# COBRAPRO: An Open-Source Software for the Doyle-Fuller-Newman Model with Co-Simulation Parameter Optimization Framework

To cite this article: Sara Ha and Simona Onori 2024 *J. Electrochem. Soc.* **171** 090522

*Journal of The Electrochemical Society*, 2024 **171** 090522
1945-7111/2024/171(9)/090522/28/$40.00 © 2024 The Electrochemical Society ("ECS"). Published on behalf of ECS by IOP Publishing Limited.


# COBRAPRO: An Open-Source Software for the Doyle-Fuller-Newman Model with Co-Simulation Parameter Optimization Framework

**Sara Ha**[1] **and Simona Onori**[2,z]

[1]*Mechanical Engineering, Stanford University, Stanford 94305, California, United States of America*
[2]*Energy Science and Engineering, Stanford University, Stanford 94305, Califronia, United States of America*

This paper presents COBRAPRO, a new open-source Doyle-Fuller-Newman (DFN) model software package with an integrated closed-loop parameter optimization routine. A key challenge in DFN model parameterization is that parameters measured from cell tear-down experiments cannot be directly used in simulations, and parameter identification is required to accurately reflect real-world battery dynamics However, existing open-source DFN codes lack the capability to perform parameter identification and operate in open-loop mode. COBRAPRO addresses this gap by implementing a systematic parameterization pipeline to accurately determine parameters using battery current and voltage data. Concepts from structural and practical identifiability are utilized to determine parameters that can be fixed to their experimental values and parameters that are suitable for optimization. In the parameter identification process, particle swarm optimization is used to minimize the error between experimental data and simulation results. Additionally, COBRAPRO incorporates a robust method to determine consistent initial conditions and utilizes a fast numerical solver for improved performance. We demonstrate COBRAPRO's parameter identification framework on reference performance test data obtained from LG INR21700-M50T cells. The parameterized model is validated against driving cycle data, showing good agreement between the experimental and simulation results.
 [DOI: 10.1149/1945-7111/ad7292]

Manuscript submitted March 18, 2024; revised manuscript received August 3, 2024. Published September 25, 2024.

## List of Symbols

| | |
|---|---|
| $a_i$ | Specific interfacial area [$m^2/m^3$], $i \in (p, n)$ |
| $brugg_i$ | Bruggeman coefficient [−], $i \in (p, s, n)$ |
| $c_{e,i}$ | Electrolyte concentration [$mol/m^3$], $i \in (p, s, n)$ |
| $c_{s,i}$ | Solid particle concentration [$mol/m^3$], $i \in (p, n)$ |
| $c_{s,i}^{avg}$ | Volume averaged solid particle concentration [$mol/m^3$], $i \in (p, n)$ |
| $c_{s,i}^{bulk}$ | Bulk solid particle concentration [$mol/m^3$], $i \in (p, n)$ |
| $c_{s,i}^{surf}$ | Particle surface concentration [$mol/m^3$], $i \in (p, n)$ |
| $D_i$ | Electrolyte diffusion coefficient [$m^2/s$], $i \in (p, s, n)$ |
| $D_i^{eff}$ | Effective electrolyte diffusion coefficient [$m^2/s$], $i \in (p, s, n)$ |
| $D_{s,i}$ | Solid diffusion coefficient [$m^2/s$], $i \in (p, n)$ |
| $F$ | Faraday constant [C/mol] |
| $I_{app}$ | Applied current [A] |
| $j_i$ | Pore wall flux [$mol/(m^2 \cdot s)$], $i \in (p, n)$ |
| $j_{0,i}$ | Exchange current density per unit charge [$mol/(m^2 \cdot s)$], $i \in (p, n)$ |
| $J$ | Total cost function [-] |
| $J^{SOC,i}$ | Electrode cost function [-], $i \in (p, n)$ |
| $J^V$ | Voltage cost function [-] |
| $k_i$ | Reaction rate constant [$m^{2.5}/(mol^{0.5} \cdot s)$], $i \in (p, n)$ |
| $l_i$ | Length of domain [m], $i \in (p, s, n)$ |
| $N_i$ | X-direction discretization points [-], $i \in (p, s, n)$ |
| $N_{ri}$ | Radial discretization points [-], $i \in (p, n)$ |
| $q$ | Switch function discreteness factor [-] |
| $Q_{dis}$ | Discharged capacity [Ah] |
| $R$ | Ideal gas constant [J/(K·mol)] |
| $R_i$ | Particle radius [m], $i \in (p, n)$ |
| $SOC_i$ | State of charge in electrode [−], $i \in (p, n)$ |
| $t_{init}$ | Initialization time [s] |
| $T$ | Cell temperature [K] |
| $t_+$ | Transference number [−] |
| $U_i$ | Open-circuit potential [V], $i \in (p, n)$ |
| $V_{cell}$ | Cell voltage [V] |
| $V_{lower}^{cutoff}$ | Lower voltage cut-off limit [V] |
| $V_{upper}^{cutoff}$ | Upper voltage cut-off limit [V] |
| $\alpha$ | Perturbation factor [-] |
| $\beta$ | Correlation index threshold [-] |
| $\varepsilon_i$ | Porosity [−], $i \in (p, s, n)$ |
| $\varepsilon_i^{solid}$ | Solid volume fraction [−], $i \in (p, n)$ |
| $\varepsilon_i^{filler}$ | Binder volume fraction [−], $i \in (p, n)$ |
| $\kappa_i$ | Electrolyte conductivity [S/m], $i \in (p, s, n)$ |
| $\kappa_i^{eff}$ | Effective electrolyte conductivity [S/m], $i \in (p, s, n)$ |
| $\sigma_i$ | Solid conductivity [S/m], $i \in (p, n)$ |
| $\sigma_i^{eff}$ | Effective solid conductivity [S/m], $i \in (p, n)$ |
| $\phi_{e,i}$ | Electrolyte potential [V], $i \in (p, s, n)$ |
| $\phi_{s,i}$ | Solid phase potential [V], $i \in (p, n)$ |
| $\theta_i^{avg}$ | Normalized $c_{s,i}^{avg}$ with respect to $c_{s,i}^{max}$ [−], $i \in (p, n)$ |
| $\theta_i^{bulk}$ | Normalized $c_{s,i}^{bulk}$ with respect to $c_{s,i}^{max}$ [−], $i \in (p, n)$ |
| $\theta_i^{surf}$ | Normalized $c_{s,i}^{surf}$ with respect to $c_{s,i}^{max}$ [−], $i \in (p, n)$ |
| $\theta_i^{0\%}$ | Stoichiometric coefficient at 0% $SOC_i$ [−], $i \in (p, n)$ |
| $\theta_i^{100\%}$ | Stoichiometric coefficient at 100% $SOC_i$ [−], $i \in (p, n)$ |

### Acronyms

| | |
|---|---|
| AD | Automatic differentiation |
| AE | Algebraic equation |
| BMS | Battery management system |
| CV | Control volume |
| DAE | Differential algebraic equation |
| DFN | Doyle-Fuller-Newman |
| ECM | Electric circuit model |
| EIS | Electrochemical impedance spectroscopy |
| ESPM | Enhanced single particle model |
| eVTOL | Electric vertical take-off and landing |
| FDM | Finite difference method |
| FEM | Finite element method |
| FVM | Finite volume method |
| GA | Genetic algorithm |
| GITT | Galvanostatic intermittent titration technique |
| HPPC | Hybrid pulse power characterization |
| LM | Levenberg-Marquardt |
| LIB | Lithium-ion battery |
| LSA | Local sensitivity analysis |
| ML | Machine learning |
| OCP | Open-circuit potential |

[z]E-mail: sonori@stanford.edu

| | |
|---|---|
| OCV | Open-circuit voltage |
| ODE | Ordinary differential equation |
| PDE | Partial differential equation |
| PSO | Particle swarm optimization |
| RMSE | Root mean square error |
| RPT | Reference performance test |
| SOC | State of charge |
| SOH | State of health |
| SPM | Single particle model |
| UDDS | Urban dynamometer driving schedule |

Lithium-ion batteries (LIBs) have emerged as the predominant choice for energy storage systems due to their high energy and power densities, and long cycle life.[1] LIBs play a crucial role in enabling the electrification of transportation and grid energy storage. The U.S. Department of Energy forecasts a substantial market surge for LIBs, projecting a 5-10 times increase between 2020 and 2030, reaching up to 2500 GWh by 2030.[2]

The widespread adoption of LIBs has been closely linked with the development and utilization of battery models. These models can vary in complexity and serve diverse purposes, from optimizing battery design to enhance energy density and fast charging/power capability,[3–5] improving manufacturing processes,[6] and advancing new material development.[7] Notably, in the domain of battery design, models have accelerated design iterations, enabling the swift exploration of a wide range of design variables and scenarios, leading to cost-effective and less labor-intensive product development.

Another important application of battery models is battery monitoring systems (BMS). The widespread integration of LIBs in applications such as electric vehicles, electric vertical take-off and landing (eVTOL) aircraft, and grid energy storage necessitates robust engineering infrastructure to ensure the safety and reliable operation of batteries. The BMS, comprising a suite of hardware and software engineering components, utilizes battery models to estimate unmeasurable internal states such as state of charge (SOC) and state of health (SOH). Control strategies are deployed to regulate battery operation, encompassing cell balancing, thermal management, and charge and discharge control. These strategies aim to reduce cell aging within the battery pack[8] and prevent overcharging or discharging conditions.[9]

Three main types of battery models exist: equivalent circuit models (ECM), data-driven models, and physics-based models. ECMs use circuit elements such as ohmic resistors and resistor-capacitor pairs to model the battery dynamics. ECMs have been widely adopted in conventional BMS due to their low computational burden and relatively simple implementation.[10] However, accurate ECMs require extensive calibration and offer little to no insight into the electrochemical states of the battery.[11] Data-driven, or machine learning (ML), models have gained attention for their ability to model batteries with high accuracy and low computational burden. MLs are used for tasks such as health estimation,[12] predicting remaining useful life,[13] assisting in model identification,[14] and reconstructing images to analyze the microscale architecture of the cell electrode.[15] However, ML models function as black boxes, with their accuracy heavily depending on the quality, quantity, and diversity of the training data. As a result, ML models must be trained on datasets that closely represent the operational conditions of the targeted application.[16] Both ML models and ECMs are empirical models that can struggle with extrapolation, meaning they may not perform well outside the range for which they were trained and validated. Recently, hybrid models[17–19] have been proposed, combining physics-based and ML methods to improve the predictive capability and physical interpretability of ML models.

Physics-based models use first principles to simulate the battery's behavior, providing insights into the internal electrochemical states within the cell. The most widely accepted and well-known physics-based model is the Doyle-Fuller-Newman (DFN) or pseudo-two-dimensional (P2D) model,[20] proposed in 1993. The DFN model is based on porous electrode theory,[21] concentrated solution theory,[22] and Butler-Volmer kinetics, and is parameterized by physical factors such as geometric, stoichiometric, concentration transport, and kinetic parameters. The DFN model has been employed in model-based battery design,[3–5] as well as in advanced BMS to monitor and control internal electrochemical states for SOC/SOH estimation[23] and develop optimal charging strategies.[24]

The DFN model, also referred to as the P2D model, describes the macroscale dimension along the through-thickness direction of the cell, where electrolyte transport and charge conservation in both the solid and electrolyte occur. Meanwhile, the microscale behavior is captured by Fick's law of diffusion in the pseudo-radial dimension within the spherical particles. Due to the coupling of the micro and macroscale dimensions, the DFN model is computationally complex. The reduced-order counterparts of the DFN model, single particle model (SPM)[25] and enhanced single particle model (ESPM),[26,27] both assume the particles can be represented by one spherical particle per electrode, with SPM ignoring electrolyte dynamics and ESPM including them. The SPM has been employed in control applications due to its lower computational burden compared to the DFN model.[28]

The DFN model consists of a system of coupled nonlinear partial differential equations (PDEs) that are numerically discretized to form a differential algebraic equation (DAE) system. This DAE system includes ordinary differential equations (ODEs) and algebraic equations (AEs), with the AEs acting as constraints. The DFN model can be discretized using various spatial discretization methods, such as finite difference method (FDM), finite volume method (FVM), finite element method (FEM), and spectral methods.[29–31]

Solving the DAE system requires consistent initial conditions to satisfy the AE constraints. In the DFN model, initial guesses for the algebraic variables are made at the battery equilibrium state when the current is zero. For instance, the open-circuit potentials (OCPs) are used as the initial guesses for the solid phase potentials and the initial SOC is used to calculate the guesses for the solid particle concentrations. However, when current is applied at $t = 0$, the battery deviates from these equilibrium states, and the initial guesses no longer satisfy the AEs. Therefore, consistent initial conditions are not known *a priori* and need to be determined to solve the DFN model.[32,33]

To overcome the issue of inconsistent initial conditions, several methods have been proposed in the context of DFN modeling. Traditional methods of solving the algebraic constraints require the use of nonlinear algebraic solvers, which can suffer from convergence issues.[32,34] The perturbation approach proposed in Ref. 32 transforms AEs into ODEs using a small perturbation factor. This approach leverages the robustness and stability of ODE solvers to iteratively refine the initial conditions, proving to be more effective in achieving consistent initial conditions than nonlinear AE solvers. The single-step approach[35] improves upon the perturbation approach by applying a switch function to smoothly transition between the perturbation and simulation phases, achieving both consistent initial condition determination and model simulation in a seamless "single step" for improved computational efficiency.

DFN model parameterization is another major challenge in implementing the model, as the accuracy and reliability of physics-based models depend on the correct parameter values. DFN parameterization methods can be categorized into two approaches. The first involves cell tear-down to experimentally measure the parameters.[36–38] The second method, called parameter identification[39] or inverse problem,[40] is a noninvasive method that uses the input-output relationship of the DFN model to identify parameters using battery data. Optimization techniques are used to minimize the objective function, defined as the error between the model voltage output and the experimental voltage data, while multi-objective functions have been proposed to include surface temperature[41] or SOC in the electrodes.[42]

In cell tear-down experiments, various electrochemical and imaging techniques can be used to measure the cell parameters. For instance,

Hg-porosimetry or SEM imaging methods can be used to determine the porosity, particle radius, and tortuosity in electrodes and separator, while SEM imaging is commonly used to measure the electrode and separator thickness and cross-sectional area.[36–38] The solid diffusion coefficient in each electrode can be characterized through half-cell galvanostatic intermittent titration technique (GITT)[36–38] or electrochemical impedance spectroscopy (EIS).[36] Electrochemical properties of the electrolyte such as diffusion, conductivity, and transference number can be characterized using galvanostatic polarization experiments and EIS.[43,44] The kinetic parameters such as the reaction rate constant can be extracted from half-cell EIS measurements.[36,38] The OCP of each electrode can also be measured using half-cell GITT or low C-rate capacity test experiments.[38,45] As not all researchers have access to the necessary equipment and resources to conduct such experiments, LiionDB, an online database introduced in Ref. 46, serves as a valuable resource by offering a searchable repository of experimentally measured parameters from various literature sources.

A common misconception in DFN modeling is that using experimentally measured parameter values from cell tear-down will lead to accurate model results. The DFN model, like any mathematical model, is a simplified representation of an actual system and cannot capture all the intricate physical and chemical processes occurring at different length and time scales within a battery. Although several extensions of the DFN model have been proposed, such as the thermal DFN model,[47] mechanical DFN model,[48,49] and distributed particle-size DFN model,[50] fully accounting for all the physical phenomena occurring in batteries and the coupling effects between different physical processes remains challenging. Furthermore, the DFN model assumes perfectly spherical particles and ignores the heterogeneity in the electrodes, assuming homogeneous tortuosity and porosity distribution. These simplifications mean that even accurately measured experimental parameters may not directly translate to precise model predictions. This was demonstrated in Refs. 38 and 51 where the experimentally obtained DFN parameters had to be tuned for the simulation results to match the experimental voltage curves. Therefore, careful parameterization methods must be developed to identify and refine the parameters in the DFN model, ensuring they are optimized and validated within the modeling framework to improve the accuracy and reliability of the simulation results.

Given the overparameterized nature of the DFN model,[52] parameter identifiability should be investigated to determine an identifiable subset of parameters prior to parameter optimization. Performing parameter identification on a subset of identifiable parameters, rather than the entire parameter set, can improve optimization convergence times and reduce overfitting issues.[53]

Identifiability analysis can be categorized into structural and practical identifiability.[53–57] Structural identifiability refers to the inherent mathematical properties of a model and whether the model allows for unique parameter estimation, regardless of the data or estimation method used.[58] Practical identifiability, on the other hand, assesses whether parameters can be reliably estimated from actual experimental data, considering real-world constraints such as measurement errors, finite data length, and data informativeness.[56] According to Ref. 54, performing both structural and practical identifiability can improve the reliability of parameter estimation.

The DFN model, a complex system with four coupled nonlinear PDEs, has been shown to be structurally unidentifiable.[59,60] In Ref. 61, the structural identifiability of a decoupled, linearized DFN model around a specific SOC point was assessed using the system's transfer function, but the simplifications of linearization limit its applicability. In Refs. 59 and 60, parameter aggregation was used to improve identifiability by combining two or more DFN parameters with indistinguishable effects on the model's output. This process involves normalizing the DFN model equations with respect to the electrode and separator thicknesses and radius of the particles to determine lumped parameters. While parameter lumping does not make the DFN model structurally identifiable, it helps improve identifiability by removing redundant parameters from a structural identifiability standpoint.

Practical identifiability refers to the ability to reliably estimate model parameters from experimental data. Unlike structural identifiability, which focuses on the mathematical properties of the model, practical identifiability considers real-world constraints such as measurement errors, finite data length, and informativeness of the data.[55] Practical identifiability often requires empirical assessments, such as sensitivity analysis, to evaluate the feasibility of parameter estimation given input-output data. Practical identifiability involves not only assessing parameter sensitivity but also determining whether the impact of each parameter on the model output is distinguishable from the impact of other parameters.[54,62] Goshtasbi et al.[63] utilized extended sensitivity analysis, averaging sensitivity across multiple local points, and the collinearity index to compute the similarity between any two parameters. In Refs. 42 and 64, local sensitivity and correlation analysis were conducted to determine a group of highly sensitive and non-correlated parameters. In Ref. 65, the QR factorization proposed in Ref. 66 was used to rank the sensitivity of non-correlated parameters, selecting the top five for identification. Additionally, some researchers have designed current input profiles that maximize the practical identifiability of certain DFN parameters.[67]

In the past decade, several open-source DFN modeling software packages have been released by the battery research community. The first open-source DFN code, DUALFOIL,[68] was published by John Newman in 1998 and written in FORTRAN. Since then, programming languages such as Python, MATLAB, and Julia have gained prominence in various engineering domains. Table I shows recent developments in DFN open-source software, such as LIONSIMBA,[30] DEARLIBS,[69] fastDFN,[70] PyBaMM,[31] MPET,[71] and PETLION.[72] LIONSIMBA was the first well-documented MATLAB DFN open-source code developed in 2016 with an accompanying peer-reviewed article.[30] LIONSIMBA offers various options to implement the DFN model, such as Fick's law and polynomial approximation for the microscale equation, 6th order FDM and spectral methods for the microscale numerical discretization, and the ability to define current, voltage, or power input profiles. PyBaMM,[31] released in 2019 and written in Python, is one of the most widely used open-source codes in the battery research community. With nearly 82 contributors on GitHub as of v24.1, PyBaMM is actively maintained and constantly improved with new features. PyBaMM is a highly modular code, that allows users to choose the model type (SPM, ESPM, DFN, DFN with particle size distribution), discretization method (FVM with linear interpolation or spectral method), and aging mechanism (particle cracking, solid electrolyte interphase growth, lithium plating).

As shown in Table I, COBRAPRO, PyBaMM, LIONSIMBA, PETLION, and MPET utilize the efficient SUNDIALS IDA solver. SUNDIALS is an open-source ODE, DAE, and AE solver package developed by Lawrence Livermore National Laboratory.[73,74] Since the original code is written in C, several publicly available packages provide interfaces to SUNDIALS. For instance, PETLION uses Sundials.jl,[75] a Julia package that interfaces with SUNDIALS IDA. MPET utilizes SciPy,[76] which includes a Python wrapper for SUNDIALS. PyBaMM uses SUNDIALS IDA interfaced from CasADi,[77] which offers SUNDIALS in both Python and MATLAB versions. Additionally, LIONSIMBA and COBRAPRO leverage SUNDIALS IDA through the sundialsTB v2.6.2 package,[73,74] which provides a MATLAB interface to SUNDIALS.

To determine consistent initial conditions, PyBaMM, LIONSIMBA, fastDFN, PETLION, and MPET utilize nonlinear AE solvers to solve the algebraic constraint equation in the DAE. For instance, SUNDIALS IDACalcIC, a nonlinear AE solver, is utilized by LIONSIMBA, fastDFN, and MPET while PyBaMM uses CasADi's rootfinder function, which is also a nonlinear AE solver. The fastDFN code implements its own nonlinear AE solver using Newton's method. Only DEARLIBS and COBRAPRO[a] utilize the

[a]The single-step approach[35] involves solving an implicit ODE system to determine consistent initial conditions. The MATLAB sundialsTB v2.6.2 version of SUNDIALS IDA is used because it can handle implicit ODEs. In contrast, CasADi's MATLAB version of SUNDIALS IDA is limited to solving semi-explicit ODE systems (CasADi v3.6.5).

**Table I. Comparison of the open-source DFN codes.**

| Information | COBRAPRO | DEARLIBS | PyBaMM | LIONSIMBA | fastDFN | PETLION | MPET |
|---|---|---|---|---|---|---|---|
| Language | MATLAB | MATLAB | Python | MATLAB | MATLAB | Julia | Python |
| Parameter Optimization Routine | ✓ | ✓ | | | | | |
| Parameter Identifiability Analysis | ✓ | | | | | | |
| Microscale equation | Fick's Law | Polynomial Approximation | Fick's Law, Polynomial Approximation | Fick's Law, Polynomial Approximation | 3rd order Pade Approximation | Fick's Law | Fick's Law |
| Microscale equation Discretization | FVM, FDM | NaN | FVM, Spectral Method | 6th order FDM, Spectral Method | NaN | FVM | CVM |
| Surface Concentration Approximation | 3rd Order Hermite Approximation | NaN | Linear Interpolation | NaN | NaN | Linear Interpolation | NaN |
| Macroscale equation Discretization | FVM | FDM | FVM, Spectral Method | FVM | FDM | FVM | FVM |
| DAE Consistent Initial Conditions | Single-step, SUNDIALS IDACalcIC | Single-step | CasADi rootfinder | SUNDIALS IDACalcIC | Newton's method | SUNDIALS IDACalcIC | Gradually Increase Current/Voltage from Equilibrium |
| DAE Solver | SUNDIALS IDA | ode15s | SUNDIALS IDA[a] | SUNDIALS IDA | In-house solver[b] | SUNDIALS IDA | SUNDIALS IDA |
| SUNDIALS Version | sundialsTB v2.6.2 | — | CasADi | sundialsTB v2.6.2 | — | Sundials.jl | SciPy |
| Input Options | Current | Current | Current, Voltage, Power | Current, Voltage, Power | Current | Current, Voltage, Power | Current, Voltage |

a) Note that PyBaMM offers other solvers in addition to the SUNDIALS IDA solver provided by CasADi. b) Crank-Nicolson method used to discretize the DAEs in time, resulting in a system of AEs at each time step, which are then solved using the bi-conjugate gradient stabilized method.

robust single-step approach to determine consistent initial conditions.

The open-source DFN codes, PyBaMM, LIONSIMBA, fastDFN, MPET, and PETLION, operate as open-loop software that requires known parameters to run the model. In contrast, closed-loop software, such as DEARLIBS and COBRAPRO, enable the co-optimization of parameters through an iterative process using battery data. Because open-loop software lack parameter optimization capabilities, they generally rely on uncalibrated, experimentally measured parameters reported in the literature. As a result, open-loop software cannot accurately model real-world batteries unless the correct parameters are obtained through alternative methods.

One such alternative is the co-simulation framework[78] developed in COMSOL Multiphysics®, a widely used commercial FEM software. Although COMSOL does not inherently include a parameter identification routine, the co-simulation framework developed in Ref. 78 uses LiveLink™ for MATLAB to enable communication between MATLAB and COMSOL. This setup allows for closed-loop parameter identification, where COMSOL simulates the DFN model and MATLAB's optimization tools identify the parameters.[78] However, COMSOL's expensive license fee and closed-source code pose barriers to collaboration and accessibility within the battery modeling community. GT-AutoLion[79] is another commercially available simulation software, which features a thermally coupled DFN model to predict cell performance, degradation, and safety but lacks an embedded parameter optimization routine.

As shown in Table I, DEARLIBS is the only open-source code besides COBRAPRO that includes a built-in parameter optimization feature. DEARLIBS was developed in 2021[69] and uses particle swarm optimization (PSO) to identify parameters using battery data. To demonstrate the parameter identification feature, the transport and kinetic parameters were identified using the 1C discharge curve obtained from LG INR21700-M50 cylindrical cells and validated with the urban dynamometer driving schedule (UDDS) driving cycle profile. However, DEARLIBS uses MATLAB's Symbolic Math Toolbox and the ode15s solver, which leads to significantly slow computation speeds, especially when implementing Fick's Law equation for the solid concentration in the particle.

In this paper, we present COBRAPRO (**Co**-simulation **B**atte**r**y Modeling for **A**ccelerated **Par**ameter **O**ptimization),[80,81] a new open-source DFN software with the capability to determine accurate parameters using battery current and voltage data. As experimentally measured parameters cannot be directly used, COBRAPRO systematically parameterizes the DFN model through a three-step process: structural identifiability to eliminate redundant DFN parameters, thermodynamic identification to optimize the stoichiometric coefficients, and practical identifiability to evaluate the sensitivity and correlation of parameters for the given battery data. The identifiable parameters are optimized using PSO, with COBRAPRO implementing PSO parallel computing to expedite convergence. This parameterization pipeline aims to address the overparameterization and identifiability issues in the DFN model by determining parameters that can be fixed to their measured values and parameters that can be optimized with higher confidence during parameter optimization.

To address the computational demands of the DFN model, COBRAPRO uses the SUNDIALS IDA solver[73,74] and automatic differentiation (AD) with CasADi,[77] achieving computation speeds up to three orders of magnitude faster than DEARLIBS. To determine consistent initial conditions, COBRAPRO implements both the robust single-step approach[33] and the SUNDIALS IDACalcIC AE solver, providing users with two options for DAE initialization.

The paper is structured as follows. An overview of the DFN model governing equations and parameters is provided, along with the numerical methods used to spatially discretize the equations. In COBRAPRO, FVM is used to discretize the macroscale equations, while both FVM and FDM implementations are used to discretize the microscale equations, providing users the flexibility to choose their preferred numerical approach. Subsequently, we delve into the description of the consistent initial condition methods: the single-step approach and SUNDIALS IDACalcIC. Following this section, the proposed DFN parameterization pipeline is discussed in detail. In the results, the computational efficiency of COBRAPRO is compared with other open-source DFN codes, showing comparable performance with PyBaMM and LIONSIMBA. A comparison study was conducted on the single-step approach and SUNDIALS IDACalcIC methods for determining consistent initial conditions at various C-rates and node numbers, demonstrating the robustness of the single-step approach. Finally, a case study is presented to demonstrate COBRAPRO's parameter identification pipeline on reference performance test (RPT) data[45] collected from fresh LG INR21700-M50T cells.[82] The parameterized model is validated against UDDS data, demonstrating good agreement between the experimental and simulation results.

## DFN Governing Equations

The DFN model[20] considers two porous electrodes, a separator, and electrolyte that permeates the pores in the electrode and separator, as shown in Fig. 1. The porous electrodes consist of active material particles that are assumed to be perfectly spherical. The particles are held together in a conductive matrix that acts as an electron conduction pathway from the current collectors to the active particles. The lithium-ions shuffle from one electrode to another via the electrolyte. At the particle-electrolyte interface, namely the particle surface, intercalation and deintercalation of the lithium-ions are modeled using charge transfer kinetics. In charging, the lithium-ions travel from the positive electrode to negative electrode and vice versa during discharging. As shown in Fig. 1, the DFN model consists of a macroscale dimension, which refers to the x-direction, consisting of the two electrodes and separator domains. The microscale dimension, often called the pseudo-dimension, refers to the radial direction within the particles in the electrodes.

Table II shows the DFN governing equations and boundary conditions. The equations consist of four PDEs describing the mass transport and charge conservation in the solid (active material particles) and liquid (electrolyte) phases, along with one AE for the Butler-Volmer kinetics.[83] Specifically, mass transport within the solid particles follows Fick's Law in spherical coordinates, governing the solid phase lithium concentration ($c_s$) dynamics. The lithium-ion concentration ($c_e$) dynamics in the electrolyte is governed by a diffusion equation with a forcing term. Charge conservation in both the solid and electrolyte phases is derived

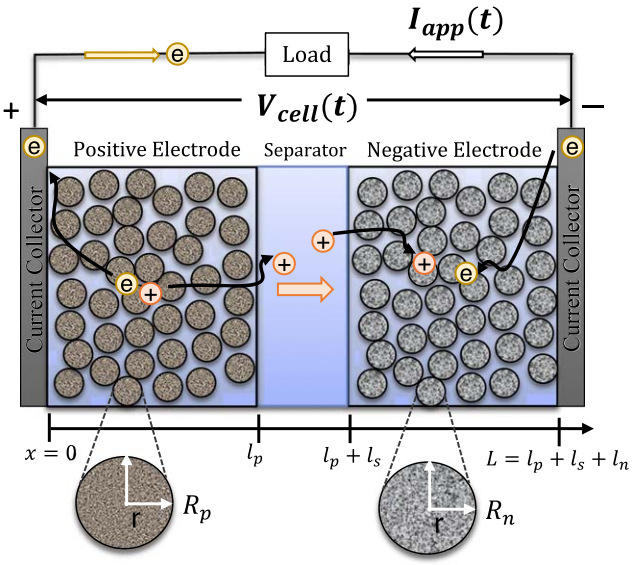


**Figure 1.** Schematic of a LIB DFN model in charging. The macroscale describes the cell through-thickness x-direction while the microscale refers to the r-direction in the particles, often referred to as the pseudo-dimension.

**Table II. DFN governing equations and boundary conditions.**

| Governing equations | Boundary conditions |
|---|---|

**Mass transport in solid phase**, $i \in (p, n)$

$$\frac{\partial c_{s,i}(r, x, t)}{\partial t} = \frac{1}{r^2}\frac{\partial}{\partial r}\left[r^2 D_{s,i}\frac{\partial c_{s,i}(r, x, t)}{\partial r}\right]$$

$$\left.\frac{\partial c_{s,i}(r, x, t)}{\partial r}\right|_{r=0} = 0$$

$$\left.\frac{\partial c_{s,i}(r, x, t)}{\partial r}\right|_{r=R_i} = -\frac{j_i(x, t)}{D_{s,i}} \quad [1]$$

**Charge conservation in solid phase,** $i \in (p, n)$

$$\frac{\partial}{\partial x}\left[\sigma_i^{eff}\frac{\partial \phi_{s,i}(x, t)}{\partial x}\right] = a_i F j_i(x, t)$$

$$\left.\sigma_p^{eff}\frac{\partial \phi_{s,p}(x, t)}{\partial x}\right|_{x=0} = -\frac{I_{app}(t)}{A_{cell}},\ \phi_{s,n}(x, t)|_{x=L} = 0$$

$$\left.\sigma_p^{eff}\frac{\partial \phi_{s,p}(x, t)}{\partial x}\right|_{x=l_p} = \left.\sigma_n^{eff}\frac{\partial \phi_{s,n}(x, t)}{\partial x}\right|_{x=l_p+l_s} = 0 \quad [2]$$

**Mass transport in electrolyte,** $i \in (p, s, n)$

$$\varepsilon_i\frac{\partial c_{e,i}(x, t)}{\partial t} = \frac{\partial}{\partial x}\left[D_i^{eff}\frac{\partial c_{e,i}(x, t)}{\partial x}\right] + \begin{cases} a_i(1 - t_+)j_i(x, t), & \text{for } i \in (p, n) \\ 0, & \text{for } i \in (s) \end{cases}$$

$$\left.\frac{\partial c_{e,p}(x, t)}{\partial x}\right|_{x=0} = \left.\frac{\partial c_{e,n}(x, t)}{\partial x}\right|_{x=L} = 0$$

$$\left.D_p^{eff}\frac{\partial c_{e,p}(x, t)}{\partial x}\right|_{x=l_p} = \left.D_s^{eff}\frac{\partial c_{e,s}(x, t)}{\partial x}\right|_{x=l_p}$$

$$\left.D_s^{eff}\frac{\partial c_{e,s}(x, t)}{\partial x}\right|_{x=l_p+l_s} = \left.D_n^{eff}\frac{\partial c_{e,n}(x, t)}{\partial x}\right|_{x=l_p+l_s} \quad [3]$$

**Charge conservation in electrolyte,** $i \in (p, s, n)$

$$\frac{\partial}{\partial x}\left[\kappa_i^{eff}\frac{2RT(1 - t_+)}{F}\frac{\partial \ln c_{e,i}(x, t)}{\partial x}\right] - \frac{\partial}{\partial x}\left[\kappa_i^{eff}\frac{\partial \phi_{e,i}(x, t)}{\partial x}\right] = \begin{cases} a_i F j_i(x, t), & \text{for } i \in (p, n) \\ 0, & \text{for } i \in (s) \end{cases}$$

$$\left.\frac{\partial \phi_{e,p}(x, t)}{\partial x}\right|_{x=0} = \left.\frac{\partial \phi_{e,n}(x, t)}{\partial x}\right|_{x=L} = 0$$

$$\left.\kappa_p^{eff}\frac{\partial \phi_{e,p}(x, t)}{\partial x}\right|_{x=l_p} = \left.\kappa_s^{eff}\frac{\partial \phi_{e,s}(x, t)}{\partial x}\right|_{x=l_p}$$

$$\left.\kappa_s^{eff}\frac{\partial \phi_{e,s}(x, t)}{\partial x}\right|_{x=l_p+l_s} = \left.\kappa_n^{eff}\frac{\partial \phi_{e,n}(x, t)}{\partial x}\right|_{x=l_p+l_s} \quad [4]$$

**Charge transfer kinetics,** $i \in (p, n)$

$$j_i(x, t) = j_{0,i}(x, t)\left[\exp\left(\frac{\alpha_{a,i}F}{RT}\eta_i(x, t)\right) - \exp\left(-\frac{\alpha_{c,i}F}{RT}\eta_i(x, t)\right)\right]$$

$$j_{0,i}(x, t) = k_i[(c_{s,i}^{\max} - c_{s,i}^{surf}(x, t))c_{e,i}(x, t)]^{\alpha_{a,i}}c_{s,i}^{surf}(x, t)^{\alpha_{c,i}}$$

$$\eta_i(x, t) = \phi_{s,i}(x, t) - \phi_{e,i}(x, t) - U_i(\theta_i^{surf}) \quad [5]$$

from Ohm's law, describing the dynamics for the solid ($\phi_s$) and electrolyte ($\phi_e$) potentials, respectively. The interfacial pore wall flux ($j$) at the particle surface is governed by the Butler-Volmer equation. Refer to Table III for additional DFN equations, including those for cell voltage and SOC in the electrodes.

To reduce the computational complexity, several approximations of Fick's law have been proposed.[84] In DEARLIBS, the mass transport in the particles is approximated using a two-term polynomial approximation (Table I), which assumes the solid concentration profile to be a parabolic function and solves for the volume averaged solid concentration ($c_{s,i}^{avg}$) and surface concentration ($c_{s,i}^{surf}$) in the form of two ODEs[84] given as

$$\frac{\partial c_{s,i}^{avg}(x, t)}{\partial t} = -3\frac{j_i(x, t)}{R_i}$$

$$c_{s,i}^{surf}(x, t) - c_{s,i}^{avg}(x, t) = -\frac{R_i}{D_{s,i}}\frac{j_i(x, t)}{5} \quad [1]$$

where $i \in (p, n)$. Although this method significantly reduces the computation burden by eliminating the $c_{s,i}$ variables along the r-direction, the polynomial approximation was shown to display low accuracy at high C-rates.[30] Fick's law should be implemented to accurately model the non-linear solid concentration profile at high C-rates. In COBRAPRO, Fick's law is implemented while maintaining computational efficiency by leveraging the SUNDIALS IDA solver.[73,74] In the **Results** section, we compare the computational speeds of COBRAPRO and DEARLIBS using Fick's Law implementations, demonstrating that the SUNDIALS IDA solver used in COBRAPRO significantly outperforms the MATLAB ode15s implementation in DEARLIBS.

Table IV presents the parameters required to solve the DFN model, categorized into geometric, transport, kinetic, concentration, stoichiometric, OCP, and electric parameters. While transport and kinetic parameters such as electrolyte diffusion ($D$), electrolyte conductivity ($\kappa$), solid phase diffusion coefficients ($D_{s,p}$, $D_{s,n}$), and reaction rate coefficients ($k_p$, $k_n$) typically vary with temperature and are often modeled using the Arrhenius equation,[85] COBRAPRO implements an isothermal model and assumes temperature-invariant parameters. It is also important to remember that the OCP function is a function of temperature and the OCP must be experimentally measured from half-cell GITT or low C-rate capacity test experiments at the cell operating temperature. Furthermore, electrolyte parameters such as the transference number ($t_+$), $D$, and $\kappa$ depend on the electrolyte concentration as shown in Ref. 43. DEARLIBS uses constant values for $t_+$, $D$, and $\kappa$, whereas COBRAPRO provides the flexibility to define these parameters as functions of the electrolyte concentration.

Although the solid diffusion and reaction rate coefficients vary with SOC in the electrodes,[86,87] COBRAPRO treats $D_{s,p}$, $D_{s,n}$, $k_p$,

**Table III. Additional equations in the DFN model.**

| Additional equations | |
|---|---|
| **Effective electrolyte diffusion,** $i \in (p, s, n)$ | |
| $D_i^{eff} = \varepsilon_i^{brugg_i} D_i$ | [6] |
| **Effective electrolyte conductivity,** $i \in (p, s, n)$ | |
| $\kappa_i^{eff} = \varepsilon_i^{brugg_i} \kappa_i$ | [7] |
| **Solid volume fraction,** $i \in (p, n)$ | |
| $\varepsilon_i^{solid} = 1 - \varepsilon_i - \varepsilon_i^{filler}$ | [8] |
| **Effective electrode conductivity,** $i \in (p, n)$ | |
| $\sigma_i^{eff} = \sigma_i \varepsilon_i^{solid}$ | [9] |
| **Specific interfacial area,** $i \in (p, n)$ | |
| $a_i = 3\frac{\varepsilon_i^{solid}}{R_i}$ | [10] |
| **Volume averaged solid particle concentration** | |
| $c_{s,p}^{avg}(x, t) = \frac{3}{R_p}\int_0^{R_p} c_{s,p}(r, x, t) r^2 \, dr$ | |
| $c_{s,n}^{avg}(x, t) = \frac{3}{R_n}\int_0^{R_n} c_{s,n}(r, x, t) r^2 \, dr$ | [11] |
| **Normalized surface solid particle concentration** | |
| $\theta_p^{surf}(x, t) = \frac{c_{s,p}^{surf}(x, t)}{c_{s,p}^{max}}, \ \theta_p^{100\%} < \theta_p^{surf} < \theta_p^{0\%}$ | |
| $\theta_n^{surf}(x, t) = \frac{c_{s,n}^{surf}(x, t)}{c_{s,n}^{max}}, \ \theta_n^{0\%} < \theta_n^{surf} < \theta_n^{100\%}$ | [12] |
| **Normalized bulk solid particle concentration** | |
| $\theta_p^{bulk}(t) = \frac{1}{c_{s,p}^{\max} l_p}\int_0^{l_p} c_{s,p}^{avg}(x, t) \, dx$ | |
| $\theta_n^{bulk}(t) = \frac{1}{c_{s,n}^{\max} l_n}\int_{l_p+l_s}^{L} c_{s,n}^{avg}(x, t) \, dx$ | [13] |
| **State of Charge** | |
| $SOC_p(t) = \frac{\theta_p^{0\%} - \theta_p^{bulk}(t)}{\theta_p^{0\%} - \theta_p^{100\%}}$ | |
| $SOC_n(t) = \frac{\theta_n^{100\%} - \theta_n^{bulk}(t)}{\theta_n^{100\%} - \theta_n^{0\%}}$ | [14] |
| **Cell Voltage** | |
| $V_{cell}(t) = \phi_{s,p}(x, t)\|_{x=0} - \phi_{s,n}(x, t)\|_{x=L} - I_{app}(t)R_c$ | [15] |

and $k_n$ as constants to simplify the implementation. Future extensions to COBRAPRO could incorporate the SOC dependence of these coefficients. In Ref. 88, the SOC-dependence of the solid diffusion coefficient is implemented by reformulating Fick's law using chemical potential as the driving force for diffusion. This allows a single coefficient to represent the solid diffusion coefficient as a function of SOC by incorporating the OCP data into the solid phase diffusion model. In Ref. 87, the SOC-dependence of the reaction rate constant is addressed using the multiple-species multiple-reactions (MSMR) approach, instead of the traditional Butler-Volmer model, to account for multiple electrochemical reactions and species within the electrode material.

## Numerical Methods

***Discretization methods.***—To solve the DFN model, the four PDEs shown in Table II require spatial discretization, transforming them to ODEs and AEs. FDM is the simple to implement but may require a large number of nodes to ensure mass conservation during cycling simulations.[89] FVM uses the volume integral form of the governing equation within each control volume (CV), setting the flux entering a CV equal to the flux leaving the adjacent CV, thereby ensuring mass conservation by construction.[30,89]

Figure 2 shows a schematic of the discretization method used in COBRAPRO. FVM is used to discretize the macroscale variables along the x-direction, namely the variables $c_e$, $\phi_e$, and $\phi_s$. Each CV along the x-direction has an associated center indicated by $k$ that spans the interval $[x_{k-1}, \ x_{k+1}]$ given $k \in [1, N_{psn}]$ where $N_{psn} = N_p + N_s + N_n$. Each CV has a length of $\Delta x_i = l_i/N_i$ for $i \in (p, s, n)$. In Fig. 2, $\bar{\psi}$ refers to the variables $\bar{c}_e$, $\bar{\phi}_e$, and $\bar{\phi}_s$, where the $\bar{(\cdot)}$ notation refers to the volume averaged value of the variable within the CV centered at index $k$.

For the microscale equations, both FDM and FVM implementations are used to discretize $c_{s,i}$ along the particle's radial direction, where each particle is located at $x_k$ for $k \in [1, N_p]$ given $i = p$ and for $k \in [N_{ps} + 1, N_{psn}]$ given $i = n$. FDM nodes along the r-direction are indexed from $k_r \in [1, N_{ri}]$ with equally spaced sections each with a length of $\Delta r_i = R_i/N_{ri}$ for $i \in (p, n)$. When using FDM in the radial discretization, users should check for mass conservation, especially when simulating multiple cycles.[89,90] For FVM, the particle is spatially discretized into spherical shells along the radial direction.[89] Each spherical shell is considered a CV, which has a center at index $k_r$ spanning the interval $[r_{k_r-1/2}, \ r_{k_r+1/2}]$ with a radial thickness of $\Delta r_i = R_i/N_{ri}$. Since surface concentration cannot be directly determined using FVM, an approximation technique is needed to estimate the surface value at $k = N_{ri} + 1/2$. COBRAPRO uses the third-order Hermite polynomial approximation, as proposed in Ref. 89, due to its higher accuracy compared to the linear interpolation implemented in PyBaMM.

Refer to Appendix A for the complete discretized DFN equations in state-space form (Tables A·1, A·2, A·3, and A·4). Appendix A also includes a demonstration of the FVM derivation for the mass conservation in the electrolyte.

***Differential Algebraic Equations (DAEs).***—The discretized DFN equations constitute a set of ODEs and AEs that can be represented as a DAE system.[91,92] This DAE representation enables the use of various numerical methods available to solve a dynamical system.[93,94] A DAE in its most general form can be written in implicit form as

$$\boldsymbol{F}(\dot{\boldsymbol{x}}(t), \boldsymbol{x}(t), \boldsymbol{z}(t), u(t)) = 0 \qquad [2]$$

where $\boldsymbol{x}(t)$ is the state vector, $\dot{\boldsymbol{x}}(t)$ is the time derivative of the state vector, $\boldsymbol{z}(t)$ is the algebraic variable vector, and $u(t)$ is the system input. The Jacobian matrix $(\partial \boldsymbol{F}/\partial \dot{\boldsymbol{x}})$ of Eq. (2) is singular because of the AEs present in the DAE system.

The discretized DFN equations can be written in semi-explicit DAE form as

$$\begin{aligned} \dot{\boldsymbol{x}}(t) &= \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{z}(t), u(t)), \quad \boldsymbol{x}(t=0) = \boldsymbol{x}_0 \\ \boldsymbol{0} &= \boldsymbol{g}(\boldsymbol{x}(t), \boldsymbol{z}(t), u(t)), \quad \boldsymbol{z}(t=0) = \boldsymbol{z}_0, \end{aligned} \qquad [3]$$

where $\boldsymbol{f}$ is the right-hand side of the differential equation, $\boldsymbol{g}$ is the AE, $\boldsymbol{x_0}$ is the initial state vector, and $\boldsymbol{z_0}$ is the initial algebraic vector. To solve a DAE system, it is essential to determine the exact values of $\boldsymbol{z_0}$ that satisfy the AEs at $t = 0$, represented as $\boldsymbol{g}(\boldsymbol{x_0}, \boldsymbol{z_0}, u_0)$. Consistent initial conditions for $\boldsymbol{z_0}$ are identified through a DAE initialization process, which is further detailed in the **Consistent Initial Conditions** section.

The state and algebraic variable vectors and system input in COBRAPRO are presented as

$$\begin{aligned} \boldsymbol{x} &= [\boldsymbol{c_{s,p}}, \boldsymbol{c_{s,n}}, \boldsymbol{c_{e,p}}, \boldsymbol{c_{e,s}}, \boldsymbol{c_{e,n}}]^T \\ \boldsymbol{z} &= [\boldsymbol{\phi_{s,p}}, \boldsymbol{\phi_{s,n}}, \boldsymbol{\phi_{e,p}}, \boldsymbol{\phi_{e,s}}, \boldsymbol{\phi_{e,n}}, \boldsymbol{j_p}, \boldsymbol{j_n}]^T \\ u(t) &= I_{app}(t), \end{aligned} \qquad [4]$$

where $\boldsymbol{c_{s,i}}$ for $i \in (p, n)$ is the solid lithium concentration vector, $\boldsymbol{c_{e,i}}$ for $i \in (p, s, n)$ is the electrolyte lithium-ion concentration vector, $\boldsymbol{\phi_{s,i}}$ for $i \in (p, n)$ is the solid potential vector, $\boldsymbol{\phi_{e,i}}$ for $i \in (p, s, n)$ is the electric potential vector, $\boldsymbol{j_i}$ for $i \in (p, n)$ is the pore wall flux vector, and $I_{app}(t)$ is the applied current vector as a function of time. Note that current is not

**Table IV. Summary of DFN model parameters.**

| Category | Symbol | Units | Description |
|---|---|---|---|
| **Geometric** | | | |
| | $l_p$ | m | Positive electrode thickness |
| | $l_s$ | m | Separator thickness |
| | $l_n$ | m | Negative electrode thickness |
| | $R_p$ | m | Positive electrode particle radius |
| | $R_n$ | m | Negative electrode particle radius |
| | $\varepsilon_p$ | – | Positive electrode porosity |
| | $\varepsilon_s$ | – | Separator porosity |
| | $\varepsilon_n$ | – | Negative electrode porosity |
| | $\varepsilon_p^{filler}$ | – | Binder volume fraction in positive electrode |
| | $\varepsilon_n^{filler}$ | – | Binder volume fraction in negative electrode |
| | $brugg_p$ | – | Bruggeman coefficient in positive electrode |
| | $brugg_s$ | – | Bruggeman coefficient in separator |
| | $brugg_n$ | – | Bruggeman coefficient in negative electrode |
| | $A_{cell}$ | $m^2$ | Cell cross-sectional area |
| **Transport** | | | |
| | $t_+$ | – | Transference number |
| | $D$ | $m^2/s$ | Electrolyte diffusion |
| | $\kappa$ | S/m | Electrolyte conductivity |
| | $D_{s,p}$ | $m^2/s$ | Solid diffusion in positive electrode |
| | $D_{s,n}$ | $m^2/s$ | Solid diffusion in negative electrode |
| | $\sigma_p$ | S/m | Solid conductivity in positive electrode |
| | $\sigma_n$ | S/m | Solid conductivity in negative electrode |
| **Kinetic** | | | |
| | $k_p$ | $m^{2.5}/(mol^{0.5}s)$ | Reaction rate in positive electrode |
| | $k_n$ | $m^{2.5}/(mol^{0.5}s)$ | Reaction rate in negative electrode |
| | $\alpha_{a,p}$ | – | Anodic charge transfer coefficient in positive electrode[a] |
| | $\alpha_{a,n}$ | – | Anodic charge transfer coefficient in negative electrode[a] |
| | $\alpha_{c,p}$ | – | Cathodic charge transfer coefficient in positive electrode[a] |
| | $\alpha_{c,n}$ | – | Cathodic charge transfer coefficient in negative electrode[a] |
| **Concentration** | | | |
| | $c_{s,p}^{max}$ | $mol/m^3$ | Maximum solid concentration in positive electrode |
| | $c_{s,n}^{max}$ | $mol/m^3$ | Maximum solid concentration in negative electrode |
| | $c_0$ | $mol/m^3$ | Equilibrium lithium-ion concentration in electrolyte |
| **Stoichiometric** | | | |
| | $\theta_p^{0\%}$ | – | Stoichiometric coefficient in positive electrode at 0% SOC |
| | $\theta_p^{100\%}$ | – | Stoichiometric coefficient in positive electrode at 100% SOC |
| | $\theta_n^{0\%}$ | – | Stoichiometric coefficient in negative electrode at 0% SOC |
| | $\theta_n^{100\%}$ | – | Stoichiometric coefficient in negative electrode at 100% SOC |
| **OCP** | | | |
| | $U_p$ | V | OCP function in positive electrode |
| | $U_n$ | V | OCP function in negative electrode |
| **Electric** | | | |
| | $R_c$ | Ω | Contact resistance |

a) Charge transfer coefficients set to 0.5 for LIBs.[11]

the only valid input for the DFN model. As shown in Table I, PyBaMM, LIONSIMBA, PETLION, and MPET support various input modalities, including voltage and power, providing simulation operating conditions beyond current-driven scenarios.

## Consistent Initial Conditions

COBRAPRO offers two approaches to determine consistent initial conditions for the DAE system: SUNDIALS IDACalcIC and single-step approach.[35] Both approaches require the user to provide an initial estimate, given as the open-circuit voltage (OCV) conditions when the battery is at rest. Initial guesses for the solid particle concentration ($c_{s,i,0}$) are calculated using the initial SOC ($SOC_0$) and the stoichiometric values pertaining to the electrode, $\theta_i^{100\%}$ and $\theta_i^{0\%}$ for $i \in (p, n)$. The solid potential initial guesses are then calculated using OCP values at $c_{s,i,0}$, given as $U_i(c_{s,i,0}/c_{s,p}^{max})$. The initial guess for the electrolyte concentration is set equal to the initial concentration in the electrolyte $c_0$. The initial guesses for the pore wall flux and electrolyte potential are given as zero as shown in Table V.

***Initialization with SUNDIALS IDACalcIC.***—The SUNDIALS IDA library provides an initialization function called IDACalcIC that determines the initial algebraic variable values. According to SUNDIALS IDA documentation,[95] IDACalcIC operates by reformulating the semi-explicit DAE given in Eq. (3) into a fully implicit DAE form given as

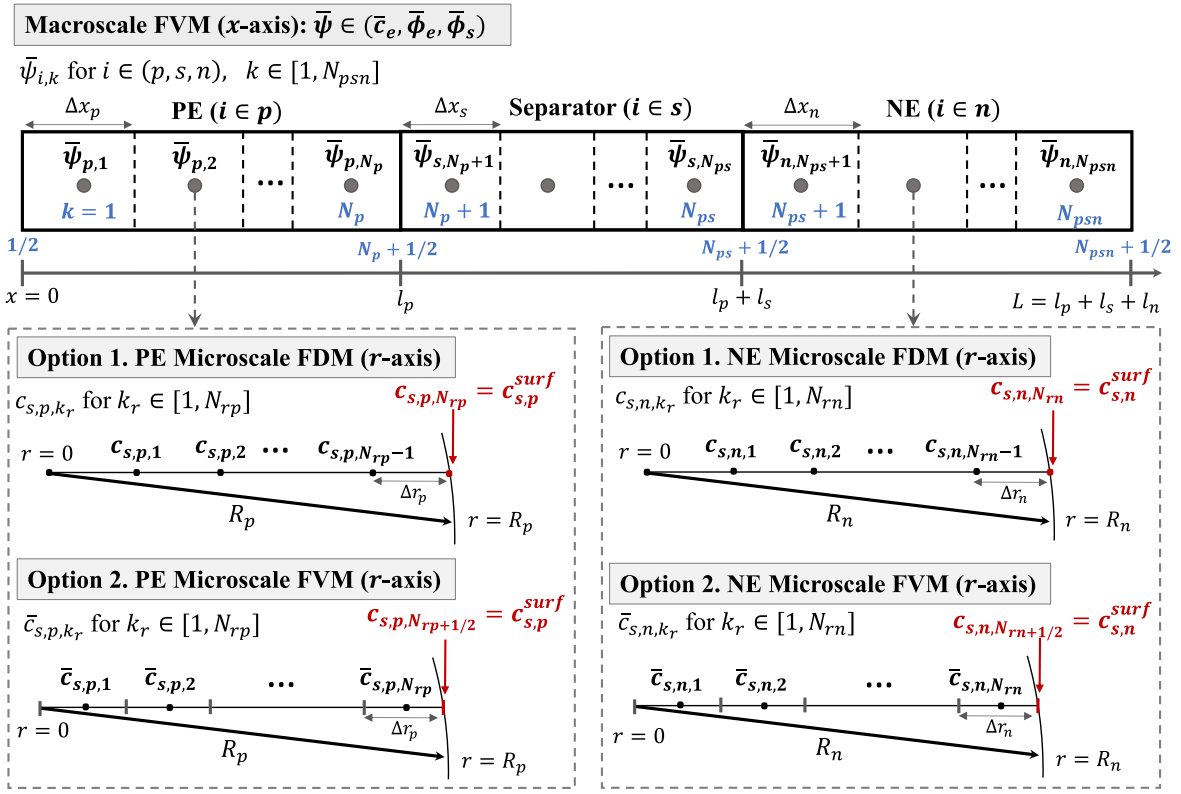


**Figure 2.** Discretization of LIB cell in the x and r-directions. The macroscale x-direction is discretized using FVM, where each CV is associated with a center $x_k$ and spans the interval $[x_{k-1}, x_{k+1}]$ where $k \in [1, N_{psn}]$ given $N_{psn} = N_p + N_s + N_n$. In the microscale r-direction, the discretization is carried out using either FDM, where nodes are indexed as $k_r = [1, N_{ri}]$, or FVM, where the center of each CV is indexed as $k_r = [1, N_{ri}]$ given $i \in (p, n)$.

$$\boldsymbol{F}\left(\begin{bmatrix}\dot{\boldsymbol{x}}_0\\ \boldsymbol{z}_0\end{bmatrix}\right) \equiv \begin{bmatrix}\dot{\boldsymbol{x}}_0 - \boldsymbol{f}(\boldsymbol{x}_0, \boldsymbol{z}_0, u)\\ \boldsymbol{g}(\boldsymbol{x}_0, \boldsymbol{z}_0, u)\end{bmatrix} = \boldsymbol{0}. \quad [5]$$

Since the initial differential variables $\boldsymbol{x}_0 = [c_{s,p,0}, c_{s,n,0}, c_{e,p,0}, c_{e,s,0}, c_{e,n,0}]^T$ are known (Table V), Eq. (5) simultaneously solves for $\dot{\boldsymbol{x}}_0$ and $\boldsymbol{z}_0$ using a nonlinear solver that employs the modified version of Newton's method.[34] The **Comparison of Initialization Methods** section compares the IDACalcIC and single-step approach at different C-rates and node numbers.

***Initialization with single-step approach.—***In contrast to the SUNDIALS IDACalcIC method, the single-step approach modifies the AEs into implicit ODEs, eliminating the need for a nonlinear algebraic solver for the constraint equation $\boldsymbol{g}(\boldsymbol{x}_0, \boldsymbol{z}_0, u_0)$. As shown in Ref. 35, a perturbation coefficient ($\alpha$) is applied to the AEs to convert them to implicit ODEs while a hyperbolic tangent switch function $T_H = 1/2(1 + \tanh(q(t - t_{init}))$ is applied to the original ODEs shown in Eq. (3). Therefore, the DAE system in Eq. (3) is converted to a newly constructed nonlinear ODE system given as

$$\dot{\boldsymbol{x}}(t) = \overbrace{\frac{1}{2}(1 + \tanh(q(t - t_{init})))}^{T_H} \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{z}(t), u(t)) \quad [6]$$

$$\alpha \cdot \dot{\boldsymbol{g}}(\boldsymbol{x}(t), \boldsymbol{z}(t), u(t)) = -\boldsymbol{g}(\boldsymbol{x}(t), \boldsymbol{z}(t), u(t)) \quad [7]$$

where $q$ is a weighting factor that controls the discreteness of the switch function and $t_{init}$ is the time allocated to find the consistent algebraic initial conditions $\boldsymbol{z}_0$. Given a sufficiently large $q$, $T_H$ equals zero for $t < t_{init}$, which effectively disables Eq. (6) and equals one for $t > t_{init}$, thereby reactivating the equation. From $t = 0$ to $t = t_{init}$, Eq. (6) is "turned off" and the system focuses on solving Eq. (7) to find consistent initial conditions. Once $t > t_{init}$, the ODEs in Eq. (6) are re-engaged, and the system solves both Eqs. (6) and (7) using the established initial conditions during $t < t_{init}$. According to Ref. 35, the duration $t_{init}$ should be adjusted based on the magnitude of the perturbation factor $\alpha$. In COBRAPRO, the default parameters are set to $q = 1000$, $t_{init} = 5$ seconds, and $\alpha = 10^{-3}$,[35] but users have the option to modify these values to better fit their simulation needs. Smaller

**Table V. Initial guess corresponding to OCV conditions.**

| Variable | Positive electrode ($i = p$) | Separator ($i = s$) | Negative electrode ($i = n$) |
|---|---|---|---|
| $c_{s,i}$ | $c_{s,p,0} = (SOC_0 \times (\theta_p^{0\%} - \theta_p^{100\%}) + \theta_p^{100\%}) \times c_{s,p}^{\max}$ | – | $c_{s,n,0} = (SOC_0 \times (\theta_n^{100\%} - \theta_n^{0\%}) + \theta_n^{0\%}) \times c_{s,n}^{\max}$ |
| $\phi_{s,i}$ | $\phi_{s,p,0} = U_p(c_{s,p,0}/c_{s,p}^{\max})$ | – | $\phi_{s,n,0} = U_n(c_{s,n,0}/c_{s,n}^{\max})$ |
| $c_{e,i}$ | $c_{e,p,0} = c_0$ | $c_{e,s,0} = c_0$ | $c_{e,n,0} = c_0$ |
| $\phi_{e,i}$ | $\phi_{e,p,0} = 0$ | $\phi_{e,s,0} = 0$ | $\phi_{e,n,0} = 0$ |
| $j_i$ | $j_{p,0} = 0$ | – | $j_{n,0} = 0$ |

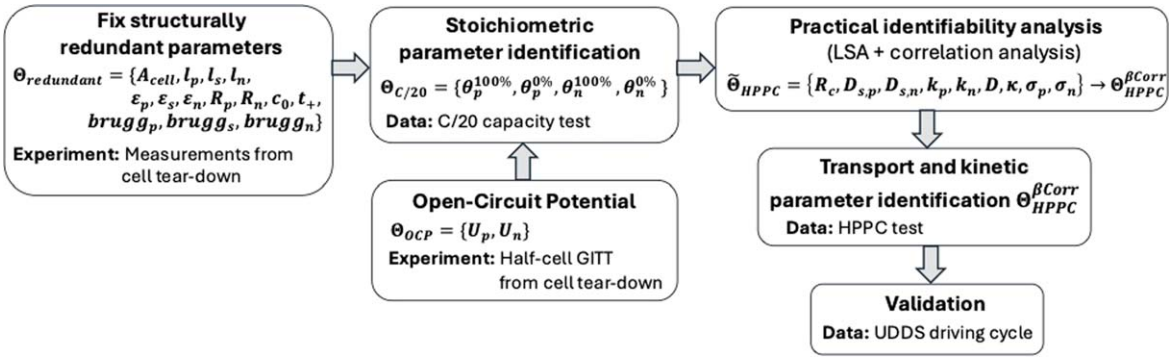


**Figure 3.** Proposed DFN parameter identification pipeline.

values of $\alpha$ improve the accuracy of the consistent initial condition solution but increase system stiffness, whereas larger $\alpha$ values decrease stiffness but reduce solution accuracy.[35]

Equations (6) and (7) can be reconfigured into an implicit ODE system in the form $\mathcal{M}(\dot{\boldsymbol{x}}(t), \boldsymbol{z}(t), \boldsymbol{u}(t), \dot{\boldsymbol{x}}(t), \dot{\boldsymbol{z}}(t), \dot{\boldsymbol{u}}(t)) = \mathbf{0}$ such that

$$\underbrace{\begin{bmatrix} \dot{\boldsymbol{x}}(t) - \frac{1}{2}(1 + \tanh(q(t - t_{init})))\boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{z}(t), \boldsymbol{u}(t)) \\ \alpha\dot{\boldsymbol{g}}(\boldsymbol{x}(t), \boldsymbol{z}(t), \boldsymbol{u}(t)) + \boldsymbol{g}(\boldsymbol{x}(t), \boldsymbol{z}(t), \boldsymbol{u}(t)) \end{bmatrix}}_{\mathcal{M}} = \mathbf{0}. \quad [8]$$

An implicit ODE solver can be used to solve the system described in Eq. (8). Although CasADi features a built-in SUNDIALS IDA solver for solving ODE and DAE systems, even in the latest version (CasADi v3.6.5), SUNDIALS IDA is limited to handling ODE and DAE systems in semi-explicit form. In the single-step approach, the DAE system is reformulated as an implicit ODE system (Eq. (8)), which cannot be expressed in semi-explicit form. Therefore, COBRAPRO uses the IDA solver from the original SUNDIALS package (sundialsTB v.2.6.2), which can handle fully implicit ODE and DAE systems.

## Parameter Identification Framework

Figure 3 shows the proposed layered parameterization framework. The first step involves fixing structurally redundant parameters, $\Theta_{redundant}$, to their cell tear-down measurement values. In the second step, the stoichiometric coefficients are identified using the C/20 capacity test data. Then, practical identifiability analysis is conducted using local sensitivity analysis (LSA) and correlation analysis to determine a subset of identifiable parameters using the HPPC profile. The identifiable parameters are then optimized using the HPPC test data. Finally, the optimized parameters are validated using the UDDS driving cycle data.

***Step 1. Reducing redundant parameters.***—In the first step of model parameterization, we implement parameter aggregation, or lumping, a technique used in structural identifiability to determine redundant parameters in the DFN model. The redundant parameters are fixed to their measured values obtained from cell tear-down experiments. This procedure ensures that the remaining parameters can be reliably estimated in subsequent parameter identification steps.

Parameter lumping involves normalizing the DFN governing equations and grouping parameters for each normalized equation.[59,60] For example, normalizing Fick's law shown in Eq. (1) reveals a lumped parameter $D_{s,i}/R_i$ for each electrode $i \in (p, n)$. The lumped parameter indicates that increasing $D_{s,i}$ and decreasing $R_i$ leads to the same solid concentration dynamics. Attempting to identify $D_{s,i}$ and $R_i$ at the same time can lead to structural identifiability issues. To circumvent this problem, one can either identify the lumped parameter $D_{s,i}/R_i$[59,60] or fix one of the parameters to its measured value. In this work, the latter approach is taken. Thus, $R_i$ is fixed to its measured value from cell tear-down,[38] thereby improving the identifiability of $D_{s,i}$ from a structural standpoint. Although $D_{s,i}$ could be fixed and $R_i$ estimated instead, typically $R_i$ is easier to measure than $D_{s,i}$, which requires additional half-cell GITT experiments and analysis.[96]

The procedure described above is repeated for the other DFN equations to determine the remaining redundant parameters. Following the process oulined in Ref. 60, the redundant parameters are identified as $\Theta_{redundant} = \{A_{cell}, l_p, l_s, l_n, \varepsilon_p, \varepsilon_s, \varepsilon_n, R_p, R_n, c_0, t_+, brugg_p, brugg_s, brugg_n\}$.[b] Therefore, all geometric parameters as well as the equilibrium electrolyte concentration $c_0$, and the transference number $t_+$ are fixed to their measured values.[38] We provide a brief overview of the redundant parameters but refer to Ref. 60 for detailed derivations.

Structural redundancy can be observed in the Bruggeman coefficients and porosity from Eqs. (6) and (7). In Eq. (6), $D_i$ is multiplied by $\varepsilon_i^{brugg_i}$ for $i \in (p, s, n)$ to yield the effective electrolyte diffusivity $D_i^{eff}$. Any combination of $D_i$ and $\varepsilon_i^{brugg_i}$ can be multiplied to yield the same value for $D_i^{eff}$. Therefore, $\varepsilon_i$ and $brugg_i$ are fixed to their measured values and $D_i$ is left to be identified in future steps. A similar approach is applied in Eq. (7), where $\kappa_i$ is treated as the free variable to be identified later. Given that $\kappa_i$ will be identified, $t_+$ becomes redundant in the term $\kappa_i^{eff}\frac{2RT(1-t_+)}{F}$ in Eq. (4). Additionally, Eq. (5) shows that $c_0$ is redundant, since $k_i$ is multiplied by $c_0$ in $j_{0,i}(t=0) = k_i[c_e(t=0)(c_{s,i}^{\max} - c_{s,i}^{surf}(t=0))c_{s,i}^{surf}(t=0)]^{0.5}$ where $c_e(t=0) = c_0$ and $c_{s,i}^{surf}(t=0) = c_{s,i,0}$ given $i \in (p, n)$.

***Step 2. Stoichiometric coefficient identification.***—When the battery is at rest, the cell is under thermodynamic conditions characterized by the electrode OCPs ($U_p$, $U_n$) and the stoichiometric coefficients ($\theta_p^{100\%}$, $\theta_p^{0\%}$, $\theta_n^{100\%}$, $\theta_n^{0\%}$), which define the upper and lower limits of the OCP windows. The electrode OCPs are typically obtained through half-cell GITT experiments that require cell tear-down, while the stoichiometric coefficients can be calibration using OCV or pseudo-OCV data.[53,97]

In this work, fresh cell C/20 capacity test data is used to identify the stoichiometric parameters: $\Theta_{C/20} = \{\theta_p^{100\%}, \theta_p^{0\%}, \theta_n^{100\%}, \theta_n^{0\%}\}$. The following assumptions are made:

[b]It is important to recognize that fixing redundant parameters does not make the DFN model fully structurally identifiable. As a system of coupled PDEs, the DFN model lacks a closed-form solution for full structural identifiability unless the equations are simplified, linearized, and decoupled.[61] The aim of this step is not to achieve complete structural identifiability but to reduce redundant parameters, thereby improving the reliability of parameter estimation of the remaining parameters.

**Assumption A1:** Voltage from the C/20 capacity test is assumed to be pseudo-OCV since the C/20 rate involves a low current and the polarization effects are minimal.[98]

**Assumption A2:** OCP functions are obtained from GITT half-cell experiments reported in Ref. 38, listed in Table C1, since the data used in this study is generated from the same cells tested in Ref. 38.

**Assumption A3:** Since the stoichiometric coefficients are identified from C/20 capacity test conducted on a fresh cell, the identified stoichiometric coefficients are only valid for a fresh cell. As the cell ages, the fresh cell OCP windows shift[99] and the stoichiometric coefficients require re-identification.

**Assumption A4:** The maximum particle concentrations ($c_{s,p}^{\max}$ and $c_{s,n}^{\max}$) are calculated using the identified stoichiometric coefficient values using the theoretical cell capacity equation. Note that this equation only holds true for a fresh cell.

Leveraging the multi-objective cost function proposed in Ref. 42 for ESPM and implemented in Ref. 89 for SPM, the following optimization problem is formulated to identify $\Theta_{C/20}$ for the DFN model:

$$\begin{aligned}
&\underset{\Theta_{C/20}}{\text{minimize}}\ J(\Theta_{C/20}) = J^{V}(\Theta_{C/20}) + J^{\text{SOC}_p}(\Theta_{C/20}) + J^{\text{SOC}_n}(\Theta_{C/20})\\
&\textit{subject to}\ (a)\ \textit{Discretized equations in Tables}\ \text{A·1, A·2, A·3, A·4}\\
&\qquad (b)\ \Theta_{C/20}^{\text{lb}} \leqslant \Theta_{C/20} \leqslant \Theta_{C/20}^{\text{ub}},\\
&\qquad (c)\ V_{\text{sim}} \geqslant V_{\text{lower}}^{\text{cutoff}},\\
&\qquad (d)\ V_{\text{sim}}^{\max} = V_{\text{exp}}^{\max},\\
&\qquad (e)\ (1-\gamma)Q_{\text{dis,exp}}^{C/20} \leqslant Q_{\text{dis,sim}}^{C/20} \leqslant (1+\gamma)Q_{\text{dis,sim}}^{C/20},\\
&\qquad (f)\ c_{s,p}^{\max} = \frac{3600Q_{\text{dis,exp}}^{C/20}}{\varepsilon_p^{\text{solid}}\text{Fl}_p A_{\text{cell}}(\theta_p^{0\%} - \theta_p^{100\%})},\\
&\qquad (g)\ c_{s,n}^{\max} = \frac{3600Q_{\text{dis,exp}}^{C/20}}{\varepsilon_n^{\text{solid}}\text{Fl}_n A_{\text{cell}}(\theta_n^{100\%} - \theta_n^{0\%})},\\
&\qquad (h)\ c_{s,i}^{\text{surf}} \leqslant c_{s,i}^{\max},\ \ i \in (p, n).
\end{aligned} \tag{9}$$

The multi-objective cost function implemented in Refs. 42 and 89 was modified for the pseudo-OCV identification problem shown in Eq. (9) by incorporating constraints (d), (e), (f), and (g). Constraint (d) ensures that the maximum (initial) voltage of the simulation matches that of the experimental data, where $V_{sim}^{\max} = V_{sim}(t=0)$ and $V_{exp}^{\max} = V_{exp}(t=0)$. Note that constraint (d) only is valid for a fresh cell, since aged cells exhibit increased polarization and do not match the OCPs and OCV curves of the fresh cells. To ensure charge conservation, constraint (e) enforces the simulated discharged capacity $Q_{dis,sim}^{C/20}$ to be within $(1-\gamma)Q_{dis,exp}^{C/20}$ and $(1+\gamma)Q_{dis,exp}^{C/20}$ of the experimental discharged capacity.[100] As denoted in Ref. 45, the discharged capacity is defined as the total capacity extracted from the battery during the C/20 capacity test. In our analysis, $\gamma = 0.01$ such that $Q_{dis,sim}^{C/20}$ is constrained within $0.99Q_{dis,sim}^{C/20}$ and $1.01Q_{dis,sim}^{C/20}$. Lastly, constraints (f) and (g) are formulated using the theoretical cell capacity equation,[11] valid for a fresh cell and given as

$$Q_i = \frac{\varepsilon_i^{solid} Fl_i A_{cell} c_{s,i}^{\max} |\theta_i^{100\%} - \theta_i^{0\%}|}{3600}, \quad i \in (p, n) \tag{10}$$

where $Q_i$ is the theoretical capacity of the cell. In this analysis, we assume that $Q_i$ is equal to $Q_{dis,exp}^{C/20}$ calculated from the C/20 capacity test. This is a valid assumption since the C/20 capacity is taken as the pseudo-OCV curve.

The optimization problem in Eq. (9) is also subject to the discretized DAE equations in constraint (a), the lower and upper bounds of $\Theta_{C/20}$ in constraint (b), and the chemistry-dependent lower voltage cut-off limit where $V_{lower}^{cutoff} = 2.5$ V for LG INR21700-M50T cells in constraint (c). The last constraint (h) prevents the surface concentration of the particles from exceeding the maximum allowable solid concentration, ensuring that the exchange current density per unit charge ($j_{0,i}$) remains positive in the Butler-Volmer equation. Note that the objective function could include minimizing the error between the experimental and simulated incremental capacity or differential voltage as demonstrated in Ref. 101.

The multi-objective cost function $J(\Theta)$[42,89] is defined in terms of the root mean square error (RMSE) between the experimental and simulated voltage, $SOC_p$, and $SOC_n$ such that

$$\begin{aligned}
J(\Theta) = &\underbrace{\sqrt{\frac{1}{N}\sum_{j=1}^{N}\left(\frac{V_{exp}(j) - V_{sim}(\Theta, j)}{V_{exp}(j)}\right)^2}}_{J^{V}(\Theta)}\\
&+ \underbrace{\sqrt{\frac{1}{N}\sum_{j=1}^{N}(SOC_{exp}(j) - SOC_{p,sim}(\Theta, j))^2}}_{J^{SOC_p}(\Theta)}\\
&+ \underbrace{\sqrt{\frac{1}{N}\sum_{j=1}^{N}(SOC_{exp}(j) - SOC_{n,sim}(\Theta, j))^2}}_{J^{SOC_n}(\Theta)}
\end{aligned} \tag{11}$$

where $\Theta$ refers to a vector of parameters to identify, $j$ is the time index, $N$ is the total number of data points, $V_{exp}$ is the experimental cell voltage, and $V_{sim}$ is the simulated cell voltage, $SOC_{exp}$ is the experimental SOC calculated from Coulomb counting, and $SOC_p$ and $SOC_n$ are the simulated SOC in the positive and negative electrodes, respectively.

To solve the optimization problem in Eq. (9), COBRAPRO implements PSO using parallel computing. PSO is a derivative-free, population-based optimization method, well-suited to find the global optimum of a high-dimensional nonlinear optimization problem.[102] The optimization process is accelerated through the use of MATLAB's Parallel Computing Toolbox, which facilitates the simultaneous simulation of multiple particles.

It is worth noting that various optimization methods have been used in the literature to perform parameter optimization. Forman et al.,[52] Zhang et al.,[41] and Rajabloo et al.[103] have used genetic algorithm (GA) while Masoudi et al.[104] used homotopy optimization and Santhanagopalan et al.[105] used Levenberg-Marquardt (LM) optimization. Kim et al.[106] proposed a deep Bayesian neural network to identify parameters that requires fewer model simulations compared to methods such as PSO, GA, and LM.

***Step 3. Practical identifiability.***—In this step, the practical identifiability of the remaining parameters, $\tilde{\Theta}_{HPPC} = \{R_c, D_{s,p}, D_{s,n}, k_p, k_n, \kappa, D, \sigma_p, \sigma_n\}$, is investigated through LSA and correlation analysis to determine a subset of identifiable parameters given the HPPC profile. The identifiable parameters are then calibrated using COBRAPRO's optimization framework using HPPC data. Refer to Appendix B for a detailed description of the LSA and correlation analysis implementation.

Given the sensitivity indices and correlation matrix as illustrated in Appendix B, Algorithm 1 identifies a subset of practically identifiable parameters using a predefined correlation coefficient threshold ($\beta$).[42] Two parameters are considered correlated if the correlation coefficient exceeds $\beta$.

**Algorithm 1.** Identify non-correlated parameters $\Theta_{HPPC}^{\beta Corr}$ [42]

1: **Input:** $\tilde{\Theta}_{HPPC}$ (vector containing parameters to analyze), $S^{V,SOC}$ (vector containing sensitivity index for each parameter in $\tilde{\Theta}_{HPPC}$), $\boldsymbol{C}$ (correlation coefficient matrix), $\beta$ (correlation coefficient threshold)
2: **Output:** $\Theta_{HPPC}^{\beta Corr}$
3: Initialize $\Theta_{HPPC}^{\beta Corr} \rightarrow \tilde{\Theta}_{HPPC}$
4: Sort $\Theta_{HPPC}^{\beta Corr}$ based on $S^{V,SOC}$ from highest to lowest sensitivity
5: **for** each highest sensitivity parameter $\theta_{highSens}$ in $\Theta_{HPPC}^{\beta Corr}$ **do**
6:  **for** each parameter $\theta$ in $\Theta_{HPPC}^{\beta Corr}$ **do**
7:  **if** $\theta \neq \theta_{highSens}$ **then**
8:  $C \rightarrow \boldsymbol{C}[\theta_{highSens}, \theta]$
9:  **if** $C > \beta$ **then**
10:  Remove $\theta$ from $\Theta_{HPPC}^{\beta Corr}$
11:  **end if**
12:  **end if**
13:  **end for**
14: **end for**
15: **return** $\Theta_{HPPC}^{\beta Corr}$

First, the algorithm takes the initial set of parameters, $\tilde{\Theta}_{HPPC}$, and their sensitivity indices, then sorts these parameters from highest to lowest sensitivity to initialize $\Theta_{HPPC}^{\beta Corr}$. Starting with the parameter with the highest sensitivity, the algorithm iterates through the other parameters to check the correlation coefficient between the highest sensitivity parameter and other parameters. If the correlation coefficient exceeds $\beta$, that parameter is removed from $\Theta_{HPPC}^{\beta Corr}$. The process is repeated until all parameters in $\Theta_{HPPC}^{\beta Corr}$ are evaluated, resulting in a final set of non-correlated parameters prioritized by their sensitivity, $\Theta_{HPPC}^{\beta Corr}$.

*Parameter optimization.*—After conducting practical identifiability analysis and identifying a subset of parameters $\Theta_{HPPC}^{\beta Corr}$ that can be effectively estimated, the HPPC data is then used to optimize the subset $\Theta_{HPPC}^{\beta Corr}$. As demonstrated in the stoichiometric identification problem, the multi-objective cost function proposed in Refs. 42 and 89 is leveraged to formulate the following optimization problem:

$$\begin{aligned}
&\underset{\Theta_{HPPC}^{\beta Corr}}{\text{minimize}}\ J(\Theta_{HPPC}^{\beta Corr}) = J^{V}(\Theta_{HPPC}^{\beta Corr}) + J^{SOC_p}(\Theta_{HPPC}^{\beta Corr}) + J^{SOC_n}(\Theta_{HPPC}^{\beta Corr}) \\
&\text{subject to (a) Discretized equations in Tables A·1, A·2, A·3, A·4} \\
&\qquad \text{(b) } \Theta_{HPPC}^{lb} \leqslant \Theta_{HPPC}^{\beta Corr} \leqslant \Theta_{HPPC}^{ub}, \\
&\qquad \text{(c) } V_{sim} \geqslant V_{lower}^{cutoff}, \\
&\qquad \text{(d) } (1-\gamma) Q_{dis,exp}^{HPPC} \leqslant Q_{dis,sim}^{HPPC} \leqslant (1+\gamma) Q_{dis,exp}^{HPPC}, \\
&\qquad \text{(e) } c_{s,i}^{surf} \leqslant c_{s,i}^{\max}, \quad i \in (p, n),
\end{aligned} \tag{12}$$

where $\gamma = 0.01$, $\Theta_{HPPC}^{lb}$ and $\Theta_{HPPC}^{ub}$ refer to the lower and upper bounds of $\Theta_{HPPC}^{\beta Corr}$, and $Q_{dis,sim}^{HPPC}$ and $Q_{dis,exp}^{HPPC}$ refer to the discharged capacity calculated from the HPPC simulation results and the HPPC experimental data. Similar to the reasoning provided for the $\Theta_{C/20}$ optimization problem, constraint (d) enforces charge conservation and constraint (e) ensures that $j_{0,i}$ is a positive number.

To solve the optimization problem in Eq. (12), the same PSO framework using parallel computing is employed as in the stoichiometric coefficient identification problem.

## Results

***Model verification.***—To verify COBRAPRO's implementation of the DFN model, the simulation results are compared against the results from COMSOL Multiphysics® at 23 °C. Figure 4a shows a comparison of the voltage curves at C/3, 1C, and 3C discharge and Fig. 4b shows a comparison of the electrolyte concentration in the positive electrode, separator, and negative electrode during 1C discharge at $t = 0$, 1800, 3600 s. The legends "COBRAPRO: FVM" and "COBRAPRO: FDMcs" refer to the FVM and FDM discretizations of $c_s$, respectively. Our simulation results match well with COMSOL and confirms the correct implementation of our model. For the purpose of comparing our results with COMSOL, both models were supplied with the LG INR21700-M50 cell parameters obtained from Ref. 38 (Appendix C). In the **Parameter Identification** section, experimental validation is provided using parameters calibrated through the parameter identification pipeline in COBRAPRO.

***Solver computation speed.***—COBRAPRO implements the open-source packages, CasADi[77] and SUNDIALS,[73,74] to achieve fast computation speed comparable to that of PyBaMM and

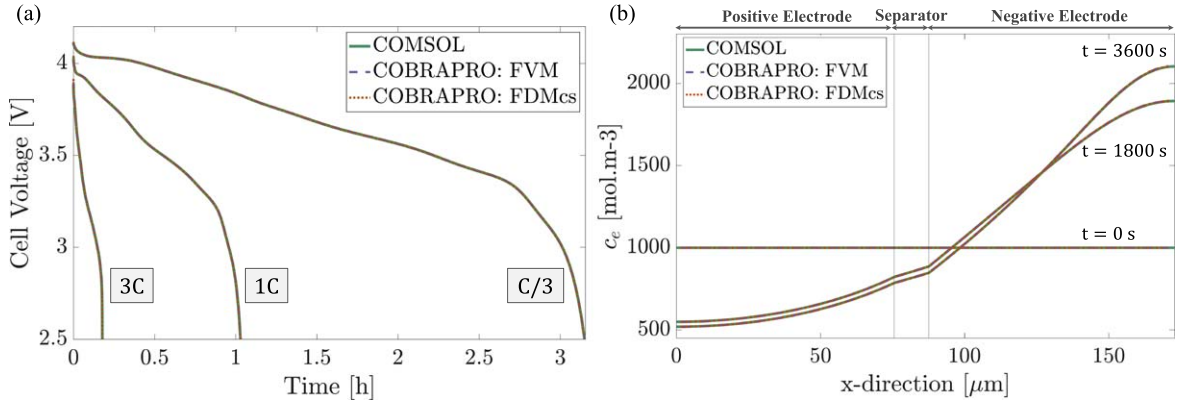


**Figure 4.** Verification of simulation results with COMSOL. (a) Cell voltage comparison at C/3, 1C, and 3C in discharge where COBRAPRO: FVM and COBRAPRO: FDMcs refer to the FVM and FDM discretizations of the solid particle concentration, respectively. (b) Electrolyte concentration at $t = 0$, 1800, 3600 s during 1C discharge.

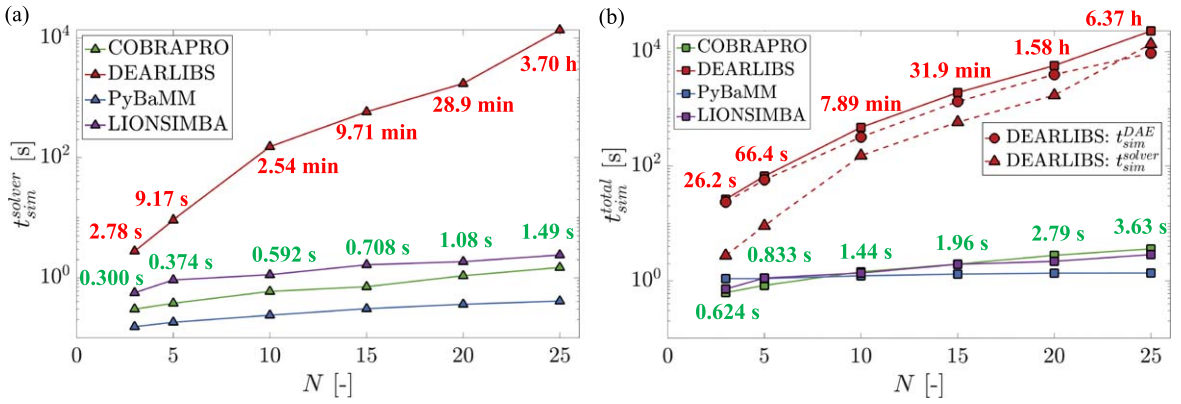


**Figure 5.** Computation speed comparison at 1C discharge for (a) DAE solver time ($t_{sim}^{solver}$) and (b) total computation time ($t_{sim}^{total}$). $N$ refers to the CVs along the x-direction in each domain, such that $N = N_p = N_s = N_n$. The radial discrete nodes were set to $N_{rp} = N_{rn} = 10$ for all simulations. In (b), the dashed red lines with circular and triangular markers denote the DAE formulation time ($t_{sim}^{DAE}$) and $t_{sim}^{solver}$ in DEARLIBS, respectively.

LIONSIMBA. CasADi's AD is used to convert the AEs to implicit ODEs in the single-step approach implemented in COBRAPRO. CasADi's AD is also used to compute the symbolic Jacobian, which helps to improve the computation speed compared to numerically approximating the Jacobian.[30] In contrast, DEARLIBS uses the MATLAB Symbolic Math Toolbox to formulate the DAE equations and ode15s solver to solve the DAE system in MATLAB symbolic form, leading to significantly longer computation times.

Figure 5 shows a comparison of the computation time of COBRAPRO, DEARLIBS, PyBaMM, and LIONSIMBA at 1C discharge for different number of CVs along the x-direction. Results are generated using Fick's law, which was implemented into DEARLIBS since the original version only included the polynomial approximation (Table I). For the COBRAPRO results presented, the solid particle concentration is discretized using FVM. In this analysis, $N$ refers to the number of CVs in each macroscale domain (positive and negative electrodes, separator) such that $N = N_p = N_s = N_n$. For this analysis, the number of CVs along the radial direction in the particles was kept constant at $N_{rp} = N_{rn} = 10$ across all simulations to maintain manageable simulation times[c] in DEARLIBS.

Figure 5a shows the DAE solver computation times ($t_{sim}^{solver}$) in logarithmic scale. The computation times for DEARLIBS and COBRAPRO are labeled in red and green, respectively. As $N$ increases, DEARLIBS experiences a significant increase in computation time, becoming impractically long due to limitations associated with the MATLAB's ode15s solver. In contrast, COBRAPRO exhibits a substantial improvement in DAE solver speed, aligning with the solver efficiencies of LIONSIMBA and PyBaMM. Compared to DEARLIBS, COBRAPRO's $t_{sim}^{solver}$ is two orders of magnitude (≈257 times) faster at $N = 10$ and three orders of magnitude (≈1608 times) faster at $N = 20$. PyBaMM consistently achieves the fastest DAE solver time across all $N$.

Figure 5b shows the total computation time ($t_{sim}^{total}$) on logarithmic scale. The $t_{sim}^{total}$ includes the total time required to execute the code, including the formulation of DAE equations ($t_{sim}^{DAE}$), determination of consistent initial conditions, and solving the DAE ($t_{sim}^{solver}$). For DEARLIBS, the significantly high $t_{sim}^{total}$ is attributed to not only $t_{sim}^{solver}$ but also $t_{sim}^{DAE}$, as depicted by the dashed red line with circle markers. This indicates that DEARLIBS' total computation time is highly impacted by the inefficiency of setting up the DAE equations through symbolic differentiation using MATLAB Symbolic Math Toolbox. In contrast, COBRAPRO's $t_{sim}^{total}$ is up to three orders of magnitude faster than that of DEARLIBS, showing comparable times to those of LIONSIMBA and PyBaMM. Notably, PyBaMM maintains a relatively constant $t_{sim}^{total}$ with increasing $N$, showing the lowest $t_{sim}^{total}$ among the compared codes at higher $N$.

***Comparison of initialization methods.***—Figure 6 shows a comparative analysis between the SUNDIALS IDACalCIC and single-step approach across various C-rates in discharging and different numbers of discretization points using parameters from Ref. 38. The $y$-axis displays the number of nodes tested, denoted as $N$ where $N = N_p = N_s = N_n = N_{rp} = N_{rn}$, and the $x$-axis denotes the C-rates from 1C to 6C in 0.1C increments. In the single-step method, the perturbation coefficient was set to $\alpha = 0.01$ while $t_{init} = 5$ s and $q = 1000$[d].

As shown in Fig. 6a, IDACalcIC is unable to determine consistent initial conditions at 1.7C across all $N$ values, with more failures observed at $N = 5$. For the single-step method, consistent initial conditions were determined for all C-rate and node numbers tested, as shown in Fig. 6b. This analysis demonstrates that the single-step approach is more robust than IDACalcIC at various C-rate and node number configurations.

***Parameter identification.***—The proposed parameter identification pipeline is demonstrated using experimental data obtained from LG INR21700-M50T cells, sourced from a publicly available dataset.[82] This dataset includes periodic RPTs, which are standardized tests designed to evaluate battery performance. The RPTs consist of the C/20 capacity test, HPPC, and EIS. Out of the ten cells tested in this dataset, cell W8 is used for this analysis. The first RPT, conducted on the fresh cell, is utilized for the identification process. Leveraging the use of RPTs for parameter identification could eliminate the need for conducting extra tests solely for parameter identification purposes.

As shown in Fig. 3, the first step of model parameterization consists of removing redundant parameters $\Theta_{redundant} = \{A_{cell}, l_p, l_s, l_n, \varepsilon_p, \varepsilon_s, \varepsilon_n, R_p, R_n, c_0, t_+, brugg_p, brugg_s, brugg_n\}$ by fixing them to their experimentally measured values reported in Ref. 38. The experimentally determined parameter values measured from cell tear-down are listed in Table C1.

*Stoichiometric coefficient identification.*—After the redundant parameters are fixed to their cell tear-down values, the C/20 capacity test is used to identify the stoichiometric parameters

[c]The total number of differential and algebraic equations in the DFN model is equal to the size of the DAE system. There are $N_{rp} \times N_p + N_{rn} \times N_n$ equations for the $c_{s,i}$ variable, $N_p + N_s + N_n$ equations for the $c_e$ variable, $N_p + N_s + N_n$ equations for the $\phi_e$ variable, $N_p + N_n$ equations for the $\phi_{s,i}$ variable, and $N_p + N_n$ for the $j_i$ variable. In summary, there are a total of $(N_{rp} + 4) \times N_p + (N_{rn} + 4) \times N_n + 2 \times N_s$ equations.

[d]Note that the perturbation coefficient $\alpha = 0.01$ was used, which yielded the highest initialization success rate while maintaining solution accuracy. We also confirmed that $t_{init} = 5$ s is sufficiently long enough for the solution to initialize given $\alpha = 0.01$. Furthermore, adjusting $q$ from $q = 1000$ and $q = 100$ did not affect the results for the determination of consistent initial conditions.

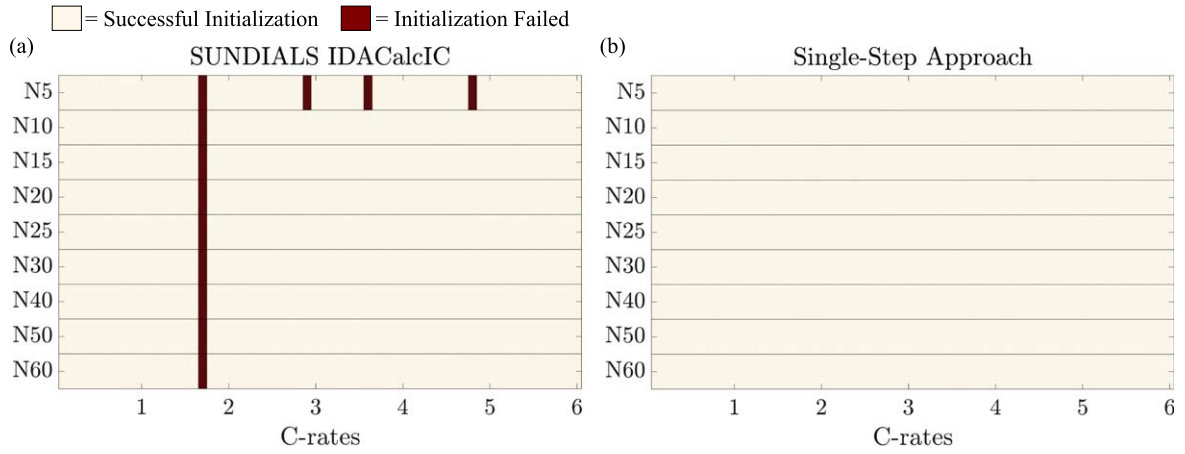


**Figure 6.** Comparison of SUNDIALS IDACalcIC and single-step approach[35] using parameters from Ref. 38. *Y*-axis shows the number of nodes tested where $N = N_p = N_s = N_n = N_{rp} = N_{rn}$ and the *x*-axis shows C-rates ranging from 1C to 6C in discharge at every 0.1C increments. (a) SUNDIALS IDACalcIC, (b) single-step approach with $\alpha = 0.01$.

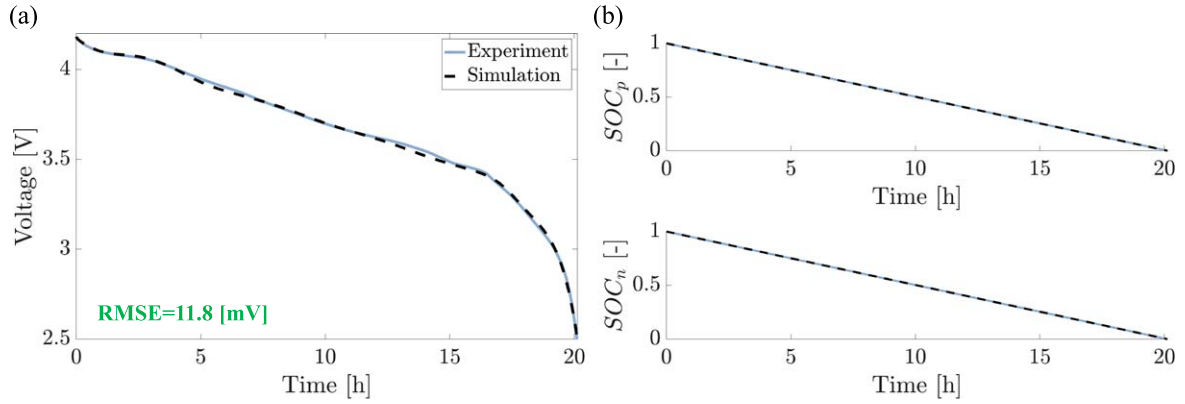


**Figure 7.** C/20 capacity test identification results.

**Table VI. Stoichiometric coefficient identification using C/20 discharge data.**

| Parameter | Unit | Lower bound | Upper bound | Initial guess | Identified value |
|---|---|---|---|---|---|
| $\theta_p^{100\%}$ | — | 0.22 | 0.34 | 0.27 | 0.2647 |
| $\theta_n^{100\%}$ | — | 0.7 | 1 | 0.9014 | 0.7784 |
| $\theta_p^{0\%}$ | — | 0.7 | 1 | 0.9084 | 0.8939 |
| $\theta_n^{0\%}$ | — | 0.015 | 0.04 | 0.0279 | 0.02982 |
| $c_{s,p}^{\max}$ [a] | mol/m$^3$ | — | — | — | 55492 |
| $c_{s,n}^{\max}$ [a] | mol/m$^3$ | — | — | — | 36690 |

a) Parameter not identified.

**Table VII. Identifiable parameter sets for given correlation index threshold values.**

| Identifiable Parameters | Correlation Coefficient Threshold |
|---|---|
| $\Theta_{HPPC}^{0.8Corr} = \{R_c, D_{s,p}, D_{s,n}\}$ | $\beta = 0.8$ |
| $\Theta_{HPPC}^{0.9Corr} = \{R_c, D_{s,p}, D_{s,n}\}$ | $\beta = 0.9$ |
| $\Theta_{HPPC}^{0.95Corr} = \{R_c, D_{s,p}, D_{s,n}, D\}$ | $\beta = 0.95$ |
| $\Theta_{HPPC}^{0.99Corr} = \{R_c, D_{s,p}, D_{s,n}, D, k_p\}$ | $\beta = 0.99$ |

$\Theta_{C/20} = \{\theta_p^{100\%}, \theta_p^{0\%}, \theta_n^{100\%}, \theta_n^{0\%}\}$. The stoichiometric parameters are identified using the optimization problem formulated in Eq. (9).

Table VI shows the initial guesses and the upper and lower bounds of the stoichiometric parameters. The initial guesses are obtained from the experimentally determined values from cell teardown reported in Ref. 38.

The identification results are shown in Fig. 7 with the simulation results and experimental data exhibiting a good match with a $J^V$ of $3.30 \times 10^{-3}$ (11.8 mV), $J^{SOC_p}$ of $3.02 \times 10^{-4}$, and $J^{SOC_n}$ of $1.09 \times 10^{-4}$ (Table VIII). Table VI reports the identified stoichiometric values and the resulting maximum concentration values determined from the constraints (f) and (g) in Eq. (12).

*HPPC identification.*—In this section, the practical identifiability of the remaining unknown parameters, $\tilde{\Theta}_{HPPC} = \{R_c, D_{s,p}, D_{s,n}, k_p, k_n, \kappa, D, \sigma_p, \sigma_n\}$, is assessed with respect to the HPPC profile to yield the identifiable parameter set $\Theta_{HPPC}^{\beta Corr}$. Subsequently, parameter identification is conducted to optimize the parameters in $\Theta_{HPPC}^{\beta Corr}$ using HPPC data.

LSA and correlation analysis are conducted on the parameters in $\tilde{\Theta}_{HPPC}$ to assess the model output change due to local perturbations

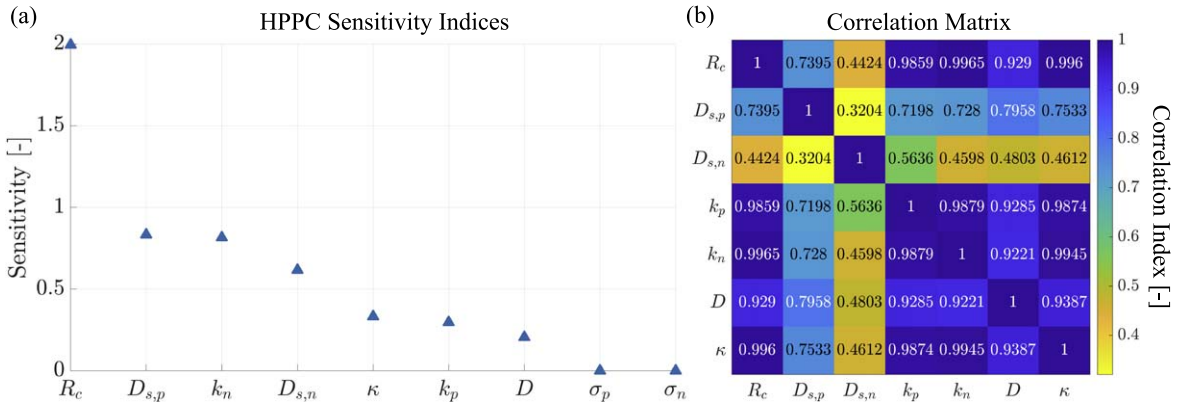


**Figure 8.** (a) Sensitivity indices of $\tilde{\Theta}_{HPPC}$ calculated using the HPPC profile and (b) correlation matrix. In the correlation analysis, $\sigma_p$ and $\sigma_n$ are removed due to their low sensitivity.

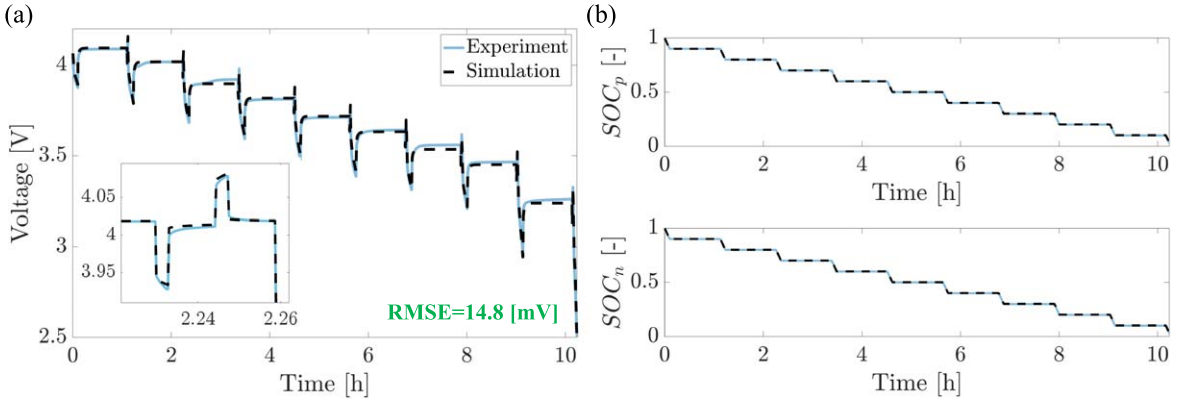


**Figure 9.** HPPC identification results for $\Theta_{HPPC}^{0.99Corr}$.

in each parameter and the degree of linear correlation between parameters. The sensitivity indices and correlation matrix for $\tilde{\Theta}_{HPPC}$ are shown in Fig. 8.

Figure 8a shows the sensitivity indices of $\tilde{\Theta}_{HPPC} = \{R_c, D_{s,p}, D_{s,n}, k_p, k_n, \kappa, D, \sigma_p, \sigma_n\}$ calculated from $S^{V,SOC}$. As expected, the sensitivities of $\sigma_p$ and $\sigma_n$ are nearly zero due to the high conductivity of the electrode compared to the electrolyte conductivity. Due to their low sensitivity, $\sigma_p$ and $\sigma_n$ are removed from the correlation analysis.

Algorithm 1 is implemented in COBRAPRO, which automatically generates a subset of uncorrelated parameters given a user-defined correlation coefficient threshold $\beta$. For demonstration purposes, the algorithm is illustrated with $\beta = 0.8$[42] using the results from Fig. 8. First, $\Theta_{HPPC}^{0.8Corr}$ is set to the initial parameter vector $\tilde{\Theta}_{HPPC}$. The parameter with the highest sensitivity in $\Theta_{HPPC}^{0.8Corr}$, in this case $R_c$, is analyzed first. As shown in Fig. 8b, the only parameters with correlation coefficients lower than 0.8 with $R_c$ are the solid phase diffusion coefficients $D_{s,p}$ and $D_{s,n}$. Therefore, all parameters

**Table VIII. Comparison of cost function values for HPPC identification and UDDS validation.**

| Parameters | Cost function | Identification | | Validation | Units |
|---|---|---|---|---|---|
| | | C/20 Discharge | HPPC | UDDS | |
| $\Theta_{HPPC}^{0.99Corr}$ | $J^V$ | 11.8 | 14.8 | 13.1 | [mV] |
| | $J^V$ | $3.30 \times 10^{-3}$ | $4.21 \times 10^{-3}$ | $3.60 \times 10^{-3}$ | [-] |
| | $J^{SOC_p}$ | $3.02 \times 10^{-4}$ | $1.33 \times 10^{-3}$ | $3.11 \times 10^{-4}$ | [-] |
| | $J^{SOC_n}$ | $1.90 \times 10^{-4}$ | $1.73 \times 10^{-3}$ | $1.62 \times 10^{-4}$ | [-] |
| $\Theta_{HPPC}^{0.95Corr}$ | $J^V$ | 11.8 | 16.1 | 13.7 | [mV] |
| | $J^V$ | $3.30 \times 10^{-3}$ | $4.97 \times 10^{-3}$ | $3.74 \times 10^{-3}$ | [-] |
| | $J^{SOC_p}$ | $3.02 \times 10^{-4}$ | $1.33 \times 10^{-3}$ | $3.20 \times 10^{-4}$ | [-] |
| | $J^{SOC_n}$ | $1.90 \times 10^{-4}$ | $1.73 \times 10^{-3}$ | $1.56 \times 10^{-4}$ | [-] |
| $\Theta_{HPPC}^{0.9Corr}$ | $J^V$ | 11.8 | 14.5 | 15.0 | [mV] |
| | $J^V$ | $3.30 \times 10^{-3}$ | $4.14 \times 10^{-3}$ | $4.11 \times 10^{-3}$ | [-] |
| | $J^{SOC_p}$ | $3.02 \times 10^{-4}$ | $1.33 \times 10^{-3}$ | $3.16 \times 10^{-4}$ | [-] |
| | $J^{SOC_n}$ | $1.90 \times 10^{-4}$ | $1.73 \times 10^{-3}$ | $1.59 \times 10^{-4}$ | [-] |

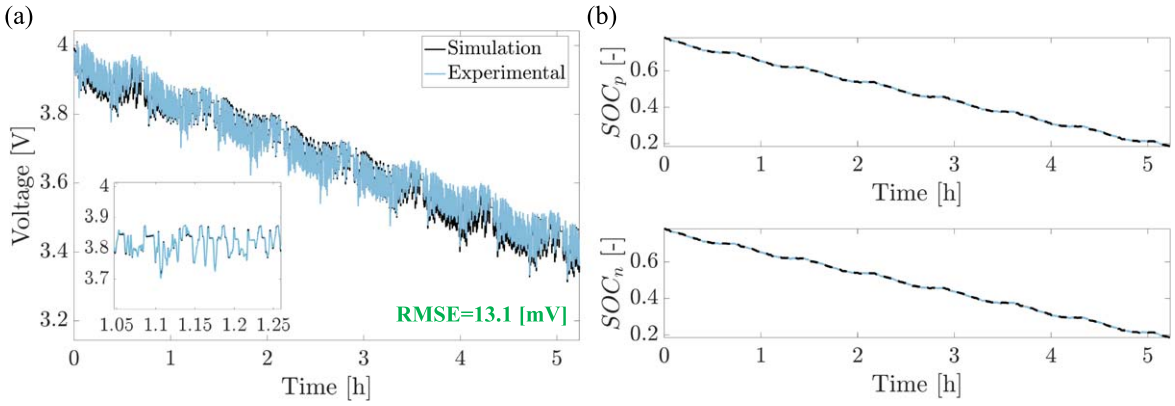


**Figure 10.** UDDS validation of identified parameters in $\Theta_{HPPC}^{0.99Corr}$.

**Table IX. Identification values for parameters in $\Theta_{HPPC}^{0.99Corr}$ using HPPC data.**

| Parameter | Unit | Lower bound | Upper bound | Initial guess | Identified value |
|---|---|---|---|---|---|
| $R_c$ | Ω | 0.001 | 0.0233 | 0.0233 | 0.0059 |
| $D_{s,p}$ | $m^2/s$ | $5.28 \times 10^{-18}$ | $3.03 \times 10^{-12}$ | $4.00 \times 10^{-15}$ | $3.03 \times 10^{-12}$ |
| $D_{s,n}$ | $m^2/s$ | $6.64 \times 10^{-17}$ | $1.64 \times 10^{-11}$ | $2.42 \times 10^{-14}$ | $2.42 \times 10^{-14}$ |
| $k_p$ | $m^{2.5}/(mol^{0.5} \cdot s)$ | $2.60 \times 10^{-14}$ | $4.89 \times 10^{-8}$ | $3.54 \times 10^{-11}$ | $2.18 \times 10^{-8}$ |
| $D$ | $m^2/s$ | $4.90 \times 10^{-12}$ | $2.89 \times 10^{-8}$ | $3.76 \times 10^{-10}$ | $7.62 \times 10^{-11}$ |

except $D_{s,p}$ and $D_{s,n}$ are removed from $\Theta_{HPPC}^{0.8Corr}$, resulting in the new vector $\Theta_{HPPC}^{0.8Corr} = \{R_c, D_{s,p}, D_{s,n}\}$. The parameter with the next highest sensitivity in $\Theta_{HPPC}^{0.8Corr}$, $D_{s,p}$, is analyzed next. Since $D_{s,p}$ and $D_{s,n}$ have correlation coefficients less than 0.8, $D_{s,n}$ is not removed from $\Theta_{HPPC}^{\beta Corr}$. Since all parameters in $\Theta_{HPPC}^{\beta Corr}$ have been analyzed, the algorithm concludes, and the identifiable parameter set for $\beta = 0.8$ is determined as $\Theta_{HPPC}^{\beta Corr} = \{R_c, D_{s,p}, D_{s,n}\}$.

Table VII shows the identifiable set of parameters determined for different $\beta$ values. The $\beta$ can be considered as a design variable that will depend on the input current and parameters being studied. In this analysis, thresholds of 0.8 and 0.9 resulted in the same set of identifiable parameters. During parameter identification, the parameters in $\Theta_{HPPC}^{\beta Corr}$ are identified, while the remaining parameters in $\tilde{\Theta}_{HPPC}$ that are not included in $\Theta_{HPPC}^{\beta Corr}$ are fixed to their experimentally measured values shown in Table C1 of Appendix C.

Among the parameter sets $\Theta_{HPPC}^{0.9Corr}$, $\Theta_{HPPC}^{0.95Corr}$, and $\Theta_{HPPC}^{0.99Corr}$, the set $\Theta_{HPPC}^{0.99Corr} = \{R_c, D_{s,p}, D_{s,n}, D, k_p\}$ resulted in the lowest RMSE for the UDDS validation results as shown in Table VIII. Table IX shows the initial guesses, lower bounds, and upper bounds for the parameters in $\Theta_{HPPC}^{0.99Corr}$. The initial guesses are obtained from the measured values from cell tear-down, with the initial guess for $R_c$ obtained from averaging the ohmic resistances extracted from EIS data at 20%, 50%, and 80% SOC.[45] The HPPC identification results for $\Theta_{HPPC}^{0.99Corr}$ are shown in Fig. 9 with the simulation and experimental results matching well with a $J^V$ of $4.21 \times 10^{-3}$ (14.8 mV), $J^{SOC_p}$ of $1.33 \times 10^{-3}$, and $J^{SOC_n}$ of $1.73 \times 10^{-3}$. The UDDS validation results shown in Fig. 10 demonstrate that the identified parameters can accurately predict cell voltage and SOC in the electrodes, with cost function values of $J^V$ of $3.60 \times 10^{-3}$ (13.1 mV), $J^{SOC_p}$ of $3.11 \times 10^{-4}$, and $J^{SOC_n}$ of $1.62 \times 10^{-4}$.

## Conclusions

In this paper, we present COBRAPRO, a new open-source battery model simulation package, capable of parameterizing the DFN model using experimental current-voltage data. A parameter identification pipeline is implemented in COBRAPRO, allowing systematic identification and determination of parameters using structural and practical identifiability analysis, combined with a closed-loop optimization routine.

COBRAPRO's parameter identification framework is demonstrated on RPT data generated from LG INR21700-M50T cells. In the first step of DFN model parameterization, redundant parameters are determined and fixed to their cell tear-down values to improve the identifiability of the remaining parameters. The C/20 capacity test is used to identify the stoichiometric parameters while the maximum concentrations are determined through the theoretical cell capacity equation. Practical identifiability of the remaining parameters is evaluated for the HPPC profile, and a subset of identifiable parameters is selected for parameter identification using HPPC data. The identified parameters are validated against experimental UDDS data, showing a good match between the simulated and experimental voltage and SOC in the electrodes.

In addition to COBRAPRO's parameterization routine, a fast DAE solver and robust framework for determining consistent initial conditions are implemented to achieve computational efficiency and ensure reliable DAE initialization. When benchmarked against other open-source DFN simulators such as LIONSIMBA and PyBaMM, COBRAPRO demonstrates comparable computation speed. To determine consistent initial conditions, the single-step approach is implemented in COBRAPRO, building upon the methodology established in DEARLIBS. For various C-rates and node numbers, the single-step approach is shown to be more robust than the nonlinear AE solver, SUNDIALS IDACalcIC.

## Acknowledgments

The authors thank the Bits and Watts Initiative within the Precourt Institute for Energy at Stanford University for its partial financial support. We thank Dr. Le Xu for all the insightful discussions on DFN modeling and parameter identification that helped improve the contents of this paper.

## Appendix A. DFN State-space Equations

***FVM derivation example: electrolyte concentration, $c_e(x, t)$.***— The PDEs defined in Eqs. (2), (3), and (4) are spatially discretized along the x-direction using FVM. To derive the FVM equations, the governing equations are integrated over each CV. The derivation is demonstrated for the electrolyte mass conservation equation.

As shown in Fig. 2, consider a CV with a center at $x_k$ spanning the interval $[x_{k-1}, x_{k+1}]$ with a length of $\Delta x_i = l_i/N_i$ for $i \in (p, s, n)$. The integral of Eq. (3) is evaluated from $x_{k-1}$ to $x_{k+1}$, resulting in

$$\Delta x_i \varepsilon_i \frac{d\bar{c}_{e,i,k}}{dt} = \left[ D^{eff}_{i,k+\frac{1}{2}} \frac{dc_{e,i,k+\frac{1}{2}}}{dx} \right] - \left[ D^{eff}_{i,k-\frac{1}{2}} \frac{dc_{e,i,k-\frac{1}{2}}}{dx} \right] + \Delta x_i a_i (1 - t_{+,k}) j_{i,k} \quad [A{\cdot}1]$$

where $\bar{c}_{e,i,k}$ is the volume-averaged value of $c_{e,i}$ for the CV centered at index $k$, given $k \in [1, N_p]$ for the positive electrode ($i = p$), $k \in [N_p + 1, N_{ps}]$ for the separator ($i = s$), and $k \in [N_{ps} + 1, N_{psn}]$ for the negative electrode ($i = n$). For conciseness, we define $N_{ps} = N_p + N_s$ and $N_{psn} = N_p + N_s + N_n$.

Using the central difference method to approximate the flux at $x_{k-1}$ and $x_{k+1}$

$$\frac{dc_{e,i,k+\frac{1}{2}}}{dx} \approx \frac{\bar{c}_{e,i,k+1} - \bar{c}_{e,i,k}}{\Delta x_i} \quad [A{\cdot}2]$$

$$\frac{dc_{e,i,k-\frac{1}{2}}}{dx} \approx \frac{\bar{c}_{e,i,k} - \bar{c}_{e,i,k-1}}{\Delta x_i} \quad [A{\cdot}3]$$

and substituting into Eq. (A·1) yields the FVM equations for the mass conservation in the electrolyte phase as follows

$$\Delta x_i \varepsilon_i \frac{d\bar{c}_{e,i,k}}{dt} = \left[ D^{eff}_{i,k+\frac{1}{2}} \frac{\bar{c}_{e,i,k+1} - \bar{c}_{e,i,k}}{\Delta x_i} \right] - \left[ D^{eff}_{i,k-\frac{1}{2}} \frac{\bar{c}_{e,i,k} - \bar{c}_{e,i,k-1}}{\Delta x_i} \right] + \Delta x_i a_i (1 - t_{+,k}) j_{i,k}, \quad i \in (p, s, n). \quad [A{\cdot}4]$$

The harmonic mean is used to calculate the effective diffusion coefficient $D_i^{eff}$ at the edges of the CVs as

$$D^{eff}_{i,k\pm\frac{1}{2}} = \frac{D^{eff}_{i,k\pm1} D^{eff}_{i,k}}{\xi_i D^{eff}_{i,k\pm1} + (1 - \xi_i) D^{eff}_{i,k}}, \quad i \in (p, s, n) \quad [A{\cdot}5]$$

where $\xi_i = \dfrac{\Delta x_{i,k\pm1}}{\Delta x_{i,k\pm1} + \Delta x_{i,k}}$.

The first and last CVs in each domain are expressed using the boundary conditions listed in Eq. (3). For the first CV at $k = 1$ in the positive electrode domain, the zero $c_e$ flux condition on the interface between the positive electrode and current collector allows us to rewrite Eq. (A·4) as

$$\Delta x_p \varepsilon_p \frac{d\bar{c}_{e,p,1}}{dt} = \left[ D^{eff}_{p,\frac{3}{2}} \frac{\bar{c}_{e,p,2} - \bar{c}_{e,p,1}}{\Delta x_p} \right] + \Delta x_p a_p (1 - t_{+,1}) j_{p,1} \quad [A{\cdot}6]$$

while the flux continuity boundary condition between the positive electrode and separator allows us to rewrite Eq. (A·4) at $k = N_p$ as

$$\Delta x_p \varepsilon_p \frac{dc_{e,p,N_p}}{dt} = \left[ D^{eff}_{p,N_p+\frac{1}{2}} \frac{\bar{c}_{e,p,N_p+1} - \bar{c}_{e,p,N_p}}{\Delta x_{\overline{ps}}} \right] - \left[ D^{eff}_{p,N_p-\frac{1}{2}} \frac{\bar{c}_{e,p,N_p} - \bar{c}_{e,p,N_p-1}}{\Delta x_p} \right] + \Delta x_p a_p (1 - t_{+,N_p}) j_{p,N_p} \quad [A{\cdot}7]$$

where $x_{\overline{ps}} = \dfrac{\Delta x_p + \Delta x_s}{2}$ since $D^{eff}_{p,N_p+\frac{1}{2}}$ refers to the interface between the last CV in the positive electrode ($k = N_p$) and the first CV in the separator ($k = N_p + 1$).

The discretized states are given in a vector as

$$\boldsymbol{c_e} = \begin{bmatrix} \boldsymbol{c_{e,p}} \\ \boldsymbol{c_{e,s}} \\ \boldsymbol{c_{e,n}} \end{bmatrix}_{N_{psn}\times 1}$$

$$\boldsymbol{c_{e,p}} = \begin{bmatrix} \bar{c}_{e,p,1} \\ \bar{c}_{e,p,2} \\ \vdots \\ \bar{c}_{e,p,N_p} \end{bmatrix}_{N_p\times 1}, \quad \boldsymbol{c_{e,s}} = \begin{bmatrix} \bar{c}_{e,s,N_p+1} \\ \bar{c}_{e,s,N_p+2} \\ \vdots \\ \bar{c}_{e,s,N_{ps}} \end{bmatrix}_{N_s\times 1},$$

$$\boldsymbol{c_{e,n}} = \begin{bmatrix} \bar{c}_{e,n,N_{ps}+1} \\ \bar{c}_{e,n,N_{ps}+2} \\ \vdots \\ \bar{c}_{e,n,N_{psn}} \end{bmatrix}_{N_n\times 1} \quad [A{\cdot}8]$$

where $\boldsymbol{c_{e,p}}$ is the state vector in the positive electrode, $\boldsymbol{c_{e,s}}$ is the state vector in the separator, and $\boldsymbol{c_{e,n}}$ is state vector in the negative electrode. The state-space equations are expressed as

$$\dot{\boldsymbol{c}}_e = \mathcal{A}_{c_e} \boldsymbol{c_e} + \mathcal{B}_{c_e} \quad [A{\cdot}9]$$

where

$$\mathcal{A}_{c_e} = \frac{1}{\varepsilon_i \Delta x_i^2} \begin{bmatrix} -D^{eff}_{p,3/2} & D^{eff}_{p,3/2} & 0 & 0 & \dots & 0 \\ D^{eff}_{p,3/2} & -(D^{eff}_{p,3/2} + D^{eff}_{p,5/2}) & D^{eff}_{p,5/2} & 0 & \dots & 0 \\ 0 & D^{eff}_{p,5/2} & -(D^{eff}_{p,5/2} + D^{eff}_{p,7/2}) & D^{eff}_{p,5/2} & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & \dots & 0 & 0 & D^{eff}_{n,N_{psn}-1/2} & -D^{eff}_{n,N_{psn}-1/2} \end{bmatrix}_{N_{psn}\times N_{psn}} \quad [A{\cdot}10]$$

$$\mathcal{B}_{c_e} = \frac{a_i}{\varepsilon_i} \begin{bmatrix} \begin{bmatrix} (1 - t_{+,1}) j_{p,1} \\ \vdots \\ (1 - t_{+,N_p}) j_{p,N_p} \end{bmatrix}_{N_p\times 1} \\ [0]_{N_s\times 1} \\ \begin{bmatrix} (1 - t_{+,N_{ps}+1}) j_{N_{ps}+1} \\ \vdots \\ (1 - t_{+,N_{psn}}) j_{N_{psn}} \end{bmatrix}_{N_n\times 1} \end{bmatrix}_{N_{psn}\times 1}. \quad [A{\cdot}11]$$

**Table A·1. State-space representation for mass transport in electrolyte using FVM.**

Mass transport in the electrolyte $i \in (p, s, n)$

$$\dot{\boldsymbol{c}}_e = \mathcal{A}_{c_e}\boldsymbol{c_e} + \mathcal{B}_{c_e}, \quad \boldsymbol{c_e} = \begin{bmatrix} [\boldsymbol{c_{e,p}}]_{N_p\times 1} \\ [\boldsymbol{c_{e,s}}]_{N_s\times 1} \\ [\boldsymbol{c_{e,n}}]_{N_n\times 1} \end{bmatrix}_{N_{psn}\times 1}$$

$$\mathcal{A}_{c_e} = \frac{1}{\varepsilon_i \Delta x_i^2}\begin{bmatrix} -D^{eff}_{p,3/2} & D^{eff}_{p,3/2} & 0 & 0 & ... & 0 \\ D^{eff}_{p,3/2} & -(D^{eff}_{p,3/2} + D^{eff}_{p,5/2}) & D^{eff}_{p,5/2} & 0 & ... & 0 \\ 0 & D^{eff}_{p,5/2} & -(D^{eff}_{p,5/2} + D^{eff}_{p,7/2}) & D^{eff}_{p,5/2} & ... & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & ... & 0 & D^{eff}_{n,N_{psn}-3/2} & -(D^{eff}_{n,N_{psn}-3/2} + D^{eff}_{n,N_{psn}-1/2}) & D^{eff}_{n,N_{psn}-1/2} \\ 0 & ... & 0 & 0 & D^{eff}_{n,N_{psn}-1/2} & -D^{eff}_{n,N_{psn}-1/2} \end{bmatrix}_{N_{psn}\times N_{psn}}$$

$$\mathcal{B}_{c_e} = \frac{a_i}{\varepsilon_i}\begin{bmatrix} (1-t_{+,1})\bar{j}_{p,1} \\ \vdots \\ (1-t_{+,N_p})\bar{j}_{p,N_p} \\ [0]_{N_s\times N_s} \\ (1-t_{+,N_{ps}+1})\bar{j}_{n,N_{ps}+1} \\ \vdots \\ (1-t_{+,N_{psn}})\bar{j}_{n,N_{psn}} \end{bmatrix}_{N_{psn}\times 1}$$

$$D^{eff}_{i,k\pm 1/2} = \frac{D^{eff}_{i,k\pm 1} D^{eff}_{i,k}}{\xi_i D^{eff}_{i,k\pm 1} + (1-\xi_i) D^{eff}_{i,k}}, \quad \xi_i = \frac{\Delta x_{i,k\pm 1}}{\Delta x_{i,k\pm 1} + \Delta x_{i,k}}, \quad \Delta x_i = \frac{l_i}{N_i} \qquad [39]$$

Mass transport in the electrolyte—Interface equations

$$\dot{\boldsymbol{c}}_{\boldsymbol{e,bc}} = \mathcal{C}_{c_e}\boldsymbol{c_e} + \begin{bmatrix} \frac{a_p}{\varepsilon_p}(1-t_{+,N_p})\bar{j}_{N_p} \\ \frac{a_s}{\varepsilon_s}(1-t_{+,N_p+1})\bar{j}_{N_p+1} \\ \frac{a_s}{\varepsilon_s}(1-t_{+,N_{ps}})\bar{j}_{N_{ps}} \\ \frac{a_n}{\varepsilon_n}(1-t_{+,N_{ps}+1})\bar{j}_{N_{ps}+1} \end{bmatrix}_{4\times 1}, \boldsymbol{c_{e,bc}} = \begin{bmatrix} \bar{c}_{e,p,N_p} \\ \bar{c}_{e,s,N_p+1} \\ \bar{c}_{e,s,N_{ps}} \\ \bar{c}_{e,n,N_{ps}+1} \end{bmatrix}_{4\times 1}$$

$$\mathcal{C}_{c_e} = \begin{bmatrix} 0 & ... & \frac{D^{eff}_{p,N_p-1/2}}{\varepsilon_p \Delta x_p^2} & \gamma_1 & \frac{D^{eff}_{p,N_p+1/2}}{\varepsilon_p \Delta x_p \Delta x_{\overline{ps}}} & 0 & ... & 0 & 0 & 0 & ... & 0 \\ 0 & ... & 0 & \frac{D^{eff}_{s,N_s+1/2}}{\varepsilon_s \Delta x_s \Delta x_{\overline{ps}}} & \gamma_2 & \frac{D^{eff}_{s,N_p+3/2}}{\varepsilon_s \Delta x_s^2} & ... & 0 & 0 & 0 & ... & 0 \\ 0 & ... & 0 & 0 & 0 & ... & \frac{D^{eff}_{s,N_{ps}-1/2}}{\varepsilon_s \Delta x_s^2} & \gamma_3 & \frac{D^{eff}_{s,N_{ps}+1/2}}{\varepsilon_s \Delta x_s \Delta x_{\overline{sn}}} & 0 & ... & 0 \\ 0 & ... & 0 & 0 & 0 & ... & 0 & \frac{D^{eff}_{n,N_{ps}+1/2}}{\varepsilon_n \Delta x_n \Delta x_{\overline{sn}}} & \gamma_4 & \frac{D^{eff}_{n,N_{ps}+3/2}}{\varepsilon_n \Delta x_n^2} & ... & 0 \end{bmatrix}_{4\times N_{psn}}$$

$$x_{\overline{ps}} = \frac{\Delta x_p + \Delta x_s}{2}, \quad x_{\overline{sn}} = \frac{\Delta x_s + \Delta x_n}{2}$$

$$\gamma_1 = -\frac{1}{\varepsilon_p \Delta x_p}\left(\frac{D^{eff}_{p,N_p-1/2}}{\Delta x_p} + \frac{D^{eff}_{p,N_p+1/2}}{\Delta x_{\overline{ps}}}\right)$$

$$\gamma_2 = -\frac{1}{\varepsilon_s \Delta x_s}\left(\frac{D^{eff}_{s,N_p+1/2}}{\Delta x_{\overline{ps}}} + \frac{D^{eff}_{s,N_p+3/2}}{\Delta x_s}\right)$$

$$\gamma_3 = -\frac{1}{\varepsilon_s \Delta x_s}\left(\frac{D^{eff}_{s,N_{ps}-1/2}}{\Delta x_s} + \frac{D^{eff}_{s,N_{ps}+1/2}}{\Delta x_{\overline{sn}}}\right)$$

$$\gamma_4 = -\frac{1}{\varepsilon_n \Delta x_n}\left(\frac{D^{eff}_{n,N_{ps}+1/2}}{\Delta x_{\overline{sn}}} + \frac{D^{eff}_{n,N_{ps}+3/2}}{\Delta x_n}\right) \qquad [40]$$

**Table A·2. State-space representation for mass transport in solid particles with FDM and FVM.**

FDM: Mass conservation in solid particles $i \in (p, n)$

$$\dot{\boldsymbol{c}}_{\boldsymbol{s},\boldsymbol{i}} = \mathcal{A}_{c_{s,i}}\boldsymbol{c}_{\boldsymbol{s},\boldsymbol{i}} + \mathcal{B}_{c_{s,i}},\ \boldsymbol{c}_{\boldsymbol{s},\boldsymbol{i}} = \begin{bmatrix} c_{s,i,1} \\ \vdots \\ c_{s,i,N_{ri}} \end{bmatrix}_{N_{ri}\times 1},\ \Delta r_i = \frac{R_i}{N_{ri}}$$

$$\mathcal{A}_{c_{s,i}} = \frac{D_{s,i}}{\Delta r_i^2}\begin{bmatrix} -2 & 2 & 0 & 0 & \dots & 0 \\ \frac{1}{2} & -2 & \frac{3}{2} & 0 & \dots & 0 \\ 0 & \frac{2}{3} & -2 & \frac{4}{3} & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & \dots & 0 & \frac{N_{ri}-3}{N_{ri}-2} & -2 & \frac{N_{ri}}{N_{ri}-1}, \\ 0 & 0 & \dots & 0 & 2 & -2 \end{bmatrix}_{N_{ri}\times N_{ri}},\quad \mathcal{B}_{c_{s,i}} = -\frac{\Delta r_i j_i}{D_{s,i}}\begin{bmatrix} 0 \\ \vdots \\ 0 \\ \frac{2N_{ri}+2}{N_{ri}} \end{bmatrix}_{N_{ri}\times 1} \qquad [41]$$

FVM: Mass conservation in solid particles $i \in (p, n)$[89]

$$\dot{\boldsymbol{c}}_{\boldsymbol{s},\boldsymbol{i}} = \mathcal{A}_{c_{s,i}}\boldsymbol{c}_{\boldsymbol{s},\boldsymbol{i}} + \mathcal{B}_{c_{s,i}},\quad \boldsymbol{c}_{\boldsymbol{s},\boldsymbol{i}} = \begin{bmatrix} \bar{c}_{s,i,1} \\ \vdots \\ \bar{c}_{s,i,N_{ri}} \end{bmatrix}_{N_{ri}\times 1},\quad \Delta r_i = \frac{R_i}{N_{ri}}$$

$$\mathcal{A}_{c_{s,i}} = \frac{4D_{s,i}}{\Delta r_i^2}\begin{bmatrix} -\frac{3}{4} & \frac{3}{4} & 0 & 0 & \cdots & 0 \\ \frac{3}{28} & -\frac{15}{28} & \frac{12}{28} & 0 & \cdots & 0 \\ 0 & \frac{12}{76} & -\frac{39}{76} & \frac{27}{76} & \cdots & 0 \\ 0 & 0 & \frac{27}{148} & -\frac{75}{148} & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & \frac{3(N_{ri}-1)^2}{12N_{ri}^2-12N_{ri}+4} & -\frac{3(N_{ri}-1)^2}{12N_{ri}^2-12N_{ri}+4} \end{bmatrix}_{N_{ri}\times N_{ri}},$$

$$\mathcal{B}_{c_{s,i}} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ -\frac{3N_{ri}^2\bar{j}_i}{\Delta r_i(3N_{ri}^2-3N_{ri}+1)} \end{bmatrix}_{N_{ri}\times 1}$$

Surface Concentration using Third-order Hermite Interpolation

$$c_{s,i}^{surf} = \left[\left(1+2\frac{R_i-r_{N_{ri}-1}}{r_{N_{ri}}-r_{N_{ri}-1}}\right)\bar{c}_{s,i,N_{ri}-1} + (R_i-r_{N_{ri}-1})\frac{\partial\bar{c}_{s,i,N_{ri}-1}}{\partial r}\right]\left(\frac{R_i-r_{N_{ri}}}{r_{N_{ri}-1}-r_{N_{ri}}}\right)^2$$
$$+\left[\left(1+2\frac{r_{N_{ri}}-R_i}{r_{N_{ri}}-r_{N_{ri}-1}}\right)\bar{c}_{s,i,N_{ri}} + (R_i-r_{N_{ri}})\frac{\partial\bar{c}_{s,i,N_{ri}}}{\partial r}\right]\left(\frac{R_i-r_{N_{ri}-1}}{r_{N_{ri}}-r_{N_{ri}-1}}\right)^2$$

$$r_{N_{ri}-1} = \Delta r_i(N_{ri}-1.5),\quad r_{N_{ri}} = \Delta r_i(N_{ri}-0.5)$$

$$\bar{c}_{s,i,N_{ri}+1} = \bar{c}_{s,i,N_{ri}} - \frac{\Delta r_i\bar{j}_i}{D_{s,i}}\ \text{(ghost node)}$$

$$\frac{\partial\bar{c}_{s,i,N_{ri}-1}}{\partial r} = \frac{\bar{c}_{s,i,N_{ri}}-\bar{c}_{s,i,N_{ri}-2}}{2\Delta r_i},\quad \frac{\partial\bar{c}_{s,i,N_{ri}}}{\partial r} = \frac{\bar{c}_{s,i,N_{ri}+1}-\bar{c}_{s,i,N_{ri}-1}}{2\Delta r_i} \qquad [42]$$

Refer to Table A·1 for the state-space equations governing the mass transport in the electrolyte. Table A·1 also includes the state-space equations for states adjacent to neighboring domains, such as $\bar{c}_{e,p,N_p}$, $\bar{c}_{e,s,N_p+1}$, $\bar{c}_{e,s,Nps}$, and $\bar{c}_{e,n,Nps+1}$, to accommodate the varying CV thicknesses across the domains as shown in Eq. (A·7).

Table A·2 shows the state-space equations resulting from FVM and FDM discretizations for the solid particle concentration equation (Eq. (1)). As shown in Fig. 2, the FVM state-space variables refer to the center of the CVs, while in FDM, they refer to the discretized node locations. In FDM, the solid surface concentration $c_{s,i}^{surf}$ is equal to the last node value at $c_{s,i,N_{ri}}$ for $i \in (p, n)$, whereas in FVM, the solid surface concentration is approximated at $c_{s,i,N_{ri}+1/2}$ for $i \in (p, n)$ using the third-order Hermite interpolation.[89]

Tables A·3 and A·4 show the FVM state-space equations for charge conservation in the solid and electrolyte phases, respectively. Note that the approximation $\frac{\partial \ln c_{e,i,k+0.5}}{\partial x} = \frac{c_{e,i,k+1}-c_{e,i,k}}{c_{e,i,k+0.5}\Delta x_i}$ is used to derive the discretized equations in Table A·4.

The Butler-Volmer equation (Eq. (5)) is algebraic and does not require spatial discretization. The volume averaged pore wall flux $\bar{j}_i$ at each CV is expressed as

**Table A·3. State-space representation for charge conservation in solid particles using FVM.**

Charge Conservation in Solid $i \in (p,\ n)$

$$\dot{\boldsymbol{\phi}}_{s,i} = \vec{0} = \mathcal{A}_{\phi_{s,i}}\boldsymbol{\phi}_{s,i} + \mathcal{B}_{\phi_{s,i}}, \quad \boldsymbol{\phi}_{s,p} = \begin{bmatrix} \phi_{s,p,1} \\ \vdots \\ \phi_{s,p,N_p} \end{bmatrix}_{N_p\times 1}, \quad \boldsymbol{\phi}_{s,n} = \begin{bmatrix} \phi_{s,n,N_{ps}+1} \\ \vdots \\ \phi_{s,n,N_{psn}} \end{bmatrix}_{N_n\times 1}$$

$$\mathcal{A}_{\phi_{s,i}} = \begin{bmatrix} -1 & 1 & 0 & 0 & ... & 0 \\ 1 & -2 & 1 & 0 & ... & 0 \\ 0 & 1 & -2 & 1 & ... & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & ... & 0 & 1 & -2 & 1 \\ 0 & ... & 0 & 0 & 1 & -1 \end{bmatrix}_{N_i\times N_i}$$

$$\mathcal{B}_{\phi_{s,p}} = -\frac{\Delta x_p{}^2 a_p F}{\sigma_p^{eff}} \begin{bmatrix} \bar{j}_{p,1} - \frac{I_{app}}{A_{cell} a_p F \Delta x_p} \\ \bar{j}_{p,2} \\ \vdots \\ \bar{j}_{p,N_p-1} \\ \bar{j}_{p,N_p} \end{bmatrix}_{N_p\times 1}, \qquad \mathcal{B}_{\phi_{s,n}} = -\frac{\Delta x_n{}^2 a_n F}{\sigma_n^{eff}} \begin{bmatrix} \bar{j}_{n,N_{ps}+1} \\ \bar{j}_{n,N_{ps}+2} \\ \vdots \\ \bar{j}_{n,N_{psn}-1} \\ 0 \end{bmatrix}_{N_{psn}\times 1} \quad [43]$$

**Table A·4. State-space representation for charge conservation in electrolyte using FVM.**

Charge Conservation in Electrolyte $i \in (p,\ s,\ n)$

$$\dot{\boldsymbol{\phi}}_e = 0 = \mathcal{A}_{\phi_e}\boldsymbol{\phi}_e + \mathcal{B}_{\phi_e} + \mathcal{C}_{\phi_e}, \quad \boldsymbol{\phi}_e = \begin{bmatrix} [\boldsymbol{\phi}_{e,p}]_{N_p\times 1} \\ [\boldsymbol{\phi}_{e,s}]_{N_s\times 1} \\ [\boldsymbol{\phi}_{e,n}]_{N_n\times 1} \end{bmatrix}_{N_{psn}\times 1}$$

$$\mathcal{A}_{\phi_e} = -\frac{1}{\Delta x_i} \begin{bmatrix} -\kappa_{p,3/2}^{eff} & \kappa_{p,3/2}^{eff} & 0 & 0 & ... & 0 \\ \kappa_{p,3/2}^{eff} & -(\kappa_{p,3/2}^{eff} + \kappa_{p,5/2}^{eff}) & \kappa_{p,5/2}^{eff} & 0 & ... & 0 \\ 0 & \kappa_{p,5/2}^{eff} & -(\kappa_{p,5/2}^{eff} + \kappa_{p,7/2}^{eff}) & \kappa_{p,7/2}^{eff} & ... & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & ... & 0 & \kappa_{n,N_{psn}-3/2}^{eff} & -(\kappa_{n,N_{psn}-3/2}^{eff} + \kappa_{n,N_{psn}-1/2}^{eff}) & \kappa_{n,N_{psn}-1/2}^{eff} \\ 0 & ... & 0 & 0 & \kappa_{n,N_{psn}-1/2}^{eff} & -\kappa_{n,N_{psn}-1/2}^{eff} \end{bmatrix}_{N_{psn}\times N_{psn}}$$

$$\mathcal{B}_{\phi_e} = \frac{2RT}{F\Delta x_i} \begin{bmatrix} \kappa_{p,3/2}^{eff}(1 - t_{+,3/2}) \frac{\bar{c}_{e,p,2} - \bar{c}_{e,p,1}}{c_{e,p,3/2}} \\ \kappa_{p,5/2}^{eff}(1 - t_{+,5/2}) \frac{\bar{c}_{e,p,3} - \bar{c}_{e,p,2}}{c_{e,p,5/2}} - \kappa_{p,3/2}^{eff}(1 - t_{+,3/2}) \frac{\bar{c}_{e,p,2} - \bar{c}_{e,p,1}}{c_{e,p,3/2}} \\ \vdots \\ -\kappa_{n,N_{psn}-1/2}^{eff}(1 - t_{+,N_{psn}-1/2}) \frac{\bar{c}_{e,n,N_{psn}} - \bar{c}_{e,n,N_{psn}-1}}{c_{e,n,N_{psn}-1/2}} \end{bmatrix}_{N_{psn}\times 1}, \quad \mathcal{C}_{\phi_e} = -a_i F \Delta x_i \begin{bmatrix} \bar{j}_{p,1} \\ \vdots \\ \bar{j}_{p,N_p} \\ [0]_{N_s\times N_s} \\ \bar{j}_{n,N_{ps}+1} \\ \vdots \\ \bar{j}_{n,N_{psn}} \end{bmatrix}_{N_{psn}\times 1}$$

$$\kappa_{i,k\pm 1/2}^{eff} = \frac{\kappa_{i,k\pm 1}^{eff}\kappa_{i,k}^{eff}}{\xi_i \kappa_{i,k\pm 1}^{eff} + (1 - \xi_i)\kappa_{i,k}^{eff}}, \quad \xi_i = \frac{\Delta x_{i,k\pm 1}}{\Delta x_{i,k\pm 1} + \Delta x_{i,k}}, \quad \Delta x_i = \frac{l_i}{N_i} \quad [44]$$

**Table A·4. (Continued).**

Charge Conservation in Electrolyte—Interface equations

$$\dot{\boldsymbol{\phi}}_{e,bc} = \mathcal{D}_{\phi_e}\boldsymbol{\phi}_e + \mathcal{E}_{\phi_e} + \begin{bmatrix} -a_p F \Delta x_p \bar{j}_{p,N_p} \\ 0 \\ 0 \\ -a_n F \Delta x_n \bar{j}_{n,N_{ps}+1} \end{bmatrix}_{4\times 1}, \quad \boldsymbol{\phi}_{e,bc} = \begin{bmatrix} \bar{\phi}_{e,p,N_p} \\ \bar{\phi}_{e,s,N_p+1} \\ \bar{\phi}_{e,s,N_{ps}} \\ \bar{\phi}_{e,n,N_{ps}+1} \end{bmatrix}_{4\times 1}$$

$$\mathcal{D}_{\phi_e} = -\left[\begin{array}{cccc|cccc|cccc} & & & & \frac{\kappa^{eff}_{p,N_p+1/2}}{\Delta x_{\overline{ps}}} & 0 & \dots & 0 & & & & \\ 0 & \dots & \frac{\kappa^{eff}_{p,N_p-1/2}}{\Delta x_p} & \gamma_1 & & & & & 0 & 0 & \dots & 0 \\ & & & & \gamma_2 & \frac{\kappa^{eff}_{s,N_p+3/2}}{\Delta x_s} & \dots & 0 & 0 & 0 & \dots & 0 \\ 0 & \dots & 0 & \frac{\kappa^{eff}_{s,N_s+1/2}}{\Delta x_{\overline{ps}}} & & & & & \frac{\kappa^{eff}_{s,N_{ps}+1/2}}{\Delta x_s \Delta x_{\overline{sn}}} & 0 & \dots & 0 \\ & & & & 0 & \dots & \frac{\kappa^{eff}_{s,N_{ps}-1/2}}{\Delta x_s} & \gamma_3 & & & & \\ 0 & \dots & 0 & 0 & & & & & \gamma_4 & \frac{\kappa^{eff}_{n,N_{ps}+3/2}}{\Delta x_n} & \dots & 0 \\ 0 & \dots & 0 & 0 & & & & & & & & \\ & & & & 0 & \dots & 0 & \frac{\kappa^{eff}_{n,N_{ps}+1/2}}{\Delta x_{\overline{sn}}} & & & & \end{array}\right]_{4\times N_{psn}}$$

$$x_{\overline{ps}} = \frac{\Delta x_p + \Delta x_s}{2}, \quad x_{\overline{sn}} = \frac{\Delta x_s + \Delta x_n}{2}$$

$$\gamma_1 = -\left(\frac{\kappa^{eff}_{p,N_p-1/2}}{\Delta x_p} + \frac{\kappa^{eff}_{p,N_p+1/2}}{\Delta x_{\overline{ps}}}\right)$$

$$\gamma_2 = -\left(\frac{\kappa^{eff}_{s,N_p+1/2}}{\Delta x_{\overline{ps}}} + \frac{\kappa^{eff}_{s,N_p+3/2}}{\Delta x_s}\right)$$

$$\gamma_3 = -\left(\frac{\kappa^{eff}_{s,N_{ps}-1/2}}{\Delta x_s} + \frac{\kappa^{eff}_{s,N_{ps}+1/2}}{\Delta x_{\overline{sn}}}\right)$$

$$\gamma_4 = -\left(\frac{\kappa^{eff}_{n,N_{ps}+1/2}}{\Delta x_{\overline{sn}}} + \frac{\kappa^{eff}_{n,N_{ps}+3/2}}{\Delta x_n}\right)$$

$$\mathcal{E}_{\phi_e} = \begin{bmatrix} \kappa^{eff}_{p,N_p+1/2}(1-t_{+,N_p+1/2})\frac{\bar{c}_{e,s,N_p+1}-\bar{c}_{e,p,N_p}}{c_{e,p,N_p+1/2}\Delta x_{\overline{ps}}} - \kappa^{eff}_{p,N_p-1/2}(1-t_{+,N_p-1/2})\frac{\bar{c}_{e,p,N_p}-\bar{c}_{e,p,N_p-1}}{c_{e,p,N_p-1/2}\Delta x_p} \\ \kappa^{eff}_{s,N_p+3/2}(1-t_{+,N_p+3/2})\frac{\bar{c}_{e,s,N_p+2}-\bar{c}_{e,s,N_p+1}}{c_{e,s,N_p+3/2}\Delta x_s} - \kappa^{eff}_{s,N_p+1/2}(1-t_{+,N_p+1/2})\frac{\bar{c}_{e,s,N_p+1}-\bar{c}_{e,p,N_p}}{c_{e,s,N_p+1/2}\Delta x_{\overline{ps}}} \\ \kappa^{eff}_{s,N_{ps}+1/2}(1-t_{+,N_{ps}+1/2})\frac{\bar{c}_{e,n,N_{ps}+1}-\bar{c}_{e,s,N_{ps}}}{c_{e,s,N_{ps}+1/2}\Delta x_{\overline{sn}}} - \kappa^{eff}_{s,N_{ps}-1/2}(1-t_{+,N_{ps}-1/2})\frac{\bar{c}_{e,s,N_{ps}}-\bar{c}_{e,s,N_{ps}-1}}{c_{e,s,N_{ps}-1/2}\Delta x_s} \\ \kappa^{eff}_{n,N_{ps}+3/2}(1-t_{+,N_{ps}+3/2})\frac{\bar{c}_{e,n,N_{ps}+2}-\bar{c}_{e,n,N_{ps}+1}}{c_{e,n,N_{ps}+3/2}\Delta x_n} - \kappa^{eff}_{n,N_{ps}+1/2}(1-t_{+,N_{ps}+1/2})\frac{\bar{c}_{e,n,N_{ps}+1}-\bar{c}_{e,s,N_{ps}}}{c_{e,n,N_{ps}+1/2}\Delta x_{\overline{sn}}} \end{bmatrix}_{N_{psn}\times 1} \quad [45]$$

$$\bar{j}_{i,k}(t) = \bar{j}_{0,i,k}\left[\exp\left(\frac{\alpha_{a,i}F}{RT}\bar{\eta}_{i,k}\right) - \exp\left(-\frac{\alpha_{c,i}F}{RT}\bar{\eta}_{i,k}\right)\right]$$
$$\bar{j}_{0,i,k} = k_{0,i}[(c_{s,i}^{\max} - c_{s,i,k}^{surf})\bar{c}_{e,i,k}]^{\alpha_{a,i}}\,(c_{s,i,k}^{surf})^{\alpha_{c,i}}$$
$$\bar{\eta}_{i,k} = \bar{\phi}_{s,i,k} - \bar{\phi}_{e,i,k} - U_i(\theta_{i,k}^{surf}) \qquad [A{\cdot}12]$$

where $c_{s,i,k}^{surf}$ and $\theta_{i,k}^{surf}$ refer to the particle surface concentration and the normalized particle surface concentration for a particle located at index $k$.

## Appendix B. Parameter Sensitivity and Correlation Analysis

In COBRAPRO, LSA and correlation analysis are implemented to assess the practical identifiability of parameters for the HPPC profile. LSA perturbs each parameter around its nominal value to quantify the impact on the model output. In LSA, the nominal values are taken from experimentally determined values from cell tear-down[38] to improve the accuracy of LSA. Correlation analysis quantifies the linear dependence between parameters.

As shown in Refs. 42 and 89, the sensitivity matrix ($\boldsymbol{S^{V,SOC}}$) is written as

$$\boldsymbol{S^{V,SOC}} = \begin{bmatrix} [\boldsymbol{S^{V}}]_{N\times m} \\ [\boldsymbol{S^{SOC_p}}]_{N\times m} \\ [\boldsymbol{S^{SOC_n}}]_{N\times m} \end{bmatrix}_{3N\times m} \qquad [B{\cdot}1]$$

where $y_V(t_j)$ is the voltage output vector at time instant $t_j$ for $j \in [1, N]$, $y_V^{nom}(t_j)$ is the nominal voltage simulated using the nominal parameter values at time instant $t_j$, $\Theta_q$ refers to the $q^{th}$ parameter for $q \in [1, m]$, and $\Theta_q^{nom}$ refers to the nominal value of the $q^{th}$ parameter. The partial derivative of the voltage with respect to the $q^{th}$ parameter is numerically calculated by taking the average of the upper and lower voltage perturbations given as

$$\frac{\partial y_V(t_j)}{\partial \Theta_q} = \frac{\left(\left|\frac{y_V^{upper}(t_j) - y_V^{nom}}{\Theta_q^{upper}}\right| + \left|\frac{y_V^{lower}(t_j) - y_V^{nom}}{\Theta_q^{lower}}\right|\right)}{2},$$
$$\begin{cases} j \in [1, N] \\ q \in [1, m] \end{cases} \qquad [B{\cdot}3]$$

where $\Theta_q^{upper} = \Theta_q^{nom} + \Delta\Theta_q$ is the upper perturbed $q^{th}$ parameter, $\Theta_q^{lower} = \Theta_q^{nom} - \Delta\Theta_q$ is the lower perturbed $q^{th}$ parameter, $y^{upper}{}_V$ refers to the voltage simulated using $\Theta_q^{upper}$, and $y^{lower}{}_V$ refers to the voltage simulated using $\Theta_q^{lower}$. In this work, the perturbation factor was set to $\Delta\Theta_q = 0.05\Theta_q^{nom}$.

Similar to $\boldsymbol{S^V}$, the elecrode *SOC* sensitivity matrices are written as

$$\boldsymbol{S^{SOC_i}} = \begin{bmatrix} \frac{\partial y_{SOC_i}(t_1)}{\partial \Theta_1}\frac{\Theta_1^{nom}}{y_{SOC_i}^{nom}(t_1)} & \frac{\partial y_{SOC_i}(t_1)}{\partial \Theta_2}\frac{\Theta_2^{nom}}{y_{SOC_i}^{nom}(t_1)} & \cdots & \frac{\partial y_{SOC_i}(t_1)}{\partial \Theta_m}\frac{\Theta_m^{nom}}{y_{SOC_i}^{nom}(t_1)} \\ \frac{\partial y_{SOC_i}(t_2)}{\partial \Theta_1}\frac{\Theta_1^{nom}}{y_{SOC_i}^{nom}(t_2)} & \frac{\partial y_{SOC_i}(t_2)}{\partial \Theta_2}\frac{\Theta_2^{nom}}{y_{SOC_i}^{nom}(t_2)} & \cdots & \frac{\partial y_{SOC_i}(t_2)}{\partial \Theta_m}\frac{\Theta_m^{nom}}{y_{SOC_i}^{nom}(t_2)} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial y_{SOC_i}(t_N)}{\partial \Theta_1}\frac{\Theta_1^{nom}}{y_{SOC_i}^{nom}(t_N)} & \frac{\partial y_{SOC_i}(t_N)}{\partial \Theta_2}\frac{\Theta_2^{nom}}{y_{SOC_i}^{nom}(t_N)} & \cdots & \frac{\partial y_{SOC_i}(t_N)}{\partial \Theta_m}\frac{\Theta_m^{nom}}{y_{SOC_i}^{nom}(t_N)} \end{bmatrix}_{N\times m}, \quad i \in (p, n) \qquad [B{\cdot}4]$$

where $\boldsymbol{S^V}$ is the voltage sensitivity matrix, $\boldsymbol{S^{SOC_p}}$ is the positive electrode *SOC* ($SOC_p$) sensitivity matrix, $\boldsymbol{S^{SOC_n}}$ is the negative electrode *SOC* ($SOC_n$) sensitivity matrix, $m$ is the total number of parameters being investigated in the LSA, and $N$ is the total number of data points in the voltage and electrode *SOC* vectors. The voltage sensitivity matrix is expressed as

$$\boldsymbol{S^V} = \begin{bmatrix} \frac{\partial y_V(t_1)}{\partial \Theta_1}\frac{\Theta_1^{nom}}{y_V^{nom}(t_1)} & \frac{\partial y_V(t_1)}{\partial \Theta_2}\frac{\Theta_2^{nom}}{y_V^{nom}(t_1)} & \cdots & \frac{\partial y_V(t_1)}{\partial \Theta_m}\frac{\Theta_m^{nom}}{y_V^{nom}(t_1)} \\ \frac{\partial y_V(t_2)}{\partial \Theta_1}\frac{\Theta_1^{nom}}{y_V^{nom}(t_2)} & \frac{\partial y_V(t_2)}{\partial \Theta_2}\frac{\Theta_2^{nom}}{y_V^{nom}(t_2)} & \cdots & \frac{\partial y_V(t_2)}{\partial \Theta_m}\frac{\Theta_m^{nom}}{y_V^{nom}(t_2)} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial y_V(t_N)}{\partial \Theta_1}\frac{\Theta_1^{nom}}{y_V^{nom}(t_N)} & \frac{\partial y_V(t_N)}{\partial \Theta_2}\frac{\Theta_2^{nom}}{y_V^{nom}(t_N)} & \cdots & \frac{\partial y_V(t_N)}{\partial \Theta_m}\frac{\Theta_m^{nom}}{y_V^{nom}(t_N)} \end{bmatrix}_{N\times m} \qquad [B{\cdot}2]$$

where $y_{SOC_i}(t_j)$ is the $SOC_i$ output vector at time instant $t_j$ for $j = [1, t_N]$ and $y_{SOC,i}^{nom}(t_j)$ is the nominal $SOC_i$ simulated using the nominal parameter values at time instant $t_j$. The partial derivative of $SOC_i$ with respect to the $q^{th}$ parameter is written using the same formulation shown in Eq. (B·3) for the voltage perturbation.

In $\boldsymbol{S^{V,SOC}}$, each column represents the time-dependent sensitivity vector for the $q^{th}$ parameter, spanning from time $t_1$ to $t_N$ for $q \in [1, m]$. The sensitivity index of the $q^{th}$ parameter is calculated by taking the L2 norm of the $q^{th}$ column of the sensitivity matrix $\boldsymbol{S^{V,SOC}}$. The sensitivity index for parameter $\Theta_q$ is given as

$$S_q^{V,SOC} = \|S_{:,q}^{V,SOC}\|, \quad q \in [1, m] \qquad [B{\cdot}5]$$

where $\|\cdot\|$ refers to the L2 norm and $S_{:,q}^{V,SOC}$ refers to the $q^{th}$ column of the total sensitivity matrix.

To quantify the correlation between parameters, the correlation matrix $\boldsymbol{C}$ is calculated as

$$\boldsymbol{C} = \begin{bmatrix} C_{1,1} & C_{1,2} & \cdots & C_{1,d} \\ C_{2,1} & C_{2,2} & \cdots & C_{2,d} \\ \vdots & \vdots & \ddots & \vdots \\ C_{b,1} & C_{j,2} & \cdots & C_{b,d} \end{bmatrix}_{j,i}, \quad b, d \in [1, m] \qquad [B{\cdot}6]$$

where $m$ equals the total number of parameters being investigated in the practical identifiability analysis.

Each element in $\boldsymbol{C}$ represents the correlation coefficient between two parameters. This coefficient is calculated by taking the absolute value of the linear correlation between two columns in the $\boldsymbol{S}^{\boldsymbol{V,SOC}}$ matrix given as

$$C_{b,d} = \left| \frac{\mathrm{cov}(\boldsymbol{S}^{\boldsymbol{V,SOC}}_{:,\boldsymbol{b}}, \boldsymbol{S}^{\boldsymbol{V,SOC}}_{:,\boldsymbol{d}})}{\sigma(\boldsymbol{S}^{\boldsymbol{V,SOC}}_{:,\boldsymbol{b}})\sigma(\boldsymbol{S}^{\boldsymbol{V,SOC}}_{:,\boldsymbol{d}})} \right| \qquad [B{\cdot}7]$$

where $\sigma(\boldsymbol{S}^{\boldsymbol{V,SOC}}_{:,\boldsymbol{b}})$ is the standard deviation of $\boldsymbol{S}^{\boldsymbol{V,SOC}}_{:,\boldsymbol{b}}$. The term $\mathrm{cov}(\boldsymbol{S}^{\boldsymbol{V,SOC}}_{:,\boldsymbol{b}}, \boldsymbol{S}^{\boldsymbol{V,SOC}}_{:,\boldsymbol{d}})$ refers to the covariance between $\boldsymbol{S}^{\boldsymbol{V,SOC}}_{:,\boldsymbol{b}}$ and $\boldsymbol{S}^{\boldsymbol{V,SOC}}_{:,\boldsymbol{d}}$ and is expressed as

$$\mathrm{cov}(\boldsymbol{S}^{\boldsymbol{V,SOC}}_{:,\boldsymbol{b}}, \boldsymbol{S}^{\boldsymbol{V,SOC}}_{:,\boldsymbol{d}}) = \frac{1}{j-1} \sum_{j=1}^{N} (\boldsymbol{S}^{\boldsymbol{V,SOC}}_{j,\boldsymbol{b}} - \mu(\boldsymbol{S}^{\boldsymbol{V,SOC}}_{:,\boldsymbol{b}}))(\boldsymbol{S}^{\boldsymbol{V,SOC}}_{j,\boldsymbol{d}} - \mu(\boldsymbol{S}^{\boldsymbol{V,SOC}}_{:,\boldsymbol{d}})) \qquad [B{\cdot}8]$$

where $N$ is the number of rows in the sensitivity matrix $\boldsymbol{S}^{\boldsymbol{V,SOC}}$ and $\mu(\boldsymbol{S}^{\boldsymbol{V,SOC}}_{:,\boldsymbol{b}})$ represents the mean of $\boldsymbol{S}^{\boldsymbol{V,SOC}}_{:,\boldsymbol{b}}$.

Since the absolute value is taken in Eq. (B·7), the correlation coefficient $C_{b,d}$ is between 0 and 1. A $C_{b,d}$ value close to 1 signifies that the $b^{th}$ and $d^{th}$ parameters are highly correlated, while a $C_{b,d}$ value near 0 means little to no correlation between the parameters. In the context of parameter identifiability analysis, a correlation coefficient threshold ($\beta$) can be defined such that if $C_{b,d} > \beta$, the parameters are considered correlated.

## Appendix C. LG INR21700-M50 Cell Parameters from Literature

In Ref. 38, cell tear-down experiments are conducted to obtain geometric, kinetic, stoichiometric, concentration, and OCP values of a LG INR21700 M50 cell[e] with a NMC811 positive electrode and silicon-graphite (Si-Gr) negative electrode. The electrolyte transport and concentration parameters, namely $t_+$, $\kappa$, $D$, and $c_0$, are obtained from the experimental results reported in Ref. 43.

[e]Note that the Bruggeman coefficients are approximated to be 1.5 following the assumption of perfectly spherical particles in the electrodes.[11]

**Table C1. LG INR21700 M50 (NMC811/Si-Gr) parameter values from Ref. 38.**

| Parameter | Units | Description | Positive Electrode ($i = p$) | Separator ($i = s$) | Negative Electrode ($i = n$) |
|---|---|---|---|---|---|
| $l_i$ | m | Thickness | $75.6 \times 10^{-6}$ | $12 \times 10^{-6}$ | $85.2 \times 10^{-6}$ |
| $R_i$ | m | Particle radius | $5.22 \times 10^{-6}$ | – | $5.86 \times 10^{-6}$ |
| $\varepsilon_i$ | – | Porosity | 0.335 | 0.47 | 0.25 |
| $\varepsilon_i^{filler}$ | – | Binder volume fraction | 0 | 0 | 0 |
| $\varepsilon_i^{solid}$ | – | Active material volume fraction | 0.665 | 0 | 0.75 |
| $brugg_i$ | – | Bruggeman coefficient | 1.5 | 1.5 | 1.5 |
| $D_{s,i}$ | $m^2/s$ | Solid phase diffusion | $4.0 \times 10^{-15}$ | – | $3.3 \times 10^{-14}$ |
| $\sigma_i$ | S/m | Solid phase conductivity | 0.2707 | – | 286.67 |
| $k_i$ | $m^{2.5}/(mol^{0.5} \cdot s)$ | Reaction rate coefficient | $3.54 \times 10^{-11}$ | – | $6.72 \times 10^{-12}$ |
| $\alpha_{a,i}$ | – | Anodic charge transfer coefficient | 0.5 | – | 0.5 |
| $\alpha_{c,i}$ | – | Cathodic charge transfer coefficient | 0.5 | – | 0.5 |
| $c_{s,i}^{max}$ | $mol/m^3$ | Maximum solid phase concentration | 63104 | – | 33133 |
| $c_0$ | $mol/m^3$ | Equilibrium concentration in electrolyte | 1000 | 1000 | 1000 |
| $\theta_i^{0\%}$ | – | Stoichiometric coefficients at 0% SOC | 0.9084 | – | 0.0279 |
| $\theta_i^{100\%}$ | – | Stoichiometric coefficients at 100% SOC | 0.27 | – | 0.9014 |
| $A_{cell}$ | $m^2$ | Cell area | | 0.1037 | |
| $F$ | C/mol | Faraday constant | | 96 485 | |
| $R$ | J/(mol·K) | Ideal gas constant | | 8.314 | |
| $T$ | K | Cell temperature | | 298.15 | |

| Parameter | Units | Description | Expression |
|---|---|---|---|
| $U_p$ | V | Open-circuit potential of NMC811 | $U_p = -0.8090 \times \theta_p + 4.4875 - 0.0428 \times \tanh(18.5138 \times (\theta_p - 0.5542)) - 17.7326 \times \tanh(15.7890 \times (\theta_p - 0.3117)) + 17.5842 \times \tanh(15.9308(\theta_p - 0.3120))$ |
| $U_n$ | V | Open-circuit potential of Si-Gr | $U_n = 1.9793 \times e^{-39.3631\theta_n} + 0.2482 - 0.0909 \times \tanh(29.8538 \times (\theta_n - 0.1234)) - 0.04478 \times \tanh(14.9159 \times (\theta_n - 0.2769)) - 0.0205 \times \tanh(30.4444(\theta_n - 0.6103))$ |
| $t_+$ | – | Transference number[43] | $t_+ = -0.1287 \times (c_e/1000)^3 + 0.4106 \times (c_e/1000)^2 - 0.4717 \times (c_e/1000) + 0.4492$ |
| $D$ | $m^2/s$ | Electrolyte diffusion[43] | $D = 8.794 \times 10^{-11} \times (c_e/1000)^2 - 3.972 \times 10^{-10} \times (c_e/1000) + 4.862 \times 10^{-10}$ |
| $\kappa$ | S/m | Electrolyte conductivity[43] | $\kappa = 0.1297 \times (c_e/1000)^3 - 2.51 \times (c_e/1000)^{1.5} + 3.329 \times (c_e/1000)$ |

## Appendix D. Model Accuracy Comparison for FVM versus FDMcs

A systematic voltage accuracy study is conducted to compare FVM and FDM for the radial discretization of Fick's law, which governs the particle concentration variables, $c_{s,i}$ , where $i \in (p, n)$. In this study, we denote "FVM" as FVM discretization of $c_{s,i}$ and "FDMcs" as FDM discretization of $c_{s,i}$. For both FVM and FDMcs schemes, FVM is used to discretize all other remaining macroscale variables along the x-direction, e.g., $c_e$, $\phi_{s,i}$, and $\phi_e$. Since COBRAPRO offers two discretization methods for the solid particle concentration, users can choose their preferred discretization method for $c_{s,i}$.

Figure 11 shows comparison results of the voltage accuracy for FVM and FDMcs at 1C discharge for different $N$ and $N_r$, where $N = N_p = N_s = N_n$ and $N_r = N_{rp} = N_{rn}$. Parameters from Ref. 38 were used for both FVM and FDMcs simulations. The baseline is the voltage simulated at $N = 100$ and $N_r = 100$ using FVM, denoted as $V^{FVM}_{N_{100},N_{r,100}}$. Figures 11a and 11b show the voltage RMSE ($V_{rmse}$) between the baseline ($V^{FVM}_{N_{100},N_{r,100}}$) and the voltage simulated at a specific $N_j$ and $N_{r,k}$, denoted as $V_{N_j,N_{r,k}}$. The voltage RMSE is defined as

$$V_{rmse} = \sqrt{\sum_{i=1}^{N} \frac{1}{N}(V^{FVM}_{N_{100},N_{r,100}}(i) - V_{N_j,N_{r,k}}(i))^2} \qquad [C\cdot1]$$

where $j, k \in (3, 5, 10, 15, 20, 25, 30, 40, 50, 60, 70, 80, 90, 100)$. Figures 11c and 11d project the three-dimensional graphs of Figs. 11a and 11b onto the two-dimensional plane ($V_{rmse}$ versus $N_r$), such that each curve represents the $V_{rmse}$ for a given $N$ as labeled in the color-coded legend.

We list key takeaway points from Fig. 11 regarding $V_{rmse}$ and $t^{total}_{sim}$, providing guidelines for selecting the optimal number of $N$ and $N_r$ to minimize $t^{total}_{sim}$ without compromising voltage accuracy. Note that the observations made here are specific to 1C discharge simulations.

- Key point 1: For both FVM and FDMcs, setting $N = 10$ for any $N_r$ maintains voltage accuracy while reducing simulation time. As shown in Figs. 11c and 11d, the $V_{rmse}$ does not improve significantly beyond $10 = N$, as indicated by the overlapping curves for $10 \leqslant N \leqslant 100$. For example, with $N_r$=10, increasing $N$ from 3 to 10 reduces $V_{rmse}$ from 0.0079 V to 0.0021 V, improving $V_{rmse}$ by

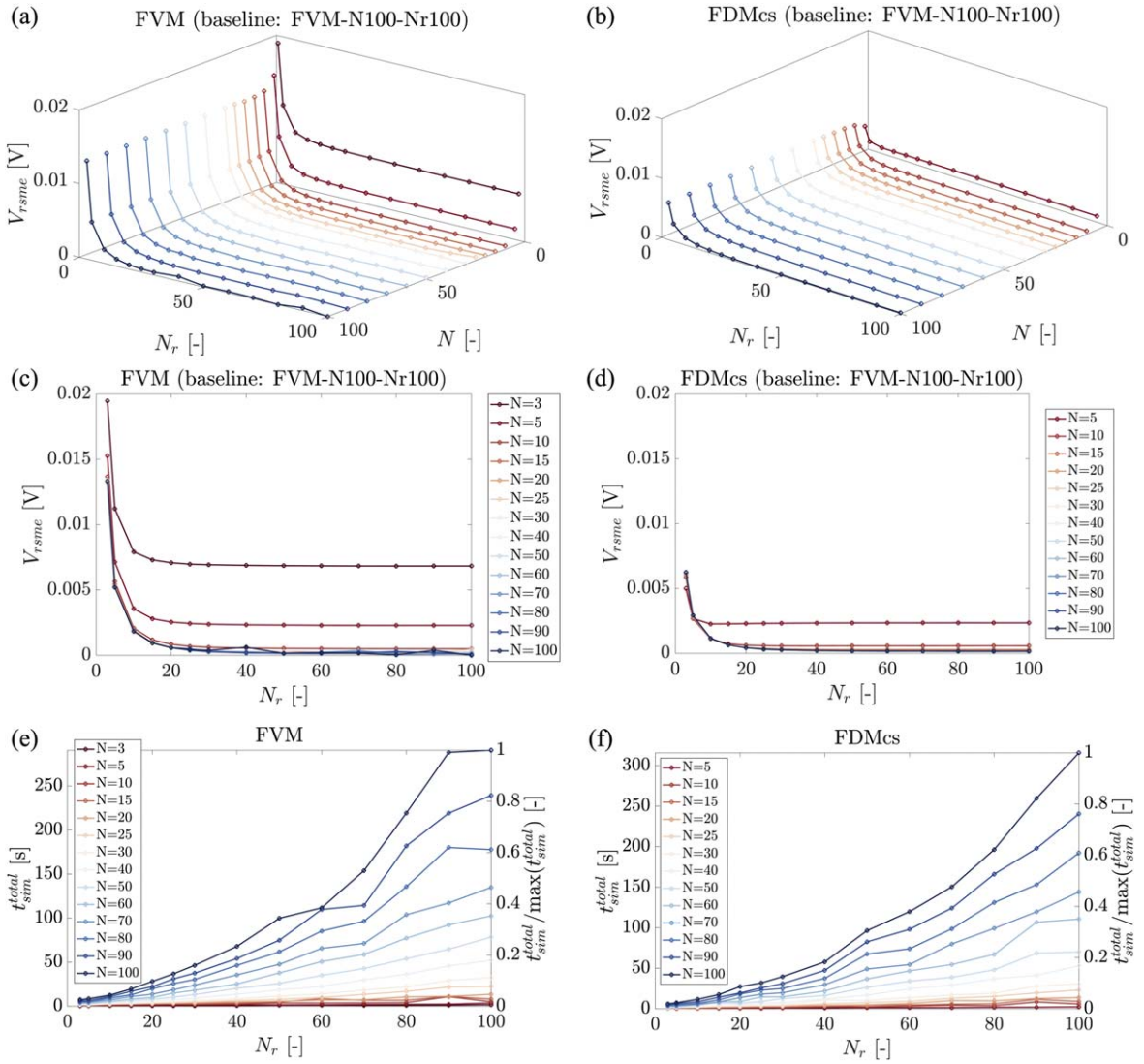


**Figure 11.** Comparison of FVM and FDM used to discretize $c_s$ at 1C discharge. FVM: (a) $V_{rmse}$ as a function of $N_r$ and $N$, (c) two-dimensional view on the $V_{rsme}$ versus $N_r$ plane, (e) total simulation time $t^{total}_{sim}$. FDMcs: (b) $V_{rmse}$ as a function of $N_r$ and $N$, (d) two-dimensional view on the $V_{rsme}$ versus $N_r$ plane, and (f) total simulation time $t^{total}_{sim}$.

73.4%. Increasing $N$ to 40 further reduces $V_{rmse}$ to 0.0019 V, but only by 9.52%, while $t_{sim}^{total}$ increases to 4.442 s (a 252% increase). Similar results apply to FDMcs, making $N = 10$ ideal for 1C discharge.[f]

- Key point 2: For both FVM and FDMcs, setting $N_r$=20 for any $N$ maintains voltage accuracy while reducing simulation time. As shown in Figs. 11c and 11d, $V_{rmse}$ decreases exponentially and shows minimal improvement beyond $N_r$=20. For example, for FVM at $N = 10$, increasing $N_r$ from 3 to 20 reduces $V_{rmse}$ from 0.0137 V to 0.0008 V, a 94.2% improvement in $V_{rmse}$. Increasing $N_r$ to 60 reduces $V_{rmse}$ by only 37.5%, with $t_{sim}^{total}$ increasing by 108%. Similar results apply to FDMcs, making $N_r$=20 ideal for 1C discharge.
- Key point 3: At the recommended $N = 10$ and $N_r = 20$, FDMcs and FVM yield comparable results, with FDMcs showing slightly better performance. At $N = 10$ and $N_r$=20, the FVM $V_{rmse} = 0.0008$ V and $t_{sim}^{total} = 1.751$ s and FDMcs $V_{rmse} = 0.0006$ V and $t_{sim}^{total} = 0.955$ s. Although FVM ensures mass conservation, FDMcs offers higher voltage accuracy because it does not require approximating the surface solid concentration.[89]

In general, for given values of $N$ and $N_r$, $V_{rmse}$ is smaller at lower C-rates because slower dynamics cause less rapid changes in the lithium concentration gradient within the particle and electrolyte. Because the maximum C-rate in the HPPC and UDDS data used in this study was 1C, we used $N = 10$ and $N_r$=20 for the C/20 capacity test, HPPC, and UDDS simulations. At C-rates exceeding 1C, more discretization points are needed due to rapid concentration gradient changes. For a given current profile, we generally recommend comparing voltage simulation results with a benchmark solution that uses many discretization points.

## ORCID

Sara Ha https://orcid.org/0009-0005-9878-3537
Simona Onori https://orcid.org/0000-0002-6556-2608

## References

1. Y. Tian et al., "Promises and Challenges of Next-Generation "Beyond Li-ion" Batteries for Electric Vehicles and Grid Decarbonization." *Chem. Rev.*, **121**, 1623 (2021).
2. J. M. Granholm, *National Blueprint for Lithium Batteries 2021-2030* (2021).
3. I. D. Campbell, K. Gopalakrishnan, M. Marinescu, M. Torchio, G. J. Offer, and D. Raimondo, "Optimising lithium-ion cell design for plug-in hybrid and battery electric vehicles." *Journal of Energy Storage*, **22**, 228 (2019).
4. Y. Dai and V. Srinivasan, "On Graded Electrode Porosity as a Design Tool for Improving the Energy Density of Batteries." *J. Electrochem. Soc.*, **163**, A406 (2016).
5. L. D. Couto, M. Charkhgard, B. Karaman, N. Job, and M. Kinnaert, "Lithium-ion battery design optimization based on a dimensionless reduced-order electrochemical model." *Energy*, **263**, 125966 (2023).
6. R. Broad and A. Fly, "Influence of geometrical manufacturing tolerances on lithium-ion battery performance." *International Journal of Energy Research*, **46**, 23824 (2022).
7. V. L. Deringer, "Modelling and understanding battery materials with machine-learning-driven atomistic simulations." *J. Phys.: Energy*, **2**, 041003 (2020).
8. M. Naguib, P. Kollmeyer, and A. Emadi, "Lithium-Ion Battery Pack Robust State of Charge Estimation, Cell Inconsistency, and Balancing: Review." *IEEE Access*, **9**, 50570 (2021).
9. I. N. Haq, E. Leksono, M. Iqbal, F. X. N. Sodami, K. D. Nugraha, and B. Yuliarto, "Development of battery management system for cell monitoring and protection." *2014 International Conference on Electrical Engineering and Computer Science (ICEECS), Kuta, Bali, Indonesia, IEEE*203 (2014).
10. M. K. Tran, M. Mathew, S. Janhunen, S. Panchal, K. Raahemifar, R. Fraser, and M. Fowler, "A comprehensive equivalent circuit model for lithium-ion batteries, incorporating the effects of state of health, state of charge, and temperature on model parameters." *Journal of Energy Storage*, **43**, 103252 (2021).
11. G. L. Plett, *Battery Management Systems: Battery Modeling* (Artech House) Vol. 1 (2015).
12. P. Gasper, A. Schiek, K. Smith, Y. Shimonishi, and S. Yoshida, "Predicting battery capacity from impedance at varying temperature and state of charge using machine learning." *Cell Reports Physical Science*, **3**, 101184 (2022).
13. X. Li, D. Yu, V. Søren Byg, and S. Daniel Ioan, "The development of machine learning-based remaining useful life prediction for lithium-ion batteries." *Journal of Energy Chemistry*, **82**, 103 (2023).

[f]The FDMcs $V_{rmse}$ at $N = 3$ is omitted as we noticed an unusual trend as a function of $N_r$ (such as $V_{rmse}$ increasing with increasing $N_r$ at $N = 3$). The simulation is unstable at $N = 3$ and we do not recommend using $N = 3$ for FDMcs.

14. P. Gasper, N. Collath, H. C. Hesse, A. Jossen, and K. Smith, "Machine-Learning Assisted Identification of Accurate Battery Lifetime Models with Uncertainty." *J. Electrochem. Soc.*, **169**, 080518 (2022).
15. O. Furat, D. P. Finegan, and D. Diercks, "Usseglio-Viretta, F., Smith, K., and Schmidt, V. Mapping the architecture of single lithium ion electrode particles in 3D, using electron backscatter diffraction and machine learning segmentation." *Journal of Power Sources*, **483**, 229148 (2021).
16. G. Pozzato, A. Allam, L. Pulvirenti, G. A. Negoita, W. A. Paxton, and S. Onori, "Analysis and key findings from real-world electric vehicle field data." *Joule*, **7**, 2035 (2023).
17. G. Pozzato and S. Onori, "Combining physics-based and machine learning methods to accelerate innovation in sustainable transportation and beyond: A control perspective." *2023 American Control Conference (ACC), San Diego, CA, USA, IEEE*640 (2023).
18. G. Pozzato, X. Li, D. Lee, J. Ko, and S. Onori, "Accelerating the transition to cobalt-free batteries: A hybrid model for LiFePO4/graphite chemistry." *npj Computational Materials*, **10**, 14 (2024).
19. D. P. Finegan, J. Zhu, X. Feng, M. Keyser, M. Ulmefors, W. Li, M. Z. Bazant, and S. J. Cooper, "The Application of Data-Driven Methods and Physics-Based Learning for Improving Battery Safety." *Joule*, **5**, 316 (2021).
20. M. Doyle, T. F. Fuller, and J. Newman, "Modeling of Galvanostatic Charge and Discharge of the Lithium/Polymer/Insertion Cell." *J. Electrochem. Soc.*, **140**, 1526 (1993).
21. J. Newman and W. Tiedemann, "Porous-electrode theory with battery applications." *AIChE J.*, **21**, 25 (1975).
22. T. F. Fuller, M. Doyle, and J. Newman, "Simulation and Optimization of the Dual Lithium Ion Insertion Cell." *J. Electrochem. Soc.*, **141**, 1 (1994).
23. S. Kolluri, S. V. Aduru, M. Pathak, R. D. Braatz, and V. R. Subramanian, "Real-time Nonlinear Model Predictive Control (NMPC) Strategies using Physics-Based Models for Advanced Lithium-ion Battery Management System (BMS)." *J. Electrochem. Soc.*, **167**, 063505 (2020).
24. B. Suthar, P. W. C. Northrop, R. D. Braatz, and V. R. Subramanian, "Optimal Charging Profiles with Minimal Intercalation-Induced Stresses for Lithium-Ion Batteries Using Reformulated Pseudo 2-Dimensional Models." *J. Electrochem. Soc.*, **161**, F3144 (2014).
25. S. Santhanagopalan, Q. Guo, P. Ramadass, and R. E. White, "Review of models for predicting the cycling performance of lithium ion batteries." *Journal of Power Sources*, **156**, 620 (2006).
26. S. J. Moura, F. B. Argomedo, R. Klein, A. Mirtabatabaei, and M. Krstic, "Battery State Estimation for a Single Particle Model With Electrolyte Dynamics." *IEEE Transactions on Control Systems Technology*, **25**, 453 (2017).
27. A. Allam and S. Onori, "An Interconnected Observer for Concurrent Estimation of Bulk and Surface Concentration in the Cathode and Anode of a Lithium-ion Battery." *IEEE Trans. Ind. Electron.*, **65**, 7311 (2018).
28. A. Jokar, B. Rajabloo, M. Désilets, and M. Lacroix, "Review of simplified Pseudo-two-Dimensional models of lithium-ion batteries." *Journal of Power Sources*, **327**, 44 (2016).
29. I. Korotkin, S. Sahu, S. E. J. O'Kane, G. Richardson, and J. M. Foster, "DandeLiion v1: An Extremely Fast Solver for the Newman Model of Lithium-Ion Battery (Dis)charge." *J. Electrochem. Soc.*, **168**, 060544 (2021).
30. M. Torchio, L. Magni, R. B. Gopaluni, R. D. Braatz, and 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**, A1192 (2016).
31. V. Sulzer, S. G. Marquis, R. Timms, M. Robinson, and S. J. Chapman, "Python battery mathematical modelling (PyBaMM)." *Journal of Open Research Software*, **9**, 14 (2021).
32. R. N. Methekar, V. Ramadesigan, J. C. Pirkle, and V. R. Subramanian, "A perturbation approach for consistent initialization of index-1 explicit differential–algebraic equations arising from battery model simulations." *Computers & Chemical Engineering*, **35**, 2227 (2011).
33. M. T. Lawder, P. W. C. Northrop, and V. R. Subramanian, "Model-Based SEI Layer Growth and Capacity Fade Analysis for EV and PHEV Batteries and Drive Cycles." *J. Electrochem. Soc.*, **161**, A2099 (2014).
34. P. N. Brown, A. C. Hindmarsh, and L. R. Petzold, "Consistent Initial Condition Calculation for Differential-Algebraic Systems." *SIAM Journal on Scientific Computing*, **19**, 1495 (1998).
35. M. T. Lawder, V. Ramadesigan, B. Suthar, and V. R. Subramanian, "Extending explicit and linearly implicit ODE solvers for index-1 DAEs. Computers &amp." *Chemical Engineering*, **82**, 283 (2015).
36. M. Ecker, S. Käbitz, I. Laresgoiti, and D. U. Sauer, "Parameterization of a Physico-Chemical Model of a Lithium-Ion Battery: II. Model Validation." *J. Electrochem. Soc.*, **162**, A1849 (2015).
37. J. Schmalstieg, C. Rahe, M. Ecker, and D. U. Sauer, "Full cell parameterization of a high-power lithium-ion battery for a physico-chemical model: part I. physical and electrochemical parameters." *J. Electrochem. Soc.*, **165**, A3799 (2018).
38. C. H. Chen, F. Brosa Planella, K. O'Regan, D. Gastol, W. D. Widanage, and E. Kendrick, "Development of experimental techniques for parameterization of multi-scale lithium-ion battery models." *J. Electrochem. Soc.*, **167**, 080534 (2020).
39. A. P. Schmidt, M. Bitzer, A. ´ W. Imre, and L. Guzzella, "Experiment-driven electrochemical modeling and systematic parameterization for a lithium-ion battery cell." *Journal of Power Sources*, **195**, 5071 (2010).
40. A. Jokar, B. Rajabloo, M. Désilets, and M. Lacroix, "An Inverse Method for Estimating the Electrochemical Parameters of Lithium-Ion Batteries: I. Methodology." *J. Electrochem. Soc.*, **163**, A2876 (2016).

41. L. Zhang, L. Wang, G. Hinds, C. Lyu, J. Zheng, and J. Li, "Multi-objective optimization of lithium-ion battery model using genetic algorithm approach." *Journal of Power Sources*, **270**, 367 (2014).
42. A. Allam and S. Onori, "Online Capacity Estimation for Lithium-Ion Battery Cells via an Electrochemical Model-Based Adaptive Interconnected Observer." *IEEE Transactions on Control Systems Technology*, **29**, 1636 (2021).
43. A. Nyman, M. Behm, and G. Lindbergh, "Electrochemical characterisation and modelling of the mass transport phenomena in LiPF6–EC–EMC electrolyte." *Electrochimica Acta*, **53**, 6356 (2008).
44. D. Sauerteig, N. Hanselmann, A. Arzberger, H. Reinshagen, S. Ivanov, and A. Bund, "Electrochemical-mechanical coupled modeling and parameterization of swelling and ionic transport in lithium-ion batteries." *Journal of Power Sources*, **378**, 235 (2018).
45. S. Ha, G. Pozzato, and S. Onori, "Electrochemical characterization tools for lithium-ion batteries." *Journal of Solid State Electrochemistry*, **28**, 1131 (2024).
46. A. A. Wang et al., "Review of parameterisation and a novel database (LiionDB) for continuum Li-ion battery models." *Progress in Energy*, **4**, 032004 (2022).
47. I. V. Thorat, D. E. Stephenson, N. A. Zacharias, K. Zaghib, J. N. Harb, and D. R. Wheeler, "Quantifying tortuosity in porous Li-ion battery materials." *Journal of Power Sources*, **188**, 592 (2009).
48. X. Zhang, W. Shyy, and A. Marie Sastry, "Numerical Simulation of Intercalation-Induced Stress in Li-Ion Battery Electrode Particles." *J. Electrochem. Soc.*, **154**, A910 (2007).
49. J. Christensen and J. Newman, "A Mathematical Model of Stress Generation and Fracture in Lithium Manganese Oxide." *J. Electrochem. Soc.*, **153**, A1019 (2006).
50. P. Mohtat, S. Lee, V. Sulzer, J. B. Siegel, and A. G. Stefanopoulou, "Differential Expansion and Voltage Model for Li-ion Batteries at Practical Charging Rates." *J. Electrochem. Soc.*, **167**, 110561 (2020).
51. W. Li, I. Demir, D. Cao, D. Jöst, F. Ringbeck, M. Junker, and D. U. Sauer, "Data-driven systematic parameter identification of an electrochemical model for lithium-ion batteries with artificial intelligence." *Energy Storage Materials*, **44**, 557 (2022).
52. J. C. Forman, S. J. Moura, J. L. Stein, and H. K. Fathy, "Genetic identification and fisher identifiability analysis of the Doyle–Fuller–Newman model from experimental cycling of a LiFePO4 cell." *Journal of Power Sources*, **210**, 263 (2012).
53. M. Andersson, M. Streb, J. Y. Ko, V. Löfqvist Klass, M. Klett, H. Ekström, M. Johansson, and G. Lindbergh, "Parametrization of physics-based battery models from input–output data: A review of methodology and current research." *Journal of Power Sources*, **521**, 230859 (2022).
54. H. Miao, X. Xia, A. S. Perelson, and H. Wu, "On Identifiability of Nonlinear ODE Models and Applications in Viral Dynamics." *SIAM Review*, **53**, 3 (2011).
55. E. Atam and L. Helsen, "Control-oriented thermal modeling of multizone buildings: methods and issues: intelligent control of a building system." *IEEE Control Systems*, **36**, 86 (2016).
56. F. G. Wieland, A. L. Hauber, M. Rosenblatt, C. Tönsing, and J. Timmer, "On structural and practical identifiability. Current Opinion in." *Systems Biology*, **25**, 60 (2021).
57. A. Raue, C. Kreutz, T. Maiwald, J. Bachmann, M. Schilling, U. Klingmüller, and J. Timmer, "Structural and practical identifiability analysis of partially observed dynamical models by exploiting the profile likelihood." *Bioinformatics*, **25**, 1923 (2009).
58. R. Bellman and K. J. Åström, "On structural identifiability." *Mathematical Biosciences*, **7**, 329 (1970).
59. Z. Khalik, M. Donkers, J. Sturm, and H. Bergveld, "Parameter estimation of the Doyle–Fuller–Newman model for Lithium-ion batteries by parameter normalization, grouping, and sensitivity analysis." *Journal of Power Sources*, **499**, 229901 (2021).
60. G. L. Plett and M. S. Trimboli, *Battery Management Systems: Physics-based Modeling* Volume 3 (2023)(Artech House).
61. R. Drummond and S. R. Duncan, "Structural Identifiability of a Pseudo-2D Li-ion Battery Electrochemical Model." *IFAC-PapersOnLine*, **53**, 12452 (2020).
62. W. Li, D. Cao, D. Jöst, F. Ringbeck, M. Kuipers, F. Frie, and D. U. Sauer, "Parameter sensitivity analysis of electrochemical model-based battery management systems for lithium-ion batteries." *Applied Energy*, **269**, 115104 (2020).
63. A. Goshtasbi, J. Chen, J. R. Waldecker, S. Hirano, and T. Ersal, "Effective Parameterization of PEM Fuel Cell Models–Part I: Sensitivity Analysis and Parameter Identifiability." *J. Electrochem. Soc.*, **167**, 044504 (2020).
64. M. F. Samadi and M. Saif, "Identifiability analysis of an electrochemical model of li-ion battery." *2016 American Control Conference (ACC), Boston, MA, USA, IEEE*pp 3107 (2016).
65. N. Jin, D. L. Danilov, P. M. J. Van Den Hof, and M. C. F. Donkers, "Parameter estimation of an electrochemistry-based lithium-ion battery model using a two-step procedure and a parameter sensitivity analysis: Parameter estimation of an electrochemistry-based lithium-ion battery model using a two-step procedure and a parameter sensitivity analysis." *International Journal of Energy Research*, **42**, 2417 (2018).
66. B. F. Lund and B. A. Foss, "Parameter ranking by orthogonalization–Applied to nonlinear mechanistic models." *Automatica*, **44**, 278 (2008).
67. M. Streb, M. Ohrelius, M. Klett, and G. Lindbergh, "Improving Li-ion battery parameter estimation by global optimal experiment design." *Journal of Energy Storage*, **56**, 105948 (2022).
68. J. Newman, "FORTRAN programs for the simulation of electrochemical systems." (1998), https://www.cchem.berkeley.edu/~jsngrp/fortran.html.
69. S. B. Lee and S. Onori, "A Robust and Sleek Electrochemical Battery Model Implementation: A MATLAB® Framework." *J. Electrochem. Soc.*, **168**, 090527 (2021).
70. S. Moura and S. Park, fastDFN (2018), https://github.com/scott-moura/fastDFN.
71. R. B. Smith and M. Z. Bazant, "Multiphase Porous Electrode Theory." *J. Electrochem. Soc.*, **164**, E3291 (2017).
72. M. D. Berliner, D. A. Cogswell, M. Z. Bazant, and R. D. Braatz, "Methods—PETLION: Open-Source Software for Millisecond-Scale Porous Electrode Theory-Based Lithium-Ion Battery Simulations." *J. Electrochem. Soc.*, **168**, 090504 (2021).
73. D. J. Gardner, D. R. Reynolds, C. S. Woodward, and C. J. Balos, "Enabling new flexibility in the SUNDIALS suite of nonlinear and differential/algebraic equation solvers." *ACM Transactions on Mathematical Software (TOMS)*, **48**, 1 (2022).
74. A. C. Hindmarsh, P. N. Brown, K. E. Grant, S. L. Lee, R. Serban, D. E. Shumaker, and C. S. Woodward, "SUNDIALS: Suite of nonlinear and differential/algebraic equation solvers." *ACM Transactions on Mathematical Software (TOMS)*, **31**, 363 (2005).
75. C. Rackauckas and Q. Nie, "Differentialequations. jl-a performant and feature-rich ecosystem for solving differential equations in julia." *Journal of Open Research Software*, **5** (2017).
76. P. Virtanen et al., "SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python." *Nature Methods*, **17**, 261 (2020).
77. J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi - A software framework for nonlinear optimization and optimal control." *Mathematical Programming Computation*, **11**, 1 (2019).
78. G. Pozzato and S. Onori, "A general matlab and comsol co-simulation framework for model parameter optimization: lithium-ion battery and gasoline particulate filter case studies." *Automotive Technical Papers*, 2023 (2023), Warrendale, Pennsylvania, United States.
79. Gt-autolion. https://www.gtisoft.com/gt-autolion Accessed 2024-07- 05.
80. S. Ha and S. Onori, "COBRAPRO: A MATLAB toolbox for physics-based battery modeling and co-simulation parameter optimization." https://github.com/COBRAPROsimulator/COBRAPRO/tree/main.
81. S. Ha and S. Onori, *arXiv* (2024), COBRAPRO: A MATLAB toolbox for Physics-based Battery modeling and co-simulation parameter optimizationarXiv.2404.10022.
82. G. Pozzato, A. Allam, and S. Onori, "Lithium-ion battery aging dataset based on electric vehicle real-driving profiles." *Data in Brief*, **41**, 107995 (2022).
83. J. Le Houx and D. Kramer, "Physics based modelling of porous lithium ion battery electrodes—A review." *Energy Reports*, **6**, 1 (2020).
84. V. R. Subramanian, V. D. Diwakar, and D. Tapriyal, "Efficient Macro-Micro Scale Coupled Modeling of Batteries." *J. Electrochem. Soc.*, **152**, A2002 (2005).
85. S. Ma, M. Jiang, P. Tao, C. Song, J. Wu, J. Wang, T. Deng, and W. Shang, "Temperature effect and thermal impact in lithium-ion batteries: A review." *Progress in Natural Science: Materials International*, **28**, 653 (2018).
86. O. Schmidt, M. Thomitzek, F. Röder, S. Thiede, C. Herrmann, and U. Krewer, "Modeling the Impact of Manufacturing Uncertainties on Lithium-Ion Batteries." *J. Electrochem. Soc.*, **167**, 060501 (2020).
87. M. Verbrugge, D. Baker, B. Koch, X. Xiao, and W. Gu, "Thermodynamic Model for Substitutional Materials: Application to Lithiated Graphite, Spinel Manganese Oxide, Iron Phosphate, and Layered Nickel-Manganese-Cobalt Oxide." *J. Electrochem. Soc.*, **164**, E3243 (2017).
88. D. R. Baker and M. W. Verbrugge, "Intercalate Diffusion in Multiphase Electrode Materials and Application to Lithiated Graphite." *J. Electrochem. Soc.*, **159**, A1341 (2012).
89. L. Xu, J. Cooper, A. Allam, and S. Onori, "Comparative Analysis of Numerical Methods for Lithium-Ion Battery Electrochemical Modeling." *J. Electrochem. Soc.*, **170**, 120525 (2023).
90. L. Mishra, A. Subramaniam, T. Jang, K. Shah, M. Uppaluri, S. A. Roberts, and V. R. Subramanian, "Perspective—Mass Conservation in Models for Electrodeposition/Stripping in Lithium Metal Batteries." *J. Electrochem. Soc.*, **168**, 092502 (2021).
91. A. M. Bizeray, S. Zhao, S. R. Duncan, and 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**, 400 (2015).
92. A. Tulsyan, Y. Tsai, R. B. Gopaluni, and R. D. Braatz, "State-of-charge estimation in lithium-ion batteries: A particle filter approach." *Journal of Power Sources*, **331**, 208 (2016).
93. W. Blajer and K. Kołodziejczyk, "A Geometric approach to solving problems of control constraints: theory and a DAE framework." *Multibody System Dynamics*, **11**, 343 (2004).
94. Y. Cao, S. Li, L. Petzold, and R. Serban, "Adjoint Sensitivity Analysis for Differential-Algebraic Equations: The Adjoint DAE System and Its Numerical Solution." *SIAM Journal on Scientific Computing*, **24**, 1076 (2003).
95. A. C. Hindmarsh, R. Serban, C. J. Balos, D. J. Gardner, D. R. Reynolds, and C. S. Woodward, "User documentation for ida." (2024), https://sundials.readthedocs.io/en/latest/ida.
96. J. Kim, S. Park, S. Hwang, and W. S. Yoon, "Principles and Applications of Galvanostatic Intermittent Titration Technique for Lithium-ion Batteries." *Journal of Electrochemical Science and Technology*, **13**, 19 (2022).

97. D. Lu, M. Scott Trimboli, G. Fan, R. Zhang, and G. L. Plett, "Implementation of a physics-based model for half-cell open-circuit potential and full-cell open-circuit voltage estimates: part I. processing half-cell data." *J. Electrochem. Soc.*, **168**, 070532 (2021).
98. A. Barai, K. Uddin, W. D. Widanage, A. McGordon, and P. Jennings, "A study of the influence of measurement timescale on internal resistance characterisation methodologies for lithium-ion cells." *Sci. Rep.*, **8**, 21 (2018).
99. C. R. Birkl, M. R. Roberts, E. McTurk, P. G. Bruce, and D. A. Howey, "Degradation diagnostics for lithium ion cells." *Journal of Power Sources*, **341**, 373 (2017).
100. G. Pozzato, A. Takahashi, X. Li, D. Lee, J. Ko, and S. Onori, "Core-shell enhanced single particle model for lithium iron phosphate batteries: model formulation and analysis of numerical solutions." *J. Electrochem. Soc.*, **169**, 063510 (2022).
101. D. Lu, M. Scott Trimboli, G. Fan, R. Zhang, and G. L. Plett, "Implementation of a physics-based model for half-cell open-circuit potential and full-cell open-circuit voltage estimates: part II. processing full-cell data." *J. Electrochem. Soc.*, **168**, 070533 (2021).
102. J. Kennedy and R. Eberhart, "Particle swarm optimization." *Proceedings of ICNN'95—International Conference on Neural Networks, Perth, WA, Australia, IEEE***volume 4** (1995).
103. B. Rajabloo, A. Jokar, M. Désilets, and M. Lacroix, "An Inverse Method for Estimating the Electrochemical Parameters of Lithium-Ion Batteries: II: Implementation." *J. Electrochem. Soc.*, **164**, A99 (2017).
104. R. Masoudi, T. Uchida, and J. McPhee, "Parameter estimation of an electrochemistry-based lithium-ion battery model." *Journal of Power Sources*, **291**, 215 (2015).
105. S. Santhanagopalan, Q. Guo, and R. E. White, "Parameter Estimation and Model Discrimination for a Lithium-Ion Cell." *J. Electrochem. Soc.*, **154**, A198 (2007).
106. M. Kim, H. Chun, J. Kim, K. Kim, J. Yu, T. Kim, and S. Han, "Data-efficient parameter identification of electrochemical lithium-ion battery model using deep Bayesian harmony search." *Applied Energy*, **254**, 113644 (2019).