

Numerical Implementation of the Doyle-Fuller-Newman (DFN) Model

Prof. Scott Moura | Energy, Controls, and Applications Lab (eCAL) | UC Berkeley

LAST UPDATED | June 12, 2017 | by Saehong Park

In this note we document the numerical implementation of the DFN model.

1 Doyle-Fuller-Newman Model

We consider the Doyle-Fuller-Newman (DFN) model in Fig. 1 to predict the evolution of lithium concentration in the solid $c_s^\pm(x, r, t)$, lithium concentration in the electrolyte $c_e(x, t)$, solid electric potential $\phi_s^\pm(x, t)$, electrolyte electric potential $\phi_e(x, t)$, ionic current $i_e^\pm(x, t)$, molar ion fluxes $j_n^\pm(x, t)$, and bulk cell temperature $T(t)$ [?]. The governing equations are given by

$$\frac{\partial c_s^\pm}{\partial t}(x, r, t) = \frac{1}{r^2} \frac{\partial}{\partial r} \left[D_s^\pm r^2 \frac{\partial c_s^\pm}{\partial r}(x, r, t) \right], \quad (1)$$

$$\varepsilon_e^j \frac{\partial c_e^j}{\partial t}(x, t) = \frac{\partial}{\partial x} \left[D_e^{\text{eff}}(c_e^j) \frac{\partial c_e^j}{\partial x}(x, t) + \frac{1 - t_c^0}{F} i_e^j(x, t) \right], \quad (2)$$

$$0 = \frac{\partial \phi_s^\pm}{\partial x}(x, t) - \frac{i_e^\pm(x, t) - I(t)}{\sigma^{\text{eff}, \pm}}, \quad (3)$$

$$0 = \kappa^{\text{eff}}(c_e) \cdot \frac{\partial \phi_e}{\partial x}(x, t) + i_e^\pm(x, t) - \kappa^{\text{eff}}(c_e) \cdot \frac{2RT}{F} (1 - t_c^0) \times \left(1 + \frac{d \ln f_{c/a}}{d \ln c_e}(x, t) \right) \frac{\partial \ln c_e}{\partial x}(x, t), \quad (4)$$

$$0 = \frac{\partial i_e^\pm}{\partial x}(x, t) - a_s^\pm F j_n^\pm(x, t), \quad (5)$$

$$0 = \frac{1}{F} i_0^\pm(x, t) \left[e^{\frac{\alpha_a F}{RT} \eta^\pm(x, t)} - e^{-\frac{\alpha_c F}{RT} \eta^\pm(x, t)} \right] - j_n^\pm(x, t), \quad (6)$$

$$\rho^{\text{avg}}_{c_P} \frac{dT}{dt}(t) = h_{\text{cell}} [T_{\text{amb}}(t) - T(t)] + I(t)V(t) - \int_{0^-}^{0^+} a_s F j_n(x, t) \Delta T(x, t) dx, \quad (7)$$

where $D_e, \kappa, f_{c/a}$ are functions of $c_e(x, t)$ and $D_e^{\text{eff}} = D_e(c_e) \cdot (\varepsilon_e^j)^{\text{brug}}$, $\sigma^{\text{eff}} = \sigma \cdot (\varepsilon_s^j + \varepsilon_f^j)^{\text{brug}}$, $\kappa^{\text{eff}} = \kappa(c_e) \cdot (\varepsilon_e^j)^{\text{brug}}$ are the effective electrolyte diffusivity, effective solid conductivity, and

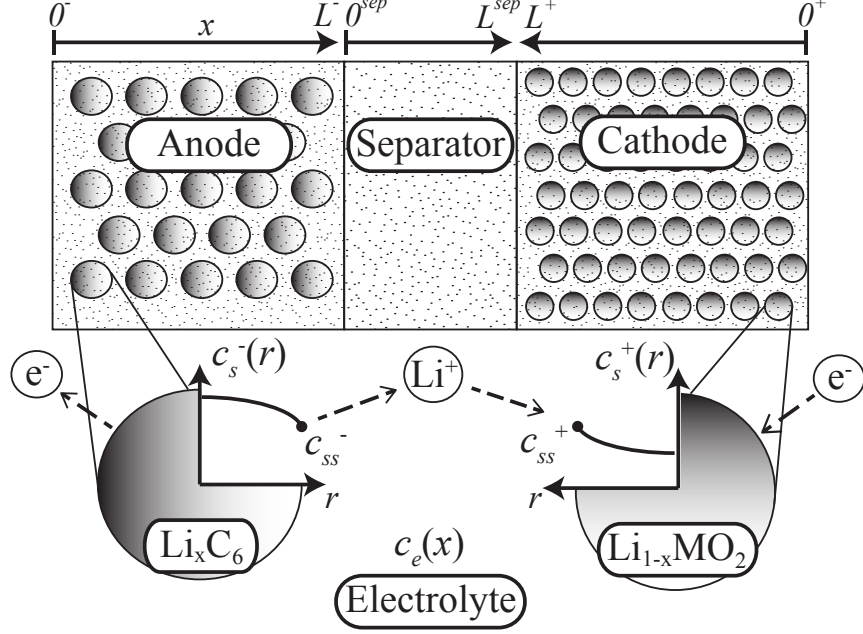


Figure 1: Schematic of the Doyle-Fuller-Newman model [?]. The model considers two phases: the solid and electrolyte. In the solid, states evolve in the x and r dimensions. In the electrolyte, states evolve in the x dimension only. The cell is divided into three regions: anode, separator, and cathode.

effective electrolyte conductivity given by the Bruggeman relationship. We also have

$$i_0^\pm(x, t) = k^\pm [c_{ss}^\pm(x, t)]^{\alpha_c} [c_e(x, t) (c_{s, \max}^\pm - c_{ss}^\pm(x, t))]^{\alpha_a}, \quad (8)$$

$$\eta^\pm(x, t) = \phi_s^\pm(x, t) - \phi_e(x, t) - U^\pm(c_{ss}^\pm(x, t)) - FR_f^\pm j_n^\pm(x, t), \quad (9)$$

$$c_{ss}^\pm(x, t) = c_s^\pm(x, R_s^\pm, t), \quad (10)$$

$$\Delta T(x, t) = U^\pm(\bar{c}_s^\pm(x, t)) - T(t) \frac{\partial U^\pm}{\partial T}(\bar{c}_s^\pm(x, t)), \quad (11)$$

$$\bar{c}_s^\pm(x, t) = \frac{3}{(R_s^\pm)^3} \int_0^{R_s^\pm} r^2 c_s^\pm(x, r, t) dr \quad (12)$$

Along with these equations are corresponding boundary and initial conditions. The boundary conditions for the solid-phase diffusion PDE (1) are

$$\frac{\partial c_s^\pm}{\partial r}(x, 0, t) = 0, \quad (13)$$

$$\frac{\partial c_s^\pm}{\partial r}(x, R_s^\pm, t) = -\frac{1}{D_s^\pm} j_n^\pm. \quad (14)$$

The boundary conditions for the electrolyte-phase diffusion PDE (2) are given by

$$\frac{\partial c_e}{\partial x}(0^-, t) = \frac{\partial c_e}{\partial x}(0^+, t) = 0, \quad (15)$$

$$D_e^{\text{eff},-}(c_e(L^-))\frac{\partial c_e}{\partial x}(L^-,t) = D_e^{\text{eff,sep}}(c_e(0^{\text{sep}}))\frac{\partial c_e}{\partial x}(0^{\text{sep}},t), \quad (16)$$

$$D_e^{\text{eff,sep}}(c_e(L^{\text{sep}}))\frac{\partial c_e}{\partial x}(L^{\text{sep}},t) = D_e^{\text{eff},+}(c_e(L^+))\frac{\partial c_e}{\partial x}(L^+,t), \quad (17)$$

$$c_e(L^-,t) = c_e(0^{\text{sep}},t), \quad (18)$$

$$c_e(L^{\text{sep}},t) = c_e(L^+,t). \quad (19)$$

The boundary conditions for the solid-phase potential ODE (3) are given by

$$\frac{\partial \phi_s^-}{\partial x}(L^-,t) = \frac{\partial \phi_s^+}{\partial x}(L^+,t) = 0. \quad (20)$$

The boundary conditions for the electrolyte-phase potential ODE (4) are given by

$$\phi_e(0^-,t) = 0, \quad (21)$$

$$\phi_e(L^-,t) = \phi_e(0^{\text{sep}},t), \quad (22)$$

$$\phi_e(L^{\text{sep}},t) = \phi_e(L^+,t). \quad (23)$$

The boundary conditions for the ionic current ODE (5) are given by

$$i_e^-(0^-,t) = i_e^+(0^+,t) = 0 \quad (24)$$

and also note that $i_e(x,t) = I(t)$ for $x \in [0^{\text{sep}}, L^{\text{sep}}]$.

In addition, the parameters $D_s^\pm, D_e, \kappa_e, k^\pm$ vary with temperature via the Arrhenius relationship:

$$\psi = \psi_{ref} \exp \left[\frac{E_\psi}{R} \left(\frac{1}{T} - \frac{1}{T_{ref}} \right) \right] \quad (25)$$

where ψ represents a temperature dependent parameter, E_ψ is the activation energy [J/mol], and ψ_{ref} is the reference parameter value at reference temperature T_{ref} .

The input to the model is the applied current density $I(t)$ [A/m²], and the output is the voltage measured across the current collectors

$$V(t) = \phi_s^+(0^+,t) - \phi_s^-(0^-,t) - R_c I(t) \quad (26)$$

Further details, including notation definitions, can be found in [?, ?].

2 Time-stepping

Ultimately, the equations are discretized to produce a DAE in the following format:

$$\dot{x} = f(x, z, u), \quad (27)$$

$$0 = g(x, z, u) \quad (28)$$

with initial conditions $x(0), z(0)$ that are consistent. That is, they verify (28). The function $f(x, z, u)$ and $g(x, z, u)$ are computed in Matlab function `dae_dfn.m`.

The time-stepping is done by solving the nonlinear equation

$$0 = F(x(t + \Delta t), z(t + \Delta t)), \quad (29)$$

$$0 = \begin{bmatrix} x(t) - x(t + \Delta t) + \frac{1}{2}\Delta t [f(x(t), z(t), u(t)) + f(x(t + \Delta t), z(t + \Delta t), u(t + \Delta t))] \\ g(x(t + \Delta t), z(t + \Delta t), u(t + \Delta t)) \end{bmatrix} \quad (30)$$

for $x(t + \Delta t), z(t + \Delta t)$. The function `cfndfn.m` returns the solution $(x(t + \Delta t), z(t + \Delta t))$ of (29)-(30), given $x(t), z(t), u(t), u(t + \Delta t)$. Note that we solve (29)-(30) using Newton's method, meaning analytic Jacobians of $F(\cdot, \cdot)$ are required w.r.t. x, z .

$$J = \begin{bmatrix} F_x^1 & F_z^1 \\ F_x^2 & F_z^2 \end{bmatrix} \quad (31)$$

$$= \begin{bmatrix} -I + \frac{1}{2}\Delta t \cdot \frac{\partial f}{\partial x}(x(t + \Delta t), z(t + \Delta t), u(t + \Delta t)) & \frac{1}{2}\Delta t \cdot \frac{\partial f}{\partial z}(x(t + \Delta t), z(t + \Delta t), u(t + \Delta t)) \\ \frac{\partial g}{\partial x}(x(t + \Delta t), z(t + \Delta t), u(t + \Delta t)) & \frac{\partial g}{\partial z}(x(t + \Delta t), z(t + \Delta t), u(t + \Delta t)) \end{bmatrix} \quad (32)$$

The analytic Jacobian is computed in two functions: (i) Function `jac_dfn_pre.m` computes Jacobian elements that are NOT state/time-dependent. (ii) Function `jac_dfn.m` computes Jacobian elements that are state/time-dependent. The secret to realizing fast and accurate simulations is carefully deriving the Jacobians and coding them in the functions above.

3 DAEs

To perform the time-stepping in the previous section, we must compute functions $f(x, z, u)$ and $g(x, z, u)$. These functions, which represent the RHS of (27)-(28), are calculated by the Matlab function `dae_dfn.m`, given the inputs x, z, u . The role of variables x, z, u are played by the DFN variables shown in Table 1.

Table 1: DAE notation for DFN states in Matlab Code

| DAE Variable | DFN Variable |
|--------------|---|
| x | $c_s^-, c_s^+, c_e = [c_e^-, c_e^{sep}, c_e^+], T$ |
| z | $\phi_s^-, \phi_s^+, i_e^-, i_e^+, \phi_e = [\phi_e^-, \phi_e^{sep}, \phi_e^+], j_n^-, j_p^+$ |
| u | I |

In the subsequent sections, we go through each DFN variable listed in Table 1 and document its numerical implementation.

4 Solid Concentration, c_s^-, c_s^+

[DONE] The PDEs (1) and BCs (13)-(14) governing Fickian diffusion in the solid phase are implemented using third order Padé approximations of the transfer function from j_n^\pm to c_{ss}^\pm [?, ?]:

$$\frac{C_{ss}^\pm(s)}{J_n^\pm(s)} = \frac{-\frac{(R_s^\pm)^3}{165(D_s^\pm)^2}s^2 - \frac{4R_s^\pm}{11D_s^\pm}s - \frac{3}{R_s^\pm}}{\frac{(R_s^\pm)^4}{3465(D_s^\pm)^2}s^3 + \frac{3(R_s^\pm)^2}{55D_s^\pm}s^2 + s}, \quad (33)$$

which we notate more simply by

$$\frac{C_{ss}^\pm(s)}{J_n^\pm(s)} = \frac{b_2s^2 + b_1s + b_0}{a_3s^3 + a_2s^2 + a_1s + a_0}. \quad (34)$$

Next we multiply top and bottom by $\frac{1}{a_3}$ to yield a unity coefficient on the highest-order term in the denominator, yielding

$$\frac{C_{ss}^\pm(s)}{J_n^\pm(s)} = \frac{\bar{b}_2s^2 + \bar{b}_1s + \bar{b}_0}{s^3 + \bar{a}_2s^2 + \bar{a}_1s + \bar{a}_0}, \quad (35)$$

where $\bar{b}_i = b_i/a_3$ and $\bar{a}_i = a_i/a_3$.

The transfer function (35) is converted into controllable canonical state-space form, thus producing the subsystem:

$$\frac{d}{dt} \begin{bmatrix} c_{s1}^\pm(t) \\ c_{s2}^\pm(t) \\ c_{s3}^\pm(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -\bar{a}_0 & -\bar{a}_1 & -\bar{a}_2 \end{bmatrix} \begin{bmatrix} c_{s1}^\pm(t) \\ c_{s2}^\pm(t) \\ c_{s3}^\pm(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} j_n^\pm(t) \quad (36)$$

$$\begin{bmatrix} c_{ss}^\pm(t) \end{bmatrix} = \begin{bmatrix} \bar{b}_0 & \bar{b}_1 & \bar{b}_2 \end{bmatrix} \begin{bmatrix} c_{s1}^\pm(t) \\ c_{s2}^\pm(t) \\ c_{s3}^\pm(t) \end{bmatrix} \quad (37)$$

for each discrete point in x .

To simplify analytical applications for the DFN model, we seek a different state-space realization in which bulk concentration $\bar{c}_s^\pm(t)$ is expressed as a state. This can be found by pursuing a Jordan-form state-space realization, which diagonalizes the system matrix. The diagonal elements represent the system eigenvalues. The zero eigenvalue corresponds to bulk concentration. All the aforementioned calculations are performed in Matlab function `c_s_mats.m`. The final result produces system matrices:

$$\frac{d}{dt} \begin{bmatrix} c_{s1}^\pm(t) \\ c_{s2}^\pm(t) \\ c_{s3}^\pm(t) \end{bmatrix} = A_{cs}^\pm \begin{bmatrix} c_{s1}^\pm(t) \\ c_{s2}^\pm(t) \\ c_{s3}^\pm(t) \end{bmatrix} + B_{cs}^\pm j_n^\pm(t) \quad (38)$$

$$\begin{bmatrix} c_{ss}^\pm(t) \\ \bar{c}_s^\pm(t) \end{bmatrix} = C_{cs}^\pm \begin{bmatrix} c_{s1}^\pm(t) \\ c_{s2}^\pm(t) \\ c_{s3}^\pm(t) \end{bmatrix} \quad (39)$$

where the second row of C_{cs}^\pm is $[C_{cs}^\pm]_2 = [0, 0, 1]$. Note we have abused notation. The states in (38)-(39) are not the same as those in (36)-(37), due to the different realizations. Also, these matrices must be computed online, due to the temperature dependence of D_s^\pm described in (25).

5 Electrolyte Concentration, c_e

[DONE] The electrolyte concentration PDE (2) combined with (5), and BCs (15)-(19) are implemented using the central difference method, which ultimately produces the matrix differential equation:

$$\frac{d}{dt}c_e^j(t) = \frac{dD_e^{\text{eff},j}}{dc_e}(c_e^j) [M_{ce}^{j,1}c_e^j(t) + M_{ce}^{j,2}c_{e,bc}^j(t)]^2 + D_e^{\text{eff},j}(c_e^j) [M_{ce}^{j,3}c_e^j(t) + M_{ce}^{j,4}c_{e,bc}^j(t)] + M_{ce}^{j,5}j_n^j(t) \quad (40)$$

where c_e, j_n^j are vectors whose elements represent discrete points along the x -dimension of the DFN model. The variable $c_{e,bc}^j$ represents the boundary values for region j , namely $c_{e,bc}^- = [c_{e,bc,1}, c_{e,bc,2}]^T$, $c_{e,bc}^{\text{sep}} = [c_{e,bc,2}, c_{e,bc,3}]^T$, $c_{e,bc}^+ = [c_{e,bc,3}, c_{e,bc,4}]^T$. The boundary values are computed by: $c_{e,bc}(t) = C_{ce} c_e(t)$ where $c_{e,bc} = [c_{e,bc,1}, c_{e,bc,2}, c_{e,bc,3}, c_{e,bc,4}]^T$.

Note that the effective diffusivity, $D_e^{\text{eff},j}(c_e^j)$, and its derivative, $\frac{dD_e^{\text{eff},j}}{dc_e}(c_e^j)$, are state-dependent and must be computed online. The matrices $M_{ce}^{j,1}, M_{ce}^{j,2}, M_{ce}^{j,3}, M_{ce}^{j,4}, M_{ce}^{j,5}, C_{ce}$ are computed offline by Matlab function `c_e_mats.m`. These matrices are given by

$$M_{ce}^{j,1} = \frac{1}{2L^j\Delta x^j} \begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ -1 & 0 & 1 & \dots & 0 \\ 0 & -1 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & & \\ 0 & 0 & \dots & \dots & 1 \\ 0 & 0 & \dots & -1 & 0 \end{bmatrix}, \quad M_{ce}^{j,2} = \frac{1}{2L^j\Delta x^j} \begin{bmatrix} -1 & 0 \\ 0 & 0 \\ \vdots & \vdots \\ 0 & 0 \\ 0 & 1 \end{bmatrix}, \quad (41)$$

$$M_{ce}^{j,3} = \frac{1}{(L^j\Delta x^j)^2} \begin{bmatrix} 1 & -2 & 1 & \dots & 0 \\ 0 & 1 & -2 & \dots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & \dots & 1 & -2 & 1 \\ 0 & 0 & \dots & 0 & 1 & -2 \end{bmatrix}, \quad M_{ce}^{j,4} = \frac{1}{(L^j\Delta x^j)^2} \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ \vdots & \vdots \\ 0 & 0 \\ 0 & 1 \end{bmatrix}, \quad (42)$$

$$M_{ce}^{j,5} = \frac{(1 - t_c^0)a_s^j}{\varepsilon_e^j}. \quad (43)$$

for $j \in \{-, \text{sep}, +\}$. The matrix C_{ce} is given by

$$C_{ce} = -(N_{ce}^2)^{-1}(N_{ce}^1) \quad (44)$$

where

$$N_{ce}^1 = \left[\begin{array}{cccc|cc} 4 & -1 & \dots & 0 & 0 & 0 & 0 & \dots \\ 0 & 0 & \dots & \frac{(\varepsilon_e^-)^{\text{brug}}}{2L^- \Delta x^-} & -4 \frac{(\varepsilon_e^-)^{\text{brug}}}{2L^- \Delta x^-} & -4 \frac{(\varepsilon_e^{\text{sep}})^{\text{brug}}}{2L^{\text{sep}} \Delta x^{\text{sep}}} & \frac{(\varepsilon_e^{\text{sep}})^{\text{brug}}}{2L^{\text{sep}} \Delta x^{\text{sep}}} & \dots \\ 4 & 0 & \dots & 0 & 0 & 0 & 0 & \dots \\ 0 & 0 & \dots & 0 & 0 & 0 & 0 & \dots \end{array} \right]$$

$$\begin{array}{ccc|ccc} \dots & 0 & 0 & 0 & 0 & \dots & 0 & 0 \\ \dots & 0 & 0 & 0 & 0 & \dots & 0 & 0 \\ \dots & \frac{(\epsilon_e^{\text{sep}})^{\text{brug}}}{2L^{\text{sep}}\Delta x^{\text{sep}}} & -4\frac{(\epsilon_e^{\text{sep}})^{\text{brug}}}{2L^{\text{sep}}\Delta x^{\text{sep}}} & -4\frac{(\epsilon_e^+)^{\text{brug}}}{2L^+\Delta x^+} & \frac{(\epsilon_e^+)^{\text{brug}}}{2L^+\Delta x^+} & \dots & 0 & 0 \\ \dots & 0 & 0 & 0 & 0 & \dots & 1 & -4 \end{array} \quad (45)$$

$$N_{ce}^2 = \begin{bmatrix} -3 & 0 & 0 & 0 & 0 \\ 0 & 3\frac{(\epsilon_e^-)^{\text{brug}}}{2L^-\Delta x^-} + 3\frac{(\epsilon_e^{\text{sep}})^{\text{brug}}}{2L^{\text{sep}}\Delta x^{\text{sep}}} & 0 & 0 & 0 \\ 0 & 0 & 3\frac{(\epsilon_e^{\text{sep}})^{\text{brug}}}{2L^{\text{sep}}\Delta x^{\text{sep}}} + 3\frac{(\epsilon_e^+)^{\text{brug}}}{2L^+\Delta x^+} & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 \end{bmatrix}. \quad (46)$$

Note, we have used second-order accurate finite difference approximations for the boundary conditions.

6 Temperature, T

[DONE] Temperature is scalar, so the ODE (7) is directly implemented as:

$$\rho^{\text{avg}} c_P \frac{dT}{dt}(t) = h_{\text{cell}} [T_{\text{amb}}(t) - T(t)] + I(t)V(t) - \int_{0^-}^{0^+} a_s F j_n(x, t) \Delta T(x, t) dx, \quad (47)$$

$$\Delta T(x, t) = U^\pm(\bar{c}_s^\pm(x, t)) - T(t) \frac{\partial U^\pm}{\partial T}(\bar{c}_s^\pm(x, t)), \quad (48)$$

$$\bar{c}_s^\pm(x, t) = \frac{3}{(R_s^\pm)^3} \int_0^{R_s^\pm} r^2 c_s^\pm(x, r, t) dr \quad (49)$$

7 Solid Potential, ϕ_s^-, ϕ_s^+

[DONE] The solid potential ODE (3) and BCs (20) are implemented using the central difference method, which ultimately produces the matrix equation:

$$\frac{d}{dt} \phi_s^-(t) = F_{psn}^1 \phi_s^-(t) + F_{psn}^2 i_{e, \text{aug}}^-(t) + G_{psn} I(t) = 0 \quad (50)$$

$$\frac{d}{dt} \phi_s^+(t) = F_{psp}^1 \phi_s^+(t) + F_{psp}^2 i_{e, \text{aug}}^+(t) + G_{psp} I(t) = 0. \quad (51)$$

where $i_{e, \text{aug}}^\pm$ are

$$i_{e, \text{aug}}^-(t) = \begin{bmatrix} 0 \\ i_e^-(x, t) \\ I(t) \end{bmatrix}, \quad i_{e, \text{aug}}^+(t) = \begin{bmatrix} I(t) \\ i_e^+(x, t) \\ 0 \end{bmatrix} \quad (52)$$

This section also computes the terminal voltage $V(t)$ from (26) using matrix equations

$$\phi_{s, bc}^-(t) = C_{psn} \phi_s^-(t) + D_{psn} I(t), \quad (53)$$

$$\phi_{s, bc}^+(t) = C_{psp} \phi_s^+(t) + D_{psp} I(t), \quad (54)$$

$$V(t) = \phi_{s,bc,2}^+(t) - \phi_{s,bc,1}^-(t) - R_c I(t) \quad (55)$$

where the following matrices are computed a priori by Matlab function `phi_s_mats.m`

$$(F1n) = (M1n) - (M2n)(N2n)^{-1}(N1n), \quad (56)$$

$$(F2n) = (M3n), \quad (57)$$

$$(Gn) = (M4n) - (M2n)(N2n)^{-1}(N3n), \quad (58)$$

$$(F1p) = (M1p) - (M2p)(N2p)^{-1}(N1p), \quad (59)$$

$$(F2p) = (M3p), \quad (60)$$

$$(Gp) = (M4p) - (M2p)(N2p)^{-1}(N3p), \quad (61)$$

$$(Cn) = -(N2n)^{-1}(N1n), \quad (62)$$

$$(Dn) = -(N2n)^{-1}(N3n), \quad (63)$$

$$(Cp) = -(N2p)^{-1}(N1p), \quad (64)$$

$$(Dp) = -(N2p)^{-1}(N3p), \quad (65)$$

where the (Mij) and $N(ij)$ matrices result from central difference approximations of the ODE in space (3) and boundary conditions (20).

$$(M1j) = \begin{bmatrix} 0 & \alpha_j & 0 & \dots & 0 \\ -\alpha_j & 0 & \alpha_j & \dots & 0 \\ 0 & -\alpha_j & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & & \\ 0 & 0 & \dots & \dots & \alpha_j \\ 0 & 0 & \dots & -\alpha_j & 0 \end{bmatrix}, \quad (M2j) = \begin{bmatrix} -\alpha_j & 0 \\ 0 & 0 \\ \vdots & \vdots \\ 0 & 0 \\ 0 & \alpha_j \end{bmatrix}, \quad (66)$$

$$(M3j) = \frac{1}{\sigma_{\text{ref},\pm}} \begin{bmatrix} 0 & -1 & 0 & \dots & 0 \\ 0 & 0 & -1 & \dots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & \dots & -1 & 0 & 0 \\ 0 & 0 & \dots & 0 & -1 & 0 \end{bmatrix}, \quad (M4j) = \frac{1}{\sigma_{\text{ref},\pm}} \mathbb{I} \quad (67)$$

$$(N1j) = \begin{bmatrix} 4\alpha_j & -\alpha_j & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & \dots & 1\alpha_j & -4\alpha_j \end{bmatrix}, \quad (N2j) = \begin{bmatrix} -3\alpha_j & 0 \\ 0 & 3\alpha_j \end{bmatrix}, \quad (68)$$

$$(N3n) = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad (N3p) = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad (69)$$

for $j \in \{n, p\}$, $\alpha_j = 1/(2L^j \Delta x^j)$. Note, we have used second-order accurate finite difference approximations for the boundary conditions.

8 Electrolyte Current, i_e^-, i_e^+

[DONE] The electrolyte current ODE (5) and BCs (24) are implemented using the central difference method, which ultimately produces the matrix equation:

$$\frac{d}{dt}i_e^-(t) = F_{ien}^{1-} i_e^-(t) + F_{ien}^{2-} j_n^-(t) + F_{ien}^{3-} I(t) \quad (70)$$

$$\frac{d}{dt}i_e^+(t) = F_{iep}^{1+} i_e^+(t) + F_{iep}^{2+} j_n^+(t) + F_{iep}^{3+} I(t) \quad (71)$$

where the following matrices are computed a priori by Matlab function `i_e_mats.m`

$$F_{ien}^{1-} = (M1n) - (M2n)(N2n)^{-1}(N1n), \quad (72)$$

$$F_{ien}^{2-} = (M3n) - (M2n)(N2n)^{-1}(N3n), \quad (73)$$

$$F_{ien}^{3-} = (M2n)(N2n)^{-1}(N4n), \quad (74)$$

$$F_{iep}^{1+} = (M1p) - (M2p)(N2p)^{-1}(N1p), \quad (75)$$

$$F_{iep}^{2+} = (M3p) - (M2p)(N2p)^{-1}(N3p), \quad (76)$$

$$F_{iep}^{3+} = (M2p)(N2p)^{-1}(N4p) \quad (77)$$

where the (Mij) and $N(ij)$ matrices result from central difference approximations of the ODE in space (5) and boundary conditions (24).

$$(M1j) = \begin{bmatrix} 0 & \alpha_j & 0 & \dots & 0 \\ -\alpha_j & 0 & \alpha_j & \dots & 0 \\ 0 & -\alpha_j & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & & \\ 0 & 0 & \dots & \dots & \alpha_j \\ 0 & 0 & \dots & -\alpha_j & 0 \end{bmatrix}, \quad (M2j) = \begin{bmatrix} -\alpha_j & 0 \\ 0 & 0 \\ \vdots & \vdots \\ 0 & 0 \\ 0 & \alpha_j \end{bmatrix}, \quad (M3j) = -\beta_j \mathbb{I}, \quad (78)$$

$$(N1j) = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 \end{bmatrix}, \quad (N2j) = \mathbb{I}, \quad (N3j) = (N1j), \quad (79)$$

$$(N4n) = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad (N4p) = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad (80)$$

for $j \in \{n, p\}$, $\alpha_j = (2L^j \Delta x^j)^{-1}$, $\beta_j = a_s^j F$.

9 Electrolyte Potential, ϕ_e

[INCOMPLETE] The electrolyte potential is implemented using the central difference method, which ultimately produces the matrix equation:

$$\frac{d}{dt}\phi_e^-(t) = F_{pe}^1(c_{e,x}) \cdot \phi_e(t) + F_{pe}^2(c_{e,x}) \cdot i_{e,x}(t) + F_{pe}^3(c_{e,x}) \cdot \ln(c_{e,x}(t)) \quad (81)$$

where vectors $i_{e,x}$ and $c_{e,x}$ represent the entire electrolyte current and concentration, respectively, across the entire battery, including boundary values,

$$i_{e,x}(t) = [0, i_e^-(x, t), I(x, t), i_e^+(x, t), 0]^T, \quad (82)$$

$$c_{e,x}(t) = [c_{e,bc,1}(t), c_e^-(x, t), c_{e,bc,2}(t), c_e^{sep}(x, t), c_{e,bc,3}(t), c_e^+(x, t), c_{e,bc,4}(t)]^T, \quad (83)$$

$$c_{e,bc}(t) = C_{ce} c_e(t) \quad (84)$$

Note that the system matrices $F_{pe}^1, F_{pe}^2, F_{pe}^3$ are state-varying. These state matrices are computed online as follows:

$$F_{pe}^1 = \kappa^{\text{eff}}(c_e) \cdot M_{pe,1} + M_{pe,2} C_{pe}, \quad (85)$$

$$F_{pe}^2 = M_{pe,3}, \quad (86)$$

$$F_{pe}^3 = \kappa_D^{\text{eff}}(c_e) M_{pe,4} \quad (87)$$

where

$$\kappa^{\text{eff}}(c_e) = \kappa(c_e) \cdot (\varepsilon_e^j)^{\text{brug}}, \quad (88)$$

$$\kappa_D^{\text{eff}}(c_e) = \kappa^{\text{eff}}(c_e) \frac{2RT}{F} (t_c^0 - 1) \left(1 + \frac{d \ln f_{c/a}}{d \ln c_e}(x, t) \right). \quad (89)$$

The matrices $M_{pe,1}, M_{pe,2}, M_{pe,3}, M_{pe,4}, C_{pe}$ are computed offline by Matlab function `phi_e_mats.m` as follows. Let

$$\alpha^j = \frac{1}{2L^j \Delta x^j} \quad (90)$$

10 Molar ion fluxes, i.e. Butler-Volmer Current, j_n^-, j_n^+

[DONE] Since the Butler-Volmer equation (6) is algebraic, and we always assume $\alpha_a = \alpha_c = 0.5 = \alpha$, it is trivially implemented as:

$$\frac{d}{dt} j_n^-(t) = \frac{2}{F} i_0^-(t) \sinh \left[\frac{\alpha F}{RT} \eta^-(t) \right] - j_n^-(t), \quad (91)$$

$$\frac{d}{dt} j_n^+(t) = \frac{2}{F} i_0^+(t) \sinh \left[\frac{\alpha F}{RT} \eta^+(t) \right] - j_n^+(t) \quad (92)$$

where

$$i_0^\pm(t) = k^\pm [c_{ss}^\pm(t) c_e(t) (c_{s,\max}^\pm - c_{ss}^\pm(t))]^\alpha, \quad (93)$$

$$\eta^\pm(t) = \phi_s^\pm(t) - \phi_e(t) - U^\pm(c_{ss}^\pm(t)) - F R_f^\pm j_n^\pm(t) \quad (94)$$

for each discrete point in x , in the electrodes only. Note that $\frac{d}{dt} j_n^\pm(t)$ is a dummy variable used to save the corresponding element of vector $g(x, z, t)$.

11 Nomenclature

See Table 2.

Table 2: Symbol Definitions

Symbols in order of appearance

| Electrochemical model states, inputs, outputs | |
|--|--|
| c_s^\pm | Lithium concentration in solid phase [mol/m ³] |
| c_e | Lithium concentration in electrolyte phase [mol/m ³] |
| ϕ_s^\pm | Solid electric potential [V] |
| ϕ_e | Electrolyte electric potential [V] |
| i_e^\pm | Ionic current [A/m ²] |
| j_n^\pm | Molar ion flux [mol/m ² -s] |
| i_0^\pm | Exchange current density [A/m ²] |
| η^\pm | Overpotential [V] |
| c_{ss}^\pm | Lithium concentration at solid particle surface [mol/m ³] |
| θ^\pm | Normalized surface concentration $c_{ss}^\pm/c_{s,\max}^\pm$ [-] |
| I | Applied current [A/m ²] |
| V | Terminal voltage [V] |
| Electrochemical model parameters | |
| D_s^\pm, D_e | Diffusivity of solid, electrolyte phase [m ² /s] |
| t_c^0 | Transference number [-] |
| $\varepsilon_s^\pm, \varepsilon_e$ | Volume fraction of solid, electrolyte phase [-] |
| F | Faraday's constant [C/mol] |
| σ^\pm | Conductivity of solid [1/Ω-m] |
| κ | Conductivity of electrolyte [1/Ω-m] |
| R | Universal gas constant [J/mol-K] |
| T | Temperature [K] |
| $f_{c/a}$ | Mean molar activity coefficient in electrolyte [-] |
| a^\pm | Specific interfacial surface area [m ² /m ³] |
| α_a, α_c | Anodic, cathodic charge transfer coefficient [-] |
| k^\pm | Kinetic reaction rate [(A/m ²)(mol ³ /mol) ^(1+α)] |
| $c_{s,\max}^\pm$ | Maximum concentration of solid material [mol/m ³] |
| U^\pm | Open circuit potential of solid material [V] |
| R_f^\pm | Solid-electrolyte interphase film resistance [Ω-m ²] |
| R_c | Resistance of connectors, current collectors [Ω-m ²] |
| R_s^\pm | Particle radius in solid phase [m] |
| L^j | Length of region $j \in \{-, \text{sep}, +\}$ |
| E_ψ | Activation energy of parameter ψ , [J/mol] |