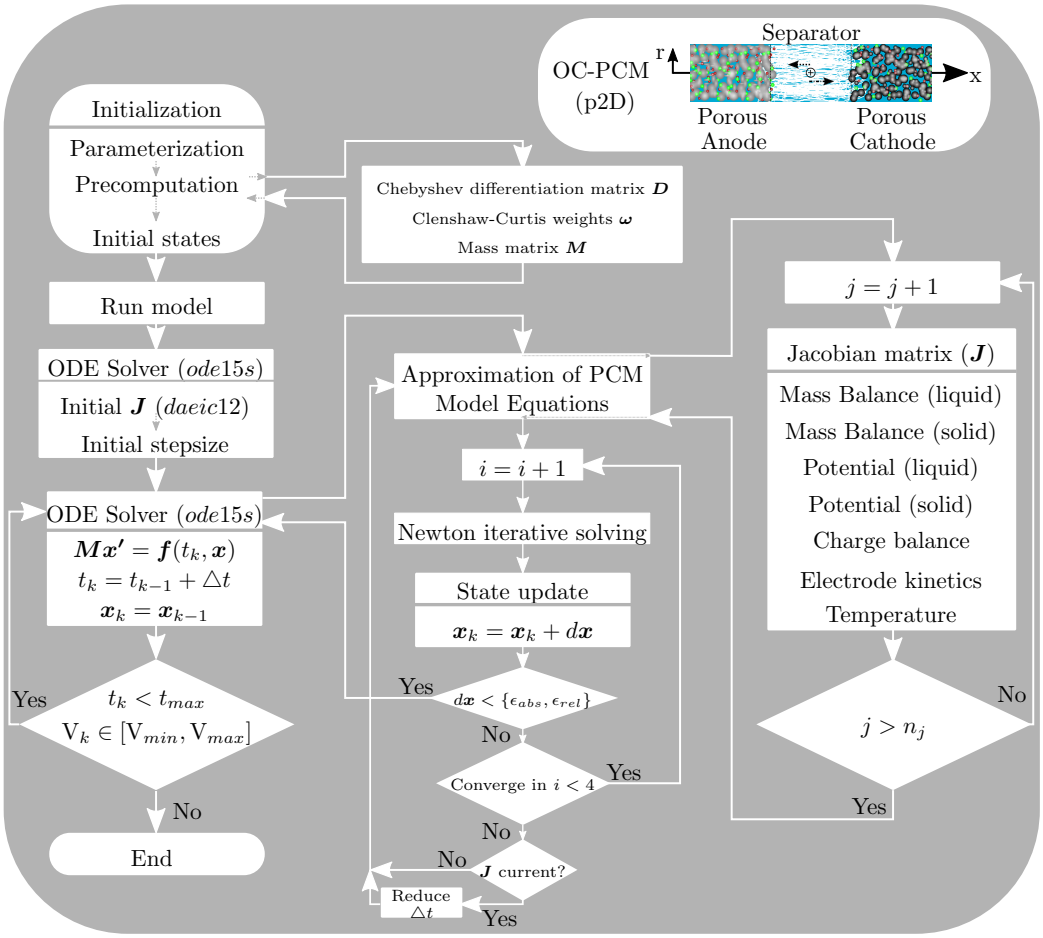

Figure 2: Simplified flow chart of the stand-alone-code of the OC-PCM implemented in MATLAB®2017b. The spatial discretization is reformulated using Chebyshev orthogonal collocation [49] and the resulting DAE system is solved using an ODE solver (*ode15s* [66, 67, 68]) of MATLAB®.

matrix $\boldsymbol{M}_i$ and the process starts again. If the jacobian is current and no convergence is expected, the step-size is decreased [66]. The solver itself uses a linearly implicit, one-step method based on numerical differentiation formulas (NDFs) implemented in backward differences [66, 67, 70], which uses

an iteration matrix $\boldsymbol{M}_i$ as

$$\boldsymbol{M}_i = \boldsymbol{M} - \frac{\triangle t}{(1-k)\boldsymbol{\gamma}_k} \cdot \boldsymbol{J}$$

to generate the state update $d\boldsymbol{x}$. The term $\boldsymbol{\gamma}_k$ represents the coefficients of the NDFs [66], k the order of the NDF and κ is a scalar factor [70]. At this point, the reader is referred to the original work [66, 67] for more information about the solver itself and the iterative state update is outlined in more detail in our supplementary part.

Note, the jacobian is calculated analytically at every spatial node j for every derivative $\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{y}}$ and passed directly to the solver instead of using the incorporated *ode15s* FDM. Thus, a discretization of 10-10-10-30 (*i.e.* 'anode-separator-cathode-particle' domain) reveals approximately a 20 times faster calculation as seen in this work. Further improvement was achieved by using sparse structure (*sparse* [69]) of the jacobian and the mass matrix. Even if the used spectral method leads to full differentiation matrices while the jacobian for the DAE system is still sparse ($\approx$4 % are non-zero), using sparse linear algebra reduces the computational cost by a factor of 4 (referring to 10-10-10-30), which tends to increase for finer discretizations.

In sum, the OC-PCM uses reformulation, which shows significant computational acceleration of the solving process on a desktop computer. In this work we evaluate the transfer of the OC-PCM into a stand-alone *C*-code including the ODE solver and the real-time ability of simulating a LIB on a microcontroller.

3.4. Rigorous COMSOL-PCM

As a benchmark, the *liion*-model [51] of COMSOL Multiphysics® is used and run via the LiveLink [44] application using MATLAB®2017b. The approximation functions are set to linear and a total 53, 8 and 40 of spatial nodes in the anode, separator and cathode domain are used with 20 nodes in the particle domain. The temperature is calculated with a single ODE (see Table A.14). The DAE system is solved with the 'Multifrontal massively parallel sparse direct solver' (MUMPS) [71] at a fixed step-size of 1 s.

3.5. Spatial configuration and DAE size of the PCMs

The spatial discretization for the PP- and EM-PCM is denoted as n_{neg} - n_{sep} - n_{pos}, which corresponds to the respective number of nodes in the anode, separator and cathode domain. The total number of DAEs calculates as

$$n_{\mathrm{DAEs}} = (n_{\mathrm{neg}} + n_{\mathrm{pos}} + 2) \cdot 6 + (n_{\mathrm{sep}} - 1) \cdot 3$$

referring to the boundary interfaces ('+2') and electrode domains ('$n_{\mathrm{neg}} + n_{\mathrm{pos}}$') with six ($c_l$, c_s, i_l, j_n, Φ_l, Φ_s) and three DAEs (c_l, i_l, Φ_l) at the internal nodes of the separator ('$n_{\mathrm{sep}} - 1$'), respectively. In terms of the OC-PCM, the spatially discretized particle domain ('n_{P}') must be included as well as the calculation of the temperature ('+1'):

$$n_{\mathrm{DAEs}} = (n_{\mathrm{neg}} + n_{\mathrm{sep}} + n_{\mathrm{pos}} - 2) \cdot 2 + (n_{\mathrm{P}} + 3) \cdot (n_{\mathrm{pos}} + n_{\mathrm{pos}}) + 1$$

Again, the term '-2' is referring to the boundary interfaces of the electrodes and the separator, which are implemented in a common node for the definition of c_l and i_l. The term '$n_{\mathrm{P}} + 3$' refers to the solid-concentration c_s and the molar flux j_n/ionic current density i_l as well as the solid-potential Φ_s.

Table 3 summarizes the spatial discretizations used in this work for the p2D-PCMs with the corresponding number of DAEs.

Table 3: Spatial discretizations of the PCMs

Model	**Indices**	**Number of spatial nodes**				**Number of DAEs**
		Anode	Separator	Cathode	Particle	
PP-PCM	1-1-1	1	1	1	PP [I]	24
	2-1-2	2	1	2		36
	5-3-5	5	3	5		78
	10-5-10	10	5	10		144
EM-PCM	1-1-1	1	1	1	EM [II]	24
	2-1-2	2	1	2		36
	5-3-5	5	3	5		78
	10-5-10	10	5	10		144
OC-PCM	5-3-5-2	5	3	5	2	73
	5-3-5-3	5	3	5	3	83
	5-3-5-5	5	3	5	5	103
	20-10-20-25	20	10	20	25	1217
COMSOL-PCM	53-8-40-20	53	8	40	20	2338 [III]

[I] Ref. [11] [II] Ref. [25] [III] referring to linear element order

4. Microcontroller implementation

Primarily the small-sized RAM of microcontrollers and low processor frequencies imply challenges for solving the p2D-PCM, which poses no challenge for a standard desktop computer equipped exemplarily with 16 GB RAM at 3.2 GHz as used in this work for the MATLAB®-code PCMs. 192 kB of RAM and a maximum of 168 MHz are offered by the chosen microcontroller to solve the *C*-code p2D-PCMs in this work.

Beside working without an operating system and with hardware modules like universal asynchronous reciever transmitter (UART) for data transmission, the transfer from the scripting language MATLAB® to the programming

language C is a significant step as some framework related options such as matrix inversion (A/b [63]), linear algebra operations (*sparse* [69]) or solvers (*ode15s* [66, 67, 68]) are not available and must be transferred without overloading the memory. Note, that most of these specific functions cannot be exported via the *MATLAB® to C* export option [72] and even if, they would not be necessarily runnable on a microcontroller.

Basically, the hardware abstraction layer (HAL) library (Cortex microcontroller software interface standard (CMSIS), ARM [73]) and the STM32 CubeMX software [74] were used to configure the system clock, peripherals and an initial code structure. The flow chart of both stand-alone C-codes (PP-/EM-PCM and OC-PCM) are shown in Figure 3. Via running the C-code PCMs on the STM32 and sending the simulation results (UART-to-USB converter) to a desktop computer, computation efficiency and simulation accuracy can be analyzed and compared to the correponding MATLAB®-code PCMs. The analysis using the MATLAB®-code PCMs on a desktop computer is not useful to evaluate the performance in low-hardware/software environment, as multi-threading calculation, oversized memory capacities, the comprehensive operating system and the framework MATLAB® itself would distort the results.

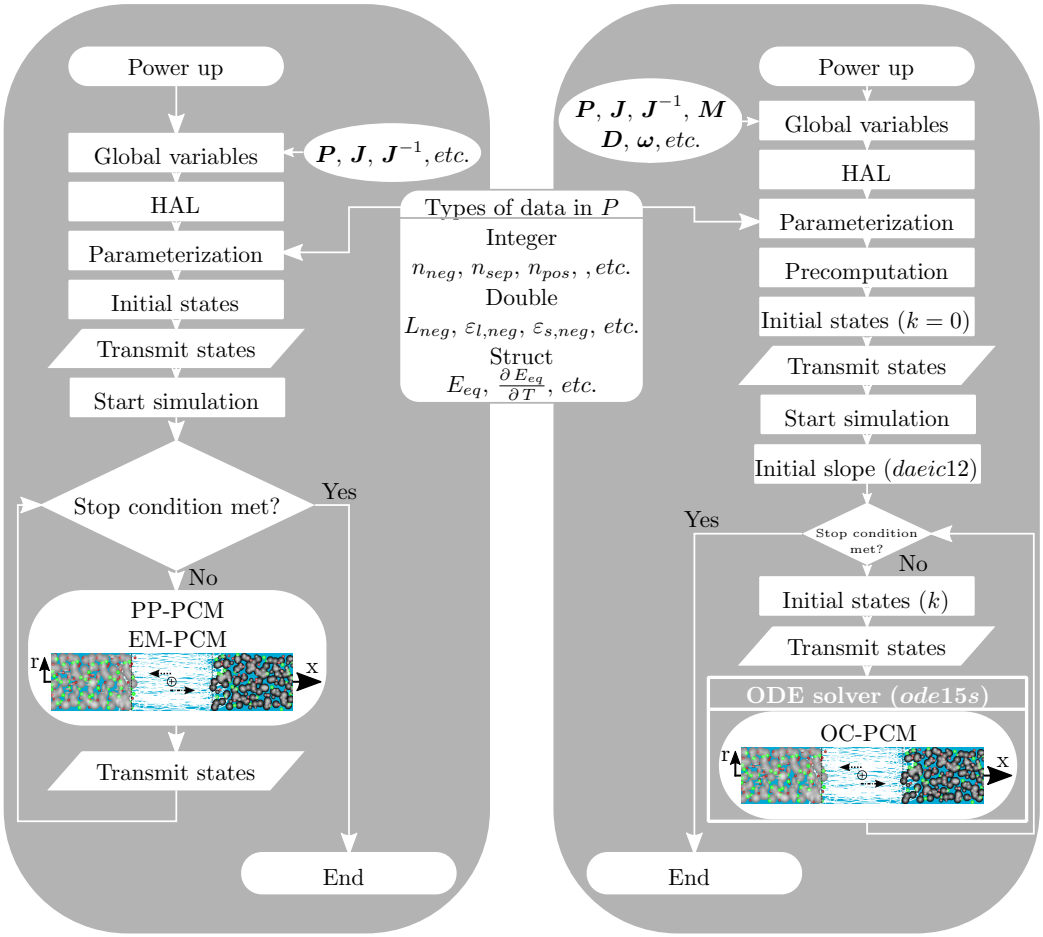

Figure 3: Flow chart of the stand-alone *C*-codes of the PP-/EM-PCM (left) and the OC-PCM (right) implemented in the microcontroller (STM32F407VGT6, STMicroelectronics [9]). The PP-/EM-PCM and OC-PCM routines refer to the p2D-PCM models shown in Figure 1 and 2. The generated simulation results are sent per converged time step via an UART-to-USB converter to a desktop computer and evaluated in terms of computation performance and simulation accuracy.

4.1. PP- and EM-PCM stand-alone C-code

After power up (see Figure 3, left), global variables with fixed memory allocation are set, which are accessible in any case whilst the remaining variables are allocated and freed with every function call. Next, the processor calls the main function to configure the system clock, initialize the peripherals, load the parameterization including analytical functions, look-up tables and single parameters, set the stop conditions and call the initialization. Via UART the messages to be transmitted are intialized and the initial states are sent to the desktop computer using the UART-to-USB converter. The main loop is entered next and the time simulation is started. It ends as soon as a stop condition is met (see Figure 1). Note, that only the current state k and the previous state $k-1$ are stored on the STM32 - otherwise the microcontroller's memory would be exceeded after a short time period. The main loop runs the PCM as depicted in Figure 1 and transmits the current states to the desktop computer at every converged time step.

In detail, the interpolation of the look-up tables E_{eq} and $\frac{\partial E_{eq}}{\partial T}$ are defined on the STM32 via spline interpolation at the junction nodes, which offers differentiability compared to piece-wise linear approximation. Based on a MATLAB® structure (*spline* [75]), the implementation includes the coefficients c, number of pieces s, order of polynomials l, range of the measured data and a pointer to the array of single intervals to define a spline as [50, 76]:

$$f(x) = c_{0,s}(x - x_s)^{l-1} + c_{1,s}(x - x_s)^{l-2} + \cdots + c_{l-1,s}(x - x_s) + c_{l,s}$$

where the coefficients differ in each knot interval of

$$x \in [x_s, x_{s+1}[$$

Not only the value of two adjacent intervals are matched but also their deriva-

tives, which is a crucial point when calculating the jacobian. To reduce memory allocation, the coefficients are stored in an array and evaluated via pointer function. Using double precision [77], a total of 751 and 41 knots are used for E_{eq} and $\frac{\partial E_{eq}}{\partial T}$ allocating 6008 and 328 Byte, respectively.

The matrix inversion to calculate the inverse of the jacobian $\boldsymbol{J}^{-1}$ uses a MATLAB® function (A/b [63]) in the MATLAB®-code PCMs. As standard C-algorithms (*e.g.* CMSIS) failed, the Gauss-Jordan (GJ) algorithm [78] was implemented, which is not a matrix-type specified algorithm like existing, tridiagonal-block-type algorithms (*e.g.* BAND(j), [79]). As floating point numbers are used in the processor, the highest accuracy is gained at low absolute values. In order to minimize the error caused by performing floating point operations during the matrix inversion, the concept of pivoting [80] is applied, which leads to lower absolute values and thus higher accuracy. In addition, the STM32 provides a floating point unit (FPU) of single precision [77]. Before the inversion starts, the matrix-entries are converted into single precision, next the inversion takes place on the FPU and the results are converted into double precision in the end. The related loss of accuracy and computation speed up is discussed in this work.

The 'Transmit states' action (see Figure 3) uses a virtual COM port between the STM32 and the desktop computer and a UART-to-USB converter (115200 bit s^{-1}), where the COM port is evaluated via a MATLAB® script. In terms of RAM, the size of n_x (= 6) state variables $\boldsymbol{x}$, both the jacobian $\boldsymbol{J}$ and its inverse $\boldsymbol{J}^{-1}$ as well as the calculation of the solid-diffusion approximation $\boldsymbol{x}_s$ determine the total size in Byte:

$$\text{size}(\boldsymbol{J}, \boldsymbol{J}^{-1}) \;=\; 2 \cdot (n_j \cdot n_x)^2 \cdot 8\,\text{Byte}$$

$$\text{size}(\boldsymbol{x}) \;=\; (n_j \cdot n_x) \cdot 2 \cdot 8\,\text{Byte}$$
$$\text{size}(\boldsymbol{x}_s) \;=\; (n_j \cdot n_s) \cdot 2 \cdot 8\,\text{Byte}$$
$$\text{with a total number of nodes:}$$
$$n_j \;=\; n_{neg} + n_{sep} + n_{pos} + 1$$

The multiplication with '2' for $\boldsymbol{x}$ and $\boldsymbol{x}_s$ is necessary for the current and previous state. The additional states n_s are two [11] for the PP- and six [25] for the EM-PCM. Table 4 shows exemplarily the possible discretizations and RAM/flash memory usage in the STM32 for the PP- and the EM-PCM. The

Table 4: Memory usage of the stand-alone *C*-code PP- and EM-PCM

Model	**PP-PCM**			**EM-PCM**		
Discretization	1-1-1	2-1-2	5-3-5	1-1-1	2-1-2	5-3-5
size($\boldsymbol{J}, \boldsymbol{J}^{-1}$) in Byte	9216	20736	112896	9216	20736	112896
size($\boldsymbol{x}$) in Byte	384	576	1344	384	576	1344
size($\boldsymbol{x}_s$) in Byte	128	192	448	384	576	1344
RAM						
Total size in Byte	9728	21504	114688	9984	21888	115584
Memory allocation in % [I]	5.1	11.2	58.8	5.2	11.4	60.2
Flash memory						
Total size in kB	173.66	173.95	174.58	177.61	177.88	178.55
Memory allocation in % [II]		$\approx$ 17 %			$\approx$ 17.4 %	

[I] Referring to the STM32 with 192 kB of RAM [I] Referring to the STM32 with 1024 kB flash memory

maximum runnable spatial configuration included 14 nodes in total as enough memory space for the variables and the solving process must be reserved. The minimum converging setup was found to be 2-1-2. The increase in memory by using the EM-approximation is negligible regarding $\boldsymbol{x}_s$ in reference to the PP-approach. The major influence is seen in the increase of spatial discretization as the jacobian size increases as well and the overall RAM usage increases

quadratically.
In sum, the PP- /EM-PCM on the microcontroller need at least $\approx$ 10 kB up to a maximum of 115 kB during calculation and the maximum flash memory allocation consumed around 174 /178 kB of the maximum 1024 kB flash memory ($\approx$ 17 /17.4 %).

4.2. OC-PCM stand-alone C-code

The structure of parameters, interpolation schemes and communication to the desktop computer of the *C*-code OC-PCM is similar to the *C*-code PP- /EM-PCM. A specified version of the *ode15s* solver [66, 67, 68] is developed in *C* offering main functionalities as

- Calculation of initial jacobian (*daeic12* [68])
- Initial step estimation $k = 0$
- Calculation of iteration matrix $\boldsymbol{M}_i$
- Iteration with simplified Newton method using GJ-inversion for $\boldsymbol{M}_i$
- Calculating new jacobian $\boldsymbol{J}$
- Adjusting step-size $\triangle t$

For calculating an initial jacobian, *daeic12* [68] was extracted from the *ode15s* solver and transferred into *C* right after setting the initial values (see Figure 3). The *sparse* function [69] was adopted to gain a sparse jacobian via neglection of any zeros and the non-zero entries are stored in an array $\frac{d\boldsymbol{f}}{d\boldsymbol{x}}$ together with the respective index coordinate pair $\frac{d\boldsymbol{f}}{d\boldsymbol{x}}|_{index}$ to save memory. The estimation of the initial step-size is performed according to Curtis *et*

al. [81]. The iteration matrix $\boldsymbol{M}_i$ is obtained at every iteration and uses the previously calculated jacobian (see Figure 2). The main loop integrates from the previous state $k-1$ to the current state k and uses simplified Newton method [67, 82] incorporating the GJ-inversion [78] for inversing $\boldsymbol{M}_i$ to generate the state update $d\boldsymbol{x}$. Step-size reduction and new jacobian calculation are implemented as described in section 3.3.

Regarding the memory allocation, the iteration matrix $\boldsymbol{M}_i$ contributes as

$$\text{size}(\boldsymbol{M}_i) \;=\; \{(n_{neg}+n_{pos})(m+5)+2n_{sep}-3\}^2 \cdot 8\,\text{Byte}$$

Similar to the jacobian of the PP-/EM-PCM, the memory usage of the $\boldsymbol{M}_i$ increases quadratically and the spatial discretization in the particle domain m is here the main driver. The analytical calculation of the jacobian $\frac{d\boldsymbol{f}}{d\boldsymbol{x}}$ is stored as array of the non-zero entries as

$$\text{size}(\tfrac{d\boldsymbol{f}}{d\boldsymbol{x}}) \;=\; \{n_{neg}(5n_{neg}+3m+6)+n_{sep}(3n_{sep}-2)+n_{pos}(5n_{pos}+3m+6)-5\}\cdot 8\,\text{Byte}$$

Therefore, the indices $\frac{d\boldsymbol{f}}{d\boldsymbol{x}}|_{index}$ of the position in the jacobian (*i.e.* row and column) are stored as integers with 4 Byte each

$$\text{size}(\tfrac{d\boldsymbol{f}}{d\boldsymbol{x}}|_{index}) \;=\; 2\cdot\{n_{neg}(5n_{neg}+3m+6)+n_{sep}(3n_{sep}-2)+n_{pos}(5n_{pos}+3m+6)-5\}\cdot 4\,\text{Byte}$$

Improvement using unsigned 16 bit integer is optional but not considered in this work. The mass matrix $\boldsymbol{M}$ is stored as integer to

$$\text{size}(\boldsymbol{M}) \;=\; \{(n_{neg}+n_{pos})(m+5)+2n_{sep}-3\}\cdot 4\,\text{Byte}$$

and the backward differences for the NDFs [66, 70] are crucial to find a consistent solution and their memory usage amounts to

$$\text{size}(\nabla^m) \;=\; 7\cdot\{(n_{neg}+n_{pos})(m+5)+2n_{sep}-3\}\cdot 8\,\text{Byte}$$

The factor '7' is required by the integration order and two additional states are saved for following iterations [66, 70]. The variable $\boldsymbol{x}$ stores the states

$$\text{size}(\boldsymbol{x}) \;=\; 2 \cdot \{(n_{neg} + n_{pos})(m + 5) + 2n_{sep} - 3\} \cdot 8\,\text{Byte}$$

and the factor '2' accounts for the previous stored iteration. The maximum runnable spatial configuration 5-3-5-5 is in the same memory range as for the PP-/EM-PCM, whereas the most coarse configuration 5-3-5-2 offering sufficient accuracy needs at least five resp. three nodes in the electrode resp. separator domain for the NMC-811/SiC parameterization. Note, that the minimum converging configuration appears to 3-2-3-2, but was neglected due to insufficient simulation accuracy. Table 5 shows the range of runnable *C*-code OC-PCMs on the STM32.

In sum, the OC-PCM on the microcontroller needs at least 54 kB RAM during calculation and the maximum allocates around 100 kB. The overall model size uses around 212 kB of the maximum 1024 kB flash memory ($\approx$ 20.7 %).

Table 5: Memory usage of the stand-alone *C*-code OC-PCM

	OC-PCM		
Discretization	5-3-5-2	5-3-5-3	5-3-5-5
size($\boldsymbol{M}_i$) in Byte	42632	55112	84872
size($\frac{d\boldsymbol{f}}{d\boldsymbol{x}}$) in Byte	3088	3328	3808
size($\frac{d\boldsymbol{f}}{d\boldsymbol{x}}\|_{index}$) in Byte	3088	3328	3808
size($\boldsymbol{M}$) in Byte	292	332	412
size(∇^m) in Byte	4088	4648	5768
size($\boldsymbol{x}$) in Byte	1168	1328	1648
RAM			
Total size in Byte	54356	68076	100316
Memory allocation in % [I]	28.3	35.5	52.2
Flash memory			
Total size in kB	212.08	212.27	212.95
Memory allocation in % [II]		$\approx$ 20.7 %	

[I] Referring to the STM32 with 192 kB RAM [II] Referring to the STM32 with 1024 kB flash memory

5. Results and discussion

5.1. Validation and computational efficiency under constant load scenarios

The experimental validation of the three PCMs and the benchmark models is shown via thermographic measurements of the INR18650-MJ1 LIB at different CC charge (0.2/0.5/1C) and discharge (0.2/0.5/1/1.5/2C) rates at 25 °C ambient temperature under convective cooling as presented in previous work [2]. A desktop computer equipped with a Intel(R) Core(TM) i5-6500 CPU at 3.20 GHz processor and 16 GB of RAM was used for calculation. The mean cell voltage error (a, b), temperature error (c, d), overall simulation time (e, f) and averaged calculation time per step (g, h) are shown in

Figure 4. Regarding CC charging in Figure 4 a and c, the mean cell voltage error ranges from 10 mV for the ECM up to 20 mV for the OC-PCM until 0.5C and increases for the ECM at 1C up to 29 mV while a decreasing trend can be seen for all PCMs. The mean temperature error appears below 0.6 K for all models. Regarding discharge, the ECM matches quite well the measurements with errors on average below 21 mV and 0.1 K until 2C while all PCMs show increasing errors for higher C-rates (*e.g.* max. 68 mV for the PP-/EM-PCM and max. 1.5 K for the OC-PCM at 2C CC). The ECM shows increased modeling errors at 1C charging, which may be linked to limitations in the pore of the porous electrodes [2] and improvements can be achieved via using physically more meaningfull, distributed-parameter ECMs [83] or adaptive, online-paramater estimation [84].

In general, all PCMs show increasing errors with higher applied C-rates (> 1C) as inhomogeneities along the electrodes (61.5×5.8 cm, width×height) increase as shown in our previous work [2], which cannot be modeled with a single-PCM as the electrode utilization is assumed to be homogeneous. Nevertheless, certain differences appear between the MATLAB®-code PCMs compared to the COMSOL-PCM. The temperature calculation is included in the jacobian of the OC-PCM, which is not implemented for the remaining PCMs and may have a significant influence on the cell voltage and temperature calculation beside the different spatial discretization and approximation schemes.

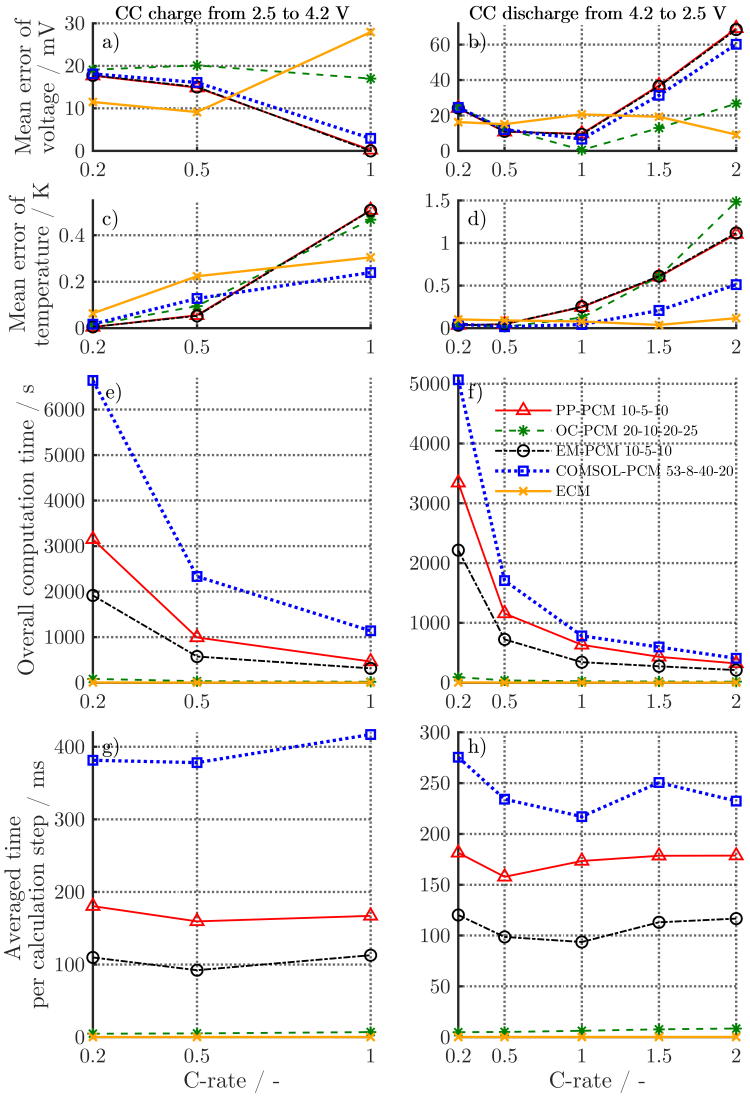

Figure 4: Experimental validation with the INR18650-MJ1 NMC-811/SiC LIB of the simulation results including the ECM, the MATLAB®-code PCMs ('PP-PCM 10-5-10', 'OC-PCM 20-10-20-25' and 'EM-PCM 10-5-10') and the rigorous benchmark PCM ('COMSOL-PCM 53-8-40-20'). The mean cell voltage error (a, b) and temperature error (c, d) are shown together with the overall computation time (e, f) and averaged calculation time per converged time step (g, h) for the different CC charge (a, c, e and g) and discharge (b, d, f and h) tests at 25 °C ambient temperature and convective cooling conditions [2].

The reformulation and solving of the solid-diffusion PDE in the OC-PCM instead of using approximations contributes to the different cell voltages compared to the PP- and EM-PCM. The error for the OC-PCM remains at ≈ 22 mV for all scenarios, whereas the used approximations (PP [11] and EM [25]) lead to increased errors for 1.5 and 2C discharge (≈ 68 mV at 2C). As the *liion*-module [51] for the COMSOL-PCM offers only the particle concentration at the center, the surface and on average, an approximation may be used here as well, which may explain the appearing deviations.

In sum, all MATLAB®-code PCMs reveal sufficient accurate simulation of the electrochemical-thermal behaviour throughout the thickness of the NMC-811/SiC electrode stack under CC charge and discharge scenarios compared to the measured electrical-thermal behaviour of the INR18650-MJ1 LIB. Increasing errors appear, when inhomogeneities of the current density [2] along the electrodes are expected to increase for high applied C-rates (> 1C).

The computation times in Figure 4 e and f of the COMSOL-PCM appear to be the slowest and the fastest appear for the ECM, as expected. The same trend can be seen for the averaged calculation time per step (see Figure 4 g and h). Approximately 6.3 ms per step are needed at 1C CC discharge for the OC-PCM, while the ECM needs only 0.2 ms. This results in an approximately 32, 470, 865 and 1085 times faster computation for the ECM compared to the OC-, EM-, PP- and COMSOL-PCM, respectively. The benefit of using explicit functions of state and input variables to solve the ECM [83] instead of solving the DAE system of the PCM is not questioned here. Even if large frameworks are used, lean computational costs confirm that implementation in the microcontroller would lead to similar results.

Therefore, the ECM is used as benchmark only and not implemented in the microcontroller.

Around 16, 67 and 26 % of the desktop computer's CPU and approximatly 660, 670 and 793 MB RAM for the PP- /EM-, OC- and COMSOL-PCM are used at a full 1C CC discharge. Considering also the computation time, the OC-PCM may be most suitable for fast computation but preferring low memory usage and CPU load, the PP- or the even faster calculating EM-PCM seem more suitable. The ECM allocates $\approx$ 1400 MB at $\approx$ 10 % CPU, which is mainly caused by using the MATLAB/Simulink framework.

The simulation results are run on a desktop computer offering sufficient computation ressources of 3.20 GHz and 16 GB RAM. The STM32 offers maximum 168 MHz and 192 kB RAM and evaluating the most suitable PCM for embedded systems must be based on adequate conditions as proposed in control devices for a BMS, which are similar to the microcontroller of this work and offer similary no commercial framework tool or a sophisticated operating system.

5.2. Reducing memory allocation via coarser spatial discretizations

For implementation in the microcontroller, the number of spatial elements in the PP-/EM-PCM and the OC-PCM is gradually reduced to save RAM and decrease computation time. The related increase of modeling error for the cell voltage V_{cell}, temperature T_{cell}, surface concentration $c_{s,s}$ and the potential drop $\Phi_s - \Phi_l$ both at the anode-separator interface is analyzed in reference to the validated MATLAB®-code PCM configurations shown in Figure 4. Figure 5 shows the increase of error exemplarily for the 1C CC charge and discharge scenario.

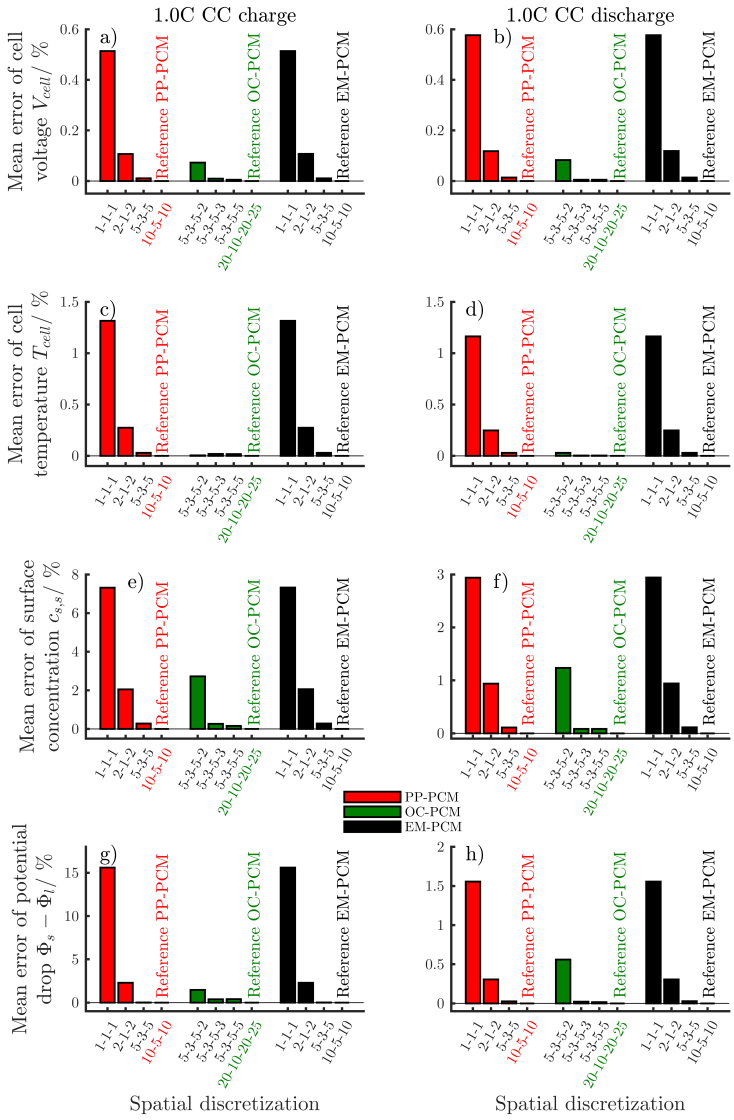

Figure 5: The relative (%) increase of modeling error for the cell voltage (a, b), the cell temperature (c, d), the surface concentration (e, f) and the potential drop $\Phi_s - \Phi_l$ (g and h) both located at the anode-separator interface $x = L_{neg}$ is shown for gradually reducing the total number of spatial nodes in the MATLAB®-code PCMs (PP-, EM- and OC-PCM) in reference to the experimentally validated PCMs shown in Figure 4('PP-PCM 10-5-10', 'EM-PCM 10-5-10' and 'OC-PCM 20-10-20-25').

Regarding the global cell variables V_{cell} and T_{cell}, similar error increase appears for charge and discharge for all PCMs. The PP- and EM-PCM show mean cell voltage error increase of $\approx$ 0.6 % for the minimum configuration (1-1-1), whereas the OC-PCM shows errors below 0.1 % in all cases. The deviance for the cell temperature is around 1.2 % for the PP- and EM-PCM, whereas nearly no deviance could be seen for the reduced configurations of the OC-PCM. Note, a minimum of 5 and 3 nodes for the electrodes and separator domain in the OC-PCM appeared for guaranteeing convergence.
Regarding the internal variables $c_{s,s}$ and $\Phi_s - \Phi_l$, higher deviances for the charge than for the discharge scenario appear with the chosen parameterization, which differs between charge and discharge to account for hysteresis effects of the open-circuit potentials [2]. Regarding the most coarse discretization, errors up to 7.3 /3 % compared to 2.7 /1.2 % for charge/discharge of the PP-/EM-PCM and the OC-PCM respectively appeared for $c_{s,s}$. In terms of $\Phi_s - \Phi_l$, the errors increase up to 16 % for charging regarding PP-/EM-PCM, whereas lower errors appear for the OC-PCM ($\approx$ 1.4 %).
The results indicate no distinct difference between the PP- and EM-PCM, when lean spatial discretization is chosen even at the lowest configuraiton of 1-1-1. The OC-PCM shows less error with decreasing number of nodes when mainly the spatial discretization in the particle is reduced - which reduces the overall size of the DAE enormously. In general, the error on global varibles such as cell voltage and temperature seems acceptable but when internal variables are used such as the potential drop at the anode-separator interface to indicate the onset of lithium plating [4, 85], disctinct errors due to a lean spatial discretization must be considered for interpreting the results

correctly.

In terms of the microcontroller, spatial configurations of 2-1-2 and 5-3-5 are used for the PP- and EM-PCM as similar mean errors regarding 2-1-2 of 4 mV (0.12 %), 7E-2 K (0.25 %), 84 $\mathrm{mol\,m^{-3}}$ (0.94 %) and 0.8 mV (0.3 %) are expected due to modeling error at 1C CC charge and discharge, which offer still sufficient accuracy to describe accurately the NMC-811/SiC INR18650-MJ1 LIB. In terms of the OC-PCM, the 5-3-5-3 and 5-3-5-5 are used which show maximum averaged errors regarding 5-3-5-3 of 0.25 mV (5E-3 %), 2E-3 K (2E-3 %), 6.7 $\mathrm{mol\,m^{-3}}$ (8E-2 %) and 6E-2 mV (2E-2 %).

5.3. Stand-alone C-code models on the microcontroller under constant load scenario

The CC charge and discharge loads as shown in section 5.1 are simulated on the STM32 using the stand-alone *C*-codes of the PP-, EM- and OC-PCM. The simulations incorporate the coarse 2-1-2, 2-1-2 and 5-3-5-3 and the maximum 5-3-5, 5-3-5 and 5-3-5-5 spatial discretizations, respectively. Figure 6 shows the mean cell voltage and temperature error in reference to the corresponding MATLAB®-code. The computational performance is analyzed via the mean iteration time and total number of iterations per 1 s time step on the STM32 processor at 168 MHz. Table 6 summarizes the analysis at 1C CC charge and discharge. The lowest mean cell voltage error for charge and discharge appears for the PP-PCM (< 0.4 mV) and the highest appears up to 3.8 mV for the EM-PCM. Reducing the number of spatial nodes (see 'Coarse configurations' in Figure 6), leads to higher deviations regarding the PP-PCM whereas the EM- and OC-PCM reveal less deviations. In terms of the cell temperature for charge and discharge, similar trends can be seen

except for the lowest temperature error, which is seen for the OC-PCM.

Table 6: Performance of *C*-code PCMs on the microcontroller at 1C CC

Model	**PP-PCM**		**EM-PCM**		**OC-PCM**	
Discretization	2-1-2	5-3-5	2-1-2	5-3-5	5-3-5-3	5-3-5-5
1C CC *charging*						
Mean $\triangle V_{cell}$ / mV [I]	0.255	0.019	0.3956	1.665	0.053	0.262
Mean $\triangle T_{cell}$ / K [I]	0.103	0.003	0.003	0.117	<1E-3	<1E-3
Mean number of iterations per step / -	4.02	4.31	2.17	2.23	1.22	1.26
Mean iteration time per step / ms	219	1052	124	560	550	932
1C CC *discharging*						
Mean $\triangle V_{cell}$ / mV [I]	0.388	0.069	2.126	3.603	0.032	0.195
Mean $\triangle T_{cell}$ / K [I]	0.08	0.08	0.10	0.17	<1E-3	<1E-3
Mean number of iterations per step / -	3.72	4.17	1.56	2.14	1.25	1.18
Mean iteration time per step / ms	205	1021	90	540	563	887

[I] in reference to the corresponding MATLAB®-code PCM with the same discretization

In sum, all *C*-code PCMs on the STM32 show sufficient accuracy below 4 mV of cell voltage and 0.4 K of cell temperature error, which is mainly caused by approximations and rounding errors. The analysis of the internal states such as concentrations and potentials is neglected here as similar low error ranges appear.

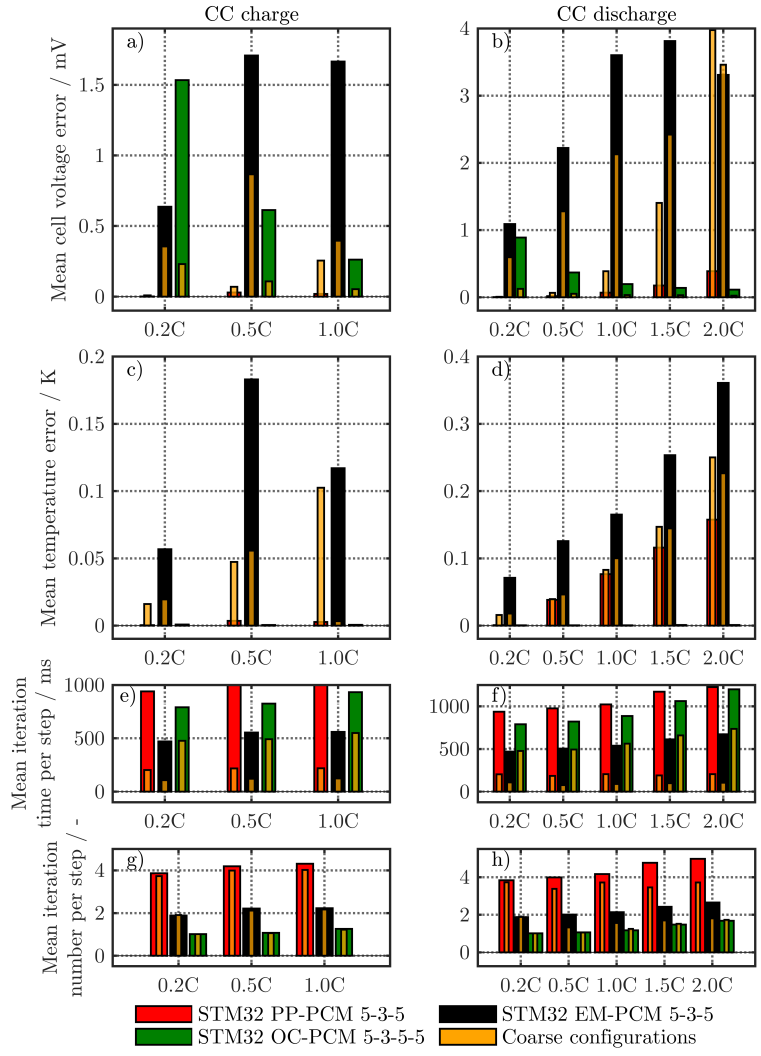

Figure 6: Evaluation of the simulation accuracy (a, b, c and d) and computational efficiency (e, f, g, and h) of the stand-alone *C*-code PP-, EM- and OC-PCM under constant current charge (0.2, 0.5, and 1C) and discharge (0.2, 0.5, 1, 1.5 and 2C) scenarios. The mean cell voltage (a, b) and temperature error (c, d) is shown in reference to the corresponding MATLAB®-code PCM for the coarse (2-1-2, 2-1-2 and 5-3-5-3 for PP-, EM- and OC-PCM, 'Coarse configurations') and maximum spatial configuration. The computational efficiency is shown similary in form of the iteration time (e, f) and the mean number of iteration per 1 s step (g, h).

Regarding the computational performance in Figure 6 e and f, the fastest computation appears for the EM-PCM (max. 560 ms per step with min. 2 iterations) whereas the slowest is seen for the PP-PCM (max. 1052 ms per step with min. 4 iterations). When the coarse discretization is used, the OC-PCM reveals the slowest calculation (max. 563 ms per 1 s step). For the OC-PCM, the 1 s time step is set as a maximum as the solver routine is implemented with the option, to reduce the step-size if no convergence appears (see Figure 2) and the OC-PCM is thus more susceptible to prolong the overall computation time. Thus, the minimal number of iterations appears for the OC-PCM and the PP-PCM needs the most iterations. The spatial discretization has less significant influence and only slightly reduces the number of iterations.

In sum, the EM-PCM is the fastest calculating model with 90 respectively 540 ms per 1 s time step for an entire 1C CC discharge using coarse respectively maximum spatial discretization in the STM32. The OC-PCM offers still calculation times below the real-time threshold even if the solver routine reduces the stepsize during runtime. The PP-PCM is most likely to require the longest calculation time and number of iterations. Under CC load scenarios, the most appropriate choice for simulating the INR18650-MJ1 LIB in real-time on the microcontroller would be the EM-PCM.

5.3.1. Influence of the processor frequency

In application, lower processor frequency results in lower energy consumption and reducing the computation power can thus pose a challenge for the *C*-code p2D-PCMs to hold real-time computability. The 32-bit ARM Cortex M4 processor [9] of the STM32 offers a frequency range up to 168 MHz and

three different configurations at 50, 109 and 168 MHz are used to simulate 1C CC charge and discharge with all *C*-code PCMs to evaluate the influence on the computation speed. The UART transfer time reveals negligible influence herein.

Figure 7 a and b show the total computation time versus the simulated time and for all frequencies, the EM-PCM shows the fastest computation, whereas the PP-PCM the slowest. A more detailed analysis of the computation performance is shown in Table 7. As the simulation accuracy and iteration number (see Table 6) are not influenced by the processor frequency, only the mean iteration and total calculation time versus the simulated time (see 'Time reduction') are shown in Table 7. Similarly to Figure 7 a and b, Figure 7 c and d illustrate the average iteration time per 1 s step. The blue horizontals in Figure 7 mark the real-time suitability, when the calculation time equals the simulated time (see 'Time factor' in Table 7). At 168 MHz, the PP-PCM is slightly atop the threshold with +8 % /+6 %, whereas the OC- and the fastest calculating EM-PCM show time reductions of 6 % /10 % and 41 % /43 % during 1C CC charge /discharge. Only the EM-PCM reveals sufficient computation speed (max. 872 ms for 1C CC charge) at 109 MHz, whereas at 50 MHz none of the PCMs can simulate in real-time. A minimum increase of 89 % (1890 ms for 1 s at 1C CC discharge) is seen for the EM-PCM and over 3 times longer computation times than the actual simulated time appar for the PP- and OC-PCM.

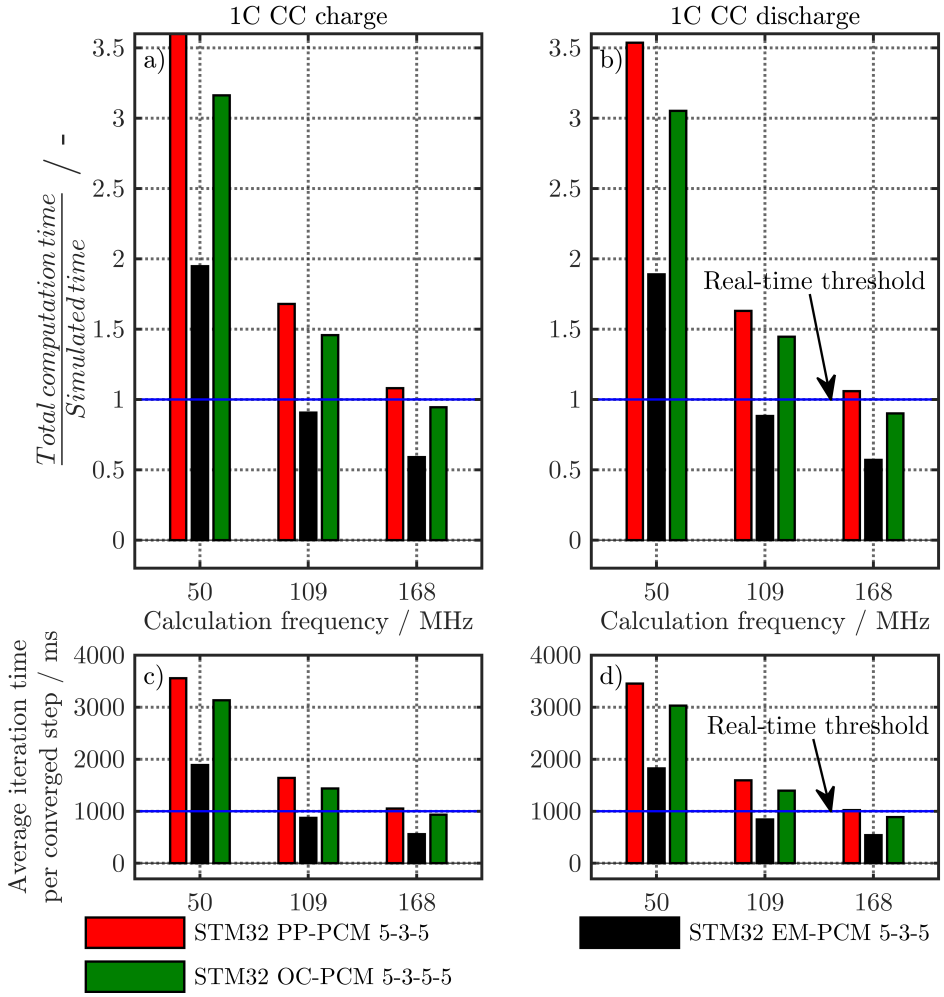

Figure 7: Computational performance of the *C*-code PP-, EM- and OC-PCM for simulating 1C CC charge (a and c) and discharge (b and d) on the STM32 at 50, 109 and 168 MHz. The horizontal line depicted in 'blue' marks the real-time threshold when the simulated time equals the total computation time (see 'Time factor = 1' in a and b) and the average iteration time per converged step equals the simulation step-size of 1 s (see '1000 ms' in c and d).

Table 7: Computational performance of C-code PCMs at different processor frequencies

Model	**PP-PCM 5-3-5**		**EM-PCM 5-3-5**		**OC-PCM 5-3-5-5**	
Load [I]	CH	DCH	CH	DCH	CH	DCH
Frequency of **168 MHz**						
Mean iteration time per step / ms [II]	1052	1021	560	540	932	887
Time factor [III]	1.08	1.06	0.59	0.57	0.94	0.90
Frequency of **109 MHz**						
Mean iteration time per step / ms [II]	1639	1592	872	841	1436	1394
Time factor [III]	1.68	1.63	0.91	0.88	1.46	1.45
Frequency of **50 MHz**						
Mean iteration time per step / ms [II]	3555	3452	1889	1823	3132	3031
Time factor [III]	3.63	3.55	1.95	1.89	3.16	3.05

[I] at 1C constant current charge (CH) and discharge (DCH)

[II] referring to a 1 s step-size [III] $\frac{Total\,calculation\,time}{Simulated\,time}$ = Time factor

In sum, real-time suitability poses a challenge for the C-code PCMs on the microcontroller and under CC loads, the necessary speed-up can be achieved via using coarser spatial discretizations (*e.g.* 2-1-2 resp. 5-3-5-3 for the PP-/EM-PCM and OC-PCM). As seen in Table 6, reducing the spatial configuration reveals the EM-PCM as fastest-calculating C-code PCM. As the accuracy is not distorted by the frequency, the most appropriate choice at low frequencies is the EM-PCM.

5.3.2. Influence of the microcontroller's accuracy

To reduce computation time, the jacobian/iteration matrix inversion is calculated using the FPU of single precision. The transfer needs two extra arrays of single precision for a current copy of the matrix and for storing the inverse. The coarse configurations 2-1-2 and 5-3-5-3 for the PP-/EM- and OC-PCM are used to avoid RAM overloads. 1C CC charge and discharge at

168 MHz are simulated and the results are shown in Table 8.

The maximum mean cell voltage error accounts to 1.862 mV for 1C CC discharge using the EM-PCM in reference to the corresponding MATLAB®-code PCM. The lowest error is seen for the OC-PCM below 0.12 mV. Negligible errors for the cell temperature (see Table 8) appear.

Comparing the results between single- (see Table 8) and double-precision (see Table 6), a trend of decreasing iterations appears ($\approx$ 19 % less) for the EM-PCM at 1C CC charge.

The benefit can be seen in the average iteration times per step in Table 8.

Table 8: Computational performance of *C*-code PCMs using FPU for 1C CC charge and discharge

Model	**PP-PCM 2-1-2**		**EM-PCM 2-1-2**		**OC-PCM 5-3-5-3**	
Load[I]	CH	DCH	CH	DCH	CH	DCH
Mean $\triangle V_{cell}$ / mV [II]	1.343	0.416	1.683	1.862	0.055	0.115
Mean $\triangle T_{cell}$ / K [II]	<1E-2	0.08	0.12	0.09	<1E-3	<1E-3
Mean number of iterations per step / -	3.90	3.33	1.75	1.59	1.17	1.30
Mean iteration time per step / ms	156	134	75	69	76	80
Time reduction [III]	29%	35%	40%	23%	86%	86%

[I] at 1C constant current charge (CH) and discharge (DCH)

[II] in reference to the corresponding MATLAB®-code PCM with identical spatial discretization

[III] compared to the calculation time without using the FPU on the STM32

The EM- and OC-PCM show calculation times around 80 ms and the PP-PCM up to 156 ms. For constant loads, minimum reduction of 29 %, 23 % and 86 % for the PP-, EM- and OC-PCM appears when the FPU is used. The mean error of the PCMs may slightly increase while using FPU, but still sufficient accuracy is offered. Regarding the maximum calculation times, sin-

gle time steps are simulated in ≈ 600 ms for a 1 s time step at a total iteration number up to 9 for the OC-PCM. This may be critical, when real-time computation must be guaranteed. The PP- and EM-PCM show lower maximum iteration numbers and computation times, where the PP-PCM appears as the overall slowest converging model.

In sum, the most benefit in using the FPU is gained with the OC-PCM but certain overshoots in calculation time and iteration number may be a problem for real-time suitablility. The EM- and PP-PCM show a more stable calculation with slightly increased errors for the EM-PCM and slower computation speed for the PP-PCM. Comparing the total calculation time to the simulated times for 1C charge and discharge, a minimum reduction of 84 %, 92 % and 92 % appears at constant load simulations for the PP-, EM- and OC-PCM, respectively.

5.4. Validation and computational efficiency under driving cycle scenario

Referring to application of LIBs in EVs, the ARTEMIS [86] drive cycle was adapted to the INR18650-MJ1 LIB current range as seen in Figure 8 a. For the experiments, a cycler (CTS, BaSyTec) paired with a climate chamber (VT 4021, Vötsch Industrietechnik GmbH) at 25 °C was used at initial 100 % SoC of the LIB. Figure 8 b illustrates the measured cell voltage together with a magnified region (Figure 8 c) showing the simulation results. The temperature profile (see Figure 8 f) and both voltage and temperature error (Figure 8 d, e and g) of the MATLAB®-code PP-, EM- and OC-PCM are shown. Table 9 summarizes the related simulation accuracy for the MATLAB®-code PCMs.

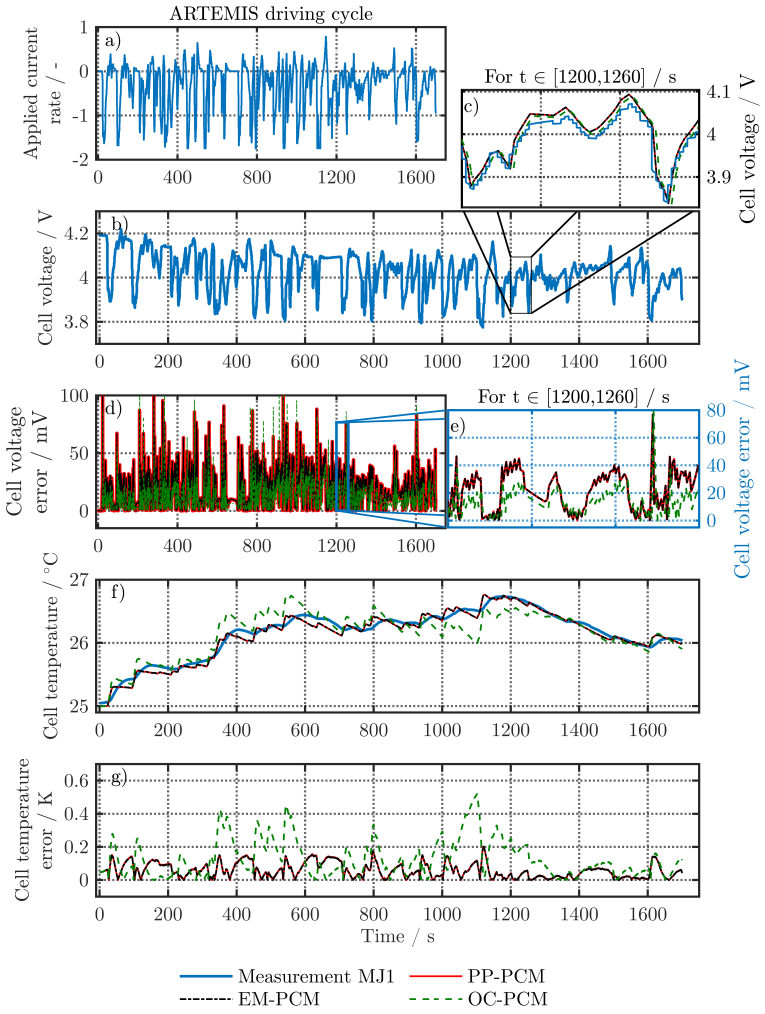

Figure 8: Measurement results of the INR18650-MJ1 LIB ('Measurement MJ1') for the applied current (a), cell voltage (b) and temperature (f) under the adapted ARTEMIS [86] driving cycle. The subplot (c) shows a magnified part from 1200 to 1260 s for the measured and simulated (MATLAB®-code 'PP-PCM 5-3-5', 'EM-PCM 5-3-5' and 'OC-PCM 5-3-5-5') cell voltages in (b). The resulting error profiles for the PCMs are shown for the cell voltage (d) with a magnified part similar to (c). Subplot (g) shows the temperature error.

In this case, averaged open-circuit potentials (E_{eq}, see supplementary part) of lithiation and delithiation [2] were used to ease the effect of measurement-related polarization and improve the simulation of dynamic loads.

The mean cell voltage error for the PP- and EM-PCM appear to be around 18.7 mV on average with a root mean squared error (RMSE) of 23.1 mV. Reducing the spatial nodes, the error slightly increases but remain below 20 mV. The OC-PCM shows no distinct difference between the coarse and the maximum configuration and reveals the most accurate simulation with a mean error of 12.4 mV at a RMSE of 16.6 mV. The temperature error for all MATLAB®-code PCMs are below 0.15 K (RMSE < 0.2), which is in the range of the measurement accuracy (*i.e.* Pt100 sensor with $\pm$ 0.15 K at 0 °C, DIN/IEC Class A). For comparison, the ECM simulation results calculated on the same desktop computer revealed a mean cell voltage error of 13.7 mV at a RMSE of 17.5 mV and a mean temperature error of 0.22 K.

The PCMs are in the same range of accuracy as the ECM and choosing the appropriate discretization or reformulation to gain real-time operability, a competitive alternative appears for model-based monitoring of LIBs. The

Table 9: Validation of MATLAB®-code PCMs under the adapted ARTEMIS driving cycle

Model	**PP-PCM**		**EM-PCM**		**OC-PCM**	
Load	**ARTEMIS driving cycle (see Figure 8 a)**					
Discretization	**2-1-2**	**5-3-5**	**2-1-2**	**5-3-5**	**5-3-5-3**	**5-3-5-5**
Mean $\triangle V_{cell}$ / mV [I]	19.3	18.7	19.3	18.7	12.4	12.4
RMSE $\triangle V_{cell}$ / mV [I]	23.8	23.1	23.8	23.1	16.6	16.6
Mean $\triangle T_{cell}$ / K [I]	0.07	0.06	0.07	0.06	0.13	0.13
RMSE $\triangle T_{cell}$ / K [I]	0.09	0.07	0.09	0.07	0.17	0.17

[I] in reference to the experimentally measured data of the INR18650-MJ1 cell

benefit of using a PCM lies in the simulated local concentrations and poten-

tials, which can be used to develop sophisticated control algorithms such as avoiding lithium plating during fast charging [85].

Finally, the performance of simulating the ARTEMIS profile with the C-code PCMs on the STM32 is evaluated. Table 10 summarizes the simulation accuracy in reference to the MATLAB®-code PCM (see Table 9) together with the computation speed. The mean cell voltage deviation is below 0.4 mV for

Table 10: Computational performance of C-code PCMs simulating the adapted ARTEMIS driving cycle

Model	**PP-PCM**		**EM-PCM**		**OC-PCM**	
Load	**ARTEMIS driving cycle (see Figure 8 a)**					
Discretization	**2-1-2**	**5-3-5**	**2-1-2**	**5-3-5**	**5-3-5-3**	**5-3-5-5**
Mean $\triangle V_{cell}$ / mV [I]	0.153	0.148	0.361	0.330	0.030	0.065
Mean $\triangle T_{cell}$ / K [I]	0.15	0.07	0.16	0.15	<1E-3	<1E-3
Mean number of iterations per step / -	3.46	4.12	2.68	3.31	3.15	3.15
Mean iteration time per step / ms	190	1009	157	838	1250	2072
Time reduction [II]	24%	n.a.	37%	n.a.	n.a.	n.a.

[I] in reference to the corresponding MATLAB®-code PCM with the identical discretization

[II] for a step-size of 250 ms

all PCMs, where the OC-PCM shows the most accurate implementation and the EM-PCM the maximum deviation to the corresponding MATLAB®-code PCM. Again, the temperature deviances are in the range of measurement accuracy. Regarding the computational performance, the OC-PCM is not able to simulate in real-time due to the step-size reduction option in the solver. The EM-PCM shows the fastest calculation and for a step-size of 250 ms, only the coarse configuration is able to simulate with a time reduction of 37 % under real-time requirements. The PP-PCM required at least 190 ms on

average, which corresponds to a time reduction of 24 %.
To conclude, the EM-PCM shows the most promising results for simulating an application-near scenario with fulfilling real-time requirements.

6. Conclusion

Trending towards high-energy LIBs, physicochemical model based monitoring can help to account for inhomogeneties on local scales and improve the state-estimation process. Efficiently reduced p2D-PCMs are evaluated on a microcontroller using either FDM together with solid-diffusion approximations or Chebyshev orthogonal collocation to reformulate particle and electrolyte domain. Experimental validation with CC charge and discharge, ARTEMIS [86] driving cycle and benchmarking to ECM and rigorous COMSOL p2D-PCM showed accurate simulation for a NMC-811/SiC LIB. In sum, the average cell voltage error of the p2D-PCMs can be summarized as modeling and parameter uncertainties, errors from spatial reduction and errors from implementation in the STM32 as shown in Table 11.
At low processor frequencies down to 50 MHz, crucial limitations appear for the p2D-PCMs to calculate in real-time. Using hardware acceleration such as the FPU, computation acceleration up to 86 % appeared, which can be recommended at low processor frequencies to gain real-time computability again. Computation analysis under CC and driving cycle loads revealed the EM-PCM as best choice for simulating a single LIB at least 37 % faster than real-time, which consumes 21.9 kB RAM for solving and 175 kB flash memory for storing the model on the microcontroller.
Future work can investigate more robust solver in terms of the OC-PCM to

Table 11: Summary of average cell voltage error for MATLAB®- and *C*-code PCMs

Model	**PP-PCM**	**EM-PCM**	**OC-PCM**	**Code**
Error from modeling and parameters / mV [I]	Compared to experimental data from 3.35 Ah NMC-811/SiC INR18650-MJ1 LIB			MATLAB
1C CC CH/DCH	<1/9.7	<1/9.3	17/<1	
Driving cycle	18.6	18.6	12.4	
Error from spatial reduction / mV [II]	4.1/4	4.1/4	<1/<1	
Error from microcontroller / mV	STM32 with max. 168 MHz @32-bit ARM® Cortex®-M4 max. 1024 kB flash memory and max. 192 kB RAM [9]			*C*
1C CC CH/DCH	<0.5/<0.5	<0.5/2.1	<0.5/<0.5	
Driving cycle		<0.5		

[I] for PP-PCM 10-5-10, EM-PCM 10-5-10 and OC-PCM 20-10-20-25

[II] for PP-PCM 10-5-10 → 2-1-2, EM-PCM 10-5-10 → 2-1-2 and OC-PCM 20-10-20-25 → 5-3-5-3

improve the performance on a microcontroller and test the *C*-code EM-PCM model-based monitoring for estimating a LIB online and develope local-anode potential based fast charging profiles to avoid lithium plating.

Acknowledgement

This work has received funding from the European Union's Horizon 2020 research and innovation programme under the grant 'Electric Vehicle Enhanced Range, Lifetime And Safety Through INGenious battery management' [EVERLASTING-713771].

Appendix A.

Table A.12: Properties of the electrolyte [2]

Electrolyte	1 M $LiPF_6$ in PC/EC/DMC
Salt diffusivity D_l [I] / $m^2\,s^{-1}$	$10\text{E-}4{\cdot}10^{-4.43-\frac{54}{T-229-5c_l}-0.22c_l}$
Ionic conductivity κ_l [I] / $S\,m^{-1}$	$0.1c_l\Big(-10.5+0.668c_l+0.494c_l^2+0.074T-0.0178c_lT$ $-8.8610^{-4}c_l^2T-6.9610^{-5}T^2+2.810^{-5}c_lT^2\Big)^2$
Activity $\frac{dlnf_\pm}{dln\,c_l}$ [I] / -	$\Big(0.601-0.24c_l^0.5+0.983\Big(1-0.0052(T-294)\Big)c_l^{1.5}\Big)$ $\cdot(1-t_+^0)^{-1}-1$
Transference t_+^0 [I] / -	0.38
Ref. concentration c_{ref} [I] / $mol\,m^{-3}$	1000

[I] Ref. [87]

Table A.13: Parameterization of the p2D-PCM for a NMC-811/SiC LIB [2]

Geometry	**Silicon-graphite (SiC)**	**Separator**	**Nickel-rich (NMC-811)**
Thickness L	86.7 µm [m]	12 µm [m]	66.2 µm [m]
Particle radius R_p	6.1 µm [m,D50]		3.8 µm [m,D50]
Active material fraction ε_s	69.4 % [e]		74.5 % [e]
Inactive fraction $\varepsilon_{s,na}$	9 % [e,*]		8.4 % [e,*]
Porosity ε_l	21.6 % [m]	45 % [e]	17.1 % [m]
Bruggeman coefficient β [VI,**]	1.5	1.5	1.85 [e]
Thermodynamics			
Equilibrium potential E_{eq}	Ref. [2] [m]		Ref. [2] [m]
Entropic coefficient $\frac{\partial E_{eq}}{\partial T}$	Ref. [2] [m]		Ref. [2] [m]
Stoichiometry 100% SoC	0.852		0.222
Stoichiometry 0% SoC	0.002		0.942
Max. theorectical loading b_g	415 mAh g^{-1} [I]		275.5 mAh g^{-1} [II]
Density ρ	2.24 g cm^{-3} [I]		4.87 g cm^{-3} [II]
Concentration $c_{s,max}$	34 684 mol m^{-3} [e]		50 060 mol m^{-3} [e]
Transport			
Solid diffusivity D_s [***]	5×10^{-14} m^2 s^{-1} [e,V]		5×10^{-13} m^2 s^{-1} [III,V]
Specific activation $\frac{E_{a,D_s}}{R}$ [***]	1200 K [e]		1200 K [e]
Solid conductivity σ_s	100 S m^{-1} [III]		0.17 S m^{-1} [e,III]
Kinetics			
Reaction rate constant k [***]	3×10^{-11} m s^{-1} [e]		1×10^{-11} m s^{-1} [e]
Specific activation $\frac{E_{a,k}}{R}$ [***]	3600 K [e]		3600 K [e]
Transfer coefficient $\alpha_{a/c}$	0.5 [e]		0.5 [e]

m = measured e = estimated * PVDF-binder/Carbon black (Ref. [88, 89])

[I] Ref. [90] [II] Ref. [91] [III] Ref. [92] [IV] Ref. [93] [V] Ref. [94] [VI] Ref. [95]

** Effective transport correction according to Bruggeman (Ref. [95]): $\Psi_{eff} = \varepsilon^{\beta} \cdot \Psi_0$

*** Arrhenius law (Ref. [96]): $k = A \cdot \exp\left(\frac{E_{a,i}}{R} \frac{(T-298[\mathrm{K}])}{T \cdot 298[\mathrm{K}]}\right)$

Table A.14: Differential algebraic equations of the p2D-PCM

Mass balance [I]	$\varepsilon_l \frac{\partial c_l(x,t)}{\partial t} = \frac{\partial}{\partial x}\left(D_l^{eff} \frac{\partial c_l(x,t)}{\partial x} + \frac{i_l(x,t)\,(1-t_+^0)}{F}\right)$
	$\varepsilon_s \frac{\partial\, c_s(x,t,r)}{\partial t} = \frac{1}{r^2}\frac{\partial}{\partial r}\left(D_s r^2 \frac{\partial\, c_s(x,t,r)}{\partial r}\right)$
Potentials [I]	$\frac{\partial\, \Phi_l(x,t)}{\partial x} = -\frac{i_l(x,t)}{\kappa_l^{eff}} + \frac{2\,R\,T}{F}\left(1-t_+^0\right)\left(1+\frac{d\,ln f_\pm}{d\,ln\, c_l(x,t)}\right)\frac{\partial\, ln\, c_l(x,t)}{\partial x}$
	$\frac{\partial\, \Phi_s(x,t)}{\partial x} = -\frac{i_{app}(t)-i_l(x,t)}{\sigma_s} \quad \text{with} \quad i_{app}(t) = i_s(x,t) + i_l(x,t) \quad \forall\, x,t$
Charge balance [I]	$\frac{\partial i_l(x,t)}{\partial x} + \frac{\partial i_s(x,t)}{\partial x} = 0 \quad \text{with} \quad \frac{\partial i_s(x,t)}{\partial x} = -\frac{3\varepsilon_s}{R_p} F j_n(x,t)$
Electrode kinetics [I]	$j_n(x,t) = \frac{i_0(x,t)}{F}\left[exp\left(\frac{\alpha_a\, F\, \eta(x,t)}{RT}\right) - exp\left(-\frac{\alpha_c\, F\, \eta(x,t)}{RT}\right)\right]$
	$\eta(x,t) = \Phi_s(x,t) - \Phi_l(x,t) - E_{eq}(x,t)$
	$i_0(x,t) = F\,k\,\left(c_{s,max} - c_{ss}(x,t)\right)^{\alpha_c}\,\left(c_{ss}(x,t)\right)^{\alpha_a}\,\left(c_l(x,t)\right)^{\alpha_a}$
Temperature [I, II]	$\frac{m\,c_p}{A_{act}}\,\frac{\partial\, T_{cell}}{\partial\, t} = -i_{app}\cdot V_{cell} + q - q_{ext} - q_\infty$
	$q = \frac{3\varepsilon_s F}{Rp}\int_{x*} j_n \cdot \left(E_{eq}\left(\frac{c_s(x,r_p)}{c_{s,max}}\right) - \frac{\partial E_{eq}\left(\frac{c_s(x,r_p)}{c_{s,max}}\right)}{\partial T}\cdot T\right)\mathrm{d}x$
	$q_{ext} = i_{app}^2 \cdot R_{ext} \qquad q_\infty = \alpha_\infty^* \frac{A_{surf}}{A_{act}}\left(T_{cell} - T_\infty\right)$

[I] Ref. [6] [II] Ref. [52, 64, 65] $\quad x* = x \in [0, L_{\text{neg}}] \wedge [L_{\text{neg}} + L_{\text{sep}}, L_{\text{neg}} + L_{\text{sep}} + L_{\text{pos}}]$

$\alpha_\infty^* = 44.3\,\mathrm{W\,m^{-2}K^{-1}}$, simplified for combining heat radiation, conduction and convection [2]

Table A.15: Nomenclature I

Greek symbols		
α		Transfer coefficient
α_∞	$\mathrm{W\,m^{-2}\,K^{-1}}$	Ambient heat transfer coefficient
β		Bruggeman coefficient
ε		Volume fraction
ϵ		Numerical tolerance
η	V	Overpotential
κ	$\mathrm{S\,m^{-1}}$	Ionic conductivity
ρ	$\mathrm{kg\,m^{-3}}$	Mass density
σ	$\mathrm{S\,m^{-1}}$	Electrical conductivity
Φ	V	Electrical potential
Ψ		Variable
$\boldsymbol{\omega}$		Clenshaw-Curtis weights

Indices	
a	Anodic reaction (oxidation)
act	Active area
app	Applied (*i.e.* current density)
c	Cathodic reaction (reduction)
eff	Transport corrected (Bruggeman correlation [95])
ext	External heat (*i.e.* from grid resistance)
l	Liquid phase (*i.e.* electrolyte)
max	Maximum
neg	Negative electrode (*i.e.* SiC)
pos	Positive electrode (*i.e.* NMC-811)
s	Solid phase (*i.e.* active particle)
sep	Separator
s,s	Solid phase (*i.e.* active particle surface)
surf	Surface

Table A.16: Nomenclature II

Latin symbols		
a	m^{-1}	Specific surface
b_g	$\mathrm{mAh\,g}^{-1}$	Maximum theoretical loading
c	$\mathrm{mol\,m}^{-3}$	Concentration of lithium cations (Li^+)
$c_{s,max}$	$\mathrm{mol\,m}^{-3}$	Maximum theoretical concentration of Li^+
c_p	$\mathrm{J\,kg}^{-1}\,\mathrm{K}^{-1}$	Heat capacity
D	$\mathrm{m}^2\,\mathrm{s}^{-1}$	Diffusion coefficient
$\boldsymbol{D}$		Differentiation matrix
E_{eq}	V	Equilibrium potential vs. Li/Li^+
$\frac{\partial E_{eq}}{\partial T}$	V/K	Entropic coefficient
$f_\pm$		Mean molar activity coefficient of electrolyte
F	$96\,485\,\mathrm{As\,mol}^{-1}$	Faraday's constant
$\boldsymbol{g}$		Non-linear equations of p2D-PCM
i	$\mathrm{A\,m}^{-2}$	Current density
i_{app}	$\mathrm{A\,m}^{-2}$	Applied current density
i_n	$\mathrm{A\,m}^{-2}$	Current density perpendicular to particle surface
i_0	$\mathrm{A\,m}^{-2}$	Exchange current density
j_n	$\mathrm{mol\,m}^{-2}\mathrm{s}^{-1}$	Pore-wall flux
$\boldsymbol{J}$		Jacobian matrix
k	$\mathrm{m\,s}^{-1}$	Reaction rate constant
L	m	Thickness
m	kg	Mass of cell
$\boldsymbol{M}$		Mass matrix
$\boldsymbol{M}_i$		Iteration matrix
r	m	r-coordinate particle domain of p2D-PCM
R	$8.314\,\mathrm{J\,mol}^{-1}\,\mathrm{K}^{-1}$	Gas constant
R_{ext}	$\Omega\,\mathrm{m}^2$	Grid resistance
R_p	m	Particle radius
q	$\mathrm{W\,m}^{-2}$	Heat generation rate per area
t	s	Time
T	K	Temperature
t_+^0		Transport number of Li^+
x	m	x-coordinate in electrolyte domain of p2D-PCM

References

[1] T. Amietszajew, E. McTurk, J. Fleming, R. Bhagat, Understanding the limits of rapid charging using instrumented commercial 18650 high-energy li-ion cells, Electrochimica Acta 263 (2018) 346–352. doi:10.1016/j.electacta.2018.01.076.

[2] J. Sturm, A. Rheinfeld, I. Zilberman, F. B. Spingler, S. Kosch, F. Frie, A. Jossen, Modeling and simulation of inhomogeneities in a 18650 nickel-rich, silicon-graphite lithium-ion cell during fast charging, Journal of Power Sources 412 (2019) 204–223. doi:10.1016/j.jpowsour.2018.11.043.

[3] C. von Lüders, V. Zinth, S. V. Erhard, P. J. Osswald, M. Hofmann, R. Gilles, A. Jossen, Lithium plating in lithium-ion batteries investigated by voltage relaxation and in situ neutron diffraction, Journal of Power Sources 342 (2017) 17–23. doi:10.1016/j.jpowsour.2016.12.032.

[4] C. von Lüders, J. Keil, M. Webersberger, A. Jossen, Modeling of lithium plating and lithium stripping in lithium-ion batteries, Journal of Power Sources 414 (2019) 41–47. doi:10.1016/j.jpowsour.2018.12.084.

[5] F. M. Kindermann, J. Keil, A. Frank, A. Jossen, A sei modeling approach distinguishing between capacity and power fade, Journal of The Electrochemical Society 164 (12) (2017) E287–E294. doi:10.1149/2.0321712jes.

[6] M. Doyle, T. F. Fuller, J. Newman, Modeling of galvanostatic charge and discharge of the lithium/polymer/insertion cell, Journal of The Electrochemical Society 140 (6) (1993) 1526–1533.

[7] V. Ramadesigan, P. W. C. Northrop, S. De, S. Santhanagopalan, R. D. Braatz, V. R. Subramanian, Modeling and simulation of lithium-ion batteries from a systems engineering perspective, Journal of The Electrochemical Society 159 (3) (2012) R32–R45.

[8] A. Jokar, B. Rajabloo, M. Désilets, M. Lacroix, Review of simplified pseudo-two-dimensional models of lithium-ion batteries, Journal of Power Sources 327 (2016) 44–55. doi:10.1016/j.jpowsour.2016.07.036.

[9] STMicroelectronics, Stm32f4discovery microcontroller, url: https://www.st.com/en/evaluation-tools/stm32f4discovery.html, STM Technical Data 2018.

[10] V. R. Subramanian, J.A. Ritter, R. E. White, Approximate solutions for galvanostatic discharge of spherical particles: 1. constant diffusion coefficient, Journal of the Electrochemical Society 148 (11) (2001) 444–449.

[11] V. R. Subramanian, V. D. Diwakar, D. Tapriyal, Efficient macro-micro scale coupled modeling of batteries, Journal of the Electrochemical Society 152 (10) (2005) 2002–2008.

[12] S. Santhanagopalan, Q. Guo, P. Ramadass, R. E. White, Review of models for predicting the cycling performance of lithium ion batteries, Journal of Power Sources 156 (2) (2006) 620–628. doi:10.1016/j.jpowsour.2005.05.070.

[13] V. R. Subramanian, V. Boovaragavan, V. D. Diwakar, Toward real-time

simulation of physics based lithium-ion battery models, Electrochemical and Solid-State Letters 10 (11) (2007) A255. doi:10.1149/1.2776128.

[14] V. R. Subramanian, V. Boovaragavan, V. Ramadesigan, M. Arabandi, Mathematical model reformulation for lithium-ion battery simulations: Galvanostatic boundary conditions, Journal of The Electrochemical Society 156 (4) (2009) A260–A271.

[15] V. Boovaragavan, V. Ramadesigan, M. V. Panchagnula, V. R. Subramanian, Continuum representation for simulating discrete events of battery operation, Journal of The Electrochemical Society 157 (1) (2010) A95–A104.

[16] R. Klein, N. A. Chaturvedi, J. Christensen, J. Ahmed, R. Findeisen, A. Kojic, State estimation of a reduced electrochemical model of a lithium-ion battery, American Control Conference (2010) 6618–6623.

[17] V. Ramadesigan, K. Chen, N. A. Burns, V. Boovaragavan, R. D. Braatz, V. R. Subramanian, Parameter estimation and capacity fade analysis of lithium-ion batteries using reformulated models, Journal of The Electrochemical Society 158 (9) (2011) A1048–A1054.

[18] T. S. Dao, C. P. Vyasarayani, J. McPhee, Simplification and order reduction of lithium-ion battery model based on porous-electrode theory, Journal of Power Sources 198 (1) (2012) 329–337.

[19] R. Klein, N. A. Chaturvedi, J. Christensen, J. Ahmed, R. Findeisen, A. Kojic, Electrochemical model based observer design for a lithium-

ion battery, IEEE Transactions on Control Systems Technology 21 (2) (2013) 289–301. doi:10.1109/TCST.2011.2178604.

[20] C. Zou, C. Manzie, S. Anwar, Control-oriented modeling of a lithium-ion battery for fast charging, IFAC Proceedings Volumes 47 (3) (2014) 3912–3917. doi:10.3182/20140824-6-ZA-1003.00829.

[21] R. Masoudi, T. Uchida, J. McPhee, Parameter estimation of an electrochemistry-based lithium-ion battery model, Journal of Power Sources 291 (2015) 215–224.
URL http://dx.doi.org/10.1016/j.jpowsour.2015.04.154

[22] C. Y. Wang, W. B. Gu, Micro-macroscopic coupled modeling of batteries and fuel cells: Model development, Journal of the Electrochemical Society 145 (10) (1998) 3407–3417.

[23] W. B. Gu, Micro-macroscopic coupled modeling of batteries and fuel cells, Journal of The Electrochemical Society 145 (10) (1998) 3418. doi:10.1149/1.1838821.

[24] W. B. Gu, C. Y. Wang, Thermal-electrochemical coupled modeling of a lithium-ion cell, Proceeding of the ECS 99 (2000) 1–16.

[25] M. Guo, R. E. White, An approximate solution for solid-phase diffusion in a spherical particle in physics-based li-ion cell models, Journal of Power Sources 198 (2012) 322–328. doi:10.1016/j.jpowsour.2011.08.096.

[26] X. Hu, S. Stanton, L. Cai, R. E. White, A linear time-invariant model for solid-phase diffusion in physics-based lithium ion cell models, Journal of Power Sources 214 (2012) 40–50. doi:10.1016/j.jpowsour.2012.04.040.

[27] X. Hu, S. Stanton, L. Cai, R. E. White, Model order reduction for solid-phase diffusion in physics-based lithium ion cell models, Journal of Power Sources 218 (2012) 212–220. doi:10.1016/j.jpowsour.2012.07.007.

[28] S. Khaleghi Rahimian, S. Rayman, R. E. White, Extension of physics-based single particle model for higher charge–discharge rates, Journal of Power Sources 224 (2013) 180–194. doi:10.1016/j.jpowsour.2012.09.084.

[29] L. Cai, R. E. White, Reduction of model order based on proper orthogonal cecomposition for lithium-ion battery simulations, Journal of The Electrochemical Society 156 (3) (2009) A154–A161.

[30] V. Ramadesigan, V. Boovaragavan, J. C. Pirkle, V. R. Subramanian, Efficient reformulation of solid-phase diffusion in physics-based lithium-ion battery models, Journal of The Electrochemical Society 157 (7) (2010) A854. doi:10.1149/1.3425622.

[31] P. W. C. Northrop, V. Ramadesigan, S. De, V. R. Subramanian, Coordinate transformation, orthogonal collocation, model reformulation and simulation of electrochemical-thermal behavior of lithium-ion battery stacks, Journal of The Electrochemical Society 158 (12) (2011) A1461–A1477. doi:10.1149/2.058112jes.

[32] B. Suthar, V. Ramadesigan, P. W. C. Northrop, B. Gopaluni, S. Santhanagopalan, R. D. Braatz, V. R. Subramanian, Optimal control and state estimation of lithium-ion batteries using reformulated models, American Control Conference (2013) 5350–5355.

[33] C. G. Mayhew, W. He, C. Kroener, R. Klein, N. A. Chaturvedi, A. Kojic, Investigation of projection-based model-reduction techniques for solid-phase diffusion in li-ion batteries, American Control Conference (2014) 123–128.

[34] P. W. C. Northrop, B. Suthar, V. Ramadesigan, S. Santhanagopalan, R. D. Braatz, V. R. Subramanian, Efficient simulation and reformulation of lithium-ion battery models for enabling electric transportation, Journal of the Electrochemical Society 161 (8) (2014) E3149–E3157. doi:10.1149/2.018408jes.

[35] M. T. Lawder, P. W. C. Northrop, V. R. Subramanian, Model-based sei layer growth and capacity fade analysis for ev and phev batteries and drive cycles, Journal of the Electrochemical Society 161 (14) (2014) A2099–A2108. doi:10.1149/2.1161412jes.

[36] K. Smith, C.-Y. Wang, Solid-state diffusion limitations on pulse operation of a lithium ion cell for hybrid electric vehicles, Journal of Power Sources 161 (1) (2006) 628–639. doi:10.1016/j.jpowsour.2006.03.050.

[37] Y. Zeng, P. Albertus, R. Klein, N. Chaturvedi, A. Kojic, M. Z. Bazant, J. Christensen, Efficient conservative numerical schemes for 1d nonlinear spherical diffusion equations with applications in battery modeling, Journal of the Electrochemical Society 160 (9) (2013) A1565–A1571. doi:10.1149/2.102309jes.

[38] P. C. Urisanga, D. Rife, S. De, V. R. Subramanian, Efficient conservative

reformulation schemes for lithium intercalation, Journal of the Electrochemical Society 162 (6) (2015) A852–A857. doi:10.1149/2.0061506jes.

[39] T. F. Fuller, M. Doyle, J. Newman, Simulation and optimization of the dual lithium ion insertion cell, Journal of The Electrochemical Society 141 (1) (1994) 1–9.

[40] M. Doyle, Design and simulation of lithium rechargeable batteries, Ph.d. thesis, University of California, Berkeley (08/1995).

[41] K. E. Thomas, J. Newman, R. M. Darling, Mathematical modeling of lithium batteries, in: W. A. van Schalkwijk, B. Scrosati (Eds.), Advances in Lithium-Ion Batteries, Springer US, Boston, MA, 2002, pp. 345–392. doi:10.1007/0-306-47508-1_13.

[42] J. Newman, Fortran programs for the simulation of electrochemical systems: dualfoil5.2.f (2014).
URL `http://www.cchem.berkeley.edu/jsngrp/`

[43] D. Zhang, B. N. Popov, R. E. White, Modeling lithium intercalation of a single spinel particle under potentiodynamic control, Journal of the Electrochemical Society 147 (3) (2000) 831. doi:10.1149/1.1393279.

[44] B. Rajabloo, M. Désilets, Choquette Y., Parameter estimation of single particle model using comsol multiphysics and matlab optimization toolbox, Proceedings of the 2015 COMSOL Conference in Boston (2010) 1–6.

[45] M. Guo, G. Sikha, R. E. White, Single-particle model for a lithium-ion

cell: Thermal behavior, Journal of The Electrochemical Society 158 (2) (2011) A122. doi:10.1149/1.3521314.

[46] J. Sturm, H. Ennifar, S. V. Erhard, A. Rheinfeld, S. Kosch, A. Jossen, State estimation of lithium-ion cells using a physicochemical model based extended kalman filter, Applied Energy 223 (2018) 103–123. doi:10.1016/j.apenergy.2018.04.011.

[47] V. R. Subramanian, J. A. Ritter, R. E. White, Approximate solutions for galvanostatic discharge of spherical particles i. constant diffusion coefficient, Journal of The Electrochemical Society 148 (11) (2001) E444. doi:10.1149/1.1409397.

[48] V. Boovaragavan, S. Harinipriya, V. R. Subramanian, Towards real-time (milliseconds) parameter estimation of lithium-ion batteries using reformulated physics-based models, Journal of Power Sources 183 (1) (2008) 361–365. doi:10.1016/j.jpowsour.2008.04.077.

[49] A. M. Bizeray, S. Zhao, S. R. Duncan, D. A. Howey, Lithium-ion battery thermal-electrochemical model-based state estimation using orthogonal collocation and a modified extended kalman filter, Journal of Power Sources 296 (2015) 400–412. doi:10.1016/j.jpowsour.2015.07.019.
URL `http://arxiv.org/pdf/1506.08689v1`

[50] The MathWorks, Matlab function reference url: https://de.mathworks.com/?s_tid=gn_logo 2017.

[51] COMSOL Multiphysics, The batteries & fuel cells module user's guide

(2017).
URL https://www.comsol.com/

[52] D. Bernardi, E. Pawlikowski, J. Newman, A general energy balance for battery systems, Journal of the Electrochemical Society 132 (1985) 5–12.

[53] C. Campestrini, T. Heil, S. Kosch, A. Jossen, A comparative study and review of different kalman filters by applying an enhanced validation method, Journal of Energy Storage 8 (2016) 142–159. doi:10.1016/j.est.2016.10.004.

[54] Z. Wei, T. M. Lim, M. Skyllas-Kazacos, N. Wai, K. J. Tseng, Online state of charge and model parameter co-estimation based on a novel multi-timescale estimator for vanadium redox flow battery, Applied Energy 172 (2016) 169–179. doi:10.1016/j.apenergy.2016.03.103.

[55] I. Zilberman, A. Rheinfeld, A. Jossen, Uncertainties in entropy due to temperature path dependent voltage hysteresis in li-ion cells, Journal of Power Sources 395 (2018) 179–184. doi:10.1016/j.jpowsour.2018.05.052.

[56] The MathWorks, Mathworks' documentation on the mkpp function, url: https://de.mathworks.com/help/matlab/ref/mkpp.html 2018.

[57] F. Zhang, M. M. U. Rehman, H. Wang, Y. Levron, G. Plett, R. Zane, D. Maksimovic, State-of-charge estimation based on microcontroller-implemented sigma-point kalman filter in a modular cell balancing system for lithium-ion battery packs, 2015 IEEE 16th Workshop on Control and Modeling for Power Electronics (COMPEL) (2015) 1–7doi:10.1109/COMPEL.2015.7236525.

[58] J. Li, Adaptive model-based state monitoring and prognostics for lithium-ion batteries, Dissertation, University of Ulm, Ulm (05.08.2016).

[59] Jorge V Barreras, Federico Ferrari, David A Howey, Development of diagnosis algorithms for li-ion batteries, implementation in a microcontroller and testing in an hardware-in-the-loop simulator, University of Oxforddoi:10.13140/RG.2.2.26819.09768.

[60] R. E. Kalman, Kalman (1960) - contributions to the theory of optimal control, Boletin de la Sociedad Matematica Mexicana 1 (5) (1960) 102–119.

[61] R. E. Kalman, A new approach to linear filtering and prediction problems, Journal of Basic Engineering 82 (1) (1960) 35. doi:10.1115/1.3662552.

[62] R. E. Kalman, R. S. Bucy, New results in linear filtering and prediction theory, Journal of Basic Engineering 83 (1) (1961) 95. doi:10.1115/1.3658902.

[63] The MathWorks, Solvers for real-time simulation url: https://www.mathworks.com/help/physmod/simscape/ug/solvers-for-real-time-simulation.html?searchhighlight=ode14x&s_tid=doc_srchtitle.

[64] J. Mao, W. Tiedemann, J. Newman, Simulation of temperature rise in li-ion cells at very high currents, Journal of Power Sources 271 (2014) 444–454. doi:10.1016/j.jpowsour.2014.08.033.

[65] J. Mao, W. Tiedemann, J. Newman, Simulation of li-ion cells by dualfoil

model under constant-resistance load, ECS Transactions 58 (48) (2014) 71–81. doi:10.1149/05848.0071ecst.

[66] L. F. Shampine, M. W. Reichelt, The matlab ode suite, SIAM Journal on Scientific Computing 18 (1) (1997) 1–22. doi:10.1137/S1064827594276424.

[67] L. F. Shampine, M. W. Reichelt, J. A. Kierzenka, Solving index-1 daes in matlab and simulink, SIAM Review 41 (3) (1999) 538–552. doi:10.1137/S003614459933425X.

[68] The MathWorks, Mathworks' documentation on 'solve stiff differential equations and daes — variable order method', url: https://www.mathworks.com/help/matlab/ref/ode15s.html 2018.

[69] The MathWorks, Mathworks' documentation on sparse matrix creation, url: https://de.mathworks.com/help/matlab/ref/sparse.html 2018.

[70] W. Klopfenstein, Numerical differentiation formulas for stiff systems of ordinary differential equations, RCA Review 32 (1971) 447–462.

[71] P. Amestoy, A. Buttari, A. Guermouche, J.-Y. L'Excellent, B. Ucar, Mumps: Multifrontal massively parallel sparse direct solver - version 5.1.2 (2017).
URL `http://mumps.enseeiht.fr/`

[72] The MathWorks, Mathworks' documentation on the a/b function, url: https://de.mathworks.com/help/matlab/ref/mldivide.html 2018.

[73] ARM, Cortex microcontroller software interface standard (cmsis) library, url: https://developer.arm.com/embedded/cmsis.

[74] STMicroelectronics, Stm32cubemx for stm32 configuration and initialization c code generation, url: https://www.st.com/content/ccc/resource/technical/document/user_manual/10/c5/1a/43/3a/70/43/7d/dm00104712.pdf/files/dm00104712.pdf/jcr:content/translations/en.dm00104712.pdf, STM Technical Data 2018.

[75] The MathWorks, Mathworks' documentation on spline interpolation functionality, url: https://de.mathworks.com/help/matlab/ref/spline.html 2018.

[76] The MathWorks, Mathworks' documentation on the c-code export function, url: https://www.mathworks.com/help/ecoder/examples/exporting-functions.html 2018.

[77] IEEE, Ieee standard for floating-point arithmetic. doi:10.1109/IEEESTD.2008.4610935.

[78] M. Neher, Descriptive higher mathematics for engineers and scientists, Lehrbuch, Springer Vieweg, Wiesbaden, 2018.
URL `http://www.springer.com/de/book/9783658194192`

[79] J. Newman, Numerical solution of coupled, ordinary differential equations, Industrial & Engineering Chemistry Fundamentals 7 (3) (1968) 514–517. doi:10.1021/i160027a025.

[80] G. Peters, J. H. Wilkinson, On the stability of gauss-jordan elimination with pivoting, Communications of the ACM 18 (1) (1975) 20–24. doi:10.1145/360569.360653.

[81] A. R. Curtis, The facsimile numerical integrator for stiff initial value problems, Atomic Energy Research Establishment (AERE-R-9352) (1979) 1–42.

[82] T. Arens, F. Hettlich, C. Karpfinger, U. Kockelkorn, K. Lichtenegger, H. Stachel, Mathematics, 2nd Edition, Spektrum Akad. Verl., Heidelberg, 2010.

[83] Y. Li, M. Vilathgamuwa, T. Farrell, S. S. Choi, N. T. Tran, J. Teague, A physics-based distributed-parameter equivalent circuit model for lithium-ion batteries, Electrochimica Acta 299 (2019) 451–469. doi:10.1016/j.electacta.2018.12.167.

[84] Z. Wei, C. Zou, F. Leng, B. H. Soong, K.-J. Tseng, Online model identification and state-of-charge estimate for lithium-ion battery with a recursive total least squares-based observer, IEEE Transactions on Industrial Electronics 65 (2) (2018) 1336–1346. doi:10.1109/TIE.2017.2736480.

[85] Z. Chu, X. Feng, L. Lu, J. Li, X. Han, M. Ouyang, Non-destructive fast charging algorithm of lithium-ion batteries based on the control-oriented electrochemical model, Applied Energy-doi:10.1016/j.apenergy.2017.03.111.

[86] M. André, The artemis european driving cycles for measuring car pol-

lutant emissions, The Science of the total environment 334-335 (2004) 73–84. doi:10.1016/j.scitotenv.2004.04.070.

[87] L. O. Valoen, J. N. Reimers, Valoen_transport properties of lipf6-based li-ion battery electrolytes, Journal of The Electrochemical Society 152 (5) (2005) A882–A891.

[88] C. M. Long, M. A. Nascarella, P. A. Valberg, Carbon black vs. black carbon and other airborne materials containing elemental carbon: physical and chemical distinctions, Environmental pollution (Barking, Essex : 1987) 181 (2013) 271–286. doi:10.1016/j.envpol.2013.06.009.

[89] G. Liu, H. Zheng, A. S. Simens, A. M. Minor, X. Song, V. S. Battaglia, Optimization of acetylene black conductive additive and pvdf composition for high-power rechargeable lithium-ion cells, Journal of The Electrochemical Society 154 (12) (2007) A1129. doi:10.1149/1.2792293.

[90] R. Dash, S. Pannala, Theoretical limits of energy density in silicon-carbon composite anode based lithium ion batteries, Scientific reports 6 (2016) 27449. doi:10.1038/srep27449.

[91] R. Jung, M. Metzger, F. Maglia, C. Stinner, H. A. Gasteiger, Oxygen release and its effect on the cycling stability of lini x mn y co z o 2 (nmc) cathode materials for li-ion batteries, Journal of The Electrochemical Society 164 (7) (2017) A1361–A1377. doi:10.1149/2.0021707jes.

[92] H.-J. Noh, S. Youn, C. S. Yoon, Y.-K. Sun, Comparison of the structural and electrochemical properties of layered li[nixcoymnz]o2 (x = 1/3, 0.5, 0.6, 0.7, 0.8 and 0.85) cathode material for

lithium-ion batteries, Journal of Power Sources 233 (2013) 121–130. doi:10.1016/j.jpowsour.2013.01.063.

[93] M. Doyle, Y. Fuentes, Computer simulations of a lithium-ion polymer battery and implications for higher capacity next-generation battery designs, Journal of The Electrochemical Society 150 (6) (2003) A706. doi:10.1149/1.1569478.

[94] D. Andre, S.-J. Kim, P. Lamp, S. F. Lux, F. Maglia, O. Paschos, B. Stiaszny, Future generations of cathode materials: an automotive industry perspective, Journal of Materials Chemistry A 3 (13) (2015) 6709–6732. doi:10.1039/C5TA00361J.

[95] D. A. G. Bruggeman, Calculation of different physical constants in heterogenous substances, Ann. Phys. 416 (7) (1935) 636–664. doi:10.1002/andp.19354160705.

[96] S. Arrhenius, About the reaction rate during acid induced inversion of sucrose, Zeitschrift für Physikalische Chemie 4 (1). doi:10.1515/zpch-1889-0116.