DandeLiion v1: An extremely fast solver for the Newman model of lithium-ion battery (dis)charge

Ivan Korotkin[1,2,5], Smita Sahu[2,3,6], Simon E. J. O'Kane[2,4,7], Giles Richardson[1,2,8], and Jamie M. Foster[2,3,9]

[1]Mathematical Sciences, University of Southampton, University Rd., SO17 1BJ, UK

[2]The Faraday Institution, Quad One, Becquerel Avenue, Harwell Campus, Didcot, OX11 0RA, UK

[3]School of Mathematics and Physics, University of Portsmouth, Lion Terrace, PO1 3HF, UK

[4]Department of Mechanical Engineering, Imperial College London, Exhibition Road, SW7 2AZ, UK

[5]i.korotkin@soton.ac.uk

[6]smita.sahu@port.ac.uk

[7]s.okane@imperial.ac.uk

[8]g.richardson@soton.ac.uk

[9]jamie.michael.foster@gmail.com

June 15, 2021

Abstract

DandeLiion (available at dandeliion.com) is a robust and extremely fast solver for the Doyle Fuller Newman (DFN) model, the standard electrochemical model for (dis)charge of a planar lithium-ion cell. DandeLiion conserves lithium, uses a second order spatial discretisation method (enabling accurate computations using relatively coarse discretisations) and is many times faster than its competitors. The code can be used 'in the cloud' and does not require installation before use. The difference in compute time between DandeLiion and its commercial counterparts is roughly a factor of 100 for the moderately-sized test case of the discharge of a single cell. Its linear scaling property means that the disparity in performance is even more pronounced for bigger systems, making it particularly suitable for applications involving multiple coupled cells. The model is characterised by a number of phenomenological parameters and functions, which may either be provided by the user or chosen from DandeLiion's library. This library contains data for the most commonly used electrolyte ($LiPF_6$) and

a number of common active material chemistries including graphite, lithium iron phosphate (LFP), nickel cobalt aluminium (NCA), and a variant of nickel cobalt manganese (NMC).

Keyword: Lithium-ion battery, Newman model, P2D model, Porous electrode theory, Stiff systems, Solver, Simulation engine, Finite elements.

1 Introduction

Lithium-ion batteries (LIBs) provide rechargeable energy storage at an unrivalled energy and power density, with a high cell voltage, and a slow loss of charge when not in use [1]. These characteristics have lead to their widespread use in consumer electronics, and their increasing dominance in electric vehicle (EV) applications and off grid storage. Driven largely by the incumbent legislation to ban the combustion engine across large parts of the world before 2040, it has been predicted that the demand for LIBs will balloon from 45 GWh/year (in 2015) to 390 GWh/year in 2030 [2]. Thus, the need to improve and optimise LIB technology is especially timely and, in particular, the development of underpinning modelling capabilities promises to significantly accelerate this process. Particularly in the case of EV applications significant challenges remain. These are associated with the demanding requirements made of vehicle batteries, including long service life, rigorous safety standards and good performance under aggressive charge/discharge regimes [3, 4].

A single LIB cell consists of two porous electrodes (an anode and a cathode) separated by a porous spacer (see figure 1) and sandwiched between two current collectors. The cell is bathed in a liquid lithium electrolyte that acts to transport charge, and lithium, between the two electrodes. Each electrode is comprised of an agglomeration of electrode particles formed from active materials into which lithium ions can intercalate. For modelling purposes electrode particles are often assumed to be spherical. Under discharge conditions Li^+ ions, which have a greater chemical energy in the anode material than the cathode material, deintercalate from the anode particles, migrate through the electrolyte and across the porous separator to the cathode where they intercalate into the cathode particles. The transport of charge, from anode to the cathode, that results from this migration of the positively charged Li-ions gives rise to a potential difference, between the two electrodes, that can be used to drive a current through an external circuit.

The electrochemistry and electrical behaviour of a LIB cell is typically modelled by the Doyle Fuller Newman (DFN) model [5, 6, 7], which is also often called porous electrode theory or the P2D model. This describes the charge transport and Li-ion migration within the cell. In particular it couples nonlinear diffusion models for Li-ion transport within the electrode particles to a semi-phenomenological model for the electrolyte, which is able to accurately capture ion transport and electrical conduction, via a Butler-Volmer model [7, 8] that quantifies the rate of (de-)intercalation of Li-ion from the surfaces of the electrode particles. More details of this model, and its relationship to the underlying physics and chemistry of the device, can be found in [8, 9, 10].

Although the DFN model is to some extent the gold standard in engineering simulations of LIBs it is nevertheless prohibitively computationally expensive in many applications. In

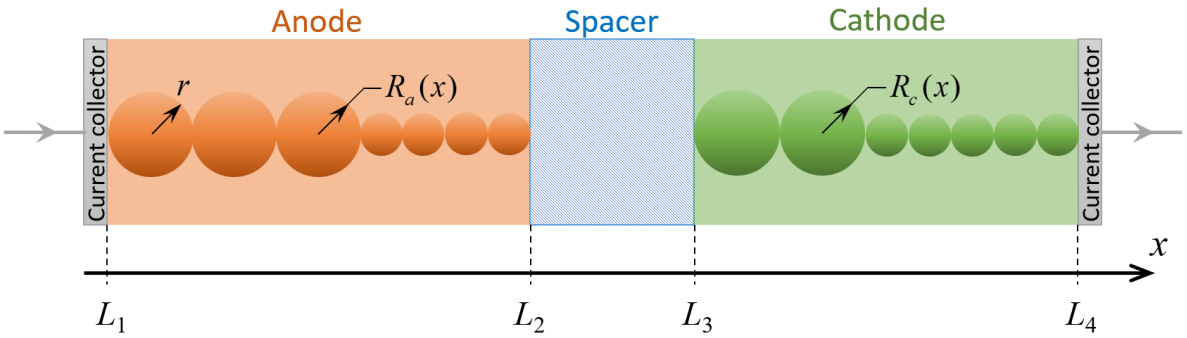

Figure 1: Schematic of a planar LIB cell. The macroscopic and microscopic coordinates, x and r, are indicated, along with the particle radii and positions between the different cell components.

particular, composite cells such as pouch and cylindrical cells have heterogeneous temperature distributions and therefore required a DFN solution to be carried out at each point in space and coupled to a three-dimensional heat transport equation. This results in a five-dimensional problem that is extremely compuationally challenging. Various approaches are adopted to reduce the complexity of this problem including equivalent circuit modelling of the cell and more recently systematic (single particle) asymptotic reductions of the DFN model which reduce the dimension of the DFN model by one, see [11, 12]. Rather than simplify the model, our approach here is to tailor highly-efficient numerical methods to the DFN model and thereby reduce the computational time to a level that is acceptable for the type of computations that we might wish to perform.

The numerical software presented in this work (DandeLiion v1) is designed to solve the DFN model and is motivated by the pressing need for fast, and powerful, numerical code that is capable of solving computationally expensive problems in battery design, such as the simulation of the thermally coupled electrochemical behaviour of composite cells (*e.g.* pouch cells and jelly-roll cells), battery modules, and (even) entire battery packs. It also has the potential to significantly enhance other computationally expensive tasks such as the optimisation of cell design and estimation of parameters from experimental data.

The rest of this work is devoted to the description of the numerical procedure, adopted in DandeLiion, for the solution of the DFN model of charge transport in a single LIB cell.

1.1 Software performance and operation

The DandeLiion solver is based upon a method of lines approach to the solution of the system of mixed parabolic-elliptic partial differential equations (PDEs) that comprise the DFN model. In this approach the DFN PDEs are first discretised in space to yield a large system of coupled time-evolving ODEs and algebraic equations. This system of differential algebraic equations (DAEs) is efficiently solved (with the default absolute and relative tolerances of 10^{-6}) using an in-house solver, written by the authors and based upon Backward Differentiation Formulae and adaptive time stepping [13, 14].

The DandeLiion solver has been validated against (i) in-house code implemented in MAT-

LAB [15], (ii) the Battery Library in Dymola [18], a proprietary code, (iii) COMSOL Multiphysics, a commercial package [19], (iv) PyBaMM, an open source project [20], and (v) the experiments and simulations described in the work of Ecker *et al.* [16, 17]. For the cross-verification with Dymola, the code was parametrised using the same model and battery properties as described in [21]. In [11] it has been compared to (vi) an approximate, simplified, reduced-order battery cell model, showing very good agreement between the two different approaches, even for relatively high discharge rates up to around 12C. An example comparison between PyBaMM, experiment and DandeLiion is shown below in §5 and further work [11, 22] also verify DandeLiion against other experiments and simulations.

The DandeLiion solver works very much faster than: (i) our previous MATLAB implementation of a DFN solver (which is based on `ode15s` with the analytic Jacobian pattern provided for the solver for increased speed); (ii) the implementation of the model in the Dymola Battery Library [18]; (iii) the implementation in COMSOL Multiphysics [19]; and (iv) than the open source implementation PyBaMM [20]. To quantify this, our MATLAB implementation, which is comparable in speed to Dymola, was outperformed by DandeLiion by a factor of around 100 in terms of reduced computational time for a single cell discharge. Another direct comparison has been carried out with both COMSOL Multiphysics 5.3a and 5.6 using a Kokam 7.5Ah pouch cell parameterised according to Ecker *et al.* [17] with non-linear diffusivity in solid particles. COMSOL Multiphysics takes between 3 minutes 45 seconds and 4 minutes to run a set of three discharge curves (1C, 3C and 5C) at 298K with 40 mesh points in the r direction (particles) for the negative electrode and 20 mesh points for the positive. In the x direction (electrolyte) the number of mesh points is roughly 50 in the negative electrode, 10 in separator and 40 in the positive electrode. For DandeLiion, it takes only about 1.7 seconds (over 100 times faster) to solve all the three cases with the same parametrisation and computational grid. When simulations are submitted to our server, computations are carried out 'in the cloud' and there is a fixed overhead of around 7-10 seconds associated with setting up of the simulation and delivery of the results. Even if one were to include these overheads in the compute time, DandeLiion would still perform this example faster than its counterparts. Moreover, we emphasize that these overheads are independent of the size of the computation, and are therefore insignificant in sizable problems.

The disparity in performance is significantly greater for larger problems, such as pouch and cylindrical cells where the advantages of the linear scaling properties of DandeLiion become even more pronounced. Furthermore, in contrast to Dymola and PyBaMM, DandeLiion uses a second order spatial discretisation and therefore requires many fewer space points to achieve the same accuracy as these other solvers, which are only first order accurate in space. A full discharge cycle of a battery at a moderate (1C) discharge rate that involves solution of a system of approximately 7000 coupled nonlinear DAEs takes less than a second of simulation time for DandeLiion on a standard desktop computer. For comparison, the solution of the same problem (using the same hardware) takes around one minute, or even more in MATLAB and this gap in code performance becomes more pronounced for bigger systems. The number of DAEs that the DandeLiion solver can handle on a desktop computer with 16 Gb of RAM (available in most standard desktops) is enough to solve approximately 2×10^7 DAEs and can be increased beyond 10^8 depending on the machine's RAM. Furthermore code performance is not hampered when the number of DAEs increases, and the simulation time scales *linearly* with the number of DAEs being solved (see Figure 2). This is

in contrast to most other DFN codes whose simulation times scale *quadratically* with system size. Such codes are therefore prohibitively computationally expensive when used for large computations.

In Figure 2, the total simulation time on a standard desktop (with 16 Gb of RAM) is plotted against the total number of DAEs on a log-log scale. For small numbers of equations ($< 10^4$) the log-log plot is not linear, due to the cost of fixed overheads, but above 10^4 DAEs we were able to verify that the solver demonstrates linear scalability up to at least 2×10^7 coupled equations. Note that each point in the plot represents an individual simulation with up to 128 battery cells (DFN models) in a stack, and it shows that even with the increasing number of coupled (electrically, via potentiostatic boundary conditions) DFN models the solver performance does not degrade and the linear trend is preserved. Importantly, this opens the possibility of simulating multidimensional systems. In particular, it will enable fully thermally coupled simulations of 3D composite cells, such as pouch cells, which are made by stacking a large number of (typically around 50) individual cells on top of each other; this is to be the subject of an upcoming publication. Heat generation within such cells can lead to significant heterogeneities in the temperature distribution, which in turn leads to heterogeneities in the electrochemical properties of individual cells (which are highly sensitive to temperature). Since the DFN model for a single isothermal cell is two-dimensional in space (one micro dimension measuring distance from the centre of an electrode particle and one macroscopic cell dimension measuring distance across the cell) the thermally coupled model that needs to be solved for a composite cell is five dimensional (3 macroscopic pack dimensions, 1 macroscopic cell and 1 micro dimension). Such problems are extremely computationally challenging and require fast and efficient solvers, such as DandeLiion. Other computationally intensive applications for which efficient code is highly desirable include parameter estimation and cell optimisation routines, both of which require that multiple simulations, using different sets of parameters, are performed on a single cell.

DandeLiion does not require installation and is available to be used 'in the cloud' at `dandeliion.com`. The website hosts comprehensive documentation as well as a series of tutorial videos aimed at educating new users on the use of this tool, which cover a range of topics, including how to simulate a full discharge cycle, modification of the model parameters and functions, how to simulate drive cycles, creating and adding a user-defined electrode chemistries, and setting up graded electrodes (electrodes with different particle sizes). Several pre-defined examples are available, and these are intended to serve as templates which can be adapted for specific customised simulations thereby lowering the barrier to entry for new users.

2 The DFN model and software implementation

DandeLiion is a framework for simulating LIB cell charge and discharge. It solves a 1+1D (pseudo 2D) DFN porous electrode model that was established in [6, 5, 7] and is reviewed in detail in [10]. The current version allows the user to choose from a library of preprogrammed parameterisations for the electrolyte and electrode chemistries or, alternatively, to specify their own parametrisations. The model parameters may be changed by the user by editing a simple web form. Complete web forms submit a simulation to the queue and computations

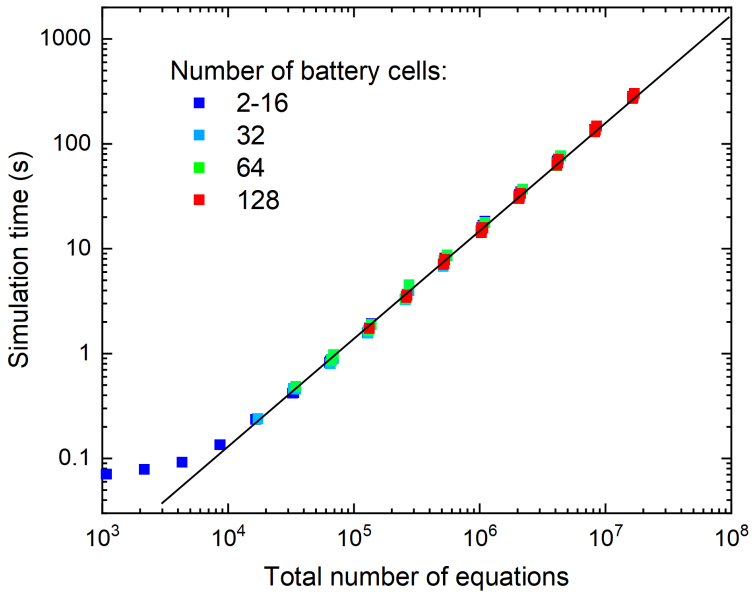

Figure 2: Simulation time in seconds vs the total number of differential-algebraic equations (DAEs) in the system. In this test, a stack of up to 128 battery cells (DFN models) has been simulated at 1C discharge rate from fully pre-charged state until fully discharged using different computational grids. Each point in the plot represents a separate simulation with one particular grid size. The density of the spatial grid varies from relatively coarse, 10 nodes in the electrolyte and 50 in each electrode particle, up to relatively fine, 1280 nodes in the electrolyte and 800 nodes in each particle. Each battery cell in a stack was parametrised according to [16, 17]. All cells are connected via potentiostatic boundary conditions (parallel battery connection). The test was performed on a desktop with `Intel Core i7-8700` CPU and 16 Gb of RAM.

are carried out on our dedicated server free of charge. A concise view of the results of finished computations can be viewed live in the web browser and we also provide an option for users to download the raw output of simulations for in-depth analysis using the software of their choice.

The core DandeLiion code is written in C++ which is partially responsible for its fast performance. It is based on a 2nd order spatial discretisation of the system of PDEs which comprise the DFN model. A finite element discretisation is employed in the macroscopic dimension, x (which measures distance across the cell), as described in detail below and in the microscopic dimension, r (which measures distance from the centre of each electrode particle), we use a control volume method described in [23]. These methods are both 2nd order accurate and, crucially, are conservative. The latter point means that when they are applied in tandem to the macroscopic and microscopic lithium conservation equations, that form part of the DFN model, they ensure global lithium conservation throughout the device. This property is particularly important when multiple (dis)charge cycles are performed and ensures that the battery's capacity is retained.

The finite element and control volume methods are applied to the DFN model equations and in doing so spatial derivatives are removed. The spatially-discretised DFN model can be written as a large coupled system of time-dependent DAEs, i.e. $\mathbf{M}\dot{\mathbf{u}} = \mathbf{f}(\mathbf{u})$, where $\mathbf{u}$ is a vector containing the time-dependent values of the model variables at the computational grid points, the mass matrix is denoted by $\mathbf{M}$, and the "right-hand side function" is denoted by $\mathbf{f}(\mathbf{u})$, see equation (40). DAE systems are typically more problematic to solve than a system comprised solely of coupled ODEs [24], but there are commercial solvers aimed at solving such systems, such as MATLAB's `ode15s` [25]. Solving the DAEs that result from the spatial discretisation of the DFN model is by far the most computationally expensive part of the solution procedure and for this reason we have developed a specialised in-house DAE solver, as part of the DandeLiion code, which is based on implicit variable-order (2 to 6) backward differentiation formulae [13, 14] and an optimised Newton root-finding method. The DAE solver used by DandeLiion is unique to this project and has been optimised for the specific purpose of solving the DFN model. In this it differs significantly from other numerical tools developed by the authors.

2.1 The DFN Model

In what follows we lay out the full cell 1+1d DFN model [5, 6, 7] that is solved by DandeLiion; for a more detailed description of the physics and chemistry underlying this model the reader is referred to Newman's book [8] and the review article [10]. The version of the 1+1d DFN model considered here describes a one dimensional cell lying between $x = L_1$ and $x = L_4$ (see figure 1), consisting of

$$\begin{array}{ll} \text{an anode in} & L_1 < x < L_2, \\ \text{a separator in} & L_2 < x < L_3, \\ \text{and a cathode in} & L_3 < x < L_4. \end{array}$$

The model comprises one-dimensional macroscopic equations posed across the width of the cell $L_1 < x < L_4$. These describe electrical conduction in the solid matrices of the anode and cathode and lithium ion transport and conduction in the electrolyte that fills the pores of the

electrode matrix. They couple to one-dimensional spherically symmetric microscopic lithium transport equations posed in representative spherical electrode particles, which occupy the regions $0 \leq r < R_a(x)$ in the anode and $0 \leq r < R_c(x)$ in the cathode. Here $R_a(x)$ and $R_c(x)$, which are allowed to vary in space to allow for the possibility of particle grading, give the radii of the anode and cathode particles, respectively, as a function of x. The full cell DFN model is formulated below in equations (1)-(16); the associated model variables are listed, and described, in Table 1, and the model parameters and functions are catalogued in Table 2.

Variable	**Description**	**Units**
x	Distance across cell	m
t	Time	s
r	Distance from centre of electrode particle	m
F	Faraday constant	$\mathrm{s\,A\,mol^{-1}}$
c	Ion concentration in electrolyte	$\mathrm{mol\,m^{-3}}$
N_-	Average flux of negative counterions in electrolyte	$\mathrm{mol\,m^{-2}s^{-1}}$
Φ	Electric potential w.r.t. lithium electrode in electrolyte	V
j	Average current density in electrolyte	$\mathrm{A\,m^{-2}}$
j_n	Current density on surface of electrode particles flowing from electrode particle to electrolyte	$\mathrm{A\,m^{-2}}$
j_a	Average current density in anode	$\mathrm{A\,m^{-2}}$
j_c	Average current density in cathode	$\mathrm{A\,m^{-2}}$
Φ_a	Electric potential in anode	V
Φ_c	Electric potential in cathode	V
c_a	Lithium-ion concentration in anode particles	$\mathrm{mol\,m^{-3}}$
c_c	Lithium-ion concentration in cathode particles	$\mathrm{mol\,m^{-3}}$
η_a	Overpotential between electrolyte and anode particles	V
η_c	Overpotential between electrolyte and cathode particles	V
$V(t)$	Potential difference across device	V

Table 1: Description of the variables and constants used in the formulation of the full cell DFN model

Param./ Ftn.	Description	Units
T	Absolute temperature	K
$\mathcal{B}(x)$	Permeability factor in electrode matrix	dim'less
$\epsilon_l(x)$	Volume fraction of electrolyte in electrode matrix	dim'less
$b(x)$	Brunauer-Emmett-Teller (BET) surface area	m^{-1}
$D_{\mathrm{e}}(c)$	Ionic diffusivity of electrolyte: function of concn.	$\mathrm{m}^2\mathrm{s}^{-1}$
t_0^+	Transference number	dim'less
$\kappa(c)$	Electrolyte conductivity as function of concentration	$\mathrm{A\,m}^{-1}\mathrm{V}^{-1}$
$\sigma_a(x)$	Anode conductivity as function of concentration	$\mathrm{A\,m}^{-1}\mathrm{V}^{-1}$
$\sigma_c(x)$	Cathode conductivity as function of concentration	$\mathrm{A\,m}^{-1}\mathrm{V}^{-1}$
$R_a(x)$	Radius of anode particles as function of position	m
$R_c(x)$	Radius of cathode particles as function of position	m
$c_a^{\max}$	Max. lithium concentration in anode particles	$\mathrm{mol\,m}^{-3}$
$c_c^{\max}$	Max. lithium concentration in cathode particles	$\mathrm{mol\,m}^{-3}$
k_a	Butler-Volmer constant in anode	$\mathrm{mol}^{-1/2}\mathrm{m}^{5/2}\mathrm{s}^{-1}$
k_c	Butler-Volmer constant in cathode	$\mathrm{mol}^{-1/2}\mathrm{m}^{5/2}\mathrm{s}^{-1}$
$U_{eq,a}(c_a)$	Open-circuit voltage: function of Li^+ concn. in anode	V
$U_{eq,c}(c_c)$	Open-circuit voltage: function of Li^+ concn. in cathode	V
$D_a(c_a)$	Li^+ diffusivity anode: function of Li^+ concn.	$\mathrm{m}^2\ \mathrm{s}^{-1}$
$D_c(c_c)$	Li^+ diffusivity cathode: function of Li^+ concn.	$\mathrm{m}^2\ \mathrm{s}^{-1}$
$I(t)$	Current flow into cell	A
$\mathcal{R}_{cont}$	Total contact resistance	$\mathrm{V\,A}^{-1}$
A	Electrode cross-sectional area	m^2
c_0	Initial ionic concentration in electrolyte	$\mathrm{mol\,m}^{-3}$
$c_{a,0}$	Initial ionic concentration in anode	$\mathrm{mol\,m}^{-3}$
$c_{c,0}$	Initial ionic concentration in cathode	$\mathrm{mol\,m}^{-3}$

Table 2: User specified functions and parameters for full cell DFN model

The Macroscopic equations

$$\epsilon_l(x)\frac{\partial c}{\partial t}+\frac{\partial N_-}{\partial x}=0,\quad N_-=-\mathcal{B}(x)D_{\mathrm{e}}(c)\frac{\partial c}{\partial x}-(1-t_0^+)\frac{j}{F}\quad\text{in}\quad L_1<x<L_4. \tag{1}$$

$$\frac{\partial j}{\partial x}=b(x)j_n,\qquad j=-\mathcal{B}(x)\kappa(c)\left(\frac{\partial\Phi}{\partial x}-\frac{2RT}{F}\frac{1-t_0^+}{c}\frac{\partial c}{\partial x}\right)\quad\text{in}\quad L_1<x<L_4, \tag{2}$$

$$\frac{\partial j_a}{\partial x}=-b(x)j_n,\quad j_a=-\sigma_a\frac{\partial\Phi_a}{\partial x}\quad\text{in}\quad L_1<x<L_2, \tag{3}$$

$$\frac{\partial j_c}{\partial x}=-b(x)j_n,\quad j_c=-\sigma_c\frac{\partial\Phi_c}{\partial x}\quad\text{in}\quad L_3<x<L_4, \tag{4}$$

$$j_n=\begin{cases}2Fk_ac^{1/2}\left(c_a|_{r=R_a(x)}\right)^{1/2}\left(c_a^{\max}-c_a|_{r=R_a(x)}\right)^{1/2}\sinh\left(\dfrac{F\eta_a}{2RT}\right) & \text{in}\quad L_1\le x<L_2,\\ 0 & \text{in}\quad L_2<x<L_3,\\ 2Fk_cc^{1/2}\left(c_c|_{r=R_c(x)}\right)^{1/2}\left(c_c^{\max}-c_c|_{r=R_c(x)}\right)^{1/2}\sinh\left(\dfrac{F\eta_c}{2RT}\right) & \text{in}\quad L_3\le x<L_4,\end{cases} \tag{5}$$

$$\eta_a=\Phi_a-\Phi-U_{eq,a}(c_a|_{r=R_a(x)}),\qquad \eta_c=\Phi_c-\Phi-U_{eq,c}(c_c|_{r=R_c(x)}). \tag{6}$$

Variables in the anode and cathode are distinguished by their subscripts: we use the variables Φ_a, j_a, c_a, η_a in the anode ($L_1 < x < L_2$) and Φ_c, j_c, c_c, η_c in the cathode ($L_3 < x < L_4$). Furthermore the electrode particles in the anode and cathode have different electrical properties and so are characterised by different equilibrium potential functions, $U_{eq,a}(c_a)$ in the anode and $U_{eq,c}(c_c)$ in the cathode.

Macroscopic boundary and interface conditions Here the macroscopic boundary and interface conditions on the model are

$$j_a|_{x=L_1} = \frac{I(t)}{A}, \quad N_-|_{x=L_1} = 0, \quad j|_{x=L_1} = 0, \tag{7}$$

$$j_a|_{x=L_2} = 0, \tag{8}$$

$$j_c|_{x=L_3} = 0, \tag{9}$$

$$j_c|_{x=L_4} = \frac{I(t)}{A}, \quad N_-|_{x=L_4} = 0, \quad j|_{x=L_4} = 0. \tag{10}$$

representing galvanostatic discharge at a current $I(t)$ which flows into the anode current collector on $x = L_1$ through the anode particles and out through the cathode current collector on $x = L_4$ through the cathode particles (figure 1). No electronic current passes through the electronically insulating separator.

Microscopic equations and boundary conditions The microscopic equations and boundary conditions on the model are given by

$$\left.\begin{aligned} &\frac{\partial c_a}{\partial t} = \frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2 D_a(c_a)\frac{\partial c_a}{\partial r}\right) \quad \text{in} \quad 0 < r < R_a(x) \\ &c_a \text{ bounded on } r = 0, \qquad -D_a(c_a)\frac{\partial c_a}{\partial r}\bigg|_{r=R_a(x)} = \frac{j_n}{F} \end{aligned}\right\} \quad \text{in} \quad L_1 < x < L_2, \tag{11}$$

$$\left.\begin{aligned} &\frac{\partial c_c}{\partial t} = \frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2 D_c(c_c)\frac{\partial c_c}{\partial r}\right) \quad \text{in} \quad 0 < r < R_c(x) \\ &c_c \text{ bounded on } r = 0, \qquad -D_c(c_c)\frac{\partial c_c}{\partial r}\bigg|_{r=R_c(x)} = \frac{j_n}{F} \end{aligned}\right\} \quad \text{in} \quad L_3 < x < L_4, \tag{12}$$

where D_a and D_c are the diffusivities of Li^+ in the of the anode and cathode particles respectively.

Initial conditions Constant initial conditions are provided for the ion concentration in the electrolyte

$$c|_{t=0} = c_0, \tag{13}$$

and likewise for those in the active materials in the anode and cathode

$$c_a|_{t=0} = c_{a,0}, \qquad c_c|_{t=0} = c_{c,0}, \tag{14}$$

where c_0, $c_{a,0}$, and $c_{c,0}$ are provided by the user.

The full cell potential The results of solution to the full cell DFN model and a specified galvanostatic current $I(t)$ can be used to compute the potentials at the anode and cathode current collectors V_a and V_c, respectively via the relations

$$V_a(t) = \Phi_a\big|_{x=L_1}, \qquad V_c(t) = \Phi_c\big|_{x=L_4}. \tag{15}$$

and hence the potential drop across the full cell (*i.e.* the cell voltage) is given by

$$V(t) = V_c(t) - V_a(t) - \mathcal{R}_{cont} I(t). \tag{16}$$

where $\mathcal{R}_{cont}$ is the contact resistance of the cell.

2.2 Software functionalities

In the most basic case the user can specify a current draw from/supply to the cell and the code will solve for the internal concentration, potential and current density profiles as well as the cell voltage during discharge/charge until the device reaches a user-defined cut-off potential. The results of a simulation in such a scenario are discussed in §5. However, DandeLiion can also be used in a number of more sophisticated ways and it has the capability to simulate: (i) a variety of cell chemistries, (ii) realistic drive cycles, (iii) graded electrodes in which particle size varies across the electrode, and (iv) GITT (Galvanostatic Intermittent Titration Technique) experiments. The user may refer to the 'Getting Started' page on `dandeliion.com` for tutorials.

Materials and Electrolyte library Data for the electrolyte $LiPF_6$ in the form of functions for electrolyte diffusivity $D_e(c)$, electrolyte conductivity $\kappa(c)$ and a value for the transference number t_0^+ is provided in the DandeLiion's parameter library. Similarly data is provided for the electrode materials graphite (Li_xC_6), LNC ($Li_x(Ni_{0.4}Co_{0.6})O_2$), LFP ($Li_xFePO_4$), and NMC ($LiNi_{1-x-y}Mn_xCo_yO_2$). For each of these materials we provide the open circuit voltage $U_{eq}(c_s)$ as a function of the concentration of intercalated lithium. For graphite and LNC we also provide the lithium diffusivity $D_s(c_s)$ within the material as a function of concentration of intercalated lithium. In the case of $LiFePO_4$ we assume a constant diffusivity value $D_s = 8 \times 10^{-18}$ m^2s^{-1} [26]. In future releases this library will be expanded enlarging the choice of pre-defined chemistries and materials. All other parameter values and functions are taken from [16, 17, 27].

3 Spatial discretisation of the DFN model

In this section we discuss the second order spatial discretisation of the DFN model (1)-(16) that leads to the system of DAEs solved by DandeLiion. This is based on a control volume method for the microscopic equations (11)-(12) that has been previously given in [23] and a novel finite element method for the macroscopic equations (1)-(10), which we describe in detail below.

We note that there are numerous other approaches that have been used in the literature to discretise the DFN model and optimise its numerical solution. Some of this past work

has focussed on speeding up the solution for transport in the active particles (the most costly part of the solution process). For instance, Subramanian *et al.* [37] focussed on leveraging approximate solutions to speed up computations for lithium transport in the electrode particles, by assuming that the Li concentration is parabolic in the particle radius r. In [38] finite difference approximations and polynomial representations were combined to reduce the coupled PDEs, which comprise the DFN model, to a system of DAEs which is smaller than that obtained by direct discretisation alone. Bizeray *et al.* [39] used Chebyshev orthogonal collocation and showed that computation time could be reduced by a factor of 10-100, in comparsion to the finite difference method, where it was insisted that both methods achieve the same accuracy. Dao *et al.* [40] used a Galerkin spectral method with sinusoidal basis functions to solve the equation in the electrolyte but showed that its advantages are limited to low C-rates. Padé approximants were first used by Fathy *et al.* [42] and extended by [43] for a reduced-order model. We emphasize that whilst all of these aforementioned approaches are at least partially successful at enhancing the solution speed, in comparison to the standard finite difference method, their applicability is restricted to a linear Fickian diffusion model of lithium ion transport. They are therefore not applicable in our case where, in order to obtain good agreement with experiment, the lithium transport in the active material is modelled by a nonlinear diffusion equation. Other work has looked at strategies for optimising the solution process in the electrolyte. For example, Cai *et al.* [41] developed a reduced-order model using proper orthogonal decomposition and showed that this method is 30 times faster than the finite difference method implemented in COMSOL Multiphysics. However, they restricted their attention to linear diffusion models both in the electrode particles and in the electrolyte whereas we allow both to be nonlinear. Finally, we note that one might consider using higher order discretisations as a strategy to increase computational speed. However, we do not go beyond second order because we want to preclude the possibility of introducing spurious oscillations in the solution which frequently occur when using high order methods in problems in which their are discontinuities and/or rapid changes in coefficients (as is the case here at the interfaces between the electrodes, separator and current collectors) [44, 45].

3.1 Finite element discretisation of the macroscopic equations

Here, we discuss the spatial second-order finite element discretisation for the macroscopic equations (1)-(6), for the electrolyte and current transport in the solid parts of the anode and cathode. A similar method has been used for a related systems of equations describing charge transport in solar cells in [28]. The microscopic diffusion equations (11)-(12) are discretised using the conservative control volume method given in Zeng *et al.* [23] which is chosen both for its second order accuracy, which matches the rate convergence of the scheme that we use for the macroscopic equations, and because it gives direct access to the concentration on the surface of the particle without the need for extrapolation. This latter feature is particularly important because the charge transfer reaction rate, given by the Butler-Volmer equations (5), depends strongly upon the surface concentration of lithium and any errors made in computing its value gives rise to large errors in the transfer current undermining the quality of the simulations.

For a positive integer N, let $\{L_1 = x_0 < x_1 ... < L_4 = 1\}$ be a partition of $[L_1, L_4]$ into the

subintervals (x_{i-1}, x_i), $1 \leq i \leq N$ with grid spacing $\Delta_{i+\frac{1}{2}} = x_{i+1} - x_i$. The computational grid is comprised of $N+1$ points. We apply the approach described in [29] to derive the finite element descretisation. The idea is to approximate dependent variables as a linear combination of piecewise linear basis functions (aka 'hat' or 'tent' functions). For a generic dependent variable, say w, we write

$$w(x,t) = \sum_{i=0}^{i=N} w_i(t)\psi_i(x) \quad \text{where} \quad \psi_i(x) = \begin{cases} \frac{x-x_{i-1}}{x_i - x_{i-1}} & x \in (x_{i-1}, x_i) \\ \frac{x_{i+1}-x}{x_{i+1}-x_i} & x \in (x_i, x_{i+1}) \\ 0 & x \notin (x_{i-1}, x_{i+1}) \end{cases}, \tag{17}$$

in which $\psi_i(x)$ is referred to as the basis functions. By eliminating N_-, j, j_a, and j_c from equations (1)-(6), we obtain the following set of macroscopic PDEs:

$$\epsilon_l(x)\frac{\partial c}{\partial t} + \frac{\partial}{\partial x}\left(q_1(x,c)\frac{\partial c}{\partial x} + q_2(x,c)\frac{\partial \Phi}{\partial x}\right) = 0, \tag{18}$$

$$\frac{\partial}{\partial x}\left(q_3(x,c)\frac{\partial c}{\partial x} + q_4(x,c)\frac{\partial \Phi}{\partial x}\right) = b(x)j_n, \tag{19}$$

$$\sigma_a \frac{\partial}{\partial x}\left(\frac{\partial \Phi_a}{\partial x}\right) = b(x)j_n, \tag{20}$$

$$\sigma_c \frac{\partial}{\partial x}\left(\frac{\partial \Phi_c}{\partial x}\right) = b(x)j_n, \tag{21}$$

in which

$$\begin{aligned} &q_1(x,c) = -\mathcal{B}(x)\left(D_e(c) + \kappa(c)\frac{2RT(1-t_0^+)^2}{F^2 c}\right), \quad q_2(x,c) = \mathcal{B}(x)\kappa(c)\frac{(1-t_0^+)}{F}, \\ &q_3(x,c) = \mathcal{B}(x)\kappa(c)\frac{2RT(1-t_0^+)}{Fc}, \quad \text{and} \quad q_4(x,c) = -\mathcal{B}(x)\kappa(c). \end{aligned} \tag{22}$$

These are to be solved subject to the boundary conditions (7)-(10). Each of the equations (18)-(21) have the following form

$$\gamma_1(x)\frac{\partial w}{\partial t} = \frac{\partial}{\partial x}\left(\gamma_2(x,w)\frac{\partial w}{\partial x} + \gamma_3(x,w)\frac{\partial \phi}{\partial x}\right) + S(x,v,w). \tag{23}$$

and, in the interests of brevity, we shall now discuss how the finite element method is applied to (23) rather than discussing each of equations (18)-(21) individually.

The spatially discretised equations are obtained by using the approximation (17) in (23), multiplying by a test function $\psi_j(x)$, $j = 0, ..., N$, and integrating over the macroscopic domain $(0,1)$ to obtain

$$\begin{aligned} \sum_{i=0}^{i=N} \frac{dw_i}{dt}\int_0^1 \gamma_1(x)\psi_i\psi_j dx = &\left(\gamma_2(x,w)\frac{\partial w}{\partial x} + \gamma_3(x,w)\frac{\partial \phi}{\partial x}\right)\psi_j\Big|_{x=0}^{x=1} \\ &- \sum_{i=0}^{N}\left(w_i(t)\int_0^1 \gamma_2(x,w)\psi_i'\psi_j' dx + \phi_i(t)\int_0^1 \gamma_3(x,w)\psi_i'\psi_j' dx\right) \\ &+ \int_0^1 S(x,v,w)\psi_j dx. \end{aligned} \tag{24}$$

The first-term on the right-hand side can be calculated using the appropriate boundary conditions, (7)-(10). In general, the remaining integrals in (24) cannot be integrated exactly and further approximations are needed in order to progress. We adopt the approach given in [28] and replace the functions γ_1, γ_2, γ_3 and S appearing in the integrands by functions that are piecewise constant over each subinterval, $x \in (x_i, x_{i+1})$, and have a value equal to that of the function (17) at the midpoint of that interval. The integral on the left-hand side of (24) is treated as follows

$$\begin{aligned}\int_0^1 \gamma_1(x)\psi_i\psi_j dx &= \int_{x_{i-1}}^{x_i} \gamma_1(x)\psi_i\psi_j dx + \int_{x_i}^{x_{i+1}} \gamma_1(x)\psi_i\psi_j dx \\ &\approx \gamma_1(x_{i-1/2}) \int_{x_{i-1}}^{x_i} \psi_i\psi_j dx + \gamma_1(x_{i+1/2}) \int_{x_i}^{x_{i+1}} \psi_i\psi_j dx.\end{aligned} \tag{25}$$

The first integral on the right-hand side in equation (24) can be approximated as follows

$$\begin{aligned}\int_0^1 \gamma_2(x,w)\psi_i'\psi_j' dx &= \int_{x_{i-1}}^{x_i} \gamma_2(x,w)\psi_i'\psi_j' dx + \int_{x_i}^{x_{i+1}} \gamma_2(x,w)\psi_i'\psi_j' dx \\ &\approx \gamma_2\left(x_{i-1/2}, w\big|_{x=x_{i-1/2}}\right) \int_{x_{i-1}}^{x_i} \psi_i'\psi_j' dx \\ &+ \gamma_2\left(x_{i+1/2}, w\big|_{x=x_{i+1/2}}\right) \int_{x_i}^{x_{i+1}} \psi_i'\psi_j' dx.\end{aligned} \tag{26}$$

Treatment of second integral on right-hand side of (24) follows analogously but with $\gamma_2(x,w)$ replaced by $\gamma_3(x,w)$. Finally we approximate the final integral in (24) by writing

$$\begin{aligned}\int_0^1 S(x,v,w)\psi_j dx \approx &\frac{\Delta_{j-1/2}}{2} S\left(x_{j-1/2}, v\big|_{x=x_{j-1/2}}, w\big|_{x=x_{j-1/2}}\right) \\ &+ \frac{\Delta_{j+1/2}}{2} S\left(x_{j+1/2}, v\big|_{x=x_{j+1/2}}, w\big|_{x=x_{j+1/2}}\right)\end{aligned} \tag{27}$$

The errors incurred in using these approximations are second order (*i.e.* their error decays proportional to the square of the grid spacing), just like the piecewise linear approximation for the dependent variables embedded in (17). Hence the finite element discretisation retains its second order convergence rate despite the additional approximations. The integrals on the right-hand side of equations (25)-(26) have integrands that depend solely upon the basis functions and their derivatives, and so can be computed exactly (for details see Appendix A). This observation leaves us in a position to write down the DAE system arising from the spatial discretisation of the macroscopic PDEs (1)-(6) of the Doyle-Fuller-Newman model.

3.2 Finite element implementation

In order to write down the spatially discretised system of equations in a concise form we introduce three discrete operators: a difference operator $\mathcal{D}_i$, an operator for evaluation of dependent variables at a mid point $\mathcal{J}_i$ and a linear operator $\mathcal{L}_i$. These act on a column vector $\mathbf{w}$ with the entries

$$w_i = w|_{x=x_i}, \quad \text{for} \quad i = 0, ..., N \tag{28}$$

for a generic dependent variable w they are defined as follows:

$$\begin{aligned}\left.\frac{\partial w}{\partial x}\right|_{x=x_{i+1/2}} &\approx \mathcal{D}_{i+1/2}(\mathbf{w}) = \frac{w_{i+1}-w_i}{\Delta_{i+1/2}} \\ w\big|_{x=x_{i+1/2}} &\approx \mathcal{J}_{i+1/2}(\mathbf{w}) = \frac{w_{i+1}+w_i}{2} \\ \mathcal{L}_i(\mathbf{w}) &= \frac{1}{6}\Delta_{i+1/2}w_{i+1} + \frac{1}{3}(\Delta_{i+1/2}+\Delta_{i-1/2})w_i + \frac{1}{6}\Delta_{i-1/2}w_{i-1}.\end{aligned} \tag{29}$$

Let $\mathbf{x}$ be a column vector of nodal points with the i^{th} entry x_i for $i = 0, ..., N$. We seek to predict the electrolyte lithium concentration $c(x,t)$ and so, following (17) and (28), we aim to find $\mathbf{c}(t)$ whose i^{th} entry is $c_i = c(x_i,t)$ for $i = 0, ..., N$. The same is true for the electrolyte whose time-dependent values at the $N+1$ nodes x_i, $i = 0, 1, ..., N$, are collated in the column vector $\mathbf{\Phi}(t)$. Similarly for the anode and cathode potentials whose time-dependent values at the N_a+1 and N_c+1 nodes respectively are collated in the column vectors $\mathbf{\Phi_a}(t)$ and $\mathbf{\Phi_c}(t)$. In addition, the values of the four quantities $q_k(x,c)$ (for $k = 1, .., 4$) at the $N+1$ nodes x_i, $i = 0, 1, ..., N$ are stored in the vectors $\mathbf{q}^{(k)}(t)$.

We are now in a position to write down the spatially discretised equations arising from the macroscopic PDEs (18)-(21) and their boundary conditions (7)-(10). We begin with the ODEs that govern the evolution of the lithium concentration in the electrolyte, and which are obtained from the spatial discretisation of (18) and boundary conditions (7b) and (10b). These take the form

$$\begin{aligned}&\Delta_{1/2}\left[\frac{1}{3}\frac{dc_0}{dt} + \frac{1}{6}\frac{dc_1}{dt}\right] = -\frac{1}{\epsilon_{1/2}}N_{1/2} \\ &\mathcal{L}_i\left(\frac{d\mathbf{c}}{dt}\right) = -\left[\frac{1}{\epsilon_{i+1/2}}N_{i+1/2} - \frac{1}{\epsilon_{i-1/2}}N_{i-1/2}\right], \text{ for } i = 1, ..., N-1 \\ &\Delta_{N-1/2}\left[\frac{1}{6}\frac{dc_{N-1}}{dt} + \frac{1}{3}\frac{dc_N}{dt}\right] = \frac{1}{\epsilon_{N-1/2}}N_{N-1/2}.\end{aligned} \tag{30}$$

where $N_{i+1/2}$ is given by

$$N_-|_{x=x_{i+1/2}} \approx N_{i+1/2} = -\left[\mathcal{J}_{i+1/2}(\mathbf{q}^{(1)})\mathcal{D}_{i+1/2}(\mathbf{c}) + \mathcal{J}_{i+1/2}(\mathbf{q}^{(2)})\mathcal{D}_{i+1/2}(\mathbf{\Phi})\right] \tag{31}$$

and $\epsilon_{i+1/2} = \epsilon|_{x=x_{i+1/2}}$. The algebraic equations for the electrolyte potential Φ, which result from the discretisation of equation (19) and the boundary conditions (7c) and (10c), are

$$\begin{aligned}0 &= \Phi_0 \\ 0 &= -\left[j_{i+1/2} - j_{i-1/2}\right] + b_{i+1/2}\frac{\Delta_{i+1/2}}{2}j^n_{i+1/2} + b_{i-1/2}\frac{\Delta_{i-1/2}}{2}j^n_{i-1/2}, \text{ for } i = 1, ..., N-1 \\ 0 &= j_{N-1/2} + \frac{\Delta_{N-1/2}}{2}j^n_{N-1/2},\end{aligned} \tag{32}$$

where $j_{i+1/2}$ and $j^n_{i+1/2}$ are given by

$$j|_{x=x_{i+1/2}} \approx j_{i+1/2} = \left[\mathcal{J}_{i+1/2}(\mathbf{q}^{(3)})\mathcal{D}_{i+1/2}(\mathbf{c}) + \mathcal{J}_{i+1/2}(\mathbf{q}^{(4)})\mathcal{D}_{i+1/2}(\mathbf{\Phi})\right], \tag{33}$$

$$j_n\big|_{x=x_{i+1/2}} \approx j^n_{i+1/2} = j_n\left(\mathcal{J}_{i+1/2}(\mathbf{c}), c_a|_{r=R_a(x_{i+1/2})}, c_c|_{r=R_c(x_{i+1/2})}, \mathcal{J}_{i+1/2}(\mathbf{\Phi}_a), \mathcal{J}_{i+1/2}(\mathbf{\Phi}_c)\right), \tag{34}$$

and $b_{i+1/2} = b|_{x=x_{i+1/2}}$. The first equation in (32) is required to set a reference value for the potential, and we select the value of zero at $x = L_1$ for convenience and without loss of generality. The algebraic equations for the potential in anode Φ_a, which result from discretisation of equation (20) and the boundary conditions (7a) and (8), are

$$\begin{aligned} 0 &= -\frac{I}{A} + j^a_{1/2} - \frac{\Delta_{1/2}}{2} b_{1/2} j^n_{1/2} \\ 0 &= -\left[j^a_{i+1/2} - j^a_{i-1/2}\right] - \frac{\Delta_{i+1/2}}{2} b_{i+1/2} j^n_{i+1/2} - \frac{\Delta_{i-1/2}}{2} b_{i-1/2} j^n_{i-1/2}, \text{ for } i = 1, ..., N-1 \\ 0 &= -j^a_{N-1/2} - \frac{\Delta_{N-1/2}}{2} b_{N-1/2} j^n_{N-1/2}, \end{aligned} \tag{35}$$

where

$$j_a|_{x=x_{i+1/2}} \approx j^a_{i+1/2} = -\sigma_a \mathcal{D}_{i+1/2}(\mathbf{\Phi}_a). \tag{36}$$

The algebraic equations for the potential in cathode Φ_c, which result from discretisation of equation (20) and the boundary conditions (9) and (10a), are

$$\begin{aligned} 0 &= -j^c_{1/2} - \frac{\Delta_{1/2}}{2} b_{1/2} j^n_{1/2} \\ 0 &= -\left[j^c_{i+1/2} - j^c_{i-1/2}\right] - b_{i+1/2} \frac{\Delta_{i+1/2}}{2} j^n_{i+1/2} - b_{i+1/2} \frac{\Delta_{i-1/2}}{2} j^n_{i-1/2}, \quad \text{for} \quad i = 1, ..., N-1 \\ 0 &= -\frac{I}{A} + j^c_{N-1/2} - b_{N-1/2} \frac{\Delta_{N-1/2}}{2} j^n_{N-1/2}. \end{aligned} \tag{37}$$

where

$$j_c|_{x=x_{i+1/2}} \approx j^c_{i+1/2} = -\sigma_c \mathcal{D}_{i+1/2}(\mathbf{\Phi}_c). \tag{38}$$

Equations (30)-(38) comprise the discretised macroscopic equations that are implemented in DandeLiion.

3.3 Assembly of the Differential Algebraic Equations

Here we briefly describe how the system of DAEs, which are solved by DandeLiion, are assembled from the spatial discretisation of the DFN model. As stated earlier the microscopic equations (11)-(12) are discretised by application of Zeng *et al.* 's [23] control volume (CV) method, which similar to the FEM discretisation of the macroscopic equations, exhibits perfect lithium conservation and also directly evaluates the lithium-ion concentration on the electrode particle surfaces, which is important from the point of view of accurately approximating the Butler-Volmer equations.

Henceforth we refer to the combined finite element and control volume spatial discretisation as the FE+CV scheme. The total number of grid points in the macroscopic dimension, x is $N = N_a + N_s + N_c$, where N_a, N_s, N_c are the grid points in the anode, separator, and cathode respectively. At each of the $N_a + N_c$ stations in x which belong to the anode or cathode we consider a representative spherical electrode particle which is discretised using

M grid points in the radial coordinate r. We denote $r_j = jh_r$, where $h_r = 1/(M-1)$ for $j = 1, ..., M$. In total we have $(N_a + N_c) \times M$ different stations in r and at these locations we denote the value of lithium concentration in anode and cathode by $c^a_{i,j}$ and $c^c_{i,j}$ respectively. The index i indicates the representative particle's position in x whereas j labels the radial position within that particle. In total we have $2N$ functions to be determined for concentration and potential in the electrolyte, N_a and N_c unknowns for the potential in anode and cathode respectively, and $(N_a + N_c) \times M$ unknowns for the concentration in anode and cathode.

The $2N + (N_a + N_c) \times (M + 1)$ unknown functions of time are assembled into one large column vector $\mathbf{u}(t)$ as follows

$$\begin{aligned}\mathbf{u}(t) &= \left[c_0, ...c_N, \Phi_0, ..., \Phi_N, \Phi^0_a, ...\Phi^{N_a}_a, \Phi^0_c, ..., \Phi^{N_c}_c, c^0_a, ...c^{N_a}_a, c^0_c, ...c^{N_c}_c\right]^T \\ &= \left[\mathbf{c}(t)^T \mathbf{\Phi}(t)^T \mathbf{\Phi}_a(t)^T \mathbf{\Phi}_c(t)^T \mathbf{c_a}(t)^T, \mathbf{c_c}(t)^T\right]^T\end{aligned} \tag{39}$$

where the superscript T denotes a transpose. This allows the system of DAEs to be written in the concise form

$$\mathbf{M}\frac{d\mathbf{u}}{dt} = \mathbf{f}(\mathbf{u}), \quad \text{with} \quad \mathbf{u}\big|_{t=0} = \mathbf{u}_0. \tag{40}$$

Here the mass matrix $\mathbf{M}$ is a $(2N + (N_a + N_c) \times (M + 1)) \times ((2N + (N_a + N_c) \times (M + 1))$ tridiagonal matrix whose entries are coefficients of the time derivative terms in the equations (30)–(37), and control volume descretisation from [23]. The vector function $\mathbf{f}(\mathbf{u})$ is nonlinear, and has length $2N + (N_a + N_c) \times (M + 1)$. Its entries are the right-hand sides of equations (30)–(37) and the equations arising from the control volume descretisation. The DAE system (40) is integrated forward in time using DandeLiion's in-house DAE solver.

4 Verification

In this section, we demonstrate the second order convergence of our FE+CV method by benchmarking against an alternative spatial discretisation applied to the DFN model. We select a standard finite volume method, see [30], to compare against and we apply this spatial discretisation to both the macroscopic and microscopic components of the model, *i.e.* (1)-(14). As such, we will henceforth refer to this approach as the FV+FV method which is expected to, and indeed does, exhibit first order convergence. Since the FE+CV method is comprised of a combination of two different methods for spatial discretisation (finite elements and control volumes) we will validate the overall second order convergence rate in two steps. First we demonstrate that the application of the CV method to a nonlinear spherical diffusion equation exhibits second order converge as the number of grid points M is increased. Then, we verify the second order convergence rate for the FE discretisation by refining the number of grid points N, in the macroscopic dimension x, whilst taking M the number of grid points in the microscopic dimension r to be large enough such that the numerical errors arising from the discretisation of the microscopic equations are negligible. An analogous two-stage strategy is used for the FV+FV method. Throughout all our spatial convergence testing we set the error tolerances on the DAE integrator to be sufficiently stringent that time integration errors can also be assumed to be negligible.

Our benchmarking protocol will be based on a cell parameterised with the data in Ecker *et al.* [16, 17] for a single full discharge cycle at 4C. Due to the lack of an exact solution a reference solution, computed on a very refined grid, is used to assess the errors. For some scalar quantity w (which could be $c, \Phi, \Phi_a, \Phi_c, c_a, c_c$) evaluated at fixed spatial and temporal values, we can define the numerical error of a simulation as

$$\mathcal{E}(w, N, N_{\text{ref}}) = |w^{(N)} - w^{(N_{ref})}|, \tag{41}$$

where $|\cdot|$ is the absolute value operator and $w^{(N_{ref})}$ is the approximation to the exact solution found by using a highly refined grid.

We first investigate the dependence of the numerical convergence on M, the number of grid points used to discretise the particle diffusion equations (11)-(12), for a fixed value of j_n. We compute a good approximation to the exact solution solution by taking a large value, in this instance $M_{\text{ref}} = 2561$. In Figure 3, the top left plot shows the logarithm of the absolute errors $\mathcal{E}(c_a(r = R_a/2, t = t_f), M, M_{\text{ref}})$ and $\mathcal{E}(c_c(r = R_c/2, t = t_f), M, M_{\text{ref}})$, for $t_f = 800$ s plotted versus $\log(M)$ for the concentration in anode and cathode using FV and CV methods. We emphasize that in this test j_n is taken to be constant and as such there is no need to evaluate c_a or c_c at a specified x (they too are independent of x). As expected the straight line fit to the CV method has a gradient of ≈ -2 corresponding to second-order accuracy of the scheme while the straight line fit to FV method has a gradient of ≈ -1 corresponding to first-order accuracy of the scheme. The numerical order of convergences are summarised in Table 3.

Next we investigate the dependence of the numerical convergence on N, the number of grid points used to discretise the macroscopic DFN equations (1)-(6). We compute a reference solution, corresponding to a good approximation to the exact solution, by using a large number of grid points, in this case $N_{\text{ref}} = 1723$ for FE+CV method and $N = N_{\text{ref}} = 2560$ for FV+FV method. Throughout the tests to assess converge in N we fix the number of grid points $M = 640$ which is sufficiently large that errors stemming from the solution to the diffusion equations describing transport in the electrode particles are negligible.

In Figure 3 the top right plot shows the logarithms of the absolute errors for electrolyte concentration c, electrolyte potential Φ and anode potential Φ_a at the midpoint of the anode plotted against $\log(N)$ using the FE+CV method and using the FV+FV method. The same least square fitting procedure is used as above to assess the numerical order of convergence from the variation in error with radial grid spacing and the results are displayed in Table 4. In particular it shows that FE+CV method is second order while the FV+FV lies between first and second order. The theoretical convergence rate for the elliptic PDEs (equations for the potential in the electrolyte and solid particles) may reach second order for the FV method if implemented properly, but in this case the theoretical rate is not achieved because the elliptic equations are strongly coupled with the (first-order) parabolic equations (for the concentrations in the electrolyte and solid particles).

Finally, in the lower panel of Figure 3, we show the error in the output voltage which is a function of time and can therefore be assessed using an error defined as

$$\mathcal{E}^p(V(t), N, N_{\text{ref}}) = \left\| V(t)^{(N)} - V(t)^{(N_{ref})} \right\|, \tag{42}$$

where $p = 1, 2, \infty$. As expected the FE+CV shows second-order convergence, corresponding

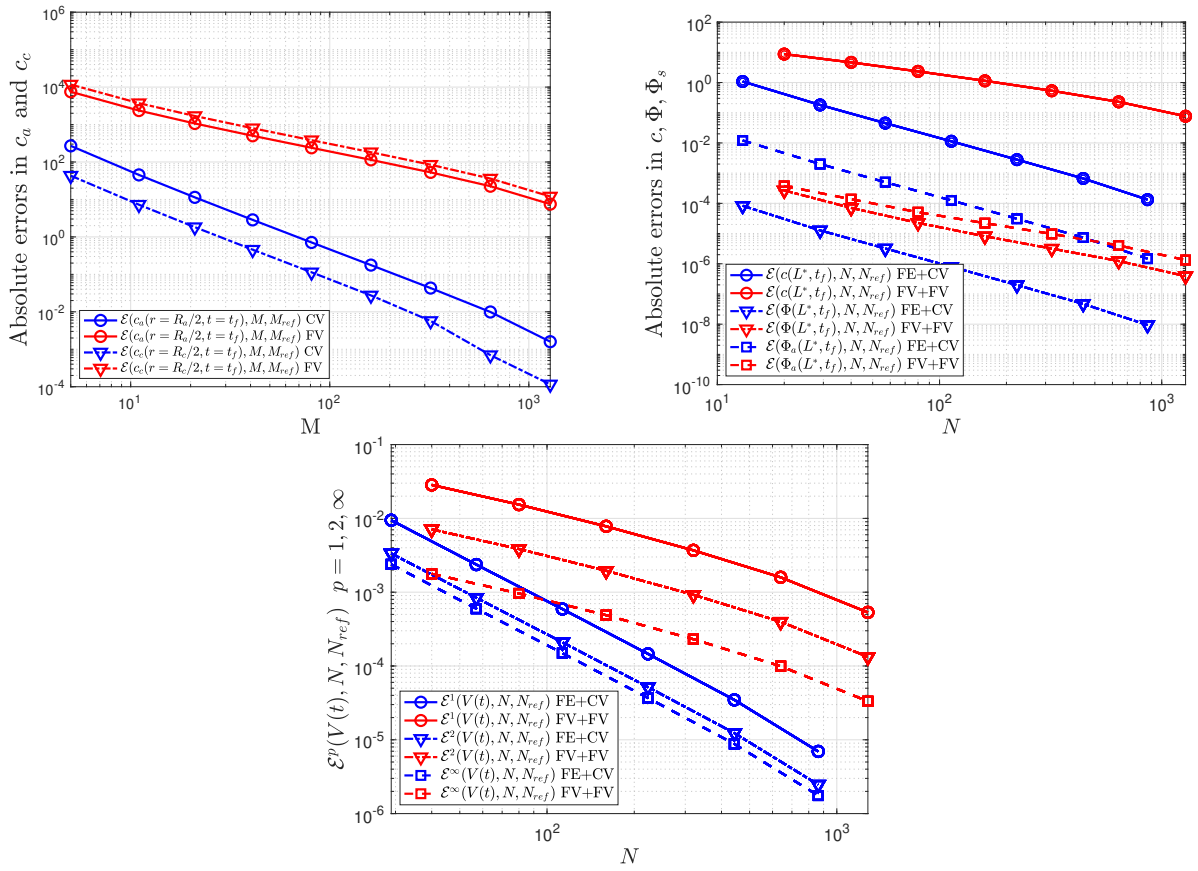

Figure 3: The top left panel is a plot of the absolute errors $\mathcal{E}(c_a(r = R_a/2, t = t_f), M, M_{\text{ref}})$ and $\mathcal{E}(c_c(r = R_c/2, t = t_f), M, M_{\text{ref}})$ for concentration in anode and cathode with $t_f = 800$ s. The top right panel is a plot of the absolute errors $\mathcal{E}(c(x = L^*, t = t_f), N, N_{\text{ref}})$, $\mathcal{E}(\Phi(x = L^*, t = t_f), N, N_{\text{ref}})$ and $\mathcal{E}(\Phi_a(x = L^*, t = t_f), N, N_{\text{ref}})$ for concentration in electrolyte, potential in electrolyte and potential in solid for $t_f = 800$ s and $L^* = (L_1 + L_2)/2$. The bottom panel is a plot of the errors $\mathcal{E}^p(V(t), N, N_{\text{ref}})$ where $p = 1, 2, \infty$ for the cell voltage.

to a straight line with gradient -2 in the $\log(\mathcal{E}^p)$ vs $\log(N)$ plot, and FV+FV shows only first order convergence (see Table 5).

Method	**Order for** $c_a(r = R_a/2, t = t_f)$	**Order for** $c_c(r = R_c/2, t = t_f)$
FV	1.19	1.19
CV	2.12	2.27

Table 3: Numerical orders of convergence for concentration in anode $c_a(r = R_a/2, t = t_f)$ and concentration in cathode $c_c(r = R_c/2, t = t_f)$ by using FV and CV methods with $t_f = 800$ s. The orders are arrived at by fitting a straight line (using least squares) to the data in Figure 3 (top left panel).

Methods	Order for $c(x = L^*, t = t_f)$	Order for $\Phi(x = L^*, t = t_f)$	Order for $\Phi_a(x = L^*, t = t_f)$
FV+FV	1.05	1.53	1.30
FE+CV	2.08	2.10	2.08

Table 4: Numerical orders of convergence for concentration in electrolyte for $c(x = L^*, t = t_f)$, potential in electrolyte $\Phi(x = L^*, t = t_f)$ and potential in solid $\Phi_s(x = L^*, t = t_f)$ by using FV+FV and FE+CV methods with $t_f = 800$ s and $L^* = (L_1 + L_2)/2$. The orders are arrived at by fitting a straight line (using least squares) to the data in Figure 3 (top right panel).

Methods	Order for $V(t)$, L^1	Order for $V(t)$, L^2	Order for $V(t)$, L^∞
FV+FV	1.04	1.04	1.04
FE+CV	2.09	2.09	2.08

Table 5: Numerical orders of convergence for voltage $V(t)$ in L^1, L^2 and L^∞ norm by using FV+FV and FE+CV methods. The orders are arrived at by fitting a straight line (using least squares) to the data in Figure 3 (bottom panel).

5 Illustrative examples

To demonstrate the practical utility of DandeLiion we show a single discharge cycle based on Graphite-Silicon/$LiNi_{1-x-y}Mn_xCo_yO_2$ LG M50 battery cell chemistry [27] and a simulation of a charge/discharge current profile applied to the cell at different (dis)charge rates.

The DFN model implemented in DandeLiion was fully parametrised according to [27]. All the parameters, including functions (*e.g.* open circuit voltages, diffusivity and conductivity in the electrolyte, see Figure 4) were filled directly in the web forms provided by the simulation engine on the DandeLiion website [31]. The computational grid can be defined by the user as well, and for the purposes of this demonstration we set up 50 grid points in the electrolyte in each electrode, 30 points across the separator, and 100 nodes in each solid particle. The authors in [27] test their parametrisation using 0.5C, 1C, and 1.5C constant discharge currents followed by a relaxation period. As a first example, we simulate a full 1C discharge with two-hour relaxation. For the chosen discretisation the total number of DAEs to be solved reaches 10^4, but the compute time remains very manageable at around 1 second. When running this on the server there is a fixed (independent of simulation size/complexity) overhead of around 7-10 seconds which is associated with setting up the simulation in the cloud, checking the user-defined parameterisation, code compilation, saving the data, creating a zip archive and generating a permanent webpage displaying the results.

After the job is complete DandeLiion users can see a set of preliminary plots (the output of the simulation described here is shown in Figure 5). These plots show the total voltage and user-defined current against time, as well as the Li ion concentration in the electrolyte and within two representative particles; one in the anode and another in the cathode, as well as the potential distribution in the electrolyte. Below these plots, a link is provided to

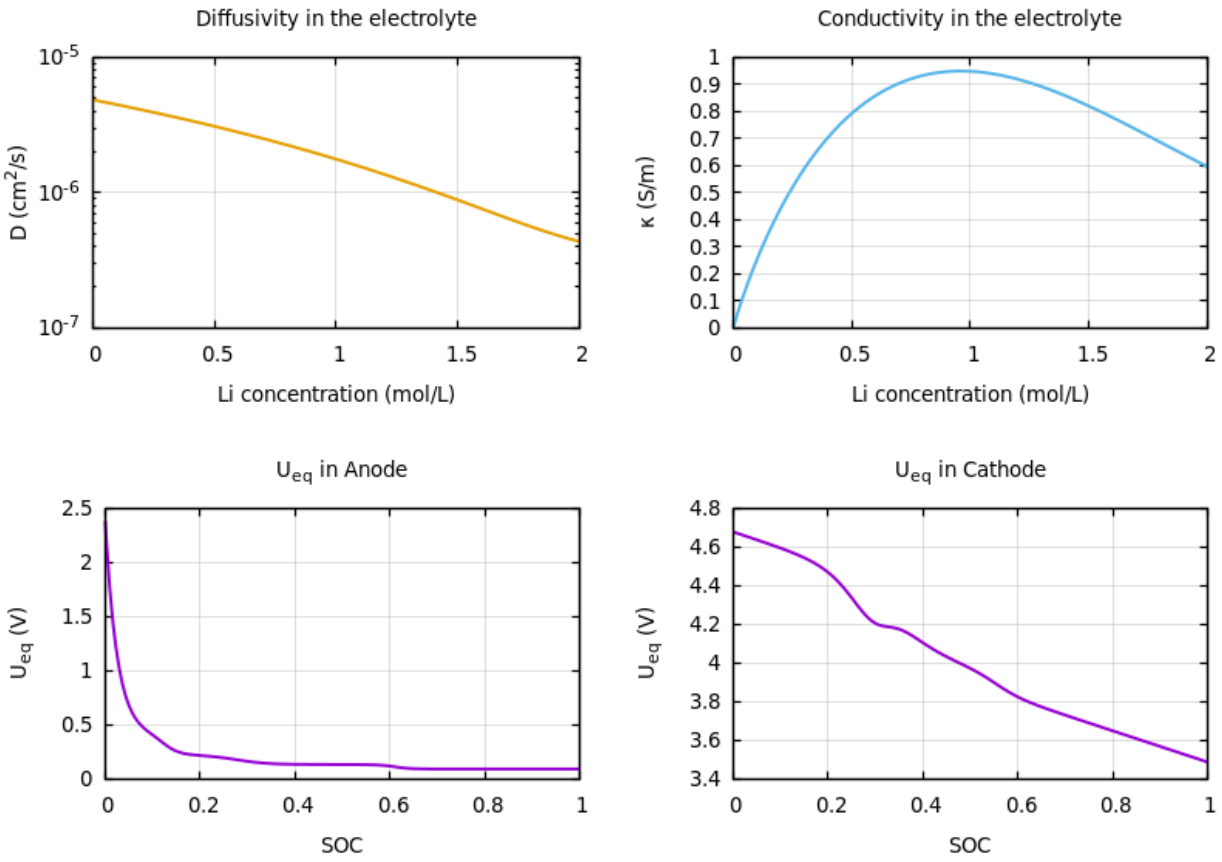

Figure 4: Ionic diffusivity (top left) and conductivity (top right) of the electrolyte, open circuit voltages of graphite-silicon anode (bottom left) and $LiNi_{1-x-y}Mn_xCo_yO_2$ cathode (bottom right). Experimental data from from [27].

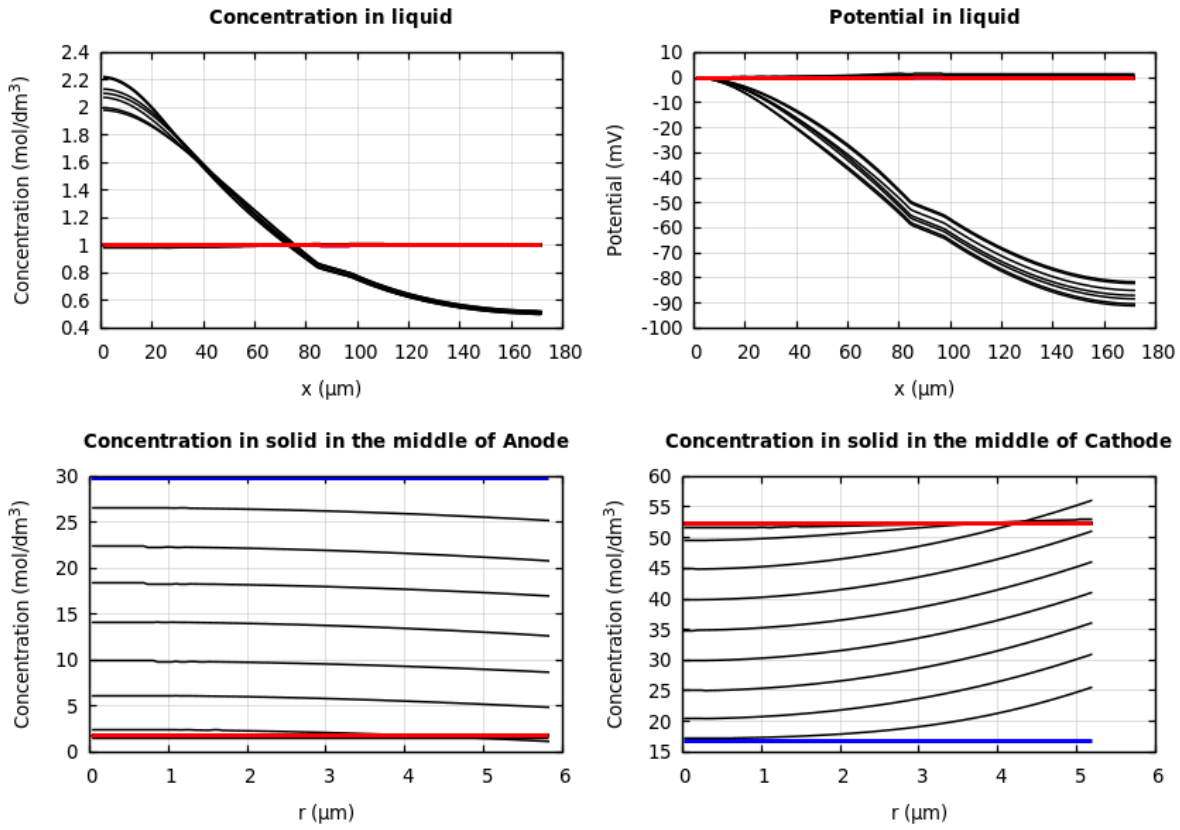

Figure 5: Concentration of Li (top left) and potential distribution (top right) in the electrolyte across the battery cell width, and concentrations of Li in two representative particles in anode (bottom left) and cathode (bottom right). The blue line shows the initial state of the battery, black thin lines correspond to the snapshots at 500 s intervals, and the red line is the final state.

download the raw data files for plotting using any other software of choice, *e.g.* Microsoft Excel, MATLAB, etc.

This simulation was used to further validate DandeLiion. The voltage is compared with both experiment and simulation results from [27] and Figure 6 shows that good agreement is obtained.

After the simulation is complete, the user may change any of the parameters and resubmit the simulation (this can be done even *during* the simulation, a new instance of the simulation will be created and sent into the queue). There is no need to complete the parametrisation form from scratch; all parameters are stored on the server and can be re-used by clicking on the 'Review all parameters & Resubmit the simulation' button. The server will create a permanent link for each parametrisation so that it can be bookmarked for future use.

DandeLiion allows the user to define different particle sizes in each electrode thereby allowing simulation of so-called graded electrodes which might have larger particles adjacent to the separator than those near the current collector, or vice versa. As a demonstration of this functionality, we take the parameter set in [27] and increase the particle size in anode

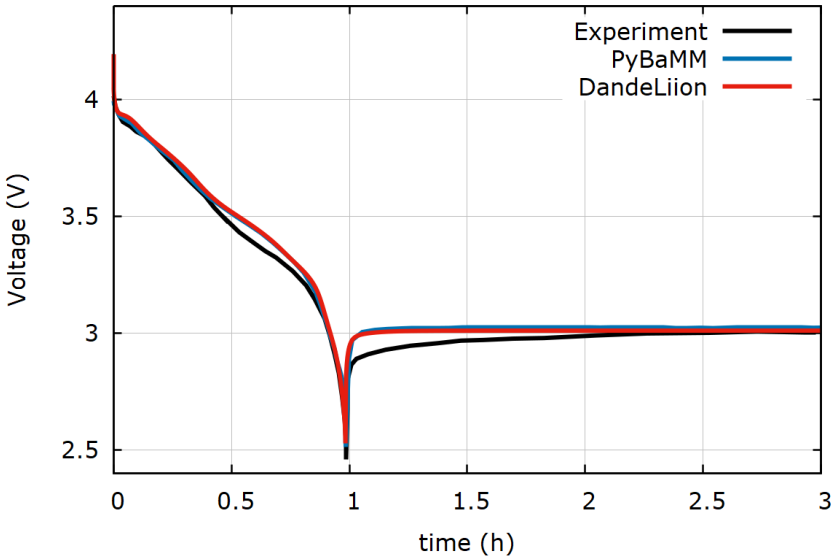

Figure 6: Cell voltage profiles computed from DandeLiion in comparison with the experimental and simulation data from [27].

near the separator by a factor of three so that those particles in $L_1 < x < (0.1L_1 + 0.9L_2)$ are of size R_a and those in $(0.1L_1 + 0.9L_2) < x < L_2$ are of size $3R_a$. The increased size of particle was accommodated in the electrode by decreasing the number of particles, as well as the particle surface area (per unit volume) $b(x)$, in $(0.1L_1 + 0.9L_2) < x < L_2$ by the same factor of three. The inclusion of the graded electrode functionality is motivated by the clear variation in particle sizes seen in microscopy data of real electrodes, see [27, 32] for examples. The importance of capturing these variations is spoken to by the quality of the agreement between DandeLiion and experiment [33] shown in Figure 7. We emphasize the improvement in fit between Figures 7 and 6 is due to the variation in particle sizes that is accounted for in the former, but not the latter.

Both simulation examples including the parametrisation and corresponding current profiles are available on the DandeLiion website [31].

6 Conclusions

This work describes the release of novel software that is able to solve the most ubiquitous electrochemical cell-scale LIB model, namely the DFN model, extremely quickly. DandeLiion is a cloud-based service, accessible via `dandeliion.com`, where users can submit their jobs via an easy-to-use web interface and can collect results both in the browser and in-full by downloading raw output. It comes equipped with comprehensive documentation, a set of video tutorials aimed at new users and a library of chemistries to construct common cell architectures. A set of pre-defined simulations are available on the website that can be adapted to suit user's specific needs. In the future we aim to expand upon the existing material library, implement additional physics including thermal coupling across multiple cells and add more functionality to the Web user interface including event handling, live

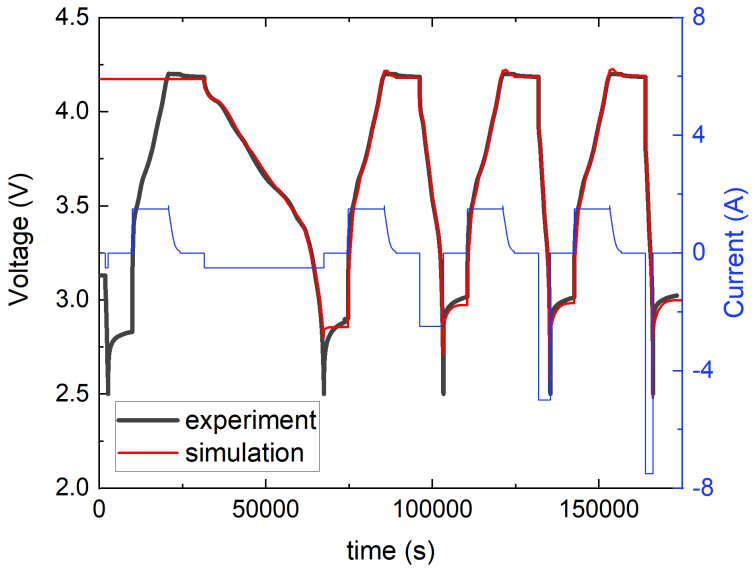

Figure 7: Cell voltage and the current vs time comparing DandeLiion simulations and experiment taken from [33]. The current varies between (dis)charge rates ranging between ± 1.5C.

feedback showing the current simulation stage in real time, etc.

DandeLiion has the capability of making rapid predictions of LIB (dis)charge behaviour and arms both academics and industrialists with the means of solving a model which has been demonstrated to accurately predict device behaviour across a range of operating protocols and a variety of device designs [10, 34]. The ability to rapidly solve large numbers of DAEs opens the door to being able to investigate multi-dimensional thermally coupled problems in composite cells (*e.g.* pouch cells and cylindrical cells), battery modules and even in entire battery packs, using a realistic electrochemical representation of the cell (rather than relatively crude equivalent circuit models). It will also enable modern optimisation techniques to be applied to electrochemical models of the cell and used to design optimal cell structures and furthermore it opens the way to using parameter estimation techniques to deduce cell properties from real cell data. It also facilitates finding solutions in computationally intensive settings, such as a realistic drive cycle.

DandeLiion's functionality expedites the development of new device designs by allowing users to explore the effects of alterations to battery designs in-silico, lowering the monetary and temporal costs associated with development via physical prototyping. It therefore paves the way for significant improvements in LIB performance, lifetime and safety, especially in the context of their use in EVs and other high-power applications. Ultimately this significant advance in LIB simulation software is expected to lead to substantial benefits to industry and increase the impetus for the creation of new products and procedures.

7 Conflict of Interest

We wish to confirm that there are no known conflicts of interest associated with this publication and there has been no significant financial support for this work that could have influenced its outcome.

Acknowledgements

The work of all the authors was supported by the Faraday Institution Multi-Scale Modelling (MSM) project (grant number EP/S003053/1). The authors would like to thank Debora Corbin for suggesting the name of the software as well as Ferran Brosa Planella and Emma Kendrick for providing the experimental data (on the LG M50 battery) used here for validation.

References

[1] G. E. Blomgren, The development and future of lithium ion batteries, Journal of The Electrochemical Society, 164 (2017), pp. A5019–A5025.

[2] G. Zubi, R. Dufo-Lopez, M. Carvalho, and G. Pasaoglu, The lithium-ion battery: State of the art and future perspectives, Renewable and Sustainable Energy Reviews, 89 (2018), pp. 292–308.

[3] J. Vetter, P. Novák, M. R. Wagner, C. Veit, K.-C. Möller, J. Besenhard, M. Winter, M. Wohlfahrt-Mehrens, C. Vogler, and A. Hammouche, Ageing mechanisms in lithium-ion batteries, Journal of Power Sources, 147 (2005), pp. 269–281.

[4] Q. Wang, P. Ping, X. Zhao, G. Chu, J. Sun, and C. Chen, Thermal runaway caused fire and explosion of lithium ion battery, Journal of Power Sources, 208 (2012), pp. 210–224.

[5] M. Doyle, J. Newman, A. S. Gozdz, C. N. Schmutz, and J.-M. Tarascon, Comparison of modeling predictions with experimental data from plastic lithium ion cells, Journal of the Electrochemical Society, **143**, (1996), pp. 1890–1903.

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

[7] T. F. Fuller, M. Doyle, and J. Newman, Simulation and optimization of the dual lithium ion insertion cell, Journal of the Electrochemical Society, **141**, (1994), pp. 1–10.

[8] J. Newman and K.E. Thomas-Alyea, Electrochemical Systems, vol. 1, Prentice Hall, New Jersey, 2004.

[9] J. Newman and W. Tiedemann, Porous-electrode theory with battery applications, AIChE Journal, 21 (1975), pp. 25–41.

[10] G. W. Richardson, J. M. Foster and R. Ranom, C. P. Please, & A. M. Ramos, Charge transport modelling of lithium ion batteries, arXiv preprint arXiv:2002.00806 (2020).

[11] G. Richardson, I. Korotkin, R. Ranom, M. Castle, and J. M. Foster, Generalised single particle models for high-rate operation of graded lithium-ion electrodes: systematic derivation and validation, Electrochimica Acta, 39:135862 (2020).

[12] S. G. Marquis, V. Sulzer, R. Timms, C. P. Please, and S. J. Chapman, An asymptotic derivation of a single particle model with electrolyte, Journal of The Electrochemical Society **166(15)**:A3693 (2019).

[13] Iserles, Arieh (1996), A First Course in the Numerical Analysis of Differential Equations, Cambridge University Press, ISBN 978-0-521-55655-2.

[14] Süli, Endre; Mayers, David (2003), An Introduction to Numerical Analysis, Cambridge University Press, ISBN 0-521-00794-1.

[15] MathWorks MATLAB (2019), `https://uk.mathworks.com/products/matlab.html`

[16] M. Ecker, T. K. D. Tran, P. Dechent, S. Käbitz, A. Warnecke, and D. U. Sauer, Parameterization of a physico-chemical model of a lithium-ion battery i. determination of parameters, Journal of The Electrochemical Society, **162**, (2015), pp. A1836–A1848.

[17] M. Ecker, S. Käbitz, I. Laresgoiti, and D. U. Sauer, Parameterization of a physico-chemical model of a lithium-ion battery ii. model validation, Journal of The Electrochemical Society, **162**, (2015), pp. A1849–A1857.

[18] Dassault Systèmes, CATIA Dymola 2019.

[19] COMSOL Multiphysics Reference Manual, COMSOL, Inc, `www.comsol.com`.

[20] V. Sulzer, S. G. Marquis, R. Timms, M. Robinson, & S. J. Chapman, Python Battery Mathematical Modelling (PyBaMM), ECSarXiv, February, 7 (2020).

[21] F. Hanke, R. L. C. Akkermans, N. Modrow, I. Korotkin, F. C. Mocanu, V. A. Neufeld, and M. Veit, Multi-Scale Electrolyte Transport Simulations for Lithium Ion Batteries, Journal of The Electrochemical Society, 167 (2020), 013522. https://doi.org/10.1149/2.0222001JES

[22] A. Zülke, I. Korotkin, H. Hoster, J. M. Foster & G. Richardson, Parameterisation of a DFN model for a commercial NCA/Si-Gr battery, in preparation.

[23] Y. Zeng, P. Albertus, R. Klein, N. Chaturvedi, A. Kojic, M. Z. Bazant, and J. Christensen, Efficient conservative numeical schemes fo 1D nonlinear spherical diffusion equations with applications in battery modelling, Journal of the Electrochemical Society, 169(9), (2013), pp. A1565–A15171.

[24] G. Wanner, E. Hairer, Solving ordinary differential equations II, Springer Berlin Heidelberg, 1996.

[25] E. A. Celaya, J. A. Aguirrezabala, P. Chatzipantelidis, Implementation of an Adaptive BDF2 Formula and Comparison with the MATLAB Ode15s, ICCS (2014).

[26] V. Srinivasan, J. Newman, Discharge Model for the Lithium Iron-Phosphate Electrode, J. Electrochem. Soc., 151 (10), A1517-A1529 (2004).

[27] C.-H. Chen, F. Brosa Planella, K. O'Regan, D. Gastol, D. Widanage, E. Kendrick, Development of Experimental Techniques for Parameterization of Multi-scale Lithium-ion Battery Models, J. Electrochem. Soc., 167, 080534 (2020).

[28] N. E. Courtier, G. Richardson, and J. M. Foster, A fast and robust numerical scheme for solving models of charge carrier transport and ion vacancy motion in perovskite solar cells, Applied Mathematical Modelling, 63, (2018), pp. 329-348. https://doi.org/10.1016/j.apm.2018.06.051

[29] C. Johnson, Solution of partial differential equations by the finite element method, Cambridge University Press, Cambridge, UK, 1987.

[30] L. Randall, Finite volume methods for hyperbolic problems, Cambridge University Press, Cambridge, UK, 2002.

[31] DandeLiion Simulation Engine webpage, `https://www.dandeliion.com/simulation`

[32] H. Liu, J. M. Foster, A. Gully, S. Krachkovskiy, M. Jiang, Y. Wu, X. Yang, B. Protas, G. R. Goward, and G. A. Botton, Three-dimensional investigation of cycling-induced microstructural changes in lithium-ion battery cathodes using focused ion beam/scanning electron microscopy, Journal of Power Sources **306**:300-308 (2016).

[33] C.-H. Chen, F. Brosa Planella, K. O'Regan, D. Gastol, D. Widanage, E. Kendrick, Experimental data for "Development of Experimental Techniques for Parameterization of Multi-scale Lithium-ion Battery Models", Zenodo, (2020), DOI: 10.5281/zenodo.4032561. https://doi.org/10.5281/zenodo.4032561

[34] A. Jokar, B. Rajabloo, M. Désilets & M. Lacroix, Review of simplified Pseudo-two-Dimensional models of lithium-ion batteries, Journal of Power Sources, 327, pp.44-55 (2016).

[35] B. D. Bruggeman, Calculation of different physical constants of heterogeneous substances. i. dielectric constants and conductivities of mixed bodies of isotropic substances, Annalen der Physik, **416**, (1935), pp. 636–664.

[36] A. Rohatgi, WebPlotDigitizer version 4.3 (2020), `https://automeris.io/WebPlotDigitizer`

[37] V. R. Subramanian, V. D. Diwakar, & D. Tapriyal,, Efficient Macro-Micro Scale Coupled Modeling of BatteriesJournal of the Electrochemical Society, 152(10), A2002 (2005).

[38] V. R. Subramanian, V. Boovaragavan, V. Ramadesigan, & M. Arabandi, Mathematical Model Reformulation for Lithium-Ion Battery Simulations: Galvanostatic Boundary Conditions, Journal of The Electrochemical Society, 156(4), A260 (2009).

[39] A. M. Bizeray, S. Duncan, S. R. Duncan & D. A. Howey, Advanced battery management systems using fast electrochemical modelling Hybrid Electr. Veh. Conf. 2013 (HEVC 2013), 978-1-84919-776-2, Institution of Engineering and Technology (2013).

[40] T. S. Dao, C. P. Vyasarayani, & J. McPhee, Simplification and Order Reduction of Lithium-ion Battery Model based on Porous-Electrode Theory Journal of Power Sources, 198, 329-337, (2012).

[41] L. Cai & R. E. White, Lithium ion cell modeling using orthogonal collocation on finite elements, Journal of Power Sources, 217, 248-255 (2012).

[42] J. C. Forman, S. Bashash, J. L.Stein & H. K. Fathy, Reduction of an Electrochemistry-Based Li-Ion Battery Model via Quasi-Linearization and Pade´Approximation, Journal of The Electrochemical Society, Volume 158, A93–A101 (2015).

[43] N. T. Tran, M. Vilathgamuwa, T. Farrell, S. S. Choi, Y. Li & J. Teague, A Pade´Approximate Model of Lithium Ion Batteries, Journal of The Electrochemical Society, 165 (7) A1409–A1421 (2018).

[44] S. Bertoluzza, S. Falletta, G. Russo, and C. Shu, Numerical solutions of partial differential equations. Springer Science & Business Media (2009).

[45] A. Quarteroni, and A. Valli, Numerical approximation of partial differential equations. Vol. 23. Springer Science & Business Media (2008).

Appendix A. Integrals required for the finite element descretisation

In section 3.1, when using the finite element method to descretise the governing system of equations (1)-(6) in space the following results are needed:

$$\int_0^1 \psi_j dx = \begin{cases} \frac{1}{2}(\Delta_{j+1/2} + \Delta_{j-1/2}) & \text{if } j = 1, .., N-1 \\ \frac{1}{2}\Delta_{1/2} & \text{if } j = 0 \\ \frac{1}{2}\Delta_{N-1/2} & \text{if } j = N \\ 0 & \text{otherwise} \end{cases}, \tag{43}$$

$$\int_0^1 \psi_i \psi_j dx = \begin{cases} \frac{1}{3}(\Delta_{j+1/2} + \Delta_{j-1/2}) & \text{if } i = j \text{ and } j = 1, .., N-1 \\ \frac{1}{3}\Delta_{1/2} & \text{if } i = j \text{ and } j = 0 \\ \frac{1}{3}\Delta_{N-1/2} & \text{if } i = j \text{ and } j = N \\ \frac{1}{6}\Delta_{j+1/2} & \text{if } i = j+1 \text{ and } j = 0, ..., N-1 \\ \frac{1}{6}\Delta_{j-1/2} & \text{if } i = j-1 \text{ and } j = 1, ..., N \\ 0 & \text{otherwise} \end{cases}, \tag{44}$$

$$\int_0^1 \psi_i' \psi_j' dx = \begin{cases} \frac{1}{\Delta_{j+1/2}} + \frac{1}{\Delta_{j-1/2}} & \text{if } i = j \text{ and } j = 1, .., N-1 \\ \frac{1}{\Delta_{1/2}} & \text{if } i = j \text{ and } j = 0 \\ \frac{1}{\Delta_{N-1/2}} & \text{if } i = j \text{ and } j = N \\ \frac{-1}{\Delta_{j+1/2}} & \text{if } i = j+1 \text{ and } j = 0, ..., N-1 \\ \frac{-1}{\Delta_{j-1/2}} & \text{if } i = j-1 \text{ and } j = 1, ..., N \\ 0 & \text{otherwise} \end{cases}, \tag{45}$$

where $\psi_i(x)$ is the basis function as defined in (17), the prime symbol ($'$) denotes derivatives with respect to x, and the indices $0 \leq i, j \leq N$.