A computationally efficient implementation of a battery pack electrochemical model using waveform relaxation

Giacomo Saccani [a,*], Gabriele Ciaramella [b], Davide M. Raimondo [a]

[a] *Department of Electrical, Computer and Biomedical Engineering, University of Pavia, 27100 Pavia, Italy*
[b] *MOX-Laboratory, Politecnico di Milano, 20133 Milano, Italy*

ARTICLE INFO

Keywords:
Lithium-ion batteries
Waveform relaxation
SPMeT with aging
Fast simulation

ABSTRACT

Electrochemical models (EMs) can be used to accurately describe the phenomena occurring inside the cells composing a battery pack. However, the computational cost required for their numerical simulation grows exponentially with system size. The aim of this work is to propose a method based on the waveform relaxation framework able to provide a significant speed up in the simulation of large battery packs. The iterative approach we propose is based on the decomposition of the model in smaller submodels that are solved in parallel. The methodology is general and can be used in principle with any EM. A battery pack composed of parallel connected cells, modeled with a Single Particle Model with Electrolyte, Thermal and aging dynamics (SPMeT with aging), is used in this work as a proof of concept. Results show that using appropriate conditions, it is possible to obtain a significantly faster convergence than centralized methods to the solution of the original problem for realistic battery packs (e.g. Tesla Model S battery level — 74 cells in parallel) with a high level of precision.

1. Introduction

Lithium-ion (Li-ion) batteries are nowadays used in a variety of applications, both industrial and domestic, as portable energy sources. Amongst the most notable ones, we find electric and hybrid vehicles, power storage for grid applications and consumer electronics. Nowadays, battery cells are manufactured using different technologies (Nickel-Cadmium, Nickel-Metal hydride, Lead acid....), but the most used one is Li-ion because of its good safety level, reduced cost and performance characteristics [1].

In order to reach the desired levels of voltage and capacity for the different applications, single cells are connected together to form battery packs [2]. For example, a Tesla Model S battery pack is composed of 6 modules. The single module is formed by 6 levels connected in series, each containing 74 cells connected in parallel [3]. In literature, many dynamical models are available to describe the behavior of the Li-ion cell: Equivalent Circuit Models (ECMs) [4] and Electrochemical Models (EMs) [5,6] are among the most used. The first ones are typically simpler and more intuitive, containing only few states, and are well suited for real-time applications. On the other hand, EM can provide a very precise representation of the phenomena inside the cell at the price of a higher computational cost. Among the several EMs developed in the literature (see e.g. [6–8]), the most used ones are the Pseudo-Two-Dimensional (P2D) model, also known as Doyle–Fuller–Newman [9], and the Single Particle Model (SPM) [10], which is a simplification of the P2D model. Also notable are multi-scale models [11–13], that accurately take into account both microscopic and macroscopic dynamics, from atomic interactions to the whole electrodes dynamics. The use of accurate models that well describe the dynamic behavior is crucial both in the design and control aspects of a battery pack. This can be useful during the design of new types of cells, or in the design and test of Battery Management Systems (BMSs) [14]. During the design phase, having a good model allows one to reduce the experiment time and cost, while in the management phase it allows to test the effectiveness of a BMS in terms of safety [15,16], fast charging and battery life [17]. Note that safety is of particular importance, because certain faults could result in fire and explosions, compromising the functioning of the battery, but also posing a risk to the users and the environment.

The computational complexity of simulating EMs for single cells increases with their accuracy [18]. The problem is even more evident when simulating a battery pack composed of hundreds of cells. In this case, differently from the single cell, we have to deal also with Kirchhoff's laws for currents and voltages, that make the system a Differential Algebraic Equation (DAE) problem. In the literature, several works tackle the problem of reducing the complexity of EMs [19–22],

* Corresponding author.
E-mail address: giacomo.saccani01@universitadipavia.it (G. Saccani).

try to balance the trade-off between complexity and accuracy [7,23] or consider numerical methods for the efficient simulation of single cells [24–27]. The use of computational reduction is extremely important for battery simulation purposes, parameter identification, state estimation and management. In the parameter identification of a P2D battery model, for example, several approaches have been developed in order to reduce the complexity of the identification procedure, see e.g. Chun et al. [28] and Kim et al. [29], that respectively use a long short-term neural network and a Bayesian neural network. Regarding state estimation several schemes have been proposed, such as the use of Kalman filtering [30,31], particle filter [32] or a multiple parameter optimization [33].

This work is focused on the numerical solution of the electrochemical model of a battery pack in a fast and efficient way, that significantly improves the simulation scalability for large systems. In order to speed up the simulation, the method we propose is to divide the system of equations in smaller subproblems using the Waveform Relaxation (WR) method. Decomposing the system has the effect of reducing the overall complexity, because solving small subsystems, that contain fewer equations than the original problem, is more efficient than solving a large one [34]. These have to be solved iteratively in order to reach convergence to the result of the original problem [35]. WR methods are used to efficiently simulate large systems of equations. They were introduced to analyze complex integrated circuits in the time domain, hence the term *waveform*, referring to the signals [36]. As modern applications required more complex and larger integrated circuits, there was the need for a simulation technique that would be more efficient than classical and sequential integration methods. First introduced for Ordinary Differential Equation (ODE) problems [37], WR methods were extended to DAE systems, as explained and proven by Crow et al. in [38]. This methodology has been applied to many different areas such as power converters [39], reaction–diffusion systems [40], electromagnetic systems [41] and others [42–44]. To the extent of our knowledge, WR methods have never been applied to battery pack simulations. Moreover, note that WR is not a simple parallelization technique. Indeed, due to the algebraic constraints between parallel cells, an iterative scheme is required in order to guarantee the convergence to the solution of the original problem.

Starting from a standard WR approach, we will extensively explain the modifications needed to obtain a convergent iterative method for the simulation of accurate EMs: first of all we will describe the need of having overlapping subsystems. Then the idea of a correction step is introduced. Finally we will analyze numerical techniques to speed up the convergence of our WR method. In particular we will focus on a technique called Anderson Acceleration [45], that is built on top of the modified WR method and uses the results provided from previous iterations to obtain a faster convergence.

Note that the methodology we propose is general and can be used in principle to simulate any EM, providing a higher efficiency when the model complexity increases. Besides, this could be useful as an add-on to existent simulators, such as LIONSIMBA [46], BEST [47,48], MULTIBAT [49], PyBaMM [50], DandeLiion [51] and TOOFAB [7], to extend their single cell capabilities. As a proof of concept, the approach we propose has been tested on a SPMeT with aging dynamics (thermal dynamics are considered only between the cells and the coolant). In particular, a detailed analysis has been conducted on battery packs with an increasing number of parallel cells. The results show that the method we developed is very effective in simulating large systems. When the system reaches a certain number of cells, we gain a considerable advantage in computation times; for example in the case of a Tesla Model S battery level, composed of 74 cells in parallel, we obtained a reduction of computation times to a third using a 6 core processor. In addition, note that the precision of the obtained result is not affected while reducing computation times.

The paper is organized as follows: the model for the single cell and the battery pack used on which our simulator is based are presented in Section 2, the WR method and algorithm are presented in Section 3, together with the Anderson Acceleration method. Finally, in Section 4 the simulation results are presented and discussed. Section 5 contains conclusions and future developments.

2. Battery pack model

In this section, a brief overview of the model adopted for the simulations is given. The SPMeT with aging dynamics [52] consists of a simplification of the P2D model [9], where each electrode is modeled as a single porous spherical particle whose ion concentration along the radial axis, in order to reduce Fick's law into ODEs [53], is approximated with a polynomial. Moreover, Partial Differential Equations (PDEs) describing the dynamics of the electrolyte have been discretized using Finite Volumes (FV) [46]. Thermal dynamics are added, with the exchange of heat between each cell and a liquid coolant kept at constant temperature [54]. Aging dynamics are included to take into account the capacity decay over time and the growth of an internal electric resistance.

2.1. Single cell model

Consider a cell composed of its three sections, cathode, separator and anode that will be identified by the index $i \in \{p, s, n\}$, while in equations referring only to the electrodes, the index $h \in \{p, n\}$ will be used instead. The current flowing through the cell is denoted with $I_{cell}(t)$, negative during a charge cycle, positive during discharge.

The dynamics of the average stoichiometry for the cathode $\theta_p(t)$ can be defined as

$$\frac{d\theta_p(t)}{dt} = -\frac{\Delta\theta_p}{C_0} I_{cell}(t), \tag{1}$$

with $\Delta\theta_h = \theta_h^{100\%} - \theta_h^{0\%}$ (respectively fully charged and discharged electrodes stoichiometry) and C_0 being the full capacity of the cell. Then, $\theta_n(t)$ can be defined similarly

$$\frac{d\theta_n(t)}{dt} = \frac{\Delta\theta_n}{C(t)} I_{cell}(t), \tag{2}$$

with $C(t)$ being the capacity of the cell. Note that, according to the model used, see [52], the cathode average stoichiometry depends only on the full capacity of the cell, while the anode depends on the actual capacity of the cell; this is due to the fact that only the anode is affected by aging dynamics. The volume-averaged concentration fluxes are then defined

$$\frac{dq_h(t)}{dt} = -30\frac{D_{s,h}(T(t))}{R_h^2} q_h(t) - \frac{45\Delta\theta_j c_{s,h}^{max}}{6R_h C(t)} I_{cell}(t), \tag{3}$$

where $T(t)$ is the temperature of the cell and $D_{s,h}(T(t))$ is the solid diffusion coefficient of each electrode, R_h is the radius of the fictitious solid particle and $c_{s,h}^{max}$ the maximum electrode solid concentration. $D_{s,h}(T(t))$ is a nonlinear exponential function of the temperature, in accordance with Arrhenius law

$$\psi(T(t)) = \psi_0 e^{\frac{-E_{a,\psi}}{RT(t)}}, \tag{4}$$

where $E_{a,\psi}$ is the activation energy associated with the generic parameter $\psi(T(t))$, R is the universal gas constant and ψ_0 is the value of the parameter at a reference temperature. All the temperature dependent coefficients will be computed according to Eq. (4). The electrode surface stoichiometry $\theta_h^*(t)$ is a function of $\theta_h(t)$ and $q_h(t)$

$$\theta_h^*(t) = \theta_h(t) + \frac{8R_h q_h(t)}{35c_{s,h}^{max}} - \frac{R_h^2 \Delta\theta_h I_{cell}(t)}{105D_{s,h}(T(t))C(t)}. \tag{5}$$

Moving over to electrolyte dynamics, a set of PDEs describes the concentration of the electrolyte $c_{e,i}(x,t)$ in the three sections of the cell [52], where x is the longitudinal variable ($0 \leq x \leq L_i$), L_i being the length of each section

$$\epsilon_i \frac{\partial c_{e,i}(x,t)}{\partial t} = \frac{\partial}{\partial x}\left[D_{e,i}^{eff}(T(t))\frac{\partial c_{e,i}(x,t)}{\partial x}\right] + \lambda_i \frac{1-t_+}{FAL_i} I_{cell}(t), \tag{6}$$

with $\lambda_p = -1$, $\lambda_s = 0$ and $\lambda_n = 1$ and $D_{e,i}^{eff}(T(t)) = D_e(T(t))\epsilon_i^{p_i}$, calculated according to Eq. (4), ϵ_i is the porosity and p_i Bruggeman's coefficient. The area of the cell is denoted with A, F is Faraday's constant and t_+ the transference number. Each section gets divided into P non-overlapping volumes in the spatial domain along the x-axis and the system of PDEs is discretized using a FV method, where the p-th volume ($p = 1, \dots, P$) of the ith section has bounds $[x_i^{[\bar{p}]}, x_i^{[\underline{p}]}]$; the set of equations then becomes

$$\epsilon_i \frac{dc_{e,i}^{[p]}(t)}{dt} = \left[\frac{\tilde{D}_e(x,T(t))}{\Delta x_i}\frac{\partial c_{e,i}(x,t)}{\partial x}\right]\Bigg|_{x_i^{[\underline{p}]}}^{x_i^{[\bar{p}]}} + \lambda \frac{1-t_+}{FAL_i} I_{cell}(t), \tag{7}$$

where $c_{e,i}^{[p]}(t)$ is the average electrolyte concentration in the p-th volume and $\Delta x_i = L_i/P$. The electrolyte diffusion coefficients have to be adjusted in order to take into account the fact that on the border between the sections there are different materials. The harmonic mean is therefore used to calculate $\tilde{D}_e(x,T(t))$, see [46] for more details.

The terminal voltage of the cell, $V_{cell}(t)$, can be defined

$$V_{cell}(t) = U_p(t) - U_n(t) + \eta_p(t) - \eta_n(t) + \Delta\phi_e(t) - I_{cell}(t)R_{SEI}(t), \tag{8}$$

with $U_h(t)$ defined as the open circuit potential of the electrodes, calculated as a polynomial function of $\theta_h^*(t)$. The electrode overpotential is $\eta_h(t)$ and $\Delta\phi_e(t)$ the electrolyte voltage drop, both of which are nonlinear functions of the states. A full formulation for these can be found in [55].

Aging and thermal dynamics are adapted from the models in [56] and describe the degradation of capacity $C(t)$ and the growth of $R_{SEI}(t)$, the *Solid Electrolyte Interface* resistance

$$\frac{dC(t)}{dt} = \frac{3C(t)}{R_{p,n}A\Delta\theta_n c_{s,n}^{max}}\overline{J}_{side}(t), \tag{9}$$

$$\frac{dR_{SEI}(t)}{dt} = \frac{M_\omega}{\rho_n \nu}\overline{J}_{side}(t), \tag{10}$$

with $\overline{J}_{side}(t)$ being the average side reaction flux, an exponential function of $T(t)$ and $I_{cell}(t)$ (see [54] for more details). Moreover, M_ω is the molar weight, ρ_n the material density and ν the admittance. Finally, thermal dynamics can be included

$$\frac{dT(t)}{dt} = \frac{1}{C_{th}}\left(Q(t) - \frac{T(t) - T_{cool}}{R_{cool}}\right), \tag{11}$$

where C_{th} is the thermal capacity of the cell, T_{cool} the temperature of the coolant and R_{cool} its thermal resistance. The cell heat generation $Q(t)$ is

$$Q(t) = |I_{cell}(t)||V_{cell}(t) - (U_p(t) - U_n(t))|. \tag{12}$$

The set of differential equations representing a single cell will be referred to as

$$\dot{\mathbf{x}}_{cell}(t) = f_{cell}\left(\mathbf{x}_{cell}(t), I_{cell}(t)\right), \tag{13}$$

with $\mathbf{x}_{cell}(t) \in \mathbb{R}^{7+3P}$, $I_{cell}(t) \in \mathbb{R}$ and $f : \mathbb{R}^{7+3P} \times \mathbb{R} \to \mathbb{R}^{7+3P}$. The vector $\mathbf{x}_{cell}(t)$ is

$$\mathbf{x}_{cell}(t) = \left[\theta_p(t)\ \theta_n(t)\ q_p(t)\ q_n(t)\ \mathbf{c}_{e,p}(t)\ \mathbf{c}_{e,s}(t)\ \mathbf{c}_{e,n}(t)\ C(t)\ R_{SEI}(t)\ T(t)\right]^\top, \tag{14}$$

having defined $\mathbf{c}_{e,i}(t) = \left[c_{e,i}^{[1]}(t), \dots, c_{e,i}^{[P]}(t)\right]$.

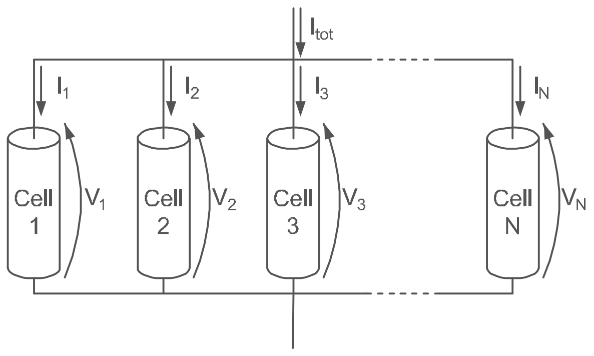


Fig. 1. Parallel connected battery pack.

2.2. Battery pack model

Cells can be connected together in order to increase the total output voltage, if a series connection is considered, or have a higher capacity, in a parallel case. In most applications, a combination of these configurations is used, to boost voltage and capacity to the desired level. In this work we consider only a parallel connection between single cells that exchange heat only with the coolant. This assumption was made according to other works (see i.e. [57–59]), where it is shown that temperature difference between different cells is in the order of $0.5K$, meaning that the thermal exchange can be neglected. The contact surface between each cell and the coolant is assumed to be the same, as well as the heat exchange coefficient. Note that, under the hypotheses made for the thermal dynamics, if we were to consider also single cells connected in series, there would not be any difference from a methodology point of view, since these would be independent from each other. The WR approach we are presenting is a proof of concept for the speed up of computations. Future works will include the extension to a full battery pack with the inclusion of thermal dynamics.

For cells connected in parallel we have that the voltage across every cell has to be equal and the total current flowing in the battery pack is divided among all the cells, according to Kirchhoff's law. Indicating with $I_{tot}(t)$ the total current we have

$$\begin{cases} \sum_{h=1}^{N} I_h(t) - I_{tot}(t) = 0 & \forall t \geq 0, \\ V_h(t) - V_{h-1}(t) = 0 & \forall t \geq 0, \forall\, h \in \{2, \dots, N\}, \end{cases} \tag{15}$$

with N being the number of parallel connected cells. The voltage $V_h(t)$ is computed according to Eq. (8) and is a function of the states $\mathbf{x}_h(t)$ and of the current $I_h(t)$ of the single cell. Note that $I_{tot}(t)$ is considered to be the known input of the system, assumed negative if the battery is charging.

Combining the state Eq. (13) for each cell, with the algebraic constraints in Eq. (15), a DAE system is obtained. In Fig. 1 a scheme of the considered battery pack is presented.

2.3. Cell and battery pack parameters

All the parameters used in the previous sections refer to a Kokam SLPB 75106100 cell and are in accordance with Ecker et al. [60,61]. They are reported in Table 1. These parameters were found experimentally by taking measures of a real cell. From a manufacturing standpoint, it is impossible to create cells that are exactly equal one another, so to simulate this fact, a variance was introduced on some of the parameters. The value for this variance was taken from the literature (see [62,63]) and the quantities to which is applied are indicated with an asterisk in Table 1. Note that not all the parameters have a variance applied to them, while in practice also these parameters may exhibit differences. This was done, once again, according to [60,61]. In addition, the focus of this work is to prove the effectiveness of the

Table 1
Cell parameters.

Parameter	Value	Unit	Parameter	Value	Unit
$\theta_p^{100\%}$	0.26	–	$\theta_p^{0\%}$	0.86	–
$\theta_n^{100\%}$	0.75	–	$\theta_n^{0\%}$	0.04	–
$c_{s,p}^{max}$	48 580	$\frac{\text{mol}}{\text{m}^3}$	$c_{s,n}^{max}$	31920	$\frac{\text{mol}}{\text{m}^3}$
$R_p^{(*)}$	6.49×10^{-6}	m	$R_n^{(*)}$	8.7×10^{-6}	m
A	0.4121	m^2	$L_p^{(*)}$	54.5×10^{-6}	m
$L_s^{(*)}$	19×10^{-6}	m	$L_n^{(*)}$	73.7×10^{-6}	m
t_+	0.26	–	$\epsilon_p^{(*)}$	0.296	–
$\epsilon_s^{(*)}$	0.508	–	$\epsilon_n^{(*)}$	0.329	–
M_ω	73×10^{-3}	$\frac{\text{kg}}{\text{mol}}$	ρ_n	2500	$\frac{\text{kg}}{\text{m}^3}$
ν	3.79×10^{-7}	$\frac{S}{m}$	C_{th}	88.26	$\frac{\text{J}}{\text{K}}$
T_{cool}	310	K	R_{cool}	41.86	$\frac{\text{J}}{\text{K}}$

methodology simulating cells that are different from each other; the fact that some parameters are constant does not affect the properties of the proposed scheme.

For what concerns the effects the variance has on the model, we have to recall once again Kirchhoff's law: in a parallel connection, if the cells are identical and assuming initial conditions equal for all of them, we would have that the current splits evenly among them, making the DAE problem trivial. Otherwise, the current would divide itself in the N branches based on the value of the cells' parameters.

2.4. DAE system

Expliciting the formulation of the system, we have that a battery pack like the one in Fig. 1 composed of N cells connected in parallel, is represented by the following set of equations

$$\begin{cases} \dot{\mathbf{x}}_1(t) = f_1\left(\mathbf{x}_1(t), I_1(t)\right), \\ \quad\vdots \\ \dot{\mathbf{x}}_N(t) = f_N\left(\mathbf{x}_N(t), I_N(t)\right), \\ \mathbf{x}_1(0) = \hat{\mathbf{x}}_1, \\ \quad\vdots \\ \mathbf{x}_N(0) = \hat{\mathbf{x}}_N, \\ \sum_{h=1}^N I_h(t) - I_{tot}(t) = 0, \\ V_1(t) - V_2(t) = 0, \\ \quad\vdots \\ V_{N-1}(t) - V_N(t) = 0, \end{cases} \tag{16}$$

where the voltage $V_h(t)$ of each cell is computed according to Eq. (8). The initial conditions $\hat{\mathbf{x}}_h$, for $h = 1, \dots, N$, are needed to make the problem solvable. We can aggregate the variables into vectors

$$\mathbf{X}(t) = \left[\mathbf{x}_1(t), \mathbf{x}_2(t), \dots, \mathbf{x}_N(t)\right]^\top, \tag{17a}$$

$$\mathbf{I}(t) = \left[I_1(t), I_2(t), \dots, I_N(t)\right]^\top, \tag{17b}$$

with $\mathbf{X}(t) \in \mathbb{R}^n$, having defined the number of states $n = N(7 + 3P)$ and $\mathbf{I}(t) \in \mathbb{R}^N$. The vector $\mathbf{X}(t)$ collects the states of every cell contained in the system, according to Eq. (14), while $\mathbf{I}(t)$ includes the algebraic variables, in our case the currents.

System (16) can be expressed using a condensed form

$$\begin{cases} \dot{\mathbf{X}}(t) = F\left(\mathbf{X}(t), \mathbf{I}(t)\right), \\ 0 = G\left(\mathbf{X}(t), \mathbf{I}(t), I_{tot}(t)\right), \\ \mathbf{X}(0) = \hat{\mathbf{X}}, \end{cases} \tag{18}$$

with $F : \mathbb{R}^n \times \mathbb{R}^N \to \mathbb{R}^n$, $G : \mathbb{R}^n \times \mathbb{R}^N \times \mathbb{R} \to \mathbb{R}^N$. The function $F\left(\mathbf{X}(t), \mathbf{I}(t)\right)$ comprises all the equations describing the model presented in Section 2.1 for each cell, while $G\left(\mathbf{X}(t), \mathbf{I}(t), I_{tot}(t)\right)$ contains the algebraic constraints previously obtained in Eq. (15), with $I_{tot}(t)$ as the input current for the entire battery pack.

2.5. Simulation

Given a finite horizon $[0, T]$, system (18) can be solved using, for example, the built-in Matlab® function *ode15s*, which is suitable to solve stiff ODE and DAE problems using implicit integration. In the following we rely on *ode15s*, but any other integrator can be used.

Initial conditions need to be provided in order for the problem to be solvable and they are assumed to be equal for all the cells, $\hat{\mathbf{x}}_1 = \hat{\mathbf{x}}_2 = \dots = \hat{\mathbf{x}}_N$. Without loss of generality, their values are defined at $t = 0$ as follows

$$\hat{\mathbf{x}}_h = [0.83\ 0.1\ 0\ 0\ 2000, \dots, 2000\ 27000\ 0.01\ 310]^\top. \tag{19}$$

Simulating the problem corresponding to Eq. (18) as a whole, will be referred to as "centralized" method, opposed to the method we are proposing of dividing the system into smaller subproblems, to reduce their complexity. In the next Section, the Waveform Relaxation (WR) approach is presented and detailed.

3. Waveform relaxation

First, the WR approach is described in detail with all the steps needed to obtain a method for the simulation of an EM-based battery pack that is faster than the centralized one. After, we will present some examples of its efficiency and precision. In fact, the purpose of this work is to obtain a method that has the same solution of the centralized one, while having a faster simulation time. The convergence of WR methods has been extensively proven in the literature (see e.g. [64]) and will be also proven in this specific case.

3.1. Waveform relaxation algorithm

Given a battery pack composed of N cells, let us consider the set of indices $\mathcal{I} := \{1, \dots, N\} \subset \mathbb{N}$, where each index corresponds to a cell. We decompose $\mathcal{I}$ in $r \in \mathbb{N}$ subsets (or *subdomains*) denoted by $\langle j \rangle \subset \mathcal{I}$, $j = 1, \dots, r$, such that $\cup_{j=1}^r \langle j \rangle = \mathcal{I}$. We assume that each subdomain contains ordered and successive indices and has cardinality s, that is $\text{card}\,\langle j \rangle = s$ for $j = 1, \dots, r$. The cardinality s is called *length of the subdomains*. Moreover, we assume that

$$\begin{aligned} \text{card}\left(\langle j \rangle \cap \langle j+1 \rangle\right) = q \text{ for } j = 1, \dots, N-1, \\ \text{card}\left(\langle j \rangle \cap \langle k \rangle\right) = 0 \text{ for } k < j-1,\ k > j+1, \end{aligned} \tag{20}$$

where the positive integer q is called *overlap*. Hence, for $q > 0$, the decomposition $\mathcal{I} = \cup_{j=1}^r \langle j \rangle$ is said to be an *overlapping decomposition* and we clearly have that each subdomain $\langle j \rangle$ overlaps with only the two neighboring subdomains $\langle j-1 \rangle$ and $\langle j+1 \rangle$. As an example, let us consider a battery pack of 7 cells as the one depicted in Fig. 2. This pack corresponds to the index set $\mathcal{I} = \{1, \dots, 7\}$. A possible overlapping decomposition of $\mathcal{I}$ is the one shown in red in Fig. 2, where we have the subdomains

$$\langle 1 \rangle = \{1, 2, 3\},\ \langle 2 \rangle = \{3, 4, 5\},\ \langle 3 \rangle = \{5, 6, 7\}, \tag{21}$$

with an overlap $q = 1$ and a length $s = 3$. The relation between r, s, q and N is given by

$$N = r(s - q) + q. \tag{22}$$

Note that, in general, the only given value is N, while the other quantities are free parameters of choice. Now, let us denote by $\langle j \rangle, \ell$ the ℓ-th index (or cell) of the jth subdomain. We can rewrite the current equilibrium $\sum_{h=1}^N I_h(t) - I_{tot}(t) = 0$ as

$$\sum_{\ell=1}^{s} I_{\langle j \rangle, \ell}(t) + I_{rest, \langle j \rangle}(t) - I_{tot}(t) = 0, \tag{23}$$

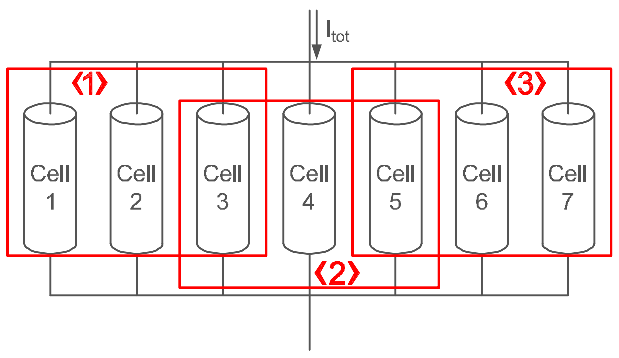


Fig. 2. Example of overlapping subsystems in a case with $N = 7$, $s = 3$ and $q = 1$.

for $j = 1, \dots, r$, where

$$I_{rest,\langle j\rangle}(t) = \sum_{h\in\mathcal{I}\setminus\langle j\rangle} I_h(t). \tag{24}$$

The term $I_{rest,\langle j\rangle}(t)$ represents the sum of the currents in all cells of the whole battery pack excluding the ones corresponding to the subdomain $\langle j\rangle$. At this point, an important remark is needed: the definition of $I_{rest,\langle j\rangle}(t)$ allows one to avoid accounting for the same currents multiple times. In other words, we want to take and sum every current only once, even if the subsystems are overlapping. Even though this is formally equivalent to the constraint on the sum of the currents in system (16), this formulation allows us to make the subsystems independent from each other, with the only condition of having $I_{rest,\langle j\rangle}(t)$ known. We will see that the WR method is iterative, and this value will be taken from the results of previous iterations.

Using the notation just presented, we can reformulate the DAE system (16) in an equivalent *domain decomposition* form:

$$\begin{cases} \dot{\mathbf{x}}_{\langle j\rangle,1}(t) = f_{\langle j\rangle,1}\left(\mathbf{x}_{\langle j\rangle,1}(t), I_{\langle j\rangle,1}(t)\right), \\ \quad\vdots \\ \dot{\mathbf{x}}_{\langle j\rangle,s}(t) = f_{\langle j\rangle,s}\left(\mathbf{x}_{\langle j\rangle,s}(t), I_{\langle j\rangle,s}(t)\right), \\ \mathbf{x}_{\langle j\rangle,1}(0) = \hat{\mathbf{x}}_{\langle j\rangle,1}, \\ \quad\vdots \\ \mathbf{x}_{\langle j\rangle,s}(0) = \hat{\mathbf{x}}_{\langle j\rangle,s}, \\ \sum_{\ell=1}^{s} I_{\langle j\rangle,\ell}(t) + I_{rest,\langle j\rangle}(t) - I_{tot}(t) = 0, \\ V_{\langle j\rangle,1}(t) - V_{\langle j\rangle,2}(t) = 0, \\ \quad\vdots \\ V_{\langle j\rangle,s-1}(t) - V_{\langle j\rangle,s}(t) = 0, \end{cases} \tag{25}$$

for $j = 1, \dots, r$. A compact formulation can be provided for this system:

$$\begin{cases} \dot{\mathbf{X}}_{\langle j\rangle}(t) = F_{\langle j\rangle}\left(\mathbf{X}_{\langle j\rangle}(t), \mathbf{I}_{\langle j\rangle}(t)\right), \\ 0 = G_{\langle j\rangle}\left(\mathbf{X}_{\langle j\rangle}(t), \mathbf{I}_{\langle j\rangle}(t), I_{rest,\langle j\rangle}(t), I_{tot}(t)\right), \\ \mathbf{X}_{\langle j\rangle}(0) = \hat{\mathbf{X}}_{\langle j\rangle}, \end{cases} \tag{26}$$

for $j = 1, \dots, r$, where the vectors $\mathbf{X}_{\langle j\rangle}(t)$ and $\mathbf{I}_{\langle j\rangle}(t)$ are defined as

$$\mathbf{X}_{\langle j\rangle}(t) = \left[\mathbf{x}_{\langle j\rangle,1}(t), \dots, \mathbf{x}_{\langle j\rangle,s}(t)\right]^{\top}, \tag{27a}$$

$$\mathbf{I}_{\langle j\rangle}(t) = \left[I_{\langle j\rangle,1}(t), \dots, I_{\langle j\rangle,s}(t)\right]^{\top}. \tag{27b}$$

Note that the function $F_{\langle j\rangle} : \mathbb{R}^{s(7+3P)} \times \mathbb{R}^{s} \to \mathbb{R}^{s(7+3P)}$ has as arguments only the states and the currents of subsystem j, while $G_{\langle j\rangle} : \mathbb{R}^{s(7+3P)} \times \mathbb{R}^{s} \times \mathbb{R} \times \mathbb{R} \to \mathbb{R}^{s}$ has as arguments the states and the currents of the jth subsystem, the current $I_{rest,\langle j\rangle}(t)$ and the total input current $I_{tot}(t)$. Notice also that $G_{\langle j\rangle}$ is different from the function G defined in Eq. (18): in Eq. (25) the current constraint has been rewritten to distinguish the currents belonging to the subsystem from all the other ones. The objective of this subdivision is to create independent subsystems that can be solved in parallel and, if their resolution is iterated, to obtain a result that will converge to the solution of the original problem (16). Moreover, because these subsystems are smaller and therefore in principle faster to solve than (16), we will see that this type of resolution can be more time efficient than the centralized method.

At this point, it is crucial to discuss the importance of having an overlapping decomposition. Clearly, the first intuitive approach would be to consider no overlap ($q = 0$), to reduce the number of cells in each subdomain that are present multiple times in different subsystems, thus reducing complexity. This is however not possible because it would result in a *domain decomposition* system not equivalent to the main problem (16). Let us explain this issue with a simple example. Consider the system of 7 cells of Fig. 2. The corresponding set $\mathcal{I} = \{1, \dots, 7\}$ is decomposed in the two non-overlapping[1] subdomains $\langle 1\rangle = \{1, 2, 3\}$ and $\langle 2\rangle = \{4, 5, 6, 7\}$, with $q = \operatorname{card}\langle 1\rangle \cap \langle 2\rangle = 0$. The first subproblem corresponding to $\langle 1\rangle$ is characterized by the algebraic constraints for the equilibrium of the voltages

$$\begin{cases} V_1(t) - V_2(t) = 0, \\ V_2(t) - V_3(t) = 0, \end{cases} \tag{28}$$

while the second subproblem corresponding to $\langle 2\rangle$ has the voltage constraints

$$\begin{cases} V_4(t) - V_5(t) = 0, \\ V_5(t) - V_6(t) = 0, \\ V_6(t) - V_7(t) = 0. \end{cases} \tag{29}$$

It is then clear that the constraint

$$V_3(t) - V_4(t) = 0 \tag{30}$$

is missing in both sets (28) and (29). Therefore, by solving the two subsystems corresponding to $\langle 1\rangle$ and $\langle 2\rangle$, it is not guaranteed that the constraint (30) is satisfied (even though the subsystems $\langle 1\rangle$ and $\langle 2\rangle$ are coupled in terms of the sum of the currents, i.e. Eq. (23)). Depending on the different properties of the single cells, it can happen that the obtained current distribution leads to voltages $V_3(t)$ and $V_4(t)$ that violate (30). On the other hand, if one enlarges by one cell the first subdomain and considers an overlapping decomposition $\langle 1\rangle = \{1, 2, 3, 4\}$ and $\langle 2\rangle = \{4, 5, 6, 7\}$, then the voltage of a cell appears in both subsystems. In the considered example we will have that $V_4(t)$ appears as $V_3(t) - V_4(t)$ in $\langle 1\rangle$ and $V_4(t) - V_5(t)$ in $\langle 2\rangle$. Doing so we ensure that all the algebraic constraints are enforced in all the subsystems. However note that this is not sufficient in order to guarantee that the solution of a *domain decomposition* based method coincides with the solution of the original problem (16) (i.e. obtain convergence). For further discussions regarding the choice of the overlap in the field of domain decomposition methods, we refer to, e.g., [40,41,65,66] and references therein.

We mentioned previously that the resolution of subsystems needs to be iterated. Before introducing the iterative algorithm, we need to discuss convergence conditions. From the *domain decomposition* systems (26) it is possible to derive a Jacobi-type WR iteration. Considering an iteration k, for a given $\mathbf{I}^{k-1}_{\langle j\rangle}(t)$, $j = 1, \dots, r$ computed in the previous iterations (or an initial guess, for the first one), the WR iteration is computed by solving (in parallel) for $\mathbf{X}^{k}_{\langle j\rangle}(t)$ and $\mathbf{I}^{k}_{\langle j\rangle}(t)$, the subproblems

$$\begin{cases} \dot{\mathbf{X}}^{k}_{\langle j\rangle}(t) = F_{\langle j\rangle}\left(\mathbf{X}^{k}_{\langle j\rangle}(t), \widetilde{\mathbf{I}}^{k}_{\langle j\rangle}(t)\right), \\ 0 = G_{\langle j\rangle}\left(\mathbf{X}^{k}_{\langle j\rangle}(t), \widetilde{\mathbf{I}}^{k}_{\langle j\rangle}(t), I^{k-1}_{rest,\langle j\rangle}(t), I_{tot}(t)\right), \\ \mathbf{X}^{k}_{\langle j\rangle}(0) = \hat{\mathbf{X}}_{\langle j\rangle}, \\ \widetilde{\mathbf{I}}^{k}_{\langle j\rangle}(t) = D_{\langle j\rangle}\mathbf{I}^{k}_{\langle j\rangle}(t) + (\mathbb{I} - D_{\langle j\rangle})\mathbf{I}^{k-1}_{\langle j\rangle}(t), \end{cases} \tag{31}$$

[1] Without loss of generality and in order to re-use the example of Fig. 2, the assumption of having $\operatorname{card}\langle j\rangle = s$ for $j = 1, \dots, r$ was neglected.

for $k = 1, 2, \dots$, where $\mathbb{I}$ is the $s \times s$ identity matrix and $D \in \mathbb{R}^{s\times s}$ is a diagonal matrix having diagonal entries equal to

$$D_{\langle j\rangle}(\ell,\ell) := \begin{cases} \omega & \text{if } \ell \in \{1,\dots,q\} \cup \{s-q+1,\dots,s\} \text{ and } j = 2\dots,r-1, \\ \omega & \text{if } \ell \in \{s-q+1,\dots,s\} \text{ and } j = 1, \\ \omega & \text{if } \ell \in \{1,\dots,q\} \text{ and } j = r, \\ 1 & \text{otherwise.} \end{cases} \tag{32}$$

Here, $\omega \in [0,1]$ is a *relaxation parameter* that can affect the convergence speed of the WR method. Notice that ω is applied only to the currents of the overlap and the definition of $D_{\langle j\rangle}$ takes into account that the first and last subsystem do not overlap. The systems defined in (31) are DAE problems of the same form of the full DAE system (18), but they have much smaller dimension since they are defined on $s \ll N$ cells. Notice how the coupling among the subsystems is obtained via the currents $I^{k-1}_{rest,\langle j\rangle}(t)$, which are computed using $I^{k-1}_{\langle j\rangle,\ell}(t)$, at the $(k-1)$-th iteration according to Eq. (24). Moreover, since the subsystems (31) have the same form of (18), one can use exactly the same numerical solver (e.g., the built-in Matlab® function *ode15s*).

Let us now briefly discuss about the choice of the relaxation parameter ω in Eqs. (31), (32). If $\omega = 0$ is chosen, then the currents in the overlap do not account for values coming from the previous iterations and the method defined by Eq. (31) cannot converge to a correct solution. This happens because the values of $I^{k}_{\langle j\rangle,\ell}(t)$ never get updated, the values $I^{k-1}_{\langle j\rangle,\ell}(t)$ are used, and therefore the currents are always equal to the initial conditions. The opposite, and intuitive, choice would be to assign $\omega = 1$, which leads to a full update of the currents in the overlap. This typical choice, which corresponds to a standard WR method, does not necessarily lead to a convergent method. For similar discussions in the context of domain decomposition methods for stationary equation we refer to [67]. If the standard WR relaxation iteration (with $\omega = 1$) is not convergent, one can try to reduce the value of ω to relax (or damp) the iterations and obtain convergence. However, even if convergence is obtained by a small ω, the iteration converges generally quite slowly. Only after introducing an appropriate *correction step*, the positive effect of ω on the convergence of our WR method becomes significant.

Obtaining convergence is a crucial step of our method: during the first tests of the WR method we observed a divergent behavior. Let us discuss the issue that we encountered when using the WR iteration (31). The algebraic constraint on the current needs to be corrected along the iterations. Since the goal is to guarantee that the current constraint (15) is satisfied, the sum

$$I^k_{sum}(t) = \sum_{j=1}^{r}\sum_{\ell=1}^{s} D_{\langle j\rangle}(\ell,\ell) I^k_{\langle j\rangle,\ell}(t) \qquad \forall t \in [0,T], \tag{33}$$

with $D_{\langle j\rangle}(\ell,\ell)$ defined in (32), should become closer to the value of $I_{tot}(t)$, i.e. the given input, as iterations go on. Notice that the value of $I^k_{sum}(t)$ is computed in such a way that the currents belonging to the cells of the overlap are not accounted twice. Instead we weight them and take half coming from a subsystem and half from the other.[2] However, in the performed tests, we computed $I^k_{sum}(t)$ at the end of each iteration and we observed a lack of convergence: the sum of the currents $I^k_{sum}(t)$ oscillates heavily around the correct value $I_{tot}(t)$ and the oscillations become larger and larger along the iterations, as shown in Fig. 3(a). In this figure, the system and the subdivision are the same of Fig. 2, where the input was set to a constant value $I_{tot}(t) = -7.5A\ \forall t \in [0,100]$. What we observe is an over-compensation: at each iteration every subsystem tries to "correct" the small error in the value of $I^k_{sum}(t)$, generated in the previous iteration and contained in the term $I^{k-1}_{rest,\langle j\rangle}(t)$. However because the subsystems are all solved at the same time, every one of them adds its own correction to compensate for the over or under estimation of $I^{k-1}_{rest,\langle j\rangle}(t)$. When one looks at the updated value of $I^k_{sum}(t)$, all these corrections sum up, creating an even bigger error. Hence, the iterations diverge and the WR method fails.

The problem essentially lies in the lack of communication among the subsystems: each subsystem can exchange only current information with its two neighbors. To solve this problem one could introduce a so-called second-level (or coarse) correction that generally consists in a modification of the result of the WR iterate; see, e.g., [67–69] and references therein. However, the definition of an appropriate correction step is a complicated and delicate task, which depends heavily on the class of problems under consideration. For the battery pack problem considered in this paper, we designed a correction step by reinforcing the algebraic current constraint at the end of each iteration. This allows to always start the successive iteration with a feasible current distribution. At iteration k, we compute the sum $I^k_{sum}(t)$ and we calculate how far $I^k_{sum}(t)$ is from $I_{tot}(t)$, that is

$$I^k_{diff}(t) = I_{tot}(t) - I^k_{sum}(t) \qquad \forall t \in [0,T], \tag{34}$$

and we divide this quantity by the number of cells N

$$\delta^k(t) = \frac{I^k_{diff}(t)}{N} \qquad \forall t \in [0,T]. \tag{35}$$

The functions $\delta^k(t)$ are used to complete the *correction step*

$$I^{*,k}_{\langle j\rangle,\ell}(t) = I^k_{\langle j\rangle,\ell}(t) + \delta^k(t) \quad \text{for } j = 1,\dots,r \quad \text{for } \ell = 1,\dots,s \quad \forall t \in [0,T], \tag{36}$$

where the currents $I^{*,k}_{\langle j\rangle,\ell}(t)$, $j = 1,\dots,r$ and $\ell = 1,\dots,s$, satisfy the current constraint defined in (33)

$$\sum_{j=1}^{r}\sum_{\ell=1}^{s} D_{\langle j\rangle}(\ell,\ell) I^{*,k}_{\langle j\rangle,\ell}(t) = I_{tot}(t) \tag{37}$$

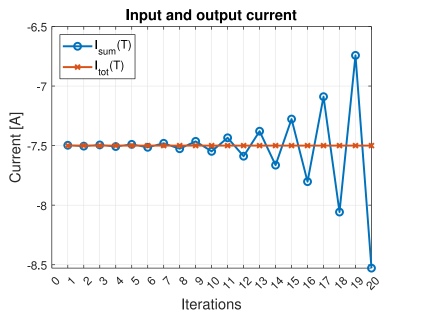
for all $t \in [0,T]$. Hence, these currents represent a feasible distribution and we can use them in place of $I^k_{\langle j\rangle,\ell}(t)$, to perform the next $(k+1)$-th iteration. In this way, we obtained that the values returned from the kth iteration, used as input of iteration $k+1$, respect the algebraic constraints. With this adjustment, we obtained the convergence of the method, as shown in Fig. 3(b), for the same conditions of Fig. 3(a).

We can now also test the method for different values of ω, comprising the *correction step*. The considered example is the same of Fig. 3(b), with $N = 7$, $s = 3$ and $q = 1$. The blue line of Fig. 4 corresponds to $\omega = 1$, i.e. the case where the overlap currents are fully updated, and the red line corresponds to $\omega = 0.5$. The use of the parameter ω $(0 < \omega \leq 1)$ has the effect of retaining more information throughout iterations, making them more efficient. The improvement is not radical, however it is interesting to see that without changing the structure of the problem and its computational complexity, we can obtain a faster convergence.

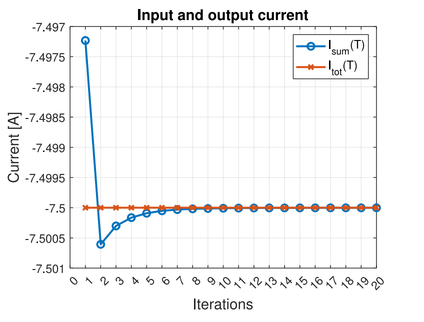
Algorithm 1 (WR Jacobi-type algorithm)

```
Require: $I_{tot}(t), \mathbf{I}^0(t)\ \forall\ t \in [0,T]$, $\mathbf{X}(0)$, a parameter $\omega \in (0,1]$, a tolerance $\epsilon$ and $k_{max}$.
1: Set $k \leftarrow 1$ and $\mathbf{I}^1 \leftarrow$ inf.
2: while $||\mathbf{I}^k - \mathbf{I}^{k-1}|| > \epsilon$ and $k < k_{max}$ do
3:   for all $j \in \{1,\dots,r\}$ do
4:     Compute $I^{k-1}_{rest,\langle j\rangle}$ using Eq. (24).
5:     Solve Eq. (31) on $[0,T]$ to get $\mathbf{X}^k_{\langle j\rangle}$ and $\mathbf{I}^k_{\langle j\rangle}$.
6:   end for
7:   Compute $I^{*,k}_{\langle j\rangle,\ell}$ for $j = 1,\dots,r$ and $\ell = 1,\dots,s$ using Eq. (35) and Eq. (36).
8:   Set $I^k_{\langle j\rangle,\ell} \leftarrow I^{*,k}_{\langle j\rangle,\ell}$ for $j = 1,\dots,r$ and $\ell = 1,\dots,s$.
9:   Set $k \leftarrow k+1$.
10: end while
```

[2] This is done by choosing $\omega = 0.5$. Any other value for ω can be chosen, however Eq. (33) would have to be adjusted accordingly.



(a) Convergence of $I_{sum}(T)$ to $I_{tot}(T)$ without the correction step applied.



(b) Convergence of $I_{sum}(T)$ to $I_{tot}(T)$ with the correction step applied.

Fig. 3. Comparison between $I_{tot}(T)$ and $I_{sum}(T)$ throughout iterations. In this example we set $I_{tot}(t) = -7.5A \ \forall t \in [0, 100]$. For this graph, the final time instant of the interval was considered, however notice that this behavior is similar for every time instant.

We can present our WR strategy for the simulation of a battery pack in Algorithm 1. The inputs needed for this algorithm are the total current $I_{tot}(t)$ for the considered interval, an initial guess for the currents $\mathbf{I}^0(t)$ over the considered horizon and the states $\mathbf{X}(0)$ at the initial time instant. Moreover we also need the parameter ω previously described, a tolerance ϵ (an arbitrarily small value), used as a criteria to stop the iterations, and a maximum value of iterations k_{max}, to avoid infinite loops. The parameters of the cells have also to be provided. Having no information on the distribution of the currents in the parallel branches of the battery pack, the initial guess is obtained by dividing $I_{tot}(t)$ by N, assuming that the input current divides itself equally among the parallels. The algorithm then starts an iterating cycle, in which every subsystem gets solved for its variables on a given time interval, while all the quantities not included in the subsystem (i.e. $I^{k-1}_{rest,\langle j\rangle}$) are taken from the previous iteration. The cycle returns the values of $\mathbf{I}^k_{\langle j\rangle}(t)$ at every time instant for every subsystem, that will be used in the successive iterates to calculate $I^{k-1}_{rest,\langle j\rangle}(t)$. This gets repeated until the current modulus of the difference between successive iterates gets smaller than a certain threshold ϵ, or the number of iterations k exceeds the maximum allowed.

For what concerns the precision of the proposed method, it has been tested by comparing the current distribution (i.e. the current flowing in each of the parallel branches) obtained by the centralized method, denoted with I_{centr}, with the one obtained by the WR approach, denoted with I_{WR}. In order to know how precise I_{WR} is with respect to I_{centr} (that will be taken as the reference solution), we compute the absolute error as

$$E_{abs} = \|I_{centr} - I_{WR}\|$$

and the relative error as

$$E_{rel} = \frac{\|I_{centr} - I_{WR}\|}{\|I_{centr}\|}.$$

For the example presented above, with $N = 7$, $s = 3$, $q = 1$, a time horizon of $[0, 100]$ and $I_{tot} = -7.5A$, the absolute error is $E_{abs} = 8.98 \times 10^{-5}$ and the relative error is $E_{rel} = 1.023 \times 10^{-5}$. We have to keep in mind that the solution provided by the WR method depends on the threshold set for stopping the iterations (indicated in the algorithm with ε and here set to $\varepsilon = 10^{-6}$). Therefore, even if in some cases the accuracy of the solution appears to be low, one could lower the threshold, reducing the absolute and relative errors.

Remark 1. Convergence of the problem to the true global solution is guaranteed if the problem is well posed (i.e. has a unique solution) and the method converges. In our particular case, the succession generated by the method converges (for continuity of the considered functions), therefore it converges also to the true global solution of the problem.

In order to further improve convergence speed of the methodology presented, in the following we suggest the use of an acceleration scheme, built on top of the WR iteration.

3.2. Anderson Acceleration

The WR method presented in the previous section can be further improved by a numerical strategy called Anderson Acceleration (AA) [45], which is used to accelerate fixed-point iterations. The WR method is exactly a fixed-point iteration. Given a current distribution $\mathbf{I}^{k-1}$, the new value $\mathbf{I}^k$ is computed by one cycle of the while loop in Algorithm 1. In a compact notation, this single iteration can be written as

$$\mathbf{I}^k = S(\mathbf{I}^{k-1}),$$

where the function S is known implicitly if Algorithm 1 is available in the form of a routine.

Denote by $\mathbf{I}^j \in \mathbb{R}^N$ the approximations to a solution $\mathbf{I}$ (i.e. the correct current distribution) obtained at iterations $j = 1, \dots, k$, and by $\boldsymbol{\gamma}^j := S(\mathbf{I}^j) - \mathbf{I}^j$ the corresponding residuals. Let $m \in \mathbb{N}^+$ and define $m_k = \min(m, k)$. According to the AA strategy, the new approximation $\mathbf{I}^{k+1}$ is computed as

$$\mathbf{I}^{k+1} = \sum_{j=0}^{m_k} \alpha^k_j S(\mathbf{I}^{k-m_k+j}), \tag{38}$$

where the coefficients α^k_j for $j = 0, \dots, m_k$ are obtained as the solution to the following minimization problem

$$\min_{\alpha^k_0,\dots,\alpha^k_{m_k}} \left\| \sum_{j=0}^{m_k} \alpha^k_j \boldsymbol{\gamma}^{k-m_k+j} \right\| \quad \text{s.t.} \ \sum_{j=0}^{m_k} \alpha^k_j = 1. \tag{39}$$

In other words, what we are doing is extending the previously explained concept of using ω to weight the results coming from previous iterations, to take account also for all the previously obtained current distributions. Many algorithms and different implementations of the Anderson Acceleration method are available in the literature and they are fully described in [70], depending on the norm used in Eq. (39).

Algorithm 2 shows a possible implementation of the Anderson Acceleration.

Notice that iterations require the storage of the history of residuals of depth at most $m + 1$. This fact is crucial when the dimension N is large. In this case one has to consider small values of m to reduce the amount of information stored during the iterations.

A crucial step in the Anderson strategy is the solution of the minimization problem (39), where any norm could be used. We decided to use the Euclidean norm, so the first-order sufficient optimality condition for the minimization problem is a linear system [71]. Different approaches are considered to solve (39); see, e.g., [72]. An efficient implementation of Anderson algorithm is proposed in [70], where (39)

Algorithm 2 Anderson Acceleration.

```
Require: Fixed-point operator $S$, initial guess $\mathbf{I}^0$, $m$, $k_{\max}$, $\epsilon$.
1: Set $k \leftarrow 1$, $\mathbf{I}^1 = S(\mathbf{I}^0)$.
2: while $k < k_{\max}$ and $||\mathbf{I}^k - \mathbf{I}^{k-1}|| > \epsilon$ do
3:     Set $m_k \leftarrow \min(m, k)$.
4:     Compute $s_k = S(\mathbf{I}^k)$.
5:     Set $\gamma^k \leftarrow s_k - \mathbf{I}^k$.
6:     Solve the minimization problem $|| \sum_{j=0}^{m_k} \alpha_j^k \gamma^{k-m_k+j}||$ s.t. $\sum_{j=0}^{m_k} \alpha_j^k = 1$.
7:     Set $\mathbf{I}^{k+1} \leftarrow \sum_{j=0}^{m_k} \alpha_j^k s_{k-m_k+j}$.
8:     Set $k \leftarrow k+1$.
9: end while
```

is transformed into an unconstrained problem. To do so, one defines $\Delta\gamma^i = \gamma^{i+1} - \gamma^i$ for each i and set $\tilde{\Gamma}_k = [\Delta\gamma^{k-m_k}, \dots, \Delta\gamma^{k-1}]$. Then the least squares problem (39) is equivalent to

$$\min_{\eta=[\eta_0,\dots,\eta_{m_k-1}]^\top} ||\gamma^k - \tilde{\Gamma}_k \eta||_2. \tag{40}$$

Notice that the coefficients α_j in (39) and η_j are related by $\alpha_0 = \eta_0$, $\alpha_i = \eta_i - \eta_{i-1}$ for $1 \le i \le m_k - 1$ and $\alpha_{m_k} = 1 - \eta_{m_k-1}$. The unconstrained least-squares problem (40) leads to a modified form of Anderson acceleration given in Algorithm 3, where the least-squares solution is denoted by $\eta^k = [\eta_0^k, \dots, \eta_{m_k-1}^k]^\top \in \mathbb{R}^{m_k}$, and we have

$$\begin{aligned}\mathbf{I}^{k+1} &= S(\mathbf{I}^k) - \sum_{i=0}^{m_k-1} \eta_i^k [S(\mathbf{I}^{k-m_k+i-1}) - S(\mathbf{I}^{k-m_k+i})] \\ &= S(\mathbf{I}^k) - \mathcal{S}_k \eta^k,\end{aligned} \tag{41}$$

where $\mathcal{S}_k = [\Delta S_{k-m_k}, \dots, \Delta S_{k-1}] \in \mathbb{R}^{m_k \times m_k}$ with $\Delta S_i = S(\mathbf{I}^{i+1}) - S(\mathbf{I}^i)$ for each i. As the algorithm proceeds, the successive least-squares problems in step 7 are solved by a QR-factorization. The details can be found in [70,73].

Algorithm 3 Modified Anderson Acceleration.

```
Require: Fixed-point operator $S$, initial guess $\mathbf{I}^0$, $m$, $k_{\max}$, $\epsilon$.
1: Set $k \leftarrow 1$, $\mathbf{I}^1 = S(\mathbf{I}^0)$, $\gamma^0 = \mathbf{I}^1 - \mathbf{I}^0$ and $s_0 \leftarrow \mathbf{I}^1$.
2: while $k < k_{\max}$ and $||\gamma^k|| > \epsilon$ do
3:     Set $m_k \leftarrow \min(m, k)$.
4:     Compute $s_k = S(\mathbf{I}^k)$.
5:     Set $\gamma^k \leftarrow s_k - \mathbf{I}^k$.
6:     Set $\Delta\gamma^{k-1} \leftarrow \gamma^k - \gamma^{k-1}$ and $\tilde{\Gamma}_k \leftarrow [\Delta\gamma^{k-m_k}, \dots, \Delta\gamma^{k-1}]$.
7:     Solve the LS problem $\eta^k = [\eta_0^k, \dots, \eta_{m_k-1}^k]^\top = \mathrm{argmin}_\eta ||\gamma^k - \tilde{\Gamma}_k \eta||_2$.
8:     Set $\Delta S_{k-1} \leftarrow s_k - s_{k-1}$ and $\mathcal{S}_k \leftarrow [\Delta S_{k-m_k}, \dots, \Delta S_{k-1}]$.
9:     Set $\mathbf{I}^{k+1} \leftarrow s_k - \mathcal{S}_k \eta^k$.
10:    Set $k \leftarrow k+1$.
11: end while
```

It is important to notice that the Anderson Acceleration strategy uses the WR Algorithm 1 as a black-box. Therefore, each time the fixed-point function S is applied to a current distribution, this corresponds to one iteration of the WR Algorithm 1. This means that, if the index m_k does not get too large (as observed for the class of problems considered in this work), the effort required by one iteration of the Anderson strategy is essentially equal to the one corresponding to one iteration of the WR Algorithm 1. More details on these algorithms and their convergence proofs can be found in [70,73,74]. The improvement brought by the Anderson acceleration method is tremendous and can be appreciated in Fig. 4, where it is compared with the previously described methods. We can see that for the first iterations it has a similar behavior, but then the gradient of descent becomes much steeper. The stagnation we see at the end is due to the tolerance used by the numerical integrator. It was noted that reducing the tolerance, the value at which stagnation is reached becomes lower.

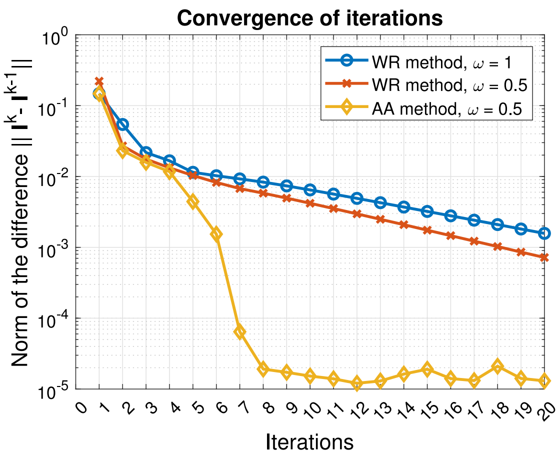


Fig. 4. Comparison of the convergence for all the presented methods.

4. Simulation results and discussion

In this section we will show that the WR method improved with the AA can be useful to speed up the simulation of a battery pack with respect to a centralized method.[3] In all the following examples, without loss of generality, the time horizon is set to $[0, 100]$. The value of the total current $I_{tot}(t)$ was chosen proportional to the number of cells in the system and constant $\forall t$ in the considered time interval. The threshold for stopping iterations is set to $\epsilon = 10^{-6}$.

The first simulations that are presented in Fig. 5, refer to the centralized approach and show the scaling of computation times with an increasing number of parallel connected cells. It is possible to note that the scaling is exponential, with a steeper growth for the cases with higher values of P (i.e. the number of finite volumes used for the discretization of the electrolyte dynamics equation). The number of discretization volumes P was chosen in such a way that further increasing its value would not result in any sensible difference in the behavior of the other states and of the output voltage.[4] Note that the centralized method uses an algorithm (*ode15*) that is serial and does not benefit of the parallel capabilities of the processor. This was verified by forcing the centralized method to run on a single processor and comparing it with the computation times of simulations where it was allowed to freely allocate resources. No difference was noted. This has been further verified in Fig. 6, where it is possible to note that for the single subsystem, the computation time is exactly equal for all the curves (i.e. simulating the same system using a different number of processors).

The algorithm we propose has the advantage, with respect to the centralized approach, of solving smaller subproblems in parallel, as we have $s \ll N$. However, in order for the WR method to be effective, we have to take into account the drawback of having to iterate the resolution of the subproblems to obtain the same current distribution of the original problem. As explained previously, the algorithm relies on the parallel capabilities of the processor: having up to 6 cores available means that up to 6 subsystems can be solved simultaneously.[5] The advantage of having a parallel algorithm can be appreciated in Fig. 6, where a single iteration of the WR method is simulated using

[3] All the simulations presented were carried out on a Windows 10 machine with 8 Gbytes of RAM and an Intel® Core™ i7-9750H 2.60 GHz 6-core processor.

[4] Proven experimentally by simulating the same system with different values of P.

[5] If the number of subsystems is greater than the number of cores, they are solved in groups in a sequential way.

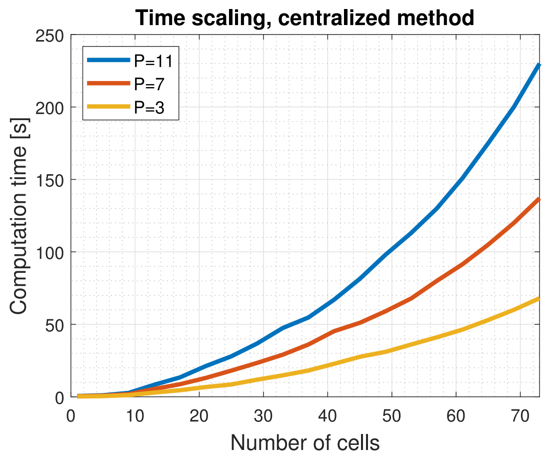


Fig. 5. Time scaling for a centralized simulation with different values of P with an increasing number of parallel cells.

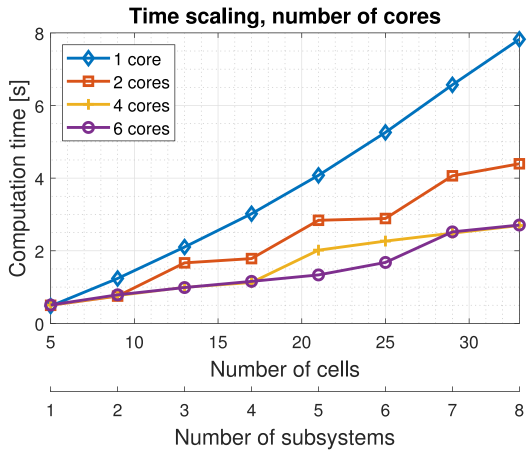


Fig. 6. Scaling of simulation time for the resolution of a single iteration with increasing number of available cores in the processor. The size of the subsystems is fixed to $s = 5$.

a different number of processors, having fixed the subsystem size and the overlap ($s = 5$, $q = 1$). Note that on the x-axis there are the number of cells N and the number of subsystems r. The blue line represents the single core performance and we can see that it is significantly higher than the others. The other lines show a step behavior, where there are almost horizontal traits interspersed with steeper increases in computation times. These latter are present when the number of subsystems r becomes equal to a multiple of the number of available cores. As example take the red line, corresponding to a simulation performed with two cores. Every addition of two subsystems there is a step up in the computation time. The same holds for the other lines. Note that the ideal behavior would exhibit a constant simulation time until the number of subsystems is equal to the number of processors. This does not happen because of parallelization overhead.

Fig. 6 demonstrates the advantage of using a parallel algorithm, as computation times are largely reduced. In this example, a single iteration of the method was simulated gradually increasing the number of subsystems present, while their size was set to $s = 5$. Potentially, by increasing the number of cores, one would be able to extend the length of the almost horizontal traits, further reducing the computation times for large systems. Note that the choice of P here is not important: increasing or decreasing it would not affect the relative position between the lines.

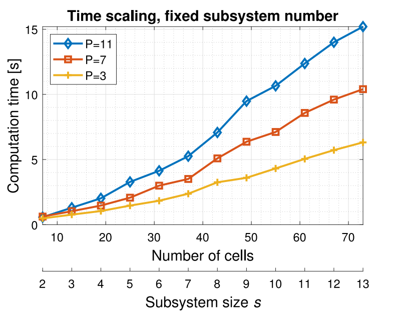
It is therefore clear that the most benefits are obtained when the number of subsystems is equal to the number of cores. Fixing then $r = 6$ and assuming N as known, we can then calculate the subsystem size s:

$$s = \frac{N - q}{r} + q, \tag{42}$$

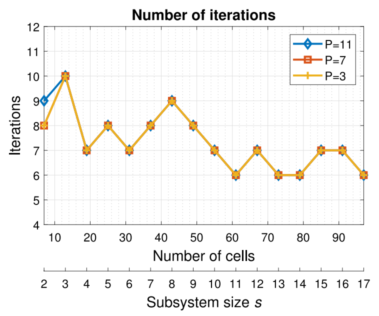
with q chosen in such a way that $s \in \mathbb{N}$. With these assumptions, Fig. 7(a) shows the scaling of the single WR iteration computation time for different values of P. Note that in this figure, the x-axis still has the number of cells N, but differently from the previous one, it has the subsystem size s. Comparing Fig. 7(a) with Fig. 6, it is possible to note that scaling has become almost linear. To conclude and summarize, the optimal subsystem size s can be chosen in such a way that the number of subsystems r is equal to the number of available processors. For what concerns the choice of the overlap q, its optimal value is always 1, as it cannot be reduced to 0, otherwise we would lose convergence; on the other hand, increasing its value would not provide any benefit to the convergence speed, only increasing the computational burden.

Remark 2. Memory usage can be a concern in some situations, as we have to ensure that it is not greedy with respect to a centralized method. In our case, it depends exclusively from the value of the overlap q. In fact, the difference between our method and a centralized one in terms of memory usage is only in the number of cells that get solved multiple times in the overlap. Previously we discussed the choice of q in terms of convergence and noted that the optimal choice was having $q = 1$ and could not be reduced further. This choice is optimal also in terms of memory size, as it reduces the memory needed to the essential minimum.

Now that we have a reference value for the time cost of each iteration, we move to the analysis of the simulation results for the WR method. In particular we are interested in the number of iterations of the AA algorithm needed to reach convergence with the desired tolerance. The first observation we can make looking at Fig. 7(b) is that the number of iterations does not grow with system size, instead it stabilizes around 6–7 iterations. The second one is that the number of iterations does not depend on the complexity of the system; this is a very important point, because even though the cost of iterations increases with system size, their number does not seem to be affected. This implies that the method is more efficient as the complexity of the system grows. Finally, in Fig. 8 a direct comparison between the centralized resolution and the WR method improved with the AA is presented, for different values of P. What we can see here is that for small systems, the centralized approach works better in terms of computation times. This is due to the need of iterating the resolution: if the system is small enough there is no advantage in making it smaller. However when the number of cells increases, the proposed method behaves better, leading to reduced computation times. Moreover, it is possible to note that the crossover point, where the WR method becomes better than the centralized approach, shifts to the left when the complexity of the system increases. This proves the effectiveness of the proposed method. Considering the practical example presented before of a Tesla Model S level with 74 cells connected in parallel [75], we have that with the proposed method, simulation times would be almost a third of a centralized method in the case with $P = 11$. Instead, having $P = 3$ allows us to halve the simulation time. This was done using 6 cores. Note that there is still room for improvement: if more cores are used for the simulation, this would allow to further reduce the total time. Moreover, we can see that as the model gets more complicated (in our case we used P to increase the number of equations in the system as a proof of concept), our method becomes more efficient. This means that using more complex models (such as P2D, for example), or adding more dynamics, such as temperature, the advantage would be more evident.



(a) Scaling of computation times for the single iteration having fixed $r = 6$ and $q = 1$.



(b) Number of iterations as a function of N and s for different values of P.

Fig. 7. Time scaling and number of iterations having fixed the subsystem number r.

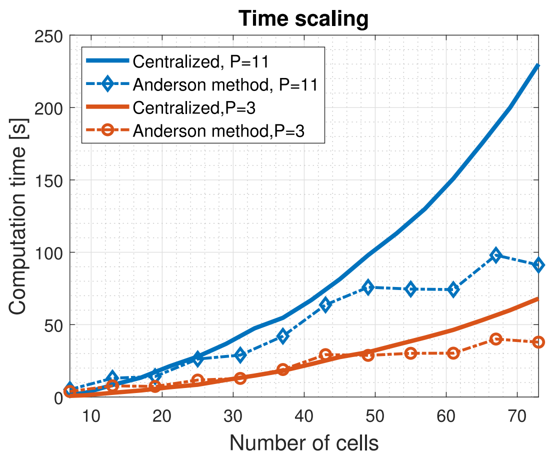


Fig. 8. Comparison between the centralized method computation times (continuous lines) and the AA method computation times (dotted lines).

5. Conclusions

In this work we considered the use of the Waveform Relaxation method to speed up the simulation of the battery pack. Convergence could not be obtained by applying a standard Waveform Relaxation, so we designed an appropriate correction step. In order to further decrease the simulation time, the Anderson Acceleration strategy was also applied. The results we found demonstrate the effectiveness of the proposed method, the precision in obtaining the same result of the centralized approach and the convergence to the true global solution. In addition, they can also serve as a proof of concept for other observations: if the simulations are carried out on a system with more processor cores, the time scaling would be even better, as we would be able to solve more subsystems at the same time. As an example, we were able to simulate a Tesla Model S battery level in a third of the time with respect to a centralized method. Moreover it was shown that this method performs better as the computational complexity grows (in our case we increased it using P), so we expect to obtain a better performance using a more complex model, e.g. the one proposed in [76]. Future developments might also include thermal coupling between cells.

CRediT authorship contribution statement

Giacomo Saccani: Methodology, Software, Validation, Investigation, Writing – original draft, Writing – review & editing, Visualization. **Gabriele Ciaramella:** Conceptualization, Methodology, Validation, Investigation, Writing – review & editing, Supervision. **Davide M. Raimondo:** Conceptualization, Methodology, Formal analysis, Investigation, Writing – review & editing, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work has been (partially) supported by the Italian Ministry for Research in the framework of the 2017 Program for Research Projects of National Interest (PRIN), Grant no. 2017YKXYXJ.

G. Ciaramella is a member of INdAM GNCS.

References

[1] B. Diouf, R. Pode, Potential of lithium-ion batteries in renewable energy, Renew. Energy 76 (2015) 375–380.

[2] R.C. Balch, A. Burke, A.A. Frank, The affect of battery pack technology and size choices on hybrid electric vehicle performance and fuel economy, in: Sixteenth Annual Battery Conference on Applications and Advances, Proceedings of the Conference (Cat. No.01TH8533), 2001, pp. 31–36.

[3] F.E. Hust, G. Ascheid, D.U. Sauer, Physico-Chemically Motivated Parameterization and Modelling of Real-Time Capable Lithium-Ion Battery Models: a Case Study on the Tesla Model S Battery, Tech. rep., Lehrstuhl für Elektrochemische Energiewandlung und Speichersystemtechnik, 2019.

[4] X. Hu, S. Li, H. Peng, A comparative study of equivalent circuit models for li-ion batteries, J. Power Sources 198 (2012) 359–367.

[5] P.M. Gomadam, J.W. Weidner, R.A. Dougal, R.E. White, Mathematical modeling of lithium-ion and nickel battery systems, J. Power Sources 110 (2) (2002) 267–284.

[6] S. Santhanagopalan, Q. Guo, P. Ramadass, R.E. White, Review of models for predicting the cycling performance of lithium ion batteries, J. Power Sources 156 (2) (2006) 620–628.

[7] Z. Khalik, M. Donkers, H. Bergveld, Model simplifications and their impact on computational complexity for an electrochemistry-based battery modeling toolbox, J. Power Sources 488 (2021) 229427.

[8] A. Jokar, B. Rajabloo, M. Désilets, M. Lacroix, Review of simplified pseudo-two-dimensional models of lithium-ion batteries, J. Power Sources 327 (2016) 44–55.

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

[10] S.K. Rahimian, S. Rayman, R.E. White, Extension of physics-based single particle model for higher charge–discharge rates, J. Power Sources 224 (2013) 180–194.

[11] C. Wang, V. Srinivasan, Computational battery dynamics (CBD)—electrochemical/thermal coupled modeling and multi-scale modeling, J. Power Sources 110 (2) (2002) 364–376.

[12] S. Shi, J. Gao, Y. Liu, Y. Zhao, Q. Wu, W. Ju, C. Ouyang, R. Xiao, Multi-scale computation methods: Their applications in lithium-ion battery research and development, Chin. Phys. B 25 (1) (2015) 018212.

[13] G. Richardson, G. Denuault, C. Please, Multiscale modelling and analysis of lithium-ion battery charge and discharge, J. Eng. Math. 72 (1) (2012) 41–72.

[14] I.D. Campbell, K. Gopalakrishnan, M. Marinescu, M. Torchio, G.J. Offer, D. Raimondo, Optimising lithium-ion cell design for plug-in hybrid and battery electric vehicles, J. Energy Storage 22 (2019) 228–238.

[15] S. Wang, C. Fernandez, Y. Fan, J. Feng, C. Yu, K. Huang, W. Xie, A novel safety assurance method based on the compound equivalent modeling and iterate reduce particle-adaptive Kalman filtering for the unmanned aerial vehicle lithium ion batteries, Energy Sci. Eng. 8 (5) (2020) 1484–1500.

[16] D.H. Doughty, E.P. Roth, A general discussion of Li ion battery safety, Electrochem. Soc. Interface 21 (2) (2012) 37.

[17] H. Rahimi-Eichi, U. Ojha, F. Baronti, M. Chow, Battery management system: An overview of its application in the smart grid and electric vehicles, IEEE Ind. Electron. Mag. 7 (2) (2013) 4–16.

[18] P. Kemper, S.E. Li, D. Kum, Simplification of pseudo two dimensional battery model using dynamic profile of lithium concentration, J. Power Sources 286 (2015) 510–525.

[19] E. Prada, D.D. Domenico, Y. Creff, J. Bernard, V. Sauvant-Moynot, F. Huet, Simplified electrochemical and thermal model of LiFePO4-graphite Li-Ion batteries for fast charge applications, J. Electrochem. Soc. 159 (9) (2012) A1508–A1519.

[20] J.C. Forman, S. Bashash, J.L. Stein, H.K. Fathy, Reduction of an electrochemistry-based li-ion battery model via quasi-linearization and pade approximation, J. Electrochem. Soc. 158 (2) (2010) A93.

[21] M. Ohlberger, S. Rave, F. Schindler, Model reduction for multiscale lithium-ion battery simulation, in: Numerical Mathematics and Advanced Applications ENUMATH 2015, Springer, 2016, pp. 317–331.

[22] L. Xia, E. Najafi, Z. Li, H. Bergveld, M. Donkers, A computationally efficient implementation of a full and reduced-order electrochemistry-based model for li-ion batteries, Appl. Energy 208 (2017) 1285–1296.

[23] K. Gopalakrishnan, G.J. Offer, A composite single particle lithium-ion battery model through system identification, IEEE Trans. Control Syst. Technol. (2021) 1–13.

[24] P.W. 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, J. Electrochem. Soc. 158 (12) (2011) A1461.

[25] D. Howey, S. Duncan, A. Bizeray, Advanced battery management systems using fast electrochemical modelling, in: IET Conference Publications, Vol. 2013, 2013.

[26] L. Cai, R.E. White, Lithium ion cell modeling using orthogonal collocation on finite elements, J. Power Sources 217 (2012) 248–255.

[27] S. Han, Y. Tang, S. Khaleghi Rahimian, A numerically efficient method of solving the full-order pseudo-2-dimensional (P2D) Li-ion cell model, J. Power Sources 490 (2021) 229571.

[28] H. Chun, J. Kim, J. Yu, S. Han, Real-time parameter estimation of an electrochemical lithium-ion battery model using a long short-term memory network, IEEE Access 8 (2020) 81789–81799.

[29] M. Kim, H. Chun, J. Kim, K. Kim, J. Yu, T. Kim, S. Han, Data-efficient parameter identification of electrochemical lithium-ion battery model using deep Bayesian harmony search, Appl. Energy 254 (2019) 113644.

[30] C. Jiang, S. Wang, B. Wu, C. Fernandez, X. Xiong, J. Coffie-Ken, A state-of-charge estimation method of the power lithium-ion battery in complex conditions based on adaptive square root extended Kalman filter, Energy 219 (2021) 119603.

[31] S.-L. Wang, C. Fernandez, W. Cao, C.-Y. Zou, C.-M. Yu, X.-X. Li, An adaptive working state iterative calculation method of the power battery by using the improved Kalman filtering algorithm and considering the relaxation effect, J. Power Sources 428 (2019) 67–75.

[32] S.-L. Wang, C.-M. Yu, C. Fernandez, M.-J. Chen, G.-L. Li, X.-H. Liu, Adaptive state-of-charge estimation method for an aeronautical lithium-ion battery pack based on a reduced particle-unscented kalman filter, J. Power Electron. 18 (4) (2018) 1127–1139.

[33] W. Cao, S.-L. Wang, C. Fernandez, C.-Y. Zou, C.-M. Yu, X.-X. Li, A novel adaptive state of charge estimation method of full life cycling lithium-ion batteries based on the multiple parameter optimization, Energy Sci. Eng. 7 (5) (2019) 1544–1556.

[34] E. Lelarasmee, The Waveform Relaxation Method for Time Domain Analysis of Large Scale Integrated Circuits: Theory and Applications, Electronics Research Laboratory, College of Engineering, University of Berkeley, 1982.

[35] J.A. White, A. Sangiovanni-Vincentelli, F. Odeh, A. Ruehli, Waveform Relaxation: Theory and Practice, Electronics Research Laboratory, College of Engineering, UCB, 1985.

[36] E. Lelarasmee, A.E. Ruehli, A.L. Sangiovanni-Vincentelli, The waveform relaxation method for time-domain analysis of large scale integrated circuits, IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. 1 (3) (1982) 131–145.

[37] J. Sand, K. Burrage, A Jacobi waveform relaxation method for ODEs, SIAM J. Sci. Comput. 20 (2) (1998) 534–552.

[38] M.L. Crow, M. Ilić, The waveform relaxation method for systems of differential/algebraic equations, Math. Comput. Modelling 19 (12) (1994) 67–84.

[39] M. Maciejewski, I.C. Garcia, S. Schöps, B. Auchmann, L. Bortot, M. Prioli, A.P. Verweij, Application of the waveform relaxation technique to the co-simulation of power converter controller and electrical circuit models, in: 2017 22nd International Conference on Methods and Models in Automation and Robotics, MMAR, 2017, pp. 837–842.

[40] M.J. Gander, A waveform relaxation algorithm with overlapping splitting for reaction diffusion equations, Numer. Linear Algebra Appl. 6 (2) (1999) 125–145.

[41] M.J. Gander, A.E. Ruehli, Optimized waveform relaxation solution of electromagnetic and circuit problems, in: 19th Topical Meeting on Electrical Performance of Electronic Packaging and Systems, 2010, pp. 65–68.

[42] F.-Y. Chang, Waveform relaxation analysis of RLCG transmission lines, IEEE Trans. Circuits Syst. 37 (11) (1990) 1394–1415.

[43] F.-Y. Chang, The generalized method of characteristics for waveform relaxation analysis of lossy coupled transmission lines, IEEE Trans. Microw. Theory Tech. 37 (12) (1989) 2028–2038.

[44] S. Moayedi, F. Cingöz, A. Davoudi, Accelerated simulation of high-fidelity models of supercapacitors using waveform relaxation techniques, IEEE Trans. Power Electron. 28 (11) (2013) 4903–4909.

[45] D.G. Anderson, Iterative procedures for nonlinear integral equations, J. ACM 12 (4) (1965) 547–560.

[46] M. Torchio, L. Magni, R.B. Gopaluni, R.D. Braatz, D.M. Raimondo, Lionsimba: a matlab framework based on a finite volume model suitable for li-ion battery design, simulation, and control, J. Electrochem. Soc. 163 (7) (2016) A1192.

[47] G.B. Less, J.H. Seo, S. Han, A.M. Sastry, J. Zausch, A. Latz, S. Schmidt, C. Wieser, D. Kehrwald, S. Fell, Micro-scale modeling of Li-ion batteries: parameterization and validation, J. Electrochem. Soc. 159 (6) (2012) A697.

[48] A. Latz, J. Zausch, Thermodynamic consistent transport theory of Li-ion batteries, J. Power Sources 196 (6) (2011) 3296–3302.

[49] J. Feinauer, S. Hein, S. Rave, S. Schmidt, D. Westhoff, J. Zausch, O. Iliev, A. Latz, M. Ohlberger, V. Schmidt, Multibat: Unified workflow for fast electrochemical 3D simulations of lithium-ion cells combining virtual stochastic microstructures, electrochemical degradation models and model order reduction, J. Comput. Sci. 31 (2017).

[50] V. Sulzer, S.G. Marquis, R. Timms, M. Robinson, S.J. Chapman, Python battery mathematical modelling (PyBaMM), 2020, ECSarXiv.

[51] I. Korotkin, S. Sahu, S. O'Kane, G. Richardson, J.M. Foster, Dandeliion v1: An extremely fast solver for the newman model of lithium-ion battery (dis)charge, 2021, arXiv:2102.06534.

[52] S.J. Moura, F.B. Argomedo, R. Klein, A. Mirtabatabaei, M. Krstic, Battery state estimation for a single particle model with electrolyte dynamics, IEEE Trans. Control Syst. Technol. 25 (2) (2017) 453–468.

[53] V.R. Subramanian, V.D. Diwakar, D. Tapriyal, Efficient macro-micro scale coupled modeling of batteries, J. Electrochem. Soc. 152 (10) (2005) A2002.

[54] A. Pozzi, M. Zambelli, A. Ferrara, D.M. Raimondo, Balancing-aware charging strategy for series-connected lithium-ion cells: A nonlinear model predictive control approach, IEEE Trans. Control Syst. Technol. 28 (5) (2020) 1862–1877.

[55] A. Pozzi, M. Torchio, R.D. Braatz, D.M. Raimondo, Optimal charging of an electric vehicle battery pack: A real-time sensitivity-based model predictive control approach, J. Power Sources 461 (2020) 228133.

[56] P. Ramadass, B. Haran, R. White, B.N. Popov, Mathematical modeling of the capacity fade of li-ion cells, J. Power Sources 123 (2) (2003) 230–240.

[57] C. Zhu, X. Li, L. Song, L. Xiang, Development of a theoretically based thermal model for lithium ion battery pack, J. Power Sources 223 (2013) 155–164.

[58] K. Murashko, J. Pyrhönen, L. Laurila, Three-dimensional thermal model of a lithium ion battery for hybrid mobile working machines: Determination of the model parameters in a pouch cell, IEEE Trans. Energy Convers. 28 (2) (2013) 335–343.

[59] H. Sun, R. Dixon, Development of cooling strategy for an air cooled lithium-ion battery pack, J. Power Sources 272 (2014) 404–414.

[60] M. Ecker, T.K.D. Tran, P. Dechent, S. Käbitz, A. Warnecke, D.U. Sauer, Parameterization of a physico-chemical model of a lithium-ion battery, J. Electrochem. Soc. 162 (9) (2015) A1836–A1848.

[61] M. Ecker, S. Käbitz, I. Laresgoiti, D.U. Sauer, Parameterization of a physico-chemical model of a lithium-ion battery, J. Electrochem. Soc. 162 (9) (2015) A1849–A1857.

[62] K. Rumpf, A. Rheinfeld, M. Schindler, J. Keil, T. Schua, A. Jossen, Influence of cell-to-cell variations on the inhomogeneity of lithium-ion battery modules, J. Electrochem. Soc. 165 (11) (2018) A2587.

[63] T. Baumhöfer, M. Brühl, S. Rothgang, D.U. Sauer, Production caused variation in capacity aging trend and correlation to initial cell performance, J. Power Sources 247 (2014) 332–338.

[64] M.J. Gander, L. Halpern, F. Nataf, Optimal convergence for overlapping and non-overlapping Schwarz waveform relaxation, in: 11th International Conference on Domain Decomposition Methods, 1999, pp. 27–36.

[65] M.J. Gander, A.E. Ruehli, Optimized waveform relaxation methods for RC type circuits, IEEE Trans. Circuits Syst. I. Regul. Pap. 51 (4) (2004) 755–768.

[66] M.J. Gander, Schwarz methods over the course of time, Electron. Trans. Numer. Anal. 31 (2008) 228–255.

[67] G. Ciaramella, M.J. Gander, L. Halpern, J. Salomon, Methods of reflections: relations with Schwarz methods and classical stationary iterations, scalability and preconditioning, SMAI J. Comput. Math. 5 (2019) 161–193.

[68] G. Ciaramella, M.J. Gander, Analysis of the parallel Schwarz method for growing chains of fixed-sized subdomains: Part I, SIAM J. Numer. Anal. 55 (3) (2017) 1330–1356.

[69] F. Chaouqui, G. Ciaramella, M.J. Gander, T. Vanzan, On the scalability of classical one-level domain-decomposition methods, Vietnam J. Math. 46 (4) (2018) 1053–1088.

[70] H.F. Walker, Anderson Acceleration: Algorithms and Implementations, WPI Math. Sciences Dept. Report MS-6-15-50, 2011.

[71] G. Giorgi, A. Guerraggio, First order generalized optimality conditions for programming problems with a set constraint, in: S. Komlósi, T. Rapcsák, S. Schaible (Eds.), Generalized Convexity, Springer Berlin Heidelberg, Berlin, Heidelberg, 1994, pp. 171–185.

[72] W. Gander, M.J. Gander, F. Kwok, Scientific Computing-an Introduction using Maple and MATLAB, Vol. 11, Springer Science &amp; Business, 2014.

[73] H. Walker, P. Ni, Anderson acceleration for fixed-point iterations, SIAM J. Numer. Anal. 49 (2011) 1715–1735.

[74] G. Ciaramella, G. Fabrini, Multilevel Techniques for the Solution of HJB Minimum-Time Control Problems, to appear in Journal of Systems Science and Complexity, 2021-2022.

[75] T. Bruen, J. Marco, Modelling and experimental evaluation of parallel connected lithium ion cells for an electric vehicle battery system, J. Power Sources 310 (2016) 91–101.

[76] T. Weaver, A. Allam, S. Onori, A novel lithium-ion battery pack modeling framework - series-connected case study, in: 2020 American Control Conference, ACC, 2020, pp. 365–372.