Fast calibration of lithium-ion batteries

Toby Sheldon
Supervisor: Dr Florian Theil

October 12, 2022

Contents

1 Introduction

The state of charge of lithium-ion (Li-ion) batteries can be modelled using the Doyle-Fuller-Newman (DFN) model [1]. However, this approach is computationally expensive, making it impractical for real-time applications. Furthermore, in order to use the DFN model for a particular battery, one needs to know the model parameters which depend on each battery.

In this project, we implemented a simplified version of the DFN model, which used fewer parameters than the full version. This simplified model resulted in faster computation speeds, whilst still achieving high accuracy. A finite difference scheme was used to approximate solutions to the partial differential equations (PDEs) that describe the simplified model.

We then used a Levenberg-Marquardt algorithm [2] to estimate the model parameters, which allowed us to minimise the discrepancy between the simulated and observed data of battery voltage.

Thus, with sufficient training data, we can calibrate the model parameters to fit this data and then use the model to predict future behaviour of the battery's voltage when subjected to an applied current. The fast computation time of the finite difference scheme means that the parameter estimation process and the data simulation are also fast.

Also, in a similar way to estimating model parameters, we were also able to 'learn' part of of the model structure. This consisted of replacing a function which is know to be part of the model, such as $\sinh(x)$, with a polynomial $q(x)$. Then, the polynomial coefficients are estimated in the same way as other model parameters in the hope that the resulting polynomial approximates $\sinh(x)$.

2 Simplified DFN model

A Li-ion battery cell is made up of a negative and a positive electrode, with a separator in between. The lithium-ions can be found in an electrolyte solution throughout the cell; and they can also be found in the solid state in balls, located in the positive and negative electrode.

The DFN model describes how the concentration of lithium and the electric potential in the battery changes over time, given some applied current. We use a simplified version of the dimensionless DFN model [3].

We denote the concentration of lithium in the electrolyte and solid state at time t by $c_e(z,t)$ and $c_s(z,r,t)$, respectively. We also denote the electric potential by $\Phi_s(z,t)$. Here, $z \in [-1,0) \cup (0,1]$ is the mesoscale variable, representing the position in the cell; and $r \in [0,1]$ is the microscale variable, representing the distance from the centre of a ball containing Li in the solid state.

The model equations are given by

$$\begin{cases} \partial_t c_s = D_s r^{-2} \partial_r \left(r^2 \partial_r c_s \right) & \text{in } [0,1] \\ -D_s \partial_r c_s = \beta_s g & \text{for } r = 1 \\ \partial_r c_s = 0 & \text{for } r = 0 \end{cases} \tag{1}$$

$$\begin{cases} \beta_e g = \partial_t c_e - D_e \Delta_z c_e & \text{in } \Omega_\mathrm{e} \\ 0 = \partial_z c_e & \text{for } z \in \partial\Omega_\mathrm{e} \\ 1 = c_e & \text{at } t = 0 \end{cases} \tag{2}$$

where z is the mesoscale variable in $\Omega_\mathrm{e} = [-1,1]$.

$$\begin{cases} g = \alpha_s \partial_z^2 \Phi_s & \text{in } \Omega_\mathrm{e} \setminus \Omega_\mathrm{s} = [-1,1] \setminus \{0\} \\ i = \alpha_s \partial_z \Phi_s & \text{in } \partial\Omega_\mathrm{e} \\ 0 = \partial_z \Phi_s & \partial\Omega_\mathrm{s} \end{cases} \tag{3}$$

where $i : \mathbb{R}^+ \to \mathbb{R}$ is the applied current and $\Omega_\mathrm{s} = \{0\}$.

We use the following definition for g:

$$\begin{aligned} g &= j_0 \sinh\left(\tfrac{\lambda}{2} \tfrac{\eta}{1+\gamma T} \right) \\ \text{with } \eta &= \Phi_s - U_\mathrm{OCP}(c_s) \\ \text{and } j_0 &= \mu \left[(1-c_s) c_s c_e \right]^{\frac{1}{2}} \end{aligned} \tag{4}$$

where U_OCP, given in [4], is the open circuit potential.

We are interested in finding the voltage, V, of a battery. This is given by the electric potential difference between the ends of both electrodes

$$V = \Phi_s(1) - \Phi_s(-1) \tag{5}$$

3 Implementing a finite difference scheme

It is impossible to solve the system of equations (1), (2) and (3) analytically, so we use a finite difference scheme to approximate a solution to these equations.

We discretise Ω_e and Ω_s in space in the following way. Let $N > 0$ and define $h := \frac{1}{N}$. We define the spatial meshes as

$$\begin{aligned} \Omega_{\mathrm{e,h}}^- &:= \{jh \in \mathbb{R} \quad | \quad j \in \{-N, -N+1, \cdots, 0\}\} \\ \Omega_{\mathrm{e,h}}^+ &:= \{jh \in \mathbb{R} \quad | \quad j \in \{0, 1, \cdots, N\}\} \\ \Omega_{\mathrm{s,h}}^- &:= \{(jh, kh) \in \mathbb{R}^2 \quad | \quad j \in \{-N, -N+1, \cdots, 0\},\ k \in \{0, 1, \cdots, N\}\} \\ \Omega_{\mathrm{s,h}}^+ &:= \{(jh, kh) \in \mathbb{R}^2 \quad | \quad j \in \{0, 1, \cdots, N\},\ k \in \{0, 1, \cdots, N\}\} \end{aligned}$$

To simplify notation, we also write

$$c_{e,j}^m := c_e(jh, m\delta), \qquad j \in \Omega_{\mathrm{e,h}}^\pm, m \in \mathbb{N} \tag{6}$$

$$c_{s,j,k}^m := c_s(jh, kh, m\delta), \qquad (j,k) \in \Omega_{\mathrm{s,h}}^\pm, m \in \mathbb{N} \tag{7}$$

$$\Phi_{s,j}^m := \Phi_s(jh, m\delta), \qquad j \in \Omega_{\mathrm{e,h}}^\pm, m \in \mathbb{N} \tag{8}$$

$$g_j^m := g(jh, m\delta), \qquad j \in \Omega_{\mathrm{e,h}}^\pm, m \in \mathbb{N} \tag{9}$$

Where $\delta > 0$ is the time step of the discretisation. We also define

$$c_{s,h}^{m,-} := \left(c_{s,-N,0}^m,\ c_{s,-N+1,0}^m,\ \cdots,\ c_{s,0,0}^m,\ c_{s,-N,1}^m,\ \cdots,\ c_{s,0,N}^m \right)^\top$$

and $c_{s,h}^{m,+}, c_{e,h}^{m,-}, c_{e,h}^{m,+}, \Phi_{s,h}^{m,-}, \Phi_{s,h}^{m,+}$ similarly.

Given $c_{s,h}^{m,-}$ we use equations (1) - (4) and the Crank-Nicholson method [5] to find $c_{s,h}^{m+1,-}$.

First, use $c_{e,h}^{m,-}$, $c_{s,h}^{m,-}$, $\Phi_{s,h}^{m,-}$ to calculate $g_h^{m,-}$.

Then, we discretise (1) and use the Crank-Nicholson scheme to get

$$\begin{aligned}
\tfrac{c^{m+1,-}_{s,j,k}-c^{m,-}_{s,j,k,}}{\delta} &= \tfrac{1}{2}D_s\left(\tfrac{2}{kh}\tfrac{c^{m+1,-}_{s,j,k+1}-c^{m+1,-}_{s,j,k}}{h}+\tfrac{c^{m+1,-}_{s,j,k+1}-2c^{m+1,-}_{s,j,k}+c^{m+1,-}_{s,j,k-1}}{h^2}\right)\\
&\quad+\tfrac{1}{2}D_s\left(\tfrac{2}{kh}\tfrac{c^{m,-}_{s,j,k+1}-c^{m,-}_{s,j,k}}{h}+\tfrac{c^{m,-}_{s,j,k+1}-2c^{m,-}_{s,j,k}+c^{m,-}_{s,j,k-1}}{h^2}\right) && , j\in\Omega^-_{\mathrm{s,h}},\ k\in\{1,\cdots,N-1\}\\
D_s\tfrac{c^{m+1,-}_{s,j,N}-c^{m+1,-}_{s,j,N-1}}{h} &= -\beta_s g^m_h && , j\in\Omega^-_{\mathrm{s,h}}\\
\tfrac{c^{m+1,-}_{s,j,1}-c^{m+1,-}_{s,j,0}}{h} &= 0 && , j\in\Omega^-_{\mathrm{s,h}}
\end{aligned} \tag{10}$$

This can be compactly written using matrices:

$$Ac^{m+1,-}_{s,h} = Bc^{m,-}_{s,h} + D^{(m),-} \tag{11}$$

where

$$A = \begin{pmatrix}
-\frac{1}{h}I & \frac{1}{h}I & \mathbf{0} & & & \\
\mathbf{0} & I & \mathbf{0} & & & \\
\mathbf{0} & \mathbf{0} & I & & & \\
 & & & \ddots & & \\
 & & & & I & \mathbf{0}\\
 & & & & -\frac{1}{h}I & \frac{1}{h}I
\end{pmatrix}$$

$$-\frac{1}{2}D_s\frac{\delta}{h^2}\begin{pmatrix}
\mathbf{0} & \mathbf{0} & \mathbf{0} & & & \\
I & -2(\frac{1}{1}+1)I & \left(\frac{2}{1}+1\right)I & & & \\
\mathbf{0} & I & -2(\frac{1}{2}+1)I & & & \\
 & & & \ddots & & \\
 & & & & -2(\frac{1}{N-1}+1)I & \left(\frac{2}{N-1}+1\right)I\\
 & & & & \mathbf{0} & \mathbf{0}
\end{pmatrix}\in\mathbb{R}^{(N+1)^2\times(N+1)^2}$$

Where $I,\mathbf{0}\in\mathbb{R}^{(N+1)\times(N+1)}$.

Similarly,

$$B = \begin{pmatrix}
\mathbf{0} & \mathbf{0} & \mathbf{0} & & & \\
\mathbf{0} & I & \mathbf{0} & & & \\
\mathbf{0} & \mathbf{0} & I & & & \\
 & & & \ddots & & \\
 & & & & I & \mathbf{0}\\
 & & & & \mathbf{0} & \mathbf{0}
\end{pmatrix}$$

$$+\frac{1}{2}D_s\frac{\delta}{h^2}\begin{pmatrix}
\mathbf{0} & \mathbf{0} & \mathbf{0} & & & \\
I & -2(\frac{1}{1}+1)I & \left(\frac{2}{1}+1\right)I & & & \\
\mathbf{0} & I & -2(\frac{1}{2}+1)I & & & \\
 & & & \ddots & & \\
 & & & & -2(\frac{1}{N-1}+1)I & \left(\frac{2}{N-1}+1\right)I\\
 & & & & \mathbf{0} & \mathbf{0}
\end{pmatrix}\in\mathbb{R}^{(N+1)^2\times(N+1)^2}$$

and

$$D^{(m),-} = -\beta_s\begin{pmatrix}0\\ \vdots\\ 0\\ g^m_{-N}\\ \vdots\\ g^m_0\end{pmatrix}\in\mathbb{R}^{(N+1)^2}$$

Then, since A is invertible, we can use (11) to find

$$c^{m+1,-}_{s,h} = A^{-1}\left(Bc^{m,-}_{s,h}+D^{(m),-}\right)$$

In a similar way, we can calculate $c^{m+1,+}_{s,h}$ and $c^{m+1,\pm}_{e,h}$ from their values at time $t=m\delta$.

To find $\Phi_{s,h}^{m+1,-}$, we find auxiliary variables $\tilde{\Phi}_s^{m+1,-} : [-1,0) \to \mathbb{R}$ and $\tilde{\lambda}^- \in \mathbb{R}$ such that

$$\Phi_s^{m+1,-} = \tilde{\Phi}_s^{m+1,-} + \tilde{\lambda}^-$$

where $\tilde{\Phi}_s^{m+1,-}$ solves (3) with the Neumann boundary condition at $z = 0$ replaced with a Dirichlet boundary condition:

$$\begin{cases} g = \alpha_s \partial_z^2 \tilde{\Phi}_s^{m+1,\pm} & \text{in } \Omega_\text{e} \setminus \Omega_\text{s} = [-1,1] \setminus \{0\} \\ i((m+1)\delta) = \alpha_s \partial_z \tilde{\Phi}_s^{m+1,\pm} & \text{in } \partial\Omega_\text{e} \\ \tilde{\lambda}^\pm = \tilde{\Phi}_s^{m+1,\pm} & , \ z = 0^\pm \end{cases} \tag{12}$$

It is possible to find $\tilde{\Phi}_{s,h}^{m+1,-}$ using matrices as described earlier.

And $\tilde{\lambda}^-$ is the constant satisfying

$$\begin{aligned} i((m+1)\delta) &= -\int_{-1}^{0} g(z,(m+1)\delta)dz \\ &= -\int_{-1}^{0} j_0 \sinh\left(\frac{\lambda}{2}\frac{\tilde{\Phi}_s^{m+1,-} + \tilde{\lambda}^-}{1+\gamma T}\right) dz \end{aligned} \tag{13}$$

which can be derived from (3).

An initial guess of $\tilde{\lambda}^-$ needs to be made. Then one can update $\tilde{\Phi}_{s,h}^{m+1,-}$ using a discretisation of (12) and update $\tilde{\lambda}^-$ using (12).

A similar approach can be taken to find $\Phi_{s,h}^{m+1,+}$.

We now have a method to evolve the state vector

$$\boldsymbol{x}_h^m := \left(c_{e,h}^{m,-}, c_{e,h}^{m,+}, c_{s,h}^{m,-}, c_{s,h}^{m,+}, \Phi_{s,h}^{m,-}, \Phi_{s,h}^{m,+}\right)$$

from the m-th time step to the $(m+1)$-th time step. Hence, we can find an approximate solution to the system of equations (1) - (3)

4 Parameter estimation

Now we are able to model the voltage of a battery as a function of applied current, with model parameters $D_e, D_s, \alpha_s, \beta_e, \beta_s, \gamma, \lambda, \mu$. We want to know which values to take for the model parameters. To find these values, we need observed data consisting of the input (applied current) and output (voltage of the battery). Model parameters are then optimised to minimise the least error between the observed voltage and the modelled voltage of the battery.

Let us denote the observed data at time $m\delta$ by (u_m, y_m) for $m \in \mathbb{N}$, where u_m represents the applied current and y_m is the voltage. For a given set of model parameters $\boldsymbol{p} \in \mathbb{R}^{n_{\boldsymbol{p}}}$, which we would to estimate, we denote the modelled voltage at time $m\delta$ by $f(u_m; \boldsymbol{p})$. Here, $n_{\boldsymbol{p}}$ is the number of parameters which we want to estimate.

We use the quadratic loss function

$$S(\boldsymbol{p}) := \sum_{m=0}^{M} \left(y_m - f(u_m; \boldsymbol{p})\right)^2$$

to measure the discrepancy between the observed and modelled output for a given set of parameters.

To find a set of parameters which minimises this loss function, we use a Levenberg-Marquardt algorithm [2]. This requires us to know the Jacobian $J_{\boldsymbol{p}}$ of the modelled ouput with respect to some given parameters $\boldsymbol{p}$.

$$J_{\boldsymbol{p}} := \begin{pmatrix} \nabla_{\boldsymbol{p}} f(u_0, \boldsymbol{p}) \\ \nabla_{\boldsymbol{p}} f(u_1, \boldsymbol{p}) \\ \vdots \\ \nabla_{\boldsymbol{p}} f(u_M, \boldsymbol{p}) \end{pmatrix} \in \mathbb{R}^{(M+1)\times n_{\boldsymbol{p}}}$$

An approximation of the Jacobian can be computed in a similar way to how the voltage is approximated in section 3 (see appendix in [6]). Instead of finding an approximate solution to the system of equations (1), (2) and (3), we now differentiate this system of equations with respect to each model parameter and then find an approximate solution to these new equations, using a finite difference scheme similar to that of section 3.

4.1 Calculating the Jacobian

In this subsection, we explain how to find the Jacobian $J_{\boldsymbol{p}}$ of $\boldsymbol{f}(\boldsymbol{p})$ with respect $\boldsymbol{p}$, where $\boldsymbol{f}(\boldsymbol{p}) = (f(u_0, \boldsymbol{p}), \cdots, f(u_M, \boldsymbol{p}))^\top$. For simplicity, we will consider the case where $\boldsymbol{p} = (D_s)$, so $n_{\boldsymbol{p}} = 1$. Thus, we need to find $\frac{\partial f(u_m, \boldsymbol{p})}{\partial D_s}$ for $0 \leq m \leq M$.

From equation (5), we see that the voltage at time $m\delta$ is

$$\begin{aligned} f(u_m, \boldsymbol{p}) &= \Phi_{s,N}^{m,+} - \Phi_{s,-N}^{m,-} \\ \implies \frac{\partial f(u_m, \boldsymbol{p})}{\partial D_s} &= \frac{\partial \Phi_{s,N}^{m,+}}{\partial D_s} - \frac{\partial \Phi_{s,-N}^{m,-}}{\partial D_s} \end{aligned} \tag{14}$$

Thus we need to know the derivative of the state vector w.r.t. D_s, $\frac{\partial \boldsymbol{x}_h^m}{\partial D_s}$. We calculate this derivative in a similar way to how we calculated $\boldsymbol{x}_h^m$ by using another finite difference scheme to solve the following equations:

$$\begin{cases} \partial_t \partial_{D_s} c_s = r^{-2}\partial_r \left(r^2 \partial_r c_s\right) + D_s r^{-2}\partial_r \left(r^2 \partial_r \partial_{D_s} c_s\right) & \text{in } [0,1] \\ -\partial_r c_s - D_s \partial_r \partial_{D_s} c_s = \beta_s \partial_{D_s} g & \text{for } r = 1 \\ \partial_r \partial_{D_s} c_s = 0 & \text{for } r = 0 \end{cases} \tag{15}$$

$$\begin{cases} \beta_e \partial_{D_s} g = \partial_t \partial_{D_s} c_e - D_e \Delta_z \partial_{D_s} c_e & \text{in } \Omega_{\mathrm{e}} \\ 0 = \partial_z \partial_{D_s} c_e & \text{for } z \in \partial\Omega_{\mathrm{e}} \\ 0 = \partial_{D_s} c_e & \text{at } t = 0 \end{cases} \tag{16}$$

$$\begin{cases} \partial_{D_s} g = \alpha_s \partial_z^2 \partial_{D_s} \Phi_s & \text{in } \Omega_{\mathrm{e}} \setminus \Omega_{\mathrm{s}} = [-1,1] \setminus \{0\} \\ 0 = \alpha_s \partial_z \partial_{D_s} \Phi_s & \text{in } \partial\Omega_{\mathrm{e}} \\ 0 = \partial_z \partial_{D_s} \Phi_s & \partial\Omega_{\mathrm{s}} \end{cases} \tag{17}$$

Where

$$\partial_{D_s} g = \frac{\partial g}{\partial c_s}\partial_{D_s} c_s + \frac{\partial g}{\partial c_e}\partial_{D_s} c_e + \frac{\partial g}{\partial \Phi_s}\partial_{D_s}\Phi_s \tag{18}$$

This system of equations is merely the derivative of equations (1) - (4) with respect to D_s. Note that, as well as depending on $\partial_{D_s} c_s, \partial_{D_s} c_e, \partial_{D_s}\Phi_s$, equations (15) - (18) also depend on c_s, c_e, Φ_s.

Now, we can implement a finite difference scheme to approximate the solution of (15) - (18). This will tell us how to evolve

$$\partial_{D_s}\boldsymbol{x}_h^m := \left(\partial_{D_s} c_{e,h}^{m,-}, \partial_{D_s} c_{e,h}^{m,+}, \partial_{D_s} c_{s,h}^{m,-}, \partial_{D_s} c_{s,h}^{m,+}, \partial_{D_s}\Phi_{s,h}^{m,-}, \partial_{D_s}\Phi_{s,h}^{m,+}\right)$$

into $\partial_{D_s}\boldsymbol{x}_h^{m+1}$, which allows us to evaluate (14) and, hence, the Jacobian $J_{\boldsymbol{p}}$.

We discretise $\partial_{D_s} c_s$, (15), in a similar way to to the discretisation of c_s, (10). Again, we use the Crank-Nicholson scheme:

$$\begin{aligned} \tfrac{\partial_{D_s} c_{s,j,k}^{m+1,-} - \partial_{D_s} c_{s,j,k,}^{m,-}}{\delta} = \; & \tfrac{1}{2}\left(\tfrac{2}{kh}\tfrac{c_{s,j,k+1}^{m+1,-} - c_{s,j,k}^{m+1,-}}{h} + \tfrac{c_{s,j,k+1}^{m+1,-} - 2c_{s,j,k}^{m+1,-} + c_{s,j,k-1}^{m+1,-}}{h^2}\right) \\ & + \tfrac{1}{2}\left(\tfrac{2}{kh}\tfrac{c_{s,j,k+1}^{m,-} - c_{s,j,k}^{m,-}}{h} + \tfrac{c_{s,j,k+1}^{m,-} - 2c_{s,j,k}^{m,-} + c_{s,j,k-1}^{m,-}}{h^2}\right) \\ & + \tfrac{1}{2}D_s\left(\tfrac{2}{kh}\tfrac{\partial_{D_s} c_{s,j,k+1}^{m+1,-} - \partial_{D_s} c_{s,j,k}^{m+1,-}}{h} + \tfrac{\partial_{D_s} c_{s,j,k+1}^{m+1,-} - 2\partial_{D_s} c_{s,j,k}^{m+1,-} + \partial_{D_s} c_{s,j,k-1}^{m+1,-}}{h^2}\right) && , j \in \Omega_{\mathrm{s,h}}^- \\ & + \tfrac{1}{2}D_s\left(\tfrac{2}{kh}\tfrac{\partial_{D_s} c_{s,j,k+1}^{m,-} - \partial_{D_s} c_{s,j,k}^{m,-}}{h} + \tfrac{\partial_{D_s} c_{s,j,k+1}^{m,-} - 2\partial_{D_s} c_{s,j,k}^{m,-} + \partial_{D_s} c_{s,j,k-1}^{m,-}}{h^2}\right) && , k \in \{1, \cdots, N-1\} \\ -\beta_s \partial_{D_s} g_h^m = \; & \tfrac{c_{s,j,N}^{m+1,-} - c_{s,j,N-1}^{m+1,-}}{h} + D_s \tfrac{\partial_{D_s} c_{s,j,N}^{m+1,-} - \partial_{D_s} c_{s,j,N-1}^{m+1,-}}{h} && , j \in \Omega_{\mathrm{s,h}}^- \\ 0 = \; & \tfrac{\partial_{D_s} c_{s,j,1}^{m+1,-} - \partial_{D_s} c_{s,j,0}^{m+1,-}}{h} && , j \in \Omega_{\mathrm{s,h}}^- \end{aligned} \tag{19}$$

Using matrices A_1, A_2, B_1, B_2, this can be written as

$$A_1 \partial_{D_s} c_{s,h}^{m+1,-} = -A_2 c_{s,h}^{m+1,-} + B_1 \partial_{D_s} c_{s,h}^{m,-} + B_2 c_{s,h}^{m+1,-} + D^{(m),-}$$

where $D^{(m),-} = -\beta_s \left(0, \cdots, 0, \partial_{D_s} g_{-N}^m, \cdots, \partial_{D_s} g_0^m\right)^\top \in \mathbb{R}^{(N+1)^2}$.

Note that in order to calculate $\partial_{D_s} c_{s,h}^{m+1,-}$, we must first know $c_{s,h}^{m+1,-}$ and $c_{s,h}^{m,-}$, which we can calculate using the finite difference scheme in section 3.

In a similar way, we can find $\partial_{D_s} c_{s,h}^{m+1,+}$ and $\partial_{D_s} c_{e,h}^{m+1,\pm}$ in terms of $\partial_{D_s} \boldsymbol{x}_h^m$, $\boldsymbol{x}_h^m$ and $\boldsymbol{x}_h^{m+1}$.

Finally, we find $\partial_{D_s} \Phi_{s,h}^{m+1,-}$ similarly to how we found $\Phi_{s,h}^{m+1,-}$ in section 3. We use auxiliary variables $\partial_{D_s} \tilde{\Phi}_s^{m+1,-} : [-1, 0) \to \mathbb{R}$ and $\tilde{\lambda}^- \in \mathbb{R}$ such that

$$\partial_{D_s} \Phi_s^{m+1,-} = \partial_{D_s} \tilde{\Phi}_s^{m+1,-} + \tilde{\lambda}^-$$

where $\partial_{D_s} \tilde{\Phi}_s^{m+1,-}$ solves (17) with the Neumann boundary condition at $z = 0$ replaced with a Dirichlet boundary condition:

$$\begin{cases} \partial_{D_s} g = \alpha_s \partial_z^2 \left(\partial_{D_s} \tilde{\Phi}_s^{m+1,\pm} \right) & \text{in } \Omega_\text{e} \setminus \Omega_\text{s} = [-1, 1] \setminus \{0\} \\ 0 = \partial_z \left(\partial_{D_s} \tilde{\Phi}_s^{m+1,\pm} \right) & \text{in } \partial\Omega_\text{e} \\ \tilde{\lambda}^\pm = \partial_{D_s} \tilde{\Phi}_s^{m+1,\pm} & , \ z = 0^\pm \end{cases} \tag{20}$$

It is possible to find $\partial_{D_s} \tilde{\Phi}_{s,h}^{m+1,-}$ using matrices as described earlier.

And $\tilde{\lambda}^-$ is the constant satisfying

$$\begin{aligned} 0 &= \int_{-1}^{0} \partial_{D_s} g(z, (m+1)\delta) dz \\ &= \int_{-1}^{0} \left(\frac{\partial g}{\partial c_s} \partial_{D_s} c_s + \frac{\partial g}{\partial c_e} \partial_{D_s} c_e \right) dz \\ &\quad + \int_{-1}^{0} \frac{\partial g}{\partial \Phi_s} \left(\partial_{D_s} \tilde{\Phi}_s + \tilde{\lambda}^- \right) dz \\ \implies \tilde{\lambda}^- &= - \frac{\int_{-1}^{0} \left(\frac{\partial g}{\partial c_s} \partial_{D_s} c_s + \frac{\partial g}{\partial c_e} \partial_{D_s} c_e + \frac{\partial g}{\partial \Phi_s} \partial_{D_s} \tilde{\Phi}_s \right) dz}{\int_{-1}^{0} \frac{\partial g}{\partial \Phi_s} dz} \quad , \quad \text{assuming } \int_{-1}^{0} \frac{\partial g}{\partial \Phi_s} dz \neq 0 \end{aligned}$$

which can be derived from (17).

A similar approach can be taken to find $\Phi_{s,h}^{m+1,+}$.

Now that we have a method of calculating $\partial_{D_s} \boldsymbol{x}_h^{m+1}$ from $\partial_{D_s} \boldsymbol{x}_h^m$, we are able to find $\frac{\partial f(u_{m+1}, \boldsymbol{p})}{\partial D_s}$ using equation (14). Hence, we are able to calculate the Jacobian of $\boldsymbol{f}$ with respect to D_s.

4.2 Using the Jacobian to estimate the model parameters

Now that we have the Jacobian $J_{\boldsymbol{p}}$ of the simulated voltage $\boldsymbol{f}(\boldsymbol{p})$ with respect to model parameters $\boldsymbol{p}$, our goal is to find a set of parameters $\boldsymbol{p}'$ such that the cost $S(\boldsymbol{p}')$ is minimised. We use the following Levenberg-Marquardt (LM) method to iteratively find better and better parameters.

First, we make an initial estimate of the parameter values (e.g. by using trial and error)

$$\boldsymbol{p}_0 := (D_{e,0}, D_{s,0}, \alpha_{s,0}, \beta_{e,0}, \beta_{s,0}, \gamma_0, \lambda_0, \mu_0)^\top$$

and start with an initial damping factor $\Lambda > 0$ for the LM algorithm.

On the k-th iteration of the LM algorithm, $k \in \mathbb{N}$, we compute the update step, defined as

$$d_k = \left(J_{\boldsymbol{p}_k}^\top J_{\boldsymbol{p}_k} + \Lambda I \right)^{-1} J_{\boldsymbol{p}_k}^\top \left(\boldsymbol{y} - \boldsymbol{f}(\boldsymbol{p}_k) \right) \tag{21}$$

where I is the identity matrix. Then we update the parameters according to

$$\boldsymbol{p}'_{k+1} = \boldsymbol{p}_k + d_k \tag{22}$$

If $S(\boldsymbol{p}'_{k+1}) < S(\boldsymbol{p}_k)$, then we set $\boldsymbol{p}_{k+1} = \boldsymbol{p}'_{k+1}$ and reduce Λ by a factor of 3.

However, if $S(\boldsymbol{p}'_{k+1}) \geq S(\boldsymbol{p}_k)$, then we increase Λ by a factor of 2 and repeat (21) and (22).

This process is repeated to find successive collections of parameters $\boldsymbol{p}_i$, such that $S(\boldsymbol{p}_i) > S(\boldsymbol{p}_{i+1})$, and it terminates when $S(\boldsymbol{p}_k) - S(\boldsymbol{p}_{k+1}) < \varepsilon$ for some tolerance $\varepsilon > 0$.

5 Experimental Results

This section illustrates the finite difference model working on input data (applied current, sampled at a frequency of roughly 1 Hz) for a given battery, provided by W. Dhammika Widanage (WMG,

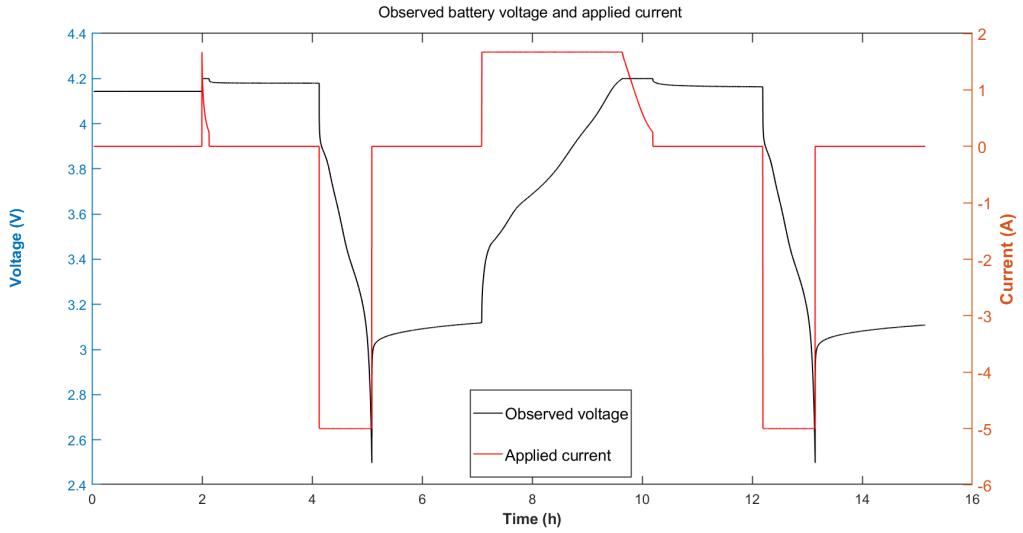

Figure 1: Plot of observed voltage (black) and applied current (red) over time.

International Manufacturing Centre, University of Warwick). We compare the modelled output (voltage of the battery) with the observed output and use a Levenberg-Marquardt method to vary the model parameters such that the quadratic loss function is minimised.

Figure 1 plots the observed voltage with the applied current as input.

We use a spatial distance in the finite difference scheme of $h = \frac{1}{10}$ and a time step of $\delta = 1$ (as the sampling frequency of the input data is 1 Hz). We also use the initial conditions

$$\begin{aligned} c_{s,j,k}^{0,-} &\equiv 0.716 \\ c_{s,j,k}^{0,+} &\equiv 0.284 \\ c_{e,j}^{0,\pm} &\equiv 1 \end{aligned}$$

We make an initial guess of the model parameters

$$\boldsymbol{p}_0 = \begin{pmatrix} D_{e,0} \\ D_{s,0} \\ T \\ \alpha_{s,0} \\ \beta_{e,0} \\ \beta_{s,0} \\ \gamma_0 \\ \lambda_0 \\ \mu_0 \end{pmatrix} = \begin{pmatrix} 10^4 \\ 10^{-3} \\ 300 \\ 10^4 \\ 1 \\ 10^{-5} \\ 10^2 \\ 10^3 \\ 10^4 \end{pmatrix} \tag{23}$$

From these initial parameters, we get a quadratic loss of $S(\boldsymbol{p}_0) = 1263$.

Figure 2 compares the simulated voltage, where the initial model parameters have been used, with the observed voltage. The simulation is, in some places, a good approximation to the observed data. However, there is a relatively large discrepancy around the six and fourteen hour mark.

Using a Levenberg-Marquardt algorithm, we find successive collections of model parameters such that $S(\boldsymbol{p}_{n+1}) < S(\boldsymbol{p}_n)$. We stop the process when $\frac{S(\boldsymbol{p}_n) - S(\boldsymbol{p}_{n+1})}{S(\boldsymbol{p}_n)} < 0.01$, when the relative reduction in cost is less than 1%. Table 1 shows the cost and relative cost reduction for each set of parameters.

Using a coarser spatial discretisation with $h = \frac{1}{5}$, we can achieve a similar level of accuracy, but with improved computation time. The finite difference scheme can simulate 54500s-worth of data from figure 1 in around 38 seconds. This means the scheme can simulate data over 1400 times faster than in real-time.

Figure 3 compares the simulated voltage, where the final model parameters have been used, with the observed voltage. After adjusting the parameters, the model has a much closer fit to the data and is clearly a better approximation to the observed data than when the initial parameters were used (see figure 4).

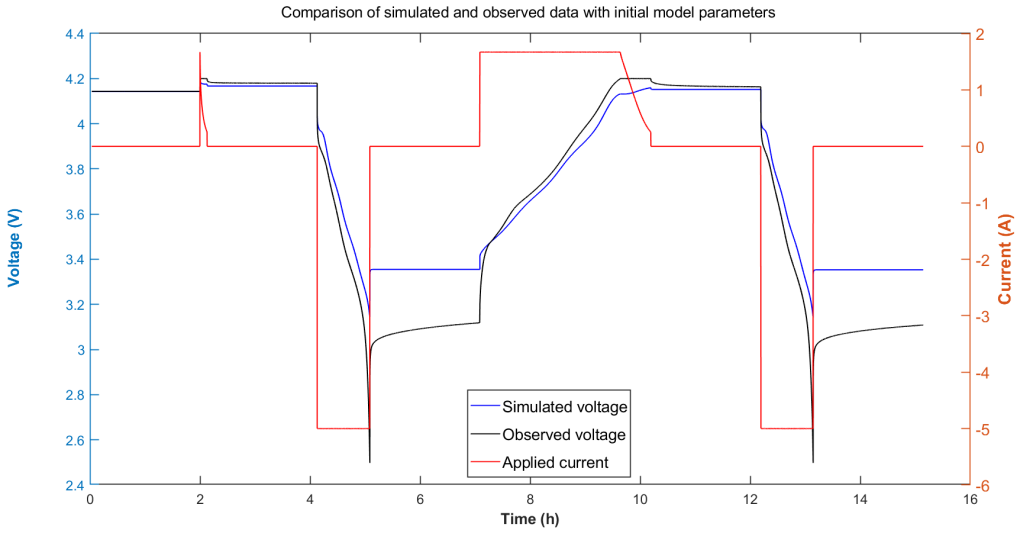

Figure 2: Plot of simulated voltage (blue), using the initial model parameters; observed voltage (black); and applied current (red) over time.

Table 1: Results of each iteration of the Levenberg-Marquardt algorithm.

Iteration	Cost	Relative cost reduction
0	1.26×10^3	–
1	414	0.672
2	241	0.418
3	199	0.172
4	115	0.424
5	85.8	0.252
6	72.3	0.158
7	70.9	0.019
8	64.4	0.092
9	64.3	0.001

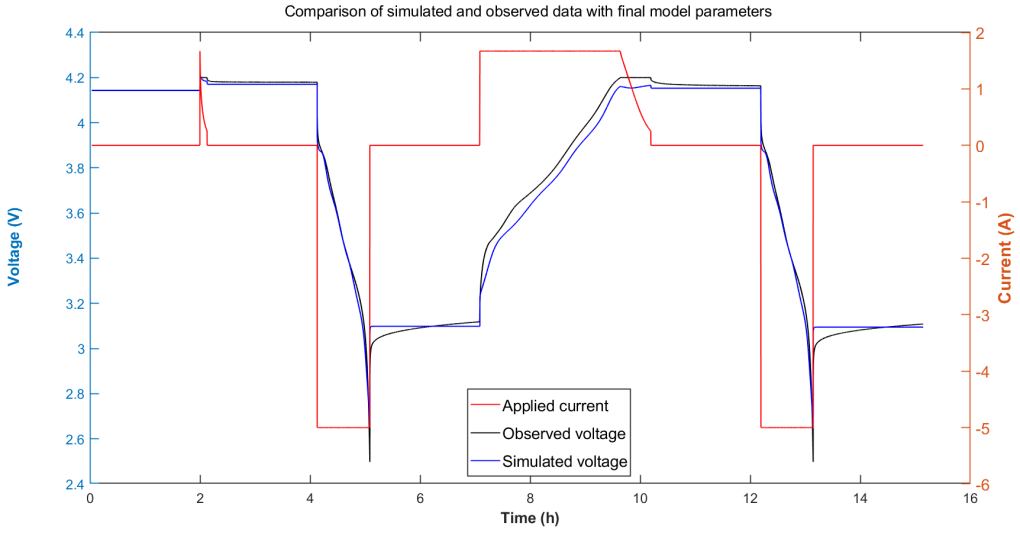

Figure 3: Plot of simulated voltage (blue), using the final model parameters; observed voltage (black); and applied current (red) over time.

6 Learning model structure

We now consider 'forgetting' part of the structure of DFN model and trying to 'relearn' it. For example, in the definition of g , (4), we can replace the square root function with some polynomial

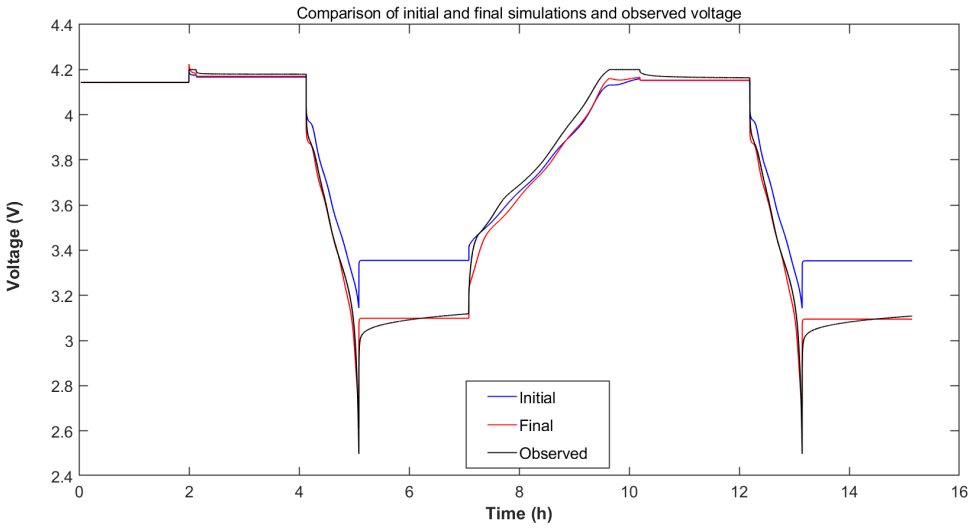

Figure 4: Comparison of the simulated voltage when using the initial model parameters (blue) and final parameters (red), with the observed voltage (black).

$q : \mathbb{R} \to \mathbb{R}$ given by

$$q(x) = \sum_{k=0}^{K} a_k x^k$$

where a_k are the polynomial coefficients which we want to estimate. Thus g becomes

$$g = \mu q(\zeta) \sinh\left(\frac{\lambda}{2}\frac{\eta}{1+\gamma T}\right), \qquad \text{where } \zeta = c_e(1-c_s)c_s$$

By treating the a_k as model parameters, we can use the same techniques mentioned in section 4 to optimise over the polynomial coefficients.

We tried to approximate $\sqrt{\zeta}$ with a polynomial of order 6, i.e $K = 6$. The following initial values for the polynomial coefficients were chosen

$$a_{k,0} := \begin{cases} 1, & k = 1 \\ 0, & k \neq 1 \end{cases}$$

meaning that our initial approximation of $\sqrt{\zeta}$ was $q_0(\zeta) := \zeta$.

Taking the same initial model parameters as in (23), we now have

$$\begin{aligned} \boldsymbol{p}_0 &= (D_{e,0}, D_{s,0}, T, \alpha_{s,0}, \beta_{e,0}, \beta_{s,0}, \gamma_0, \lambda_0, \mu_0, a_{0,0}, a_{1,0}, a_{2,0}, a_{3,0}, a_{4,0}, a_{5,0}, a_{6,0})^\top \\ &= \left(10^4, 10^{-3}, 300, 10^4, 1, 10^{-5}, 10^2, 10^3, 10^4, 0, 1, 0, 0, 0, 0, 0\right)^\top \end{aligned}$$

After estimating these parameters, using section 4, we obtained

$$g_{\text{est}}(\zeta, \eta) = 10^4 q_{\text{est}}(\zeta) \sinh\left(\frac{1000}{2}\frac{\eta}{1+100\times 300}\right)$$

where $q_{\text{est}}(\zeta) = -0.056 + 2.565\zeta - 6.255\zeta^2 + 5.598\zeta^3 + 1.521\zeta^4 - 0.850\zeta^5 - 1.007\zeta^6$. To compare, we obtained

$$\tilde{g}(\zeta, \eta) = 241\sqrt{\zeta} \sinh\left(\frac{1030}{2}\frac{\eta}{1+4.30\times 300}\right)$$

when using the correct structure for g, using the square root as in (4).

Figure 5 shows how much closer g_{est} is to $\tilde{g}$ than g_0 is to $\tilde{g}$, where g_0 is the initial approximation of $\sqrt{\zeta}$ using $q_0(\zeta) = \zeta$. Here, we are only considering the domain $\zeta \in [0, 0.3]$ because $\zeta = c_e(1-c_s)c_s$, where $c_e \approx 1$ and $c_s \in [0, 1]$. Moreover, during simulations at a 1C discharge voltage response, we found $\eta \in [-0.3, 0.3]$. Hence, figure 5 gives us a reasonable idea of well our polynomial approximation of the square root is.

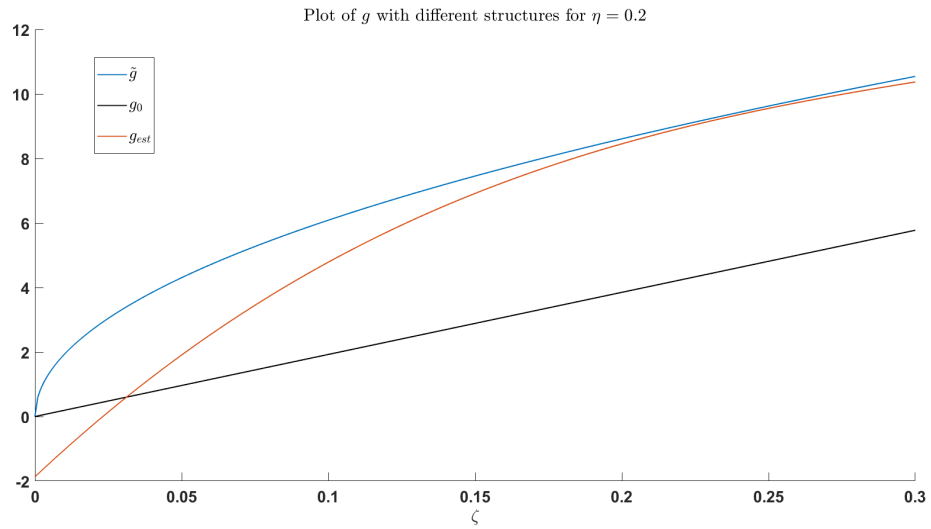

Figure 5: Plot of the different structures of $g(\zeta, \eta)$ at $\eta = 0.2$. The correct model structure is denoted $\tilde{g}$ (blue); the initial approximation of $\tilde{g}$ is denoted g_0 (black); and the final approximation is denoted g_{est} (red).

7 Conclusion

We have implemented a simplified version of the DFN model for Li-ion batteries, which allows us to forecast the state of charge of a given battery. We used a finite difference scheme to implement the model which, depending on the number of discretisation points, can simulate data approximately 1400 times faster than in real time. This computational efficiency makes the scheme suitable for real-time battery analysis.

We are also able to estimate the model parameters for any Li-ion battery, as long as there is sufficient training data provided. This allows us to calibrate the scheme to the given battery and improve the accuracy of predictions made about its state of charge. Thus, we are able to calibrate the model to a Li-ion battery and efficiently predict the state of charge for any given current applied to the battery.

Moreover, we can 'learn' part of the model structure by approximating some functions by, for example, polynomials. This enables us to accurately model certain phenomena without already knowing the theoretical model equations.

References

[1] Marc Doyle, Thomas F. Fuller, and John Newman. Modeling of galvanostatic charge and discharge of the lithium/polymer/insertion cell. *Journal of The Electrochemical Society*, 140(6), 1993.

[2] Kenneth Levenberg. A method for the solution of certain non-linear problems in least squares a method for the solution of certain non-linear problems in least squares. *Quarterly of Applied Mathematics*, 2(2):164–168, 1944.

[3] Matthew Hunt, Florian Theil, Ferran Brosa Planella, and W. Widanage. Coupling temperature distribution with the single particle model. arXiv:2208.05448. 2022.

[4] Chang-Hui Chen, Ferran Brosa Planella, Kieran O'Regan, Dominika Gastol, W. Dhammika Widanage, and Emma Kendrick. Development of experimental techniques for parameterization of multi-scale lithium-ion battery models. *J. Electrochem. Soc.*, 167(8):080534, 2020.

[5] J. Crank and P. Nicolson. A practical method for numerical evaluation of solutions of partial differential equations of the heat-conduction type. *Mathematical Proceedings of the Cambridge Philosophical Society*, 43(1):50–67, 1947.

[6] Johan Paduart, Lieve Lauwers, Jan Swevers, Kris Smolders, Johan Schoukens, and Rik Pintelon. Identification of nonlinear systems using polynomial nonlinear state space models. *Automatica*, 46(4):647–656, 2010.