Modellierung und Simulation des Ladevorgangs bei einer Lithium-Ionen-Batterie

Eric Frommherz und Arthur Messerschmidt
eric.frommherz@gmail.com arthurmesserschmidt@gmail.com

Abschlussbericht der Kooperationsphase 2024/2025 des Kurses KA19

Durchgeführt am Institut für Angewandte und Numerische Mathematik II am Karlsruher Institut für Technologie

Betreut von Prof. Dr. Willy Dörfler

6. Oktober 2025

Abstract

This paper presents a Python-based implementation for numerically simulating the charging and discharging phases of a lithium-ion battery. The simulation is based on the Doyle-Fuller-Newman (DFN) model, which simplifies the battery to a one-dimensional line, incorporating spherical particles to capture concentration flow within the electrodes. The system is governed by a set of coupled partial differential equations, which are discretized and solved using the Finite Difference Method (FDM). FDM plays a central role in the numerical approach and enables the efficient approximation of concentration and potential profiles throughout the simulation.

Inhaltsverzeichnis

1 Problemstellung und Zielsetzung

Die Ziele des Pariser Klimaabkommens [fwZuEB15] führen zu einem Umstieg auf erneuerbare Energien. In Europa erscheinen Photovoltaik und Windkraft als zwei vielversprechende Ansätze [ES+19]. Dabei handelt es sich um fluktuierende, nicht planbare Energiequellen, die bei einem vollständigen Umstieg zu einer inkonsistenten Versorgung führen. Ein Lösungsansatz für diese Problematik sind Batteriespeicher, welche bei Wind- und Lichtmangel das Stromnetz durch gespeicherte Reserven entlasten können [Son23]. Darunter stellen Lithium-basierte Akkumulatoren nach aktuellem Stand die wichtigste Technologie dar [AMWWW09]. In Branchen wie der Elektromobilität und der Mikroelektronik haben Lithium-Ionen-Akkus in den letzten zwei Jahrzehnten aufgrund ihrer hohen Energiedichte, ihres geringen Gewichts und ihrer langen Lebensdauer zunehmend Verwendung gefunden. Durch die Weiterentwicklung in der Materialforschung und der Zellchemie konnten die Kapazität und die Lebenszyklusdauer verbessert werden. Es bestehen weitere ungelöste Fragestellungen bezüglich der Wahl des Elektrodenmaterials und der Optimierung des Ionen- und Elektronentransports. Diese Herausforderungen lassen sich durch Simulationen und Modellierungen gezielt untersuchen.
Beim Ladevorgang einer Lithium-Ionen-Zelle wandern Li^+-Ionen von der Kathode zur Anode. Dort lagern sie sich in Aktivpartikeln als atomares Lithium ein. Beim Entladen wird dieses Lithium unter Energiefreigabe wieder freigesetzt. Der gesamte Vorgang lässt sich durch mathematische Modelle abbilden. Auf mikroskopischer Ebene müssten unzählige Partikel gleichzeitig simuliert werden. Dies führt zu unrealistisch langen Rechenzeiten. Eine direkte numerische Simulation aller Partikel ist für reale Zellgrößen nicht umsetzbar. Eine Lösung bietet die Homogenisierung, bei welcher der Einfluss vieler Partikel gemittelt wird, sodass exemplarisch einzelne Partikel auf makroskopischer Ebene simuliert werden können. Ein Ansatz ist das Doyle-Fuller-Newman(DFN)-Modell. Unter dessen Annahme betrachten wir kugelförmige Aktivpartikel. Das ist mathematisch günstig, da Kugeln eine konstante Krümmung aufweisen und durch die Radialsymmetrie nur den Radius als Freiheitsgrad ergänzen.
Ziel des Projekts ist es, diese sowie andere Prozesse in einer Lithium-Ionen-Batterie in einem angepassten Modell zu simulieren. Die Simulation erfolgt in einem eindimensionalen Makromodell, in das die 2D-Partikel eingebettet werden. Der Ionentransport wird durch partielle Differentialgleichungen beschrieben. Es folgt ein numerisches Modell für die Konzentrationsverteilung und den Ionentransport in Lithium-Ionen-Zellen. Das Projekt leistet einen Beitrag zur Verbesserung elektrochemischer Modellierungen.

2 Begriffserklärungen, Definitionen und Notationen

2.1 Standard-Notationen

Für eine Funktion $u : [0, L] \times [0, T] \to \mathbb{R}$ bezeichnet man die zeitliche partielle Ableitung mit $\partial_t u$ und die räumliche partielle Ableitung mit $\partial_x u$. Der Gradient ist ∇u und der Laplace-Operator $\Delta u = \partial_{xx} u$ im eindimensionalen Fall.
Den natürlichen Logarithmus bezeichnet man mit ln, wobei gilt $\ln : (0, \infty) \to \mathbb{R}$.

2.2 Geometrie und Diskretisierung

Das eindimensionale Batteriegebiet ist das Intervall $\Omega = [0, L]$ mit der Gesamtlänge L. Dieses unterteilt sich in drei Bereiche:

$$\Omega_n = [0, L_n] \quad \text{(Anodenbereich)} \tag{1}$$
$$\Omega_s = (L_n, L_n + L_s) \quad \text{(Separatorbereich)} \tag{2}$$
$$\Omega_p = [L_n + L_s, L] \quad \text{(Kathodenbereich)} \tag{3}$$

Für die Diskretisierung wird das Intervall $[0, L]$ in N gleichmäßige Zellen der Breite $h = L/N$ unterteilt. Die Gitterpunkte sind $x_i = ih$ für $i = 0, 1, \ldots, N$ mit den Zellmitten bei $x_{i+1/2} = (i + 1/2)h$. Die Zeitdiskretisierung erfolgt mit Zeitschritten τ zu den Zeitpunkten $t_n = n\tau$ für $n = 0, 1, 2, \ldots$.

2.3 Physikalische Größen

2.3.1 Konzentrationen

$c_e(x,t)$ Konzentration der Lithium-Ionen im Elektrolyt in $\mathrm{mol/m^3}$

$c_s(r,x,t)$ Konzentration der Lithium-Ionen im Aktivpartikel in $\mathrm{mol/m^3}$

$c_{\mathbf{max}}$ Maximale Lithium-Konzentration im Aktivpartikel in $\mathrm{mol/m^3}$

$c_{\mathbf{ref}}$ Referenzkonzentration für die Entdimensionierung in $\mathrm{mol/m^3}$

2.3.2 Potentiale

$\phi_e(x,t)$ Elektrochemisches Potential im Elektrolyt in V

$\phi_s(x,t)$ Elektrochemisches Potential im Aktivmaterial in V

$U_e(x,t)$ Elektrisches Potential im Elektrolyt in V

$U_{\mathbf{eq}}(c_s)$ Gleichgewichtspotential in V

η Überspannung in V

2.3.3 Transportgrößen

D_e Diffusionskoeffizient im Elektrolyt in $\mathrm{m^2/s}$

D_s Diffusionskoeffizient im Aktivpartikel in $\mathrm{m^2/s}$

$N(x,t)$ Massenfluss der Lithium-Ionen in $\mathrm{mol/(m^2 s)}$

$i(x,t)$ Elektrischer Stromfluss in $\mathrm{A/m^2}$

$i_{\mathbf{BV}}$ Butler-Volmer-Stromdichte in $\mathrm{A/m^2}$

2.3.4 Materialeigenschaften

κ_e Ionische Leitfähigkeit des Elektrolyts in S/m

σ_s Elektronische Leitfähigkeit des Aktivmaterials in S/m

a Spezifische Oberfläche (Oberfläche pro Volumeneinheit) in m^{-1}

R_p Partikelradius in der Kathode in m

R_n Partikelradius in der Anode in m

i_0 Austauschstromdichte in $\mathrm{A/m}^2$

i_{00} Referenzaustauschstromdichte in $\mathrm{A/m}^2$

2.4 Physikalische Konstanten

F Faraday-Konstante: $F = 96485,3321\,\mathrm{C/mol}$

R Universelle Gaskonstante: $R = 8,3145\,\mathrm{J/(mol \cdot K)}$

K Temperatur in Kelvin (konstant bei 293 K)

ξ Dimensionsloser Parameter: $\xi = \frac{F}{RK}$

α Symmetriefaktor in der Butler-Volmer-Gleichung (meist $\alpha = 0.5$)

2.5 Diskrete Notationen

Für die numerische Implementierung verwenden wir folgende Indizierung:

c_i^n Diskrete Konzentration am Ort x_i zur Zeit t_n

$\mathbf{C}^n$ Vektor aller Konzentrationen zur Zeit t_n

A Systemmatrix der diskretisierten Differentialgleichung

h Räumliche Schrittweite

h_s Räumliche Schrittweite im Festkörper

τ Zeitschrittweite

2.6 Operatoren und Funktionen

$\mathbf{sinh}(x)$ Hyperbolischer Sinus: $\sinh(x) = \frac{e^x - e^{-x}}{2}$

$\mathbf{tanh}(x)$ Hyperbolischer Tangens: $\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$

$U_{\mathbf{bs}}^n(z)$ Empirische Gleichgewichtspotentialfunktion für die Anode

$U_{\mathbf{bs}}^p(z)$ Empirische Gleichgewichtspotentialfunktion für die Kathode

z Normierter Ladezustand: $z = c_s/c_{\max}$

3 Batterie-Modelierung

3.1 Aufbau der Batterie

3.1.1 Grundprinzip

Ein Lithium-Ionen-Akkumulator nutzt das Prinzip der Potentialdifferenz zur Energieerzeugung und -speicherung. Die galvanische Zelle besteht aus 3 Komponenten: der Anode (negative Elektrode), der Kathode (positive Elektrode) und dem Elektrolyt als ionischer Leiter.
Entlädt man einen Akkumulator, so fließen Elektronen über einen externen Verbraucher von der Anode zur Kathode. Da die Kathode viel negativer geladen ist, wandern parallel Lithium-Kationen von der Anode zur Kathode, um die Ladungsneutralität innerhalb der Zelle aufrechtzuerhalten. Beim Laden wird dieser Vorgang durch das Anlegen einer externen Spannung umgekehrt.

3.1.2 Die Anode

Die negative Elektrode einer Lithium-Ionen-Zelle wird als Anode bezeichnet, an welcher während des Entladevorgangs die Oxidation stattfindet. Das Standardmaterial ist Graphit (C_6), ein kristallines Kohlenstoffallotrop, welches in Schichten angeordnet ist. Dies ist essenziell für die Funktionsweise, da die Ionen sich darin einlagern können, ohne die Grundstruktur zu zerstören. Bei einer vollständigen Lithiierung entsteht so die Verbindung LiC_6, bei der zwischen jeder sechsten Kohlenstoffschicht ein Lithium-Ion eingelagert ist.
Das elektrochemische Verhalten lässt sich durch eine Reduktion beschreiben:

$$Li^+ + e^- + C_6 \rightarrow LiC_6$$

Die Ionen werden aus dem Elektrolyt aufgenommen und zusammen mit den externen Elektronen in die Graphitstruktur eingelagert. Beim Entladevorgang läuft die Oxidation ab:

$$LiC_6 \rightarrow Li^+ + e^- + C_6$$

Die eingelagerten Lithium-Ionen werden wieder freigesetzt und geben ihre Elektronen an den externen Stromkreis ab.
Die theoretische Kapazität von Graphit beträgt etwa 372 mAh/g, was der vollständigen Lithiierung entspricht. In der Praxis werden jedoch nur 90–95 % der Kapazität erreicht, da nicht alle Interkalationsplätze zugänglich sind.
Es gibt Alternativen zu Graphit. Silizium (Si) hat eine deutlich höhere Kapazität von etwa 4200 mAh/g, aber leidet während des Entladens unter einer Volumenexpansion von ca. 400%, welche zur schnelleren Degradation führt.
Jedoch existieren stabile Verbindungen wie Lithiumtitanat ($Li_4Ti_5O_{12}$), welche eine Kapazität von 175 mAh/g besitzen.

3.1.3 Die Kathode

Die Kathode ist die positive Elektrode, da an ihr während des Entladevorgangs die Reduktion stattfindet. Im Unterschied zum Kohlenstoff bei der Anode basieren Katho-

denmaterialien auf komplexen Lithium-Metalloxiden oder -phosphaten. Die Wahl des Stoffes beeinflusst die Zellspannung, die Energiedichte und die Lebensdauer.
Lithium-Cobalt-Oxid ($LiCoO_2$, LCO) war eines der ersten verfügbaren Kathodenmaterialien. Es besitzt eine Energiedichte von etwa 140–150 mAh/g und eine Nennspannung von 3,7 V. Die Struktur ähnelt der des Graphits, wobei die Lithium-Ionen in den CoO_2-Schichten eingelagert werden. Größter Nachteil ist der teure und hohe Cobaltanteil, dessen Abbau zudem ethische Bedenken mit sich zieht.
Lithium-Eisenphosphat ($LiFePO_4$, LFP) ist ein sicheres und langlebiges Kathodenmaterial. Die Kristallstruktur bietet eine hohe thermische Stabilität sowie Zyklenfestigkeit. Die Kapazität beträgt theoretisch 170 mAh/g. Zusammen mit seiner flachen Entladekurve findet die Technologie weite Verbreitung. Der einzige Nachteil liegt in der vergleichsweise geringeren Kapazität.
Lithium-Nickel-Mangan-Cobalt-Oxid ($LiNiMnCoO_2$, NMC) wird in der Herstellung verwendet. Die verschiedenen Ni:Mn:Co-Verhältnisse (1:1:1, 5:3:2, 8:1:1) ermöglichen angepasste Eigenschaften. Während ein höherer Nickel-Anteil die Kapazität auf bis zu 200 mAh/g hebt, erhöhen Mangan-Anteile die Stabilität. Je nach Verwendungszweck bietet NMC eine gute Balance.
Lithium-Mangan-Oxid ($LiMn_2O_4$, LMO) ermöglicht einen sehr schnellen Lithium-Transport. Die Variante bietet 120 mAh/g Kapazität und zeichnet sich durch hohe Lade- und Entladeraten sowie niedrige Kosten aus.
Die Prozesse an der Kathode lassen sich beim Laden durch

$$Li_2MO_2 \rightarrow LiMO_2 + Li^+ + e^-,$$

sowie Entladen durch

$$LiMO_2 + Li^+ + e^- \rightarrow Li_2MO_2$$

beschreiben, wobei M für das Übergangsmetall steht. Das Material des Metalls ist maßgeblich für die Zelleigenschaft.

3.1.4 Der Elektrolyt

Der Elektrolyt ist als Medium zwischen Anode und Kathode der Grundbaustein für den Ionentransport. Basis sind Carbonat-Gemische wie Ethylencarbonat (EC) mit Dimethylcarbonat (DMC) oder Diethylcarbonat (DEC). Zusätzlich wird oft Lithiumhexafluorophosphat ($LiPF_6$) mit 1,0–1,2 mol/L beigemischt. Dieses löst sich gut in Carbonaten und ermöglicht gleichzeitig eine bessere Ionenleitfähigkeit. Es gibt viele weitere Additive wie Vinylencarbonat oder Fluorethylencarbonat, die in geringen Mengen beigemischt werden, um verschiedenste Ergebnisse zu erzielen. Die Frage nach der optimalen Zusammensetzung bleibt derzeit offen.
Grundvoraussetzung ist eine chemische Stabilität, da der Elektrolyt mit der stark reduzierten Anode als auch der stark oxidierten Kathode im Kontakt stehen muss. Eine niedrige Viskosität für die Zellbenetzung, wie auch geringe Entflammbarkeit aus Sicherheitsgründen, sind von Vorteil.

3.1.5 Der Separator

Der Separator sitzt zwischen der Anode und der Kathode und hat die Aufgabe, die beiden Elektroden elektrisch zu isolieren und einen Ionenfluss zu ermöglichen. Ohne ihn

wäre die Funktionsweise nicht gewährleistet, da die Zelle direkt kurzschließen würde.
Die meisten Separatoren bestehen aktuell aus Polyethylen (PE) und Polypropylen (PP) und sind wenige Mikrometer dick (10–25 μm). In ihnen befinden sich Poren im Nanometer-Bereich, welche den Ionen ermöglichen, zwischen den Elektroden zu diffundieren.
Der Vorteil von PE und PP liegt sicherheitstechnisch in der geringen Schmelztemperatur. Im Falle von Temperaturen über 130 °C/160 °C schmelzen die Separatoren und schließen die Poren, was den weiteren Ionenfluss verhindert. Man bezeichnet sie daher als "Shutdown-Separator".

3.1.6 Aktivpartikel

Das Verständnis des Einlagerungsprozesses der Lithium-Ionen setzt ein tieferes Verständnis der Elektroden zugrunde. Diese bestehen nicht aus massiven Schichten der Materialien, sondern treten in Form von porösen Strukturen, auch Aktivpartikel genannt, auf. Das sind Bereiche in Elektroden, an welchen die elektrochemische Reaktion und Einlagerung stattfindet. In der theoretischen Modellierung des DFN-Modells werden diese Aktivpartikel vereinfacht als perfekte Kugeln betrachtet. Das erleichtert durch ihre konstante Krümmung am Rand später die Berechnung.
In der Praxis zeigen Aktivpartikel eine Vielzahl von Morphologien auf. Da das Oberflächenverhältnis direkt die Butler-Volmer-Gleichung beeinflusst, erhalten wir bei einer Kugel und einem Ellipsoid trotz identischer Volumina eine unterschiedlich verfügbare Stromdichte. Die Partikelform bleibt eine fortwährende Forschungsfrage.

3.2 Theoretische Grundlagen und mathematische Modellierung

Der Übergang vom physikalischen Konstrukt zur mathematischen Modellierung erfordert Vereinfachungen und Idealisierungen. Diese sind häufig für ein praktikables Simulationsmodell notwendig, können allerdings physikalische Effekte vernachlässigen.

Eine fundamentale Vereinfachung der Batteriemodellierung ist die Homogenisierung. Anstatt jedes Aktivpartikel, jede lokale Konzentration oder jeden Fluss explizit zu berücksichtigen, werden die Eigenschaften auf beliebig viele Repräsentanten über das Volumen gemittelt. Dies ermöglicht die Anwendung verschiedenster Methoden, verliert aber Informationen über lokale Inhomogenitäten.

Im Folgenden wird eine Vollzelle während des Ladevorgangs betrachtet. Diese besteht aus der Anode, dem Elektrolytbereich, der Kathode und den Aktivpartikeln.

3.2.1 Kontinuitätsgleichungen

Die mathematische Beschreibung der Lithium-Ionen-Zelle basiert auf den Erhaltungssätzen von Masse sowie Ladung (vgl. [Cas16]). Betrachtet wird ein eindimensionales Modell auf dem Intervall $[0, L]$, wobei L für die Länge des Intervalls steht. $c = c(t, x)$ beschreibt die Konzentration der Lithium-Ionen innerhalb des Elektrolyts bei gegebener Zeit t und Ort x. Der Massenfluss der Ionen ist $N = N(t, x)$, der elektrische

Stromfluss ist $J = J(t, x)$. Das Intervall basiert auf dem Prinzip der Massenerhaltung und besagt, dass sich die Konzentration c nur durch Ein- oder Ausströmen des Konzentrationsflusses $\vec{N}$ lokal verändern kann. Außerdem wird innerhalb der Batterie eine Ladungsneutralität angenommen. Daraus ergeben sich diese beiden Gleichungen:

$$\partial_t c + \partial_x N = 0 \tag{4}$$
$$\partial_x J = 0 \tag{5}$$

Es ist zu beachten, dass man sich im 1-dimensionalen Raum befindet und aufgrund von $\nabla \cdot \vec{N}(t, x) = \frac{\partial N}{\partial x}(t, x) = \partial_x N(t, x)$ die Ableitung anstatt der Divergenz genügt.

3.3 Diskretisierung der Massenerhaltung

Durch die numerische Lösung der Transportgleichung (4) wird die Diffusionsgleichung hergeleitet. Dazu wird das Intervall $[0, L]$ in N gleich große Zellen der Länge $h = \frac{L}{N}$ unterteilt. Der Zellmittelpunkt der i-ten Zelle I_i sei x_i und die Zellränder $x_{i\pm1/2}$. Für die Diskretisierung der Zeit wird $t_n = n\tau$ verwendet, wobei $n \in \mathbb{N}^+$ gilt und τ für die Zeitschrittweite steht. Das nennt man auch **Finite-Differenzen-Verfahren**. Ziel ist es, die mittlere Änderungsrate in Raumzelle i und Zeitschritt n zu berechnen. Dazu wird die Gleichung (4) über Raum und Zeit integriert:

$$\int_{t_{n-1}}^{t_n} \frac{1}{h} \int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} (\partial_t c + \partial_x N) dx\, dt = 0.$$

Der Ausdruck $\frac{1}{h}$ normiert dabei das Integral, wodurch später die mittlere Konzentration pro Zelle berechnet wird.
Nun werden beide Terme getrennt betrachtet:

$$\left[\int_{t_{n-1}}^{t_n} \frac{1}{h} \int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} (\partial_t c) dx\, dt\right] + \left[\int_{t_{n-1}}^{t_n} \frac{1}{h} \int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} (\partial_x N) dx\, dt\right] = 0$$

und mit Hilfe des Hauptsatzes der Differential- und Integralrechnung diskretisiert:
a) Der Zeitterm:

$$\int_{t_{n-1}}^{t_n} \frac{1}{h} \int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} (\partial_t c) dx\, dt = \frac{1}{h} \int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \left[c(t_n, x) - c(t_{n-1}, x)\right] dx$$

$$\approx c(t_n, x_i) - c(t_{n-1}, x_i)$$

Im letzten Schritt wird über die Zelle gemittelt, wodurch der Zellmittelpunkt repräsentativ für die Konzentration innerhalb der Zelle steht. Bei der Multiplikation mit h kürzt sich der Vorfaktor heraus.

b) Der Raumterm:

$$\int_{t_{n-1}}^{t_n} \frac{1}{h} \int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} (\partial_x N) dx\, dt = \frac{1}{h} \int_{t_{n-1}}^{t_n} \left[N(t, x_{i+\frac{1}{2}}) - N(t, x_{i-\frac{1}{2}})\right] dt$$

$$\approx \tau \frac{1}{h} \left(N(t_n, x_{i+\frac{1}{2}}) - N(t_n, x_{i-\frac{1}{2}}) \right)$$

Auch hier wird approximiert, dass sich die Konzentration über die Zeitspanne τ nicht wesentlich ändert und t_n repräsentativ für das Zeitintervall steht.
Bei der Addition beider Terme ergibt sich die diskrete Version der Kontinuitätsgleichung:

$$c(t_n, x_i) - c(t_{n-1}, x_i) + \tau \frac{1}{h} \left(N(x_{i+\frac{1}{2}}, t_n) - N(x_{i-\frac{1}{2}}, t_n) \right) = 0$$

Der Fluss $N(x,t)$ lässt sich diskret in Abhängigkeit der Konzentration $c(t,x)$ ausdrücken, da er laut dem Entropiesatz immer gegen das Konzentrationsgefälle wirkt und auf Ausgleich bedacht ist. Mithilfe des Fickschen Gesetzes [Cas16] wird er umgeschrieben:

$$N(x,t) = -D \frac{\partial c}{\partial x}(x,t). \tag{6}$$

Die Flüsse an den Zellrändern $x_{i\pm 1/2}$ werden benötigt und deshalb durch einen Differenzenquotienten angenähert:

$$N\left(x_{i+\frac{1}{2}}, t_n\right) \approx -D \frac{c(x_{i+1}, t_n) - c(x_i, t_n)}{h}$$

$$N\left(x_{i-\frac{1}{2}}, t_n\right) \approx -D \frac{c(x_i, t_n) - c(x_{i-1}, t_n)}{h}$$

Durch Einsetzen in die obige Kontinuitätsgleichung

$$c(t_n, x_i) - c(t_{n-1}, x_i) + \tau \frac{1}{h} \left[-D \frac{c(x_{i+1}, t_n) - c(x_i, t_n)}{h} - \left(-D \frac{c(x_i, t_n) - c(x_{i-1}, t_n)}{h} \right) \right] = 0$$

und Umformen resultiert die implizite Diffusionsgleichung:

$$c(t_n, x_i) - c(t_{n-1}, x_i) + \frac{\tau D}{h^2} \left[-c(x_{i+1}, t_n) + 2c(x_i, t_n) - c(x_{i-1}, t_n) \right] = 0$$

Im Folgenden wird die Index-Notation verwendet und α als Variable für den Vorfaktor. Es gilt

$$c_i^n := c(x_i, t_n)$$

und

$$\alpha := \frac{\tau D}{h^2}.$$

Daraus ergibt sich:

$$c_i^n - c_i^{n-1} + \alpha \left(-c_{i+1}^n + 2c_i^n - c_{i-1}^n \right) = 0$$

und umgeformt:

$$c_i^n + \alpha(-c_{i-1}^n + 2c_i^n - c_{i+1}^n) = c_i^{n-1}.$$

Nach Umstellung der Gleichung erhält man:

$$-\alpha c_{i-1}^n + (1+2\alpha)c_i^n - \alpha c_{i+1}^n = c_i^{n-1}$$

was in Matrixschreibweise dem hier entspricht:

$$\begin{pmatrix} 1+2\alpha & -\alpha & 0 & \cdots & 0 \\ -\alpha & 1+2\alpha & -\alpha & \cdots & 0 \\ 0 & -\alpha & 1+2\alpha & \cdots & 0 \\ \vdots & \vdots & \ddots & -\alpha & \\ 0 & 0 & 0 & -\alpha & 1+2\alpha \end{pmatrix} \mathbf{c}^{n+1} = \mathbf{c}^n$$

Hierbei handelt es sich um eine Tridiagonalmatrix, welche nur Werte in ihrer Haupt- sowie den beiden Nebendiagonalen aufweist. Diese beschreibt die grundsätzliche Diffusion der Lithium-Ionen.

3.4 Diskretisierung des Stromflusses

3.4.1 Ohmsches Gesetz

Der Stromfluss $\vec{J}$ lässt sich für das Potential am Aktivpartikel durch das Ohmsche Gesetz [Cas16] zu

$$\vec{J} = -\sigma \frac{\partial \phi_s}{\partial x},$$

umformen, wobei σ die Leitfähigkeit und ϕ das Potential darstellt. Für das elektrochemische Potential im Elektrolyt gilt die Umrechnung des elektrischen Potentials U_e in das elektrochemische Potential ϕ_e, wie in Kapitel 3.4.2. Es gilt die leicht abgewandelte Gleichung

$$\vec{J} = -\kappa \frac{\partial U_e}{\partial x}$$

Im Modell werden unterschiedliche Leitfähigkeiten verwendet:
- Im Elektrolyt: κ_e (konstant über alle Bereiche)
- In der Anode: $\sigma_{s,n}$ (nur im Anodenbereich aktiv)
- In der Kathode: $\sigma_{s,p}$ (nur im Kathodenbereich aktiv)

3.4.2 Elektrochemisches Potential

Ziel ist die Herleitung des elektrochemischen Potentials aus dem elektrischen Potential. Dazu werden verschiedene Erkenntnisse kombiniert:

Die Nernst-Planck-Gleichung [ALS09] beschreibt die Ionenbewegung unter Berücksichtigung des elektrischen Feldes und lautet:

$$N_\pm = -D_\pm \, \partial_x c_\pm \; - \; z_\pm u_\pm F \, c_\pm \, \partial_x u_e$$

mit z als Ladungszahl der Ionen. Der ionische Strom wird beschrieben durch

$$i_e = F(z_+ N_+ + z_- N_-).$$

Unter der Annahme eines elektronenneutralen, symmetrischen Elektrolyts setzt man $c_+ = c_- = c_e$, $D_+ = D_- = D$, $u_+ = u_- = u$. Außerdem gilt nach der Einstein-Smoluchowski-Beziehung [Ein05] $D = u\,RK$. Damit wird aus

$$\begin{aligned} i_e &= F\Big((+1)\Big[-D\,\partial_x c_e - uF\,c_e\,\partial_x u_e\Big] + (-1)\Big[-D\,\partial_x c_e + uF\,c_e\,\partial_x u_e\Big]\Big) \\ &= -2F^2 u\,c_e\,\partial_x u_e \;-\; (RK)F\,(1-2t_+)\,\partial_x(\log(c_e)), \end{aligned}$$

wobei sich $t_+ = \frac{1}{2}$ setzen lässt (symmetrischer Elektrolyt). Damit kürzt sich der Subtrahend:

$$i_e = -2F^2 u\,c_e\,\partial_x u_e.$$

Mit $\kappa_e := 2F^2 u\,c_e$, erhält man

$$i_e = -\kappa_e\left(\partial_x u_e + \frac{RK}{2F}\,\partial_x \log(c_e)\right) = -\kappa_e\,\partial_x\left(u_e + \frac{RK}{2F}\log(c_e)\right).$$

Damit lautet die Potentialdefinition

$$\phi_e = \; u_e + \frac{1}{2\xi}\log(c_e), \quad \xi = \frac{F}{RK}.$$

3.4.3 Räumliche Diskretisierung

Für die Diskretisierung werden die aus der Kontinuitätsgleichung (5) hervorgehenden Gleichungen

$$-\sigma_s\,\frac{\partial^2 \phi_s}{\partial x^2} = 0 \quad \text{(Aktivpartikel)}$$

und

$$-\kappa_e\,\frac{\partial^2 U_e}{\partial x^2} = 0 \quad \text{(Elektrolyt)}$$

verwendet. Die Diskretisierung der beiden Gleichungen ist identisch und wird deswegen nur anhand der Gleichung des elektrischen Potentials im Elektrolyt gezeigt. Nun wird die Ausgangsgleichung analog zur Kontinuitätsgleichung der Massenerhaltung (3.3) diskretisiert. Das Intervall ist erneut $[0, L]$ in N. Die Ausgangsgleichung wird über das Raumintervall integriert:

$$-\frac{1}{h}\int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \kappa_e\,\frac{\partial^2 U_e}{\partial x^2}\,\mathrm{d}x = 0$$

Nach Anwendung des Hauptsatzes

$$-\frac{\kappa_e}{h}\left[\frac{\partial U_e}{\partial x}\right]_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} = 0$$

und Approximation durch zentrale Differenzenquotienten

$$\frac{\partial U_e}{\partial x}\left(x_{i+\frac{1}{2}}\right) \approx \frac{U_{e,i+1} - U_{e,i}}{h}$$

$$\frac{\partial U_e}{\partial x}\left(x_{i-\frac{1}{2}}\right) \approx \frac{U_{e,i} - U_{e,i-1}}{h}$$

ergibt sich im Elektrolyt die Gleichung:

$$\frac{\kappa_e}{h^2}(-U_{e,i-1} + 2U_{e,i} - U_{e,i+1}) = 0. \tag{7}$$

In Matrixschreibweise lässt sich dies so zusammenfassen:

$$\frac{\kappa_e}{h^2}\begin{pmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 2 \end{pmatrix}\begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_{N-1} \\ u_N \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix}$$

3.5 Diskretisierung des Aktivpartikels

In diesem Modell werden die Aktivpartikel als radialsymmetrische Kugeln betrachtet. Die Aktivpartikel existieren nur im Anoden- und Kathodenbereich. In diesem Bereich gibt es L_p/h und L_n/h Aktivpartikel. Aus der radialsymmetrischen Form entsteht eine Differentialgleichung für die Diffusion im Körper:

$$\frac{\partial c_s}{\partial x} - D_s\Big(\frac{\partial^2 c_s}{\partial^2 r} + \frac{2}{r}\frac{\partial c_s}{\partial r}\Big) = 0$$

Dies wird umgeformt zu

$$\frac{\partial c_s}{\partial x} - D_s\frac{\partial^2 c_s}{\partial^2 r} = \frac{2}{r}\frac{\partial c_s}{\partial r}.$$

Der rechte Teil der Gleichung wird wie Gleichung (6) diskretisiert. Für $\frac{2}{r}\frac{\partial c_s}{\partial r}$ muss über den Vorwärtsdifferenzenquotienten diskretisiert werden.

$$\frac{\partial c_s}{\partial r} = \frac{C_{l+1} - C_l}{h_s}$$

Für den Radius r wird die Länge vom Mittelpunkt zum halbzahligen Gitterpunkt zwischen l und $l{+}1$ gewählt. Es gilt $r_{l+1/2} = (l{+}1/2)h_s$. Alles in die Differentialgleichung eingesetzt ergibt:

$$\left(1 + \frac{D_s\tau}{h_s^2} + \frac{2h_s}{r_{l+1/2}}\right)C_{s,j,l}^{k+1} - \frac{D_s\tau}{h_s^2}C_{s,j,l-1}^{k+1} - \frac{D_s\tau}{h_s^2}\left(1 + \frac{2h_s}{r_{l+1/2}}\right)C_{s,j,l+1}^{k+1} = C_{s,j,l}^{k}.$$

Zunächst wird diese Gleichung als Gleichungssystem ohne Berücksichtigung der Randbedingungen formuliert. Es entsteht das System

$$\mathbf{A}\cdot\mathbf{C}^{k+1} = \mathbf{C}^k$$

mit den Vektoren

$$\mathbf{C}^{k+1} = \begin{bmatrix} C_{s,j,1}^{k+1} \\ C_{s,j,2}^{k+1} \\ \vdots \\ C_{s,j,N}^{k+1} \end{bmatrix}, \quad \mathbf{C}^{k} = \begin{bmatrix} C_{s,j,1}^{k} \\ C_{s,j,2}^{k} \\ \vdots \\ C_{s,j,N}^{k} \end{bmatrix},$$

und der Matrix

$$\mathbf{A} = \begin{bmatrix} a_1 & c_1 & 0 & \cdots & 0 \\ b_2 & a_2 & c_2 & \ddots & \vdots \\ 0 & b_3 & a_3 & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & c_{N-1} \\ 0 & \cdots & 0 & b_N & a_N \end{bmatrix},$$

wobei gilt

$$\begin{aligned} a_l &= 1 + \frac{D_s \tau}{h_s^2} + \frac{2h_s}{r_{l+1/2}}, \\ b_l &= -\frac{D_s \tau}{h_s^2}, \\ c_l &= -\frac{D_s \tau}{h_s^2} \left(1 + \frac{2h_s}{r_{l+1/2}}\right). \end{aligned}$$

Die Randbedingungen sind Neumann-Bedingungen und können in erster Ordnung diskretisiert werden, etwa durch $(u_2 - u_1)/h = g$ am linken Rand und analog am rechten Rand. Das vollständige System mit Randbedingungen ist im Anhang A.1.2 dargestellt.

3.6 Randbedingungen

3.6.1 Dirichlet-Randbedingungen

Für die Berechnung des Potentials innerhalb des Elektrolyts wird die Festsetzung des Wertes der Funktion am Rand benötigt. Es gilt:

$$u = g \quad \text{auf } \partial\Omega$$

wobei g eine vorgegebene Funktion am Rand ist. Die Festsetzung der Ladung am Rand ist für die Erdung des elektrochemischen Potentials wichtig.

3.6.2 Neumannsche-Randbedingungen

Für die Zellränder wird ein Neumannsches Randwertproblem für die Poisson-Gleichung betrachtet. Anders als bei der Dirichletrandbedingung wird hier nicht die Konzentration am Rand, sondern der Konzentrationsfluss durch den Rand vorgegeben. Es gilt:

$$\begin{aligned} -\Delta u &= f \quad \text{in } \Omega, \\ \frac{\partial u}{\partial n} &= g \quad \text{auf } \partial\Omega \end{aligned}$$

wobei Δu den Laplace-Operator und $\frac{\partial u}{\partial n}$ die Ableitung am Rand ausdrückt.
Es wird deutlich, dass, wenn u eine Lösung hat, auch $u+c$ eine Lösung für jede beliebige Konstante ($c \in \mathbb{R}$) bieten muss, da

$$\Delta(u + c) = \Delta u + \Delta c = \Delta u + 0 = \Delta u. \tag{8}$$

Die Greensche Identität lautet:

$$\int_\Omega \Delta u \, dx = \int_{\partial\Omega} \frac{\partial u}{\partial n} \, ds.$$

Setzt man $\Delta u = -f$ und $\frac{\partial u}{\partial n} = g$ ein, so ergibt sich:

$$-\int_\Omega f \, dx = \int_{\partial\Omega} g \, ds.$$

Umgestellt ergibt sich die sogenannte Kompatibilitätsbedingung:

$$\int_\Omega f \, dx + \int_{\partial\Omega} g \, ds = 0. \tag{9}$$

Diese Bedingung ist notwendig, damit eine Lösung u existieren kann. Sie stellt sicher, dass die Gesamtbilanz der Quellen und Senken im Gebiet mit dem Fluss über den Rand übereinstimmt. Alles, was im Inneren erzeugt oder vernichtet wird, muss über den Rand hinausgelangen oder eintreten. Da der Laplace-Operator mit Neumann-Randbedingungen einen nichttrivialen Kern besitzt, ist die Lösung nur bis auf eine additive Konstante bestimmt (siehe 8). Eindeutigkeit erhält man durch eine Zusatzbedingung, z. B. $u = 0$ in einem Punkt oder durch die Forderung, dass der Mittelwert von u Null ist (16).

3.6.3 Diskretisierung im eindimensionalen Fall

Für die numerische Lösung des Neumann-Randwertproblems wird im eindimensionalen Fall wieder Bezug auf das Gitter $x_0, x_1, \ldots, x_N$ mit der Schrittweite h genommen. Da der Rand selbst nicht eindeutig ableitbar ist, nutzt man zeitweise einen virtuellen Punkt außerhalb des Systems, um die Ableitung am Rand durch Differenzenquotienten zu approximieren. Dies nennt man Geisterpunkt-Verfahren. Im Folgenden wird der Randpunkt als x_B, der nächst innere Gitterpunkt als x_I und mit x_G ein „Geisterpunkt“ außerhalb des Intervalls im Abstand $|x_B - x_I|$ bezeichnet.

Das Ziel ist die Herleitung einer Formel für $u''(x)$. Dazu wird die Idee des Differenzenquotienten zweifach iteriert. Zuerst die Approximation der ersten Ableitung:

$$u'(x) \approx \frac{u(x+h) - u(x)}{h} \quad \text{(Vorwärtsdifferenz)}$$

$$u'(x) \approx \frac{u(x) - u(x-h)}{h} \quad \text{(Rückwärtsdifferenz)}$$

Mithilfe der Differenz der Vorwärts- und Rückwärtsdifferenz lässt sich die Krümmung $u''(x)$ bestimmen:

$$u''(x) \approx \frac{\frac{u(x+h)-u(x)}{h} - \frac{u(x)-u(x-h)}{h}}{h}$$

$$= \frac{1}{h}\left(\frac{u(x+h) - u(x)}{h} - \frac{u(x) - u(x-h)}{h}\right)$$

$$= \frac{1}{h^2}\left(u(x+h) - 2u(x) + u(x-h)\right)$$

$$= \frac{u(x-h) - 2u(x) + u(x+h)}{h^2} = u''(x)$$

Am Randpunkt ist $u(x-h)$ jedoch unbekannt, da dieser außerhalb des Intervalls liegt. Daher wird der Geisterpunkt x_G eingeführt und $u(x-h) \approx u_h(x)$ approximiert:

$$u''(x_B) \approx \frac{u_h(x_G) - 2u_h(x_B) + u_h(x_I)}{h^2}.$$

Durch Umstellen ergibt sich dann:

$$-u_h(x_I) + 2u_h(x_B) - u_h(x_G) = h^2 f(x_B)$$

wobei $f(x)$ die zweite Ableitung von u ist.

Die Ableitung am Rand wird weiterhin als Neumann-Bedingung ausgedrückt:

$$\frac{\partial u}{\partial n}(x_B) = g(x_B)$$

Im eindimensionalen Fall gilt am linken Rand $x = 0$:

$$-u'(0) = g(0) \quad \Rightarrow \quad u'(0) = -g(0)$$

Die erste Ableitung $u'(x_B)$ lässt sich erneut durch einen Differenzenquotienten mit dem Geisterpunkt ausdrücken:

$$u'(x_B) \approx \frac{u_h(x_I) - u_h(x_G)}{2h}$$

Eingesetzt in die Randbedingung:

$$\frac{u_h(x_I) - u_h(x_G)}{2h} = -g(x_B)$$

und umgeformt

$$-u_h(x_I) + u_h(x_G) = 2hg(x_B)$$

ergibt sich der zweite Term mit dem Geisterpunkt. Dadurch wird die Randbedingung in zweiter Ordnung diskretisiert (Fehler $O(h^2)$). Ein einfacher einseitiger Differenzenquotient würde nur erste Ordnung erreichen (Fehler $O(h)$).
$u_h(x_G)$ wird durch das Addieren beider Gleichungen eliminiert:

$$\begin{aligned}(-u_h(x_I) + 2u_h(x_B) - u_h(x_G)) + (-u_h(x_I) + u_h(x_G)) &= h^2 f(x_B) + 2hg(x_B) \\ = \quad -2u_h(x_I) + 2u_h(x_B) &= h^2 f(x_B) + 2hg(x_B).\end{aligned}$$

Auf diese Weise sind die Randbedingungen integrierbar, ohne den Punkt außerhalb des Intervalls explizit berücksichtigen zu müssen.

3.7 Butler-Volmer-Gleichung

Ausschlaggebend für den Stromfluss innerhalb des Lithium-Ionen-Akkumulators sind elektrochemische Prozesse an den Grenzflächen zwischen Elektrolyt und Aktivpartikel. Parallel zum Elektronentransport im Festkörper findet eine Einlagerung der Lithium-Ionen durch das Elektrolyt in das Aktivmaterial statt. Die zentrale Größe dieses Prozesses wird durch die Butler-Volmer-Gleichung [ZGL+24] beschrieben. Diese lässt sich aus fundamentalen physikalisch-chemischen Prinzipien herleiten.

3.7.1 Gleichgewichtspotential und Überspannung

Ein zentraler Parameter der Butler-Volmer-Gleichung ist das Gleichgewichtspotential $U_{\text{eq}}(c_s)$. Dieses beschreibt die Triebkraft der elektrochemischen Reaktion am Aktivpartikel und wird durch dessen Konzentration bestimmt. Es verbindet die elektrische Spannung mit der chemischen Konzentration.
Aus der Nernst-Gleichung [FM94] folgt eine einfache Redoxreaktion:

$$U_{\text{eq}} = U^0 + \frac{RK}{F} \ln\left(\frac{a_{\text{ox}}}{a_{\text{red}}}\right)$$

mit
- U^0 als Standardelektrodenpotential im spannungslosen Zustand und
- $a_{\text{ox}}/a_{\text{red}}$ (oft als Reaktionsquotient Q_r bezeichnet) als Stoffkonzentration des Redox-Partners.

In der Realität befinden sich die Li-Atome im Gitter jedoch nicht im idealen Zustand, sondern zeigen
- Gitterverzerrungen durch unterschiedliche Ionenradien,
- Unordnung in der Verteilung aufgrund von Li-Li-Wechselwirkungen sowie
- strukturelle Phasenübergänge bei bestimmten Konzentrationen.

Aus diesen Gründen folgt das Potential nicht der Nernst-Beziehung, sondern stellt eine komplexe materialspezifische Funktion für c_s dar.

In der Praxis wird das Gleichgewichtspotential aus experimentellen Messungen bestimmt. Dabei lädt und entlädt man eine Zelle bei so kleinen Strömen, dass annähernd von Gleichgewichtsbedingungen ausgegangen werden kann. Das gemessene Potential liefert direkt $U_{\text{eq}}(c_s)$
Für unser Modell verwenden wir zwei empirisch angepasste Formeln für Anode und Kathode nach Allaire [RJ25]:

Anode (Graphit):

$$\begin{aligned} U_{bs}^n(z) = {} & 0.6379 + 0.5416\exp(-100z) + 0.044\tanh\left(-\frac{z-0.18}{0.05}\right) \\ & + 0.2\tanh\left(-\frac{z-1.035}{0.055}\right) + 0.6875\tanh\left(-\frac{z+0.02}{0.038}\right) \\ & + 0.0175\tanh\left(-\frac{z-0.511}{0.02}\right) \end{aligned}$$

Kathode:

$$\begin{aligned} U_{bs}^{p}(z) = {} & 1091.60809 + 0.5416\exp(-100z) \\ & + 0.721524957\tanh\left(-\frac{z-0.984753725}{0.0422847481}\right) \\ & + 1089.28504\tanh\left(-\frac{z+5.39175183}{1.49617161}\right) \end{aligned}$$

wobei $z = c_s/c_{\max}$ der normierte Ladezustand ist.

3.7.2 Arrhenius-Gleichung

Die Butler-Volmer-Gleichung beschreibt die Stromdichte i durch eine Reaktion an der Grenzfläche zwischen Elektrolyt und Aktivmaterial. Ionen gehen vom Elektrolyt in den Festkörper über oder umgekehrt, begleitet von einem entgegengesetzten Elektronentransfer:

$$Li^+ + e^- \leftrightharpoons Li_{(s)}$$

Es herrscht die kathodische Reduktion während des Ladevorgangs ($Li^+ + e^- \Rightarrow Li_{(s)}$) sowie die anodische Oxidation während des Entladevorgangs ($Li_{(s)} \Rightarrow Li^+ + e^-$).
Die Arrhenius-Gleichung [Arr89] beschreibt analog zur Maxwell-Boltzmann-Verteilung die Reaktionsgeschwindigkeit proportional zur Exponentialfunktion der Aktivierungsenergie $\Delta G^{\ddagger}$. Für beide Teilreaktionen ergeben sich die Ströme:

$$i_{\text{kath}} = i_0 \cdot \exp\left(-\frac{\alpha F\eta}{RK}\right) \qquad \text{Kathodisch (Einlagerung)}$$

$$i_{\text{anod}} = i_0 \cdot \exp\left(+\frac{(1-\alpha)F\eta}{RK}\right) \qquad \text{Anodisch (Freisetzung)}$$

Der Nettostrom ist nun die Differenz beider Teilströme:

$$i = i_{\text{anod}} - i_{\text{kath}} = i_0\left[\exp\left(\frac{(1-\alpha)F\eta}{RK}\right) - \exp\left(-\frac{\alpha F\eta}{RK}\right)\right]$$

Wenn man den Symmetriekoeffizienten $\alpha = 0,5$ setzt, was eine häufig physikalisch begründete Annahme für einfache Elektronentransferreaktionen ist [GCF+14], vereinfachen sich beide Exponenten zu $\pm\frac{F\eta}{2RK}$:

$$i = 2i_0\left[\exp\left(\frac{F\eta}{2RK}\right) - \exp\left(-\frac{F\eta}{2RK}\right)\right].$$

Mit der hyperbolischen Sinusfunktion $\sinh(x)$ erhalten wir:

$$i = 2i_0\sinh\left(\frac{F\eta}{2RK}\right)$$

und mit der dimensionslosen Konstante $\xi := \frac{F}{RK}$:

$$i = 2i_0\sinh\left(\frac{\xi}{2}\eta\right).$$

Die Überspannung η ist definiert als die Abweichung des Elektrodenpotentials vom Gleichgewichtspotential. Für eine Elektrode gilt:

$$\eta = \phi_s - \phi_e - U_{\text{eq}}(\tilde{c}_s)$$

wobei ϕ_s das Potential des Aktivmaterials (Festkörper), ϕ_e das Potential des Elektrolyts und U_{eq} das Gleichgewichtspotential der Reaktion (siehe 3.7.1) darstellt. In unserem Modell verwenden wir als Überspannung unsere empirische Funktion $\text{Ubs}(c_s)$:

$$\eta = \phi_s - \phi_e - \text{Ubs}(c_s)$$

Durch Einsetzen in unsere Butler-Volmer-Gleichung ergibt sich:

$$i_{\text{BV}} = 2i_0 \cdot \sinh\left(\frac{\xi}{2}\left(\phi_s - \phi_e - \text{Ubs}(c_s)\right)\right)$$

Die Austauschstromdichte i_0 hängt von den lokalen Konzentrationen der Reaktionspartner ab. Aus der statistischen Mechanik und der Kollisionstheorie folgt, dass sie proportional zur Wurzel des Produkts aller beteiligten Stoffe ist:

c_e**:** Li^+-Konzentration im Elektrolyt (Reaktand)

c_s**:** Li-Konzentration im Festkörper (Produkt)

$c_{max} - c_s$**:** verfügbare Plätze im Festkörper für weitere Interkalation

$$i_0 = i_{00}\sqrt{c_e\, c_s\, (c_{\text{max}} - c_s)}$$

Diese Formulierung stellt sicher, dass:

bei $c_s \to 0$: Die Reaktion durch fehlende Li-Atome im Festkörper gehemmt wird,

bei $c_s \to c_{\text{max}}$: Die Reaktion stoppt, da keine Plätze mehr verfügbar sind und

bei $c_e \to 0$: Die Reaktion durch Li^+-Mangel im Elektrolyt limitiert wird.

Der Vorfaktor i_{00} ist nach Arrhenius eine materialspezifische Konstante, die den Stoßfaktor angibt. Nach Gilbert Newton Lewis ist jedoch dieser Faktor temperaturabhängig. Er ist proportional zur Wurzel der Temperatur:

$$i_{00} = u\sqrt{K}$$

Im Fall dieses Modells wird mit konstanter Temperatur gerechnet.

3.8 Entdimensionierung

Die Entdimensionierung ist ein fundamentales Werkzeug in numerischen Simulationen. Dabei verfolgt das Verfahren gleich mehrere Ziele, die für unsere Modellierung essenziell sind:

Numerische Stabilität:
Innerhalb eines Batteriesystems bewegen sich die Konstanten in ganz unterschiedlichen Größenordnungen. Konzentrationen liegen im Bereich von 10^3 bis 10^5 mol/m^3, Längen im Mikrometerbereich (10^{-6} m), und Zeiten von Sekunden bis Stunden. Diese extremen Unterschiede können zu numerischen Instabilitäten und vermeidbaren Rundungsfehlern führen, da Computer mit endlicher Präzision arbeiten.

Einfachere Gleichungssysteme:
Durch geschickte Wahl der Größen können die Koeffizienten in den Differentialgleichungen so skaliert werden, dass sie alle möglichst in der Größenordnung von 1 liegen. Das verbessert die resultierenden linearen Gleichungssysteme erheblich.

Physikalische Einsicht:
Entdimensionierte Werte geben Aufschluss über die Stärke verschiedener Effekte. In unserem Fall zeigen die dimensionslosen Parameter das Verhältnis von Diffusion zu Reaktion.

Universelle Gültigkeit:
Entdimensionierte Ergebnisse sind unabhängig von spezifischen Einheiten und können einfacher mit anderen Systemen verglichen und auf diese übertragen werden.

Die Entdimensionierung erfolgt mit folgenden charakteristischen Größen, die nach Allaire et al. [RJ25] gewählt wurden:

$$
\begin{aligned}
\mathrm{Far}_K &= 9.64853321 \cdot 10^4 \, \frac{C}{mol} && \text{(Faraday-Konstante)} \\
\mathrm{Gas}_K &= 8.3145 \, \frac{J}{mol \cdot K} && \text{(Gaskonstante)} \\
L_{\mathrm{ref}} &= 1 \cdot 10^{-4} \, \mathrm{m} && \text{(Charakteristische Zelldicke)} \\
C_{\mathrm{ref}} &= 5 \cdot 10^4 \, \mathrm{mol/m^3} && \text{(Referenzkonzentration)} \\
T_{\mathrm{ref}} &= 1000 \, \mathrm{s} && \text{(Referenzzeitskala)} \\
K_{\mathrm{ref}} &= 293 \, \mathrm{K} && \text{(Raumtemperatur)} \\
U_{\mathrm{ref}} &= \frac{Far_K}{Gas_K \cdot K_{\mathrm{ref}}} && \text{(Thermisches Potential)} \\
J_{\mathrm{ref}} &= 1 \, \mathrm{A/m^2} && \text{(Referenzstromdichte)}
\end{aligned}
$$

Nun werden alle Werte mithilfe der entsprechenden Referenzgröße definiert:

Variablen:

$$\hat{c}_e = \frac{c_e}{C_{\text{ref}}}, \quad \hat{c}_s = \frac{c_s}{C_{\text{ref}}} \qquad \text{(Konzentrationen)}$$

$$\hat{\phi}_e = \frac{\phi_e}{U_{\text{ref}}}, \quad \hat{\phi}_s = \frac{\phi_s}{U_{\text{ref}}} \qquad \text{(Potentiale)}$$

$$\hat{x} = \frac{x}{L_{\text{ref}}}, \quad \hat{t} = \frac{t}{T_{\text{ref}}} \qquad \text{(Raum und Zeit)}$$

$$\hat{i}_{\text{BV}} = \frac{i_{\text{BV}}}{J_{\text{ref}}} \qquad \text{(Stromdichte)}$$

Parameter:
Die wichtigsten Gruppen charakterisieren die relative Stärke verschiedenster Prozesse.

Diffusionskoeffizienten:

$$\hat{D}_{e,n} = 10^{-11}\frac{T_{\text{ref}}}{L_{\text{ref}}^2} \qquad \text{(Elektrolytdiffusion in Anode)}$$

$$\hat{D}_{e,p} = 10^{-11}\frac{T_{\text{ref}}}{L_{\text{ref}}^2} \qquad \text{(Elektrolytdiffusion in Kathode)}$$

$$\hat{D}_{e,s} = 10^{-11}\frac{T_{\text{ref}}}{L_{\text{ref}}^2} \qquad \text{(Elektrolytdiffusion im Separator)}$$

$$\hat{D}_{s,n} = 3.4 \cdot 10^{-14}\frac{T_{\text{ref}}}{L_{\text{ref}}^2} \qquad \text{(Festkörperdiffusion in Anode)}$$

$$\hat{D}_{s,p} = 1.6 \cdot 10^{-14}\frac{T_{\text{ref}}}{L_{\text{ref}}^2} \qquad \text{(Festkörperdiffusion in Kathode)}$$

Ionische Leitfähigkeiten:

$$\hat{\kappa}_{e,n} = 0.12\frac{U_{\text{ref}}}{L_{\text{ref}} J_{\text{ref}}} \qquad \text{(Elektrolytleitfähigkeit in Anode)}$$

$$\hat{\kappa}_{e,p} = 0.12\frac{U_{\text{ref}}}{L_{\text{ref}} J_{\text{ref}}} \qquad \text{(Elektrolytleitfähigkeit in Kathode)}$$

$$\hat{\kappa}_{e,s} = 0.12\frac{U_{\text{ref}}}{L_{\text{ref}} J_{\text{ref}}} \qquad \text{(Elektrolytleitfähigkeit im Separator)}$$

Elektronische Leitfähigkeiten:

$$\hat{\sigma}_{s,n} = 10^4\frac{U_{\text{ref}}}{L_{\text{ref}} J_{\text{ref}}} \qquad \text{(elektrische Festkörperleitfähigkeit in Anode)}$$

$$\hat{\sigma}_{s,p} = 10\frac{U_{\text{ref}}}{L_{\text{ref}} J_{\text{ref}}} \qquad \text{(elektrische Festkörperleitfähigkeit in Kathode)}$$

$$\hat{\sigma}_{s,s} = 0 \qquad \text{(Separator - kein Elektronentransport)}$$

Faraday-Parameter:

$$\hat{F} = \frac{Far_K\, L_{\text{ref}}\, C_{\text{ref}}}{T_{\text{ref}}\, J_{\text{ref}}}$$

Entdimensionierte Modellparameter:
Die physikalischen Parameter werden entsprechend der Referenzgrößen normiert:

Geometrische Parameter:

$$\hat{L} = \frac{L}{L_{\text{ref}}} = 1 \qquad \text{(Gesamtzellendicke)}$$
$$\hat{L}_n = \frac{L_n}{L_{\text{ref}}} = 0.45 \qquad \text{(Anodendicke)}$$
$$\hat{L}_p = \frac{L_p}{L_{\text{ref}}} = 0.45 \qquad \text{(Kathodendicke)}$$
$$\hat{L}_s = \frac{L_s}{L_{\text{ref}}} = 0.1 \qquad \text{(Separatordicke)}$$

Partikelgeometrie:

$$\hat{R}_n = \frac{R_n}{L_{\text{ref}}} = 0.075 \qquad \text{(Anodenpartikelradius)}$$
$$\hat{R}_p = \frac{R_p}{L_{\text{ref}}} = 0.035 \qquad \text{(Kathodenpartikelradius)}$$

Spezifische Oberflächen:

$$\hat{a}_n = a_n \, L_{\text{ref}} = 60 \qquad \text{(Anodenoberfläche)}$$
$$\hat{a}_p = a_p \, L_{\text{ref}} = 28 \qquad \text{(Kathodenoberfläche)}$$

Für kugelförmige Partikel gilt theoretisch $a = 3/R$, wodurch sich ergibt:

$$\hat{a} = a \, L_{\text{ref}} = \frac{3L_{\text{ref}}}{R} = \frac{3}{\hat{R}}$$

4 Numerische Umsetzung

Innerhalb des eindimensionalen DFN-Modells [RJ25] wird die Zelle in x-Richtung für das dimensionslose Gebiet $\Omega = (0, 1)$ beschrieben:

$$\Omega_n = (0, L_n) \qquad \text{(Anode)}$$
$$\Omega_s = (L_n, L_p) \qquad \text{(Separator)}$$
$$\Omega_p = (L_p, 1) \qquad \text{(Kathode)}$$

Die physikalischen Eigenschaften dazu wurden bereits in Kapitel 3.1 erläutert. Die Aktivpartikel werden als Kugeln mit Radius R_n (Anode) bzw. R_p (Kathode) modelliert. Die Größen C_e, ϕ_e, ϕ_s sind auf Ω definiert. Die Konzentration im Aktivpartikel C_s ist für Ω_p und Ω_n definiert. Nach der Startwertermittlung für ϕ_s und ϕ_e werden die Größen der Reihe nach für jeden Zeitschritt ermittelt und für das Interface der nachfolgenden Größen verwendet. Zuerst werden die Werte für C_e ermittelt. Anschließend werden die Potentiale im Elektrolyt und am Aktivpartikel ermittelt. Am Ende folgt die Berechnung von C_s.

4.1 Verknüpfung der physikalischen Größen

Im DFN-Modell werden die einzelnen Größen c_e, c_s, ϕ_s und ϕ_e über die Butler-Volmer-Funktion in der Anode und Kathode verknüpft. Im Separatorbereich existieren keine Aktivpartikel. Es gibt keine Verknüpfung über die Butler-Volmer-Funktion. Durch das ohmsche Gesetz [Cas16] gilt

$$i_s = -\sigma \frac{\partial \phi_s}{\partial x}.$$

Es gilt

$$\frac{\partial i_s}{\partial x} = -\sigma \frac{\partial^2 \phi_s}{\partial^2 x}.$$

Im Allgemeinen gilt für die Verbindung der beiden Stromflüsse i_s und i_e [New75]

$$\partial_x i_s + \partial_x i_e = 0$$

Für i gilt die Butler-Volmer-Gleichung. $\partial_x i$ kann durch Multiplikation mit der spezifischen Oberfläche a erreicht werden [New75]. Die Ableitung des Stromflusses wird für ϕ_s negativ definiert, ausgehend vom Zusammenhang aus der vorherigen Gleichung, um die Ableitung des Stromflusses für ϕ_e positiv zu definieren. Es gilt also:

$$\frac{\partial i_s}{\partial x} = -a \cdot i_{bv}.$$

Das Potential im Elektrolyt ϕ_e wird über die Hilfsgröße U_e berechnet. Für U_e gilt, aufgrund des ohmschen Gesetzes, ähnlich wie für das Potential am Aktivpartikel,

$$\frac{\partial i_e}{\partial x} = -\kappa \frac{\partial^2 U_e}{\partial^2 x}.$$

Für die Erfüllung der allgemeinen Gleichung der beiden Stromflüsse gilt:

$$\frac{\partial i_e}{\partial x} = a \cdot i_{bv}.$$

Die Lithium-Ionenübertragung wird über die Butler-Volmer-Funktion definiert. Es gilt weiterhin die Kontinuitätsgleichung.

$$\frac{\partial c_e}{\partial x} + \frac{\partial N}{\partial x} = 0$$

Der Massenstrom setzt sich aus dem Massenstrom der Diffusion und dem Massenstrom in den Aktivpartikeln zusammen.

$$N_{\text{gesamt}} = N_{\text{Diffusion}} - N_e$$

Der Massenstrom N_e wird mit

$$N_e = \frac{i_{bv}}{zF}$$

dargestellt [NB21]. i_{bv} stellt den Stromfluss am Aktivpartikel dar. z ist die Ladungszahl, die bei einem Lithium-Ion 1 entspricht. Durch Umstellen der Kontinuitätsgleichung entsteht

$$\frac{\partial c_e}{\partial x} + \frac{\partial N_{\text{Diffusion}}}{\partial x} = \frac{\partial N_e}{\partial x}.$$

Auflösen von N_e ergibt

$$\frac{\partial N_e}{\partial x} = \frac{\partial}{\partial x}\left(\frac{i_{bv}}{F}\right) = \frac{1}{F}\frac{\partial i_{bv}}{\partial x} = \frac{a}{F} i_{bv}.$$

Eingesetzt in die Kontinuitätsgleichung folgt

$$\frac{\partial c_e}{\partial x} - D\frac{\partial^2 c_e}{\partial^2 x} = \frac{a}{F} i_{bv}.$$

Für die Konzentration im Aktivpartikel wird die Verknüpfung über eine Flussrandbedingung gewählt. Da der Massenstrom, der das Elektrolyt verlässt, in den Aktivpartikel übergeht, muss

$$\frac{\partial c_s}{\partial x} = -\frac{1}{F} i_{bv} \text{ für } r = R$$

gelten. Für das Interface ist die Konzentration der Aktivpartikel am Rand notwendig $r = R$. Da die Batterie in drei Bereiche eingeteilt wird, gilt für das Interface F in den jeweiligen Bereichen:

$$\begin{aligned} \Omega_s \text{ (Separator)}: &\quad a = 0 \quad \text{(keine elektrochemische Reaktion)} \\ \Omega_n \text{ (Anode)}: &\quad a = a_{sn} \\ \Omega_p \text{ (Kathode)}: &\quad a = a_{sp} \end{aligned}$$

4.2 Randbedingungen des DFN-Modells

Für die Randbedingungen der Konzentration im Elektrolyt c_e gilt:

$$\frac{\partial c_e}{\partial x} = g = 0 \quad \text{auf } \partial\Omega$$

Dies gilt, da das System in sich abgeschlossen ist und keinen Verlust an Lithium-Ionen hat. Die Lithium-Ionen bewegen sich zwischen den einzelnen Teilgebieten.
Für die Konzentration im Aktivpartikel c_s gilt am inneren Rand

$$\frac{\partial c_s}{\partial r} = g = 0 \quad \text{für } r = 0.$$

Am äußeren Rand, also der Kontaktfläche zum Elektrolyt, gilt

$$\frac{\partial c_s}{\partial r} = g = -\frac{1}{F} i_{bv} \quad \text{für } r = R.$$

Dies geht daraus hervor, dass die Lithium-Ionen, welche aus dem Elektrolyt fließen, in die Aktivpartikel fließen müssen.
Für die Randbedingung des Potentials am Aktivpartikel gilt

$$\frac{\partial \phi_s}{\partial x} = g = -J \quad \text{für } x = L. \tag{10}$$

An den Rändern des Separators gilt für das Potential am Aktivpartikel

$$\frac{\partial \phi_s}{\partial r} = g = 0 \quad \text{auf } \partial\Omega_s.$$

Außerdem kann der Potentialrandwert für das DFN-Modell geerdet werden:

$$\phi_s = g = 0 \quad \text{für } x = 0.$$

Für das Potential im Elektrolyt ϕ_e gilt:

$$\frac{\partial \phi_e}{\partial x} = g = 0 \quad \text{auf } \partial\Omega$$

4.3 Potentiale ermitteln

Sowohl ϕ_e als auch ϕ_s auf Ω_p lassen sich aufgrund ihrer Neumann-Randbedingung an beiden Seiten ohne weitere Nebenbedingungen nur bis auf eine Konstante bestimmen. Für die Nebenbedingungen wird die Ladungserhaltung zu Hilfe genommen. Für Ω_g muss

$$\int_\Omega a \cdot i_{bv}\, dx = 0 \tag{11}$$

gelten, da innerhalb der Batterie keine Ladung erzeugt werden darf. Zusätzlich muss auch noch der Stromfluss in den einzelnen Gebieten Ω_p und Ω_n richtig abgebildet werden. Hierfür ist wichtig, dass die Randbedingung (10) erfüllt wird. Deswegen muss

$$\int_{\Omega_p} a \cdot i_{bv}\, dx = J \tag{12}$$

erfüllt werden. Diese beiden Gleichungen, die beide von ϕ_e und ϕ_s abhängig sind, stellen die Nebenbedingungen zur Bestimmung der Konstanten in jedem Zeitschritt dar.

4.4 Aufbau der Matrix mit Randbedingungen und Interface

Die Matrizen für die jeweiligen Differenzialgleichungen wurden in vorherigen Kapiteln erstellt. F steht für das Interface, welches aus der Butler-Volmer-Gleichung hervorgeht. f_n sind die Werte des Vektors F. G ist Teil der Randbedingung. Für C_e ergibt sich dann das System

$$A_{ce} C_e^{k+1} = C_e^k + \tau F_{ce} + \tau G_{ce}.$$

Für die Konzentrationen im Aktivpartikel ist folgendes System in jedem relevanten Gebiet Ω_p und Ω_n zu berechnen.

$$A_{cs} C_{s,j}^{k+1} = C_{s,j}^k + \tau G_{cs,j}$$

Die einzelnen Vektoren und der Aufbau befinden sich im Anhang. Die Multiplikation des F-Terms mit τ ist deswegen notwendig, da dieser bei der Diskretisierung der zeitlichen Ableitung entsteht. Das Gleichungssystem für die Konzentration im Aktivpartikel benötigt keine Interface-Bedingung. Die Verknüpfung entsteht durch die Randbedingung.

$$A_{cs}C_s^{k+1} = C_s^k + \tau G_{cs}.$$

Für ϕ_s in den relevanten Gebieten gilt

$$A_{ps,n}\phi_{s,n} = F_{\phi_{s,n}} + G_{\phi_{s,n}} \tag{13}$$

und

$$A_{ps,p}\phi_{s,p} = F_{\phi_{s,p}} + G_{\phi_{s,p}}$$

wobei $\partial_x\phi_s = 0$ auf $\partial\Omega_s$ gilt, da dort keine Aktivpartikel existieren. Diese Gleichung ist stationär. Sie hängt nicht direkt von zeitlichen Faktoren ab. Für ϕ_e gilt

$$A_{ue}U_e = F_{ue} + G_{ue}$$

Diese Gleichung stellt auch eine stationäre Problemstellung dar. Diese müssen für den Zeitpunkt $t = 0s$ gelöst werden.

5 Das Programm

Der nachfolgende Abschnitt bezieht sich auf die praktische Anwendung unseres Modells. Das resultierende Programm folgt einer klaren Struktur und separiert jede Berechnung in einzelne Funktionen. Die wichtigsten Parameter werden durch eine externe Datenbank geladen, um möglichst schnell zwischen Wertetabellen wechseln zu können. Innerhalb der Wertetabelle werden bereits die realen Werte zu den entdimensionierten Werten umgewandelt, entsprechend Kapitel 3.8. Das eigentliche Programm lässt sich in der Zeitschrittweite und der Anzahl der zu berechnenden Ortspunkte variieren. Die Vergrößerung dieser Variablen führt zum Anstieg der Laufzeit und gleichzeitig zu einem kleineren Fehler (siehe 5.6). Für die Startwertermittlung wurde eine eigene Methode implementiert.

5.1 Die Datenbank

Die Datenbank, hier database (A.2), ist eine externe Datei, um Simulationswerte sowie die Entdimensionierung auszulagern. Sie ermöglicht es, über verschiedene *cases* zwischen Simulationswerten zu wechseln. In dieser Datenbank ist nur *case_1* aufgeführt. In diesem wurden analog zur Entdimensionierung in Kapitel 3.8 alle Werte, Konstanten, Parameter sowie Formeln übertragen.

5.2 Die Startwertermittlung

Die Bestimmung geeigneter Startwerte ist entscheidend für die Qualität der restlichen Simulation. Während die Anfangskonzentration direkt aus der gegebenen Datenbank

übernommen werden kann, erfordern die Potentiale eine iterative Bestimmung.

Anfangskonzentrationen:
Die Startkonzentrationen werden basierend auf dem gewünschten Ladezustand festgelegt. Das Elektrolyt wird dabei homogen über das gesamte Intervall mit dem Wert aus der database verteilt:

```
76 ce = np.full(N,database["ce0"])
```

Analog bei der Aktivpartikelkonzentration:

```
77 cs = np.zeros(N)
78
79 # 1) Anoden-Region [0 .. L_n]:
80 cs[:L_n] = database["cs0n"]
81
82 # 2) Kathoden-Region [L_p .. N-1]:
83 cs[L_p:] = database["cs0p"]
```

Der Separatorbereich bleibt Null, da er keine Aktivpartikel hat. Innerhalb des Partikels wird der Startwert ebenfalls geladen:

```
86 matrix_cs = [np.full(N_solid, cs[i]) for i in range(N)]
```

Kombistartwert:
Die Anfangspotentiale ϕ_s und ϕ_e können nicht unabhängig voneinander gewählt werden, da sie durch die Butler-Volmer-Gleichung (3.7) miteinander gekoppelt sind. Das Problem wird in der Funktion *kombistartwert*() durch die Fixpunktiteration gelöst. Das System muss dabei sowohl die Ladungsneutralität im Gesamtsystem

$$\int_\Omega a\ i_{BV}\, dx = 0,$$

als auch die korrekte Stromdichte in der Kathode

$$\int_{\Omega_p} a\ i_{BV}\, dx = J_{ext}$$

berücksichtigen.

```
492 def kombistartwert(ps_old, pe_old):
493     global ps
494     ps = ps_old
495     for k in range(1000):
496         for i in range(100):
497
498             pe= meansol(A_u_matrix, f_u, pe_old)
499             if mag(pe - pe_old) < 1e-20:
500                 if info > 1:
501                     #print("  Anfangspotential")
502                     #print("    Konvergenz in ", i, "Schritten zu |Pe-Pe_old| = ",mag(
                            pe - pe_old))
503                     break
504             pe_old = pe
505         for i in range(100):
506             ps= berechne_ps(ce, pe, cs, ps_old)
507             if mag(ps - ps_old) < 1e-12:
508                 if info > 1:
509                     #print("  Anfangspotential")
510                     #print("    Konvergenz in ", i, "Schritten zu |Ps-Ps_old| = ",mag(
                            ps - ps_old))
511                     break
512             ps_old = ps
```

```
513         if Abbruchkriterium(pe, ps) :
514             Integral_gesamt = np.trapezoid(F_Potenzial(ce, pe, cs, ps),dx=h)
515             Integral_p = np.trapezoid(F_Potenzial(ce, pe, cs, ps)[L_p:],dx=h)
516             print("␣Anfangspotenzial:␣")
517             print("␣Konvergenz␣in␣", k,"Schritten")
518             print("I_ges:␣", Integral_gesamt)
519             print("Integral_k:␣", Integral_p)
520             break
521 
522     return ps, pe
```

Hierbei werden ϕ_e und ϕ_s im Wechsel zueinander berechnet. ϕ_e wird, wie in Kapitel 5.3 beschrieben, durch das Meansol-Verfahren und ϕ_s, wie in Kapitel 5.4 beschrieben, berechnet.
Diese Iteration wird so oft wiederholt, bis die gewählten Konvergenzkriterien unterschritten werden. Nach erfolgreicher Konvergenz stehen konsistente Startwerte für alle Feldgrößen zur Verfügung, die als Basis für die zeitabhängige Simulation dienen.

5.3 Das Meansol-Verfahren

Das Elektrolytpotential ϕ_e wird durch die Poisson-Gleichung mit reinen Neumann-Randbedingungen beschrieben (7):

$$\frac{\kappa_e}{h^2}(-\phi_{e,i-1} + 2\phi_{e,i} - \phi_{e,i+1}) = a i_{BV}. \quad \text{in } \Omega$$
$$\frac{\partial \phi_e}{\partial n} = 0 \quad \text{auf } \partial\Omega$$

wobei sich das Potential als Gleichung der Form

$$Au = f(u)$$

darstellen lässt. Dabei ist A die Diskretisierung des Problems:

$$-\Delta u = f \quad \text{in } \Omega, \tag{14}$$
$$\frac{\partial u}{\partial n} = 0 \quad \text{auf } \partial\Omega. \tag{15}$$

welches analog zu (8) nur bis auf eine additive Konstante bestimmt ist. Da A singulär ist und die Differenz von Werten an Knoten berechnet, hat es den Konstantvektor $\mathbf{1}$ als Nullvektor ($A\,\mathbf{1} = 0$). Damit ist $Au = f(u)$ nur lösbar, wenn $f(u)$ orthogonal zu $\mathbf{1}$ ist. Das bedeutet:

$$\mathbf{1}^T f(u) = 0 \tag{16}$$

wobei $\mathbf{1}^T f(u)$ die Summe aller Einträge von $f(u)$ ist. Damit spiegelt die Gleichung die Kompatibilitätsbedingung (17) wider.

Kompatibilitätsbedingung des Neumann-Problems
Ausgangssituation ist das typische Neumann-Problem (14) welches über ganz Ω integriert wird:

$$\int_\Omega (-\Delta u)\, dx = \int_\Omega f\, dx.$$

Nach dem Gaußschen Integralsatz (Divergenztheorem) gilt:

$$\int_\Omega (-\Delta u)\, dx = -\int_{\partial\Omega} \nabla u \cdot n dS$$

und durch (15):

$$-\int_{\partial\Omega} \nabla u \cdot n\, dS = 0.$$

Es folgt:

$$\int_\Omega f\, dx = 0. \tag{17}$$

Physikalisch gesehen ist das die Kompatibilitätsbedingung, welche besagt, dass das System nur dann im Gleichgewicht ist, wenn die gesamte Quelle f bei Nullbedingungen am Rand keine Netto-Ladung produziert.

Umsetzung im Programm
Zuerst wird das Startpotential von der Logarithmus-Funktion bereinigt.

```
314     u0 = u0 - 1/2 * np.log(ce)
```

Danach wird der Vektor normalisiert, damit der neue Mittelwert = 0 ist. Man darf den Vektor aufgrund (8) um jeden Wert verschieben.

```
322     u0m = np.mean(u0)
323     u0c = u0 - u0m
```

Danach wird die Kompatibilitätsbedingung (17) im Code umgesetzt:

```
326     def compat(s):
327         return np.trapezoid(f(u0c + s * ev),dx= h)
328
329     # 3) Nullstelle of compat(s) finden
330     u0m_root, = fsolve(compat, u0m)
331
332     # 4)
333     rhs = f(u0c + u0m_root * ev)  #ev ist der Einheitsvektor
```

Hier wird eine Skalarfunktion definiert und genau das s berechnet, für das

$$\int_\Omega f(u_c + s)\, dx \;=\; 0 \quad \text{gilt, was (16) entspricht.}$$

Als nächstes wird die Matrix zur Berechnung von u aufgestellt. Dafür wird A auf einen Blockvektor mit dem Einheitsvektor als Randbedingung erweitert.

```
337     # 5) Blocksystem
338     BM = np.block([
339         [A,          ev.reshape(-1,1)],
340         [ev.reshape(1,-1), np.zeros((1,1))]
341     ])
```

Die rechte Seite des Gleichungssystems besteht aus rhs und der Nullbedingung (16).

```
342     FM = np.concatenate([rhs, [0]])
```

Nun wird folgendes Gleichungssystem gelöst:

$$\begin{bmatrix} A & \mathbf{1} \\ \mathbf{1}^T & 0 \end{bmatrix} \begin{bmatrix} u \\ \lambda \end{bmatrix} = \begin{bmatrix} \text{rhs} \\ 0 \end{bmatrix}.$$

```
345     UM = np.linalg.solve(BM, FM)
```

Das Ergebnis ist der Vektor u mit Mittelwert 0, sowie der angehängte Lagrange-Multiplikator λ. Letzteres ist ein Nebenprodukt, welches auf die eindeutige Lösung zurückführen lässt. Es wird nicht benötigt, weswegen es abgetrennt wird.

```
346     u = UM[:N]
```

Vor der Rückgabe wird der neue Mittelwert mit der Logarithmus-Funktion addiert.

```
349     u_return = u + u0m_root
350     return u_return +0.5* np.log(ce)
```

5.4 Bestimmung des Aktivpartikelpotentials

Das Aktivpartikelpotential in der Anode wird durch Erdung des linken Randes berechnet und so festgesetzt. Die Berechnung von $\phi_{s,p}$ in der Kathode muss wegen einer fehlenden Festsetzung mit der zweiten Nebenbedingung [12] berechnet werden. Hierfür muss die Nullstelle der Funktion

$$f(\phi_{s,L}) = \int_{\Omega_p} a\ i_{bv}\, dx - J$$

gefunden werden. Diese wird in dem Programm durch ein Bisektionsverfahren ermittelt. Innerhalb der Bisektion muss für jeden Wert der passende Funktionswert gefunden werden. Hierfür wird zuerst mithilfe einer Fixpunktiteration das Verhalten von ϕ_s in der Anode bestimmt und anschließend mit der Trapezregel integriert.

```
269     def Integral_wert(phi_rand):
270         phi_g = Matrixloeser(phi_rand, f)
271         Funktionswert = np.trapezoid(-f(phi_g), dx=h) - I0
272         return Funktionswert
273 
274     def Matrixloeser(u_rand, f):
275         u = np.full(length, u_rand)
276         for i in range(100):
277             rhs = f(u)
278             rhs[-1] = u_rand
279             u_neu = A_ps_p_splu.solve(rhs)
280             if np.linalg.norm(u_neu - u) < 1e-10:
281                 break
282             u = u_neu
283         return u
```

Anschließend wird die eigentliche Bisektion durch Intervallhalbierung durchgeführt.

```
292     # Bisektion
293     for i in range(100):
294         Mittelwert = (bisektionswerte[0] + bisektionswerte[1]) / 2
295         if abs(Integral_wert(Mittelwert)) < 1e-10:
296             break
297         if Integral_wert(Mittelwert) * Integral_wert(bisektionswerte[0]) < 0:
```

```
298             bisektionswerte[1] = Mittelwert
299         elif Integral_wert(Mittelwert) * Integral_wert(bisektionswerte[1]) < 0:
300             bisektionswerte[0] = Mittelwert
301         else:
302             print("Fehler")
303 
304     phi_p = Matrixloeser(Mittelwert, f)
305     return phi_p
```

So kann $\phi_{s,p}$ in jedem Zeitschritt bestimmt werden. Jedoch muss noch $\phi_{s,n}$ bestimmt werden. Das passiert in der übergeordneten Funktion BERECHNE-PS():

```
240 def berechne_ps(ce, pe, cs, ps):
241     rhs_g = G_Potenzial_Solid_t_0 + F_Potenzial_Solid(ce, pe, cs, ps)
242     rhs_g[0]=0
243     f_ps_n = np.zeros(L_n)
244     for j in range(L_n):
245         f_ps_n[j] = rhs_g[j]
246     ps_n = A_ps_n_splu.solve(f_ps_n)
247     ps_p_0 = ps[L_p:]
248     ce_p = ce[L_p:]
249     cs_p = cs[L_p:]
250     pe_p = pe[L_p:]
251 
252     def f_ps_p(ps_p):
253         g_ps = F_potenzial_solid_p(ce_p, pe_p, cs_p, ps_p)
254         return g_ps
255 
256     ps_p = bisektionsverfahren(A_ps_p_matrix, f_ps_p, ps_p_0, h, J_ext)
257     N_sep = L_p - L_n
258     ps_new = np.concatenate([
259         ps_n,
260         np.zeros(N_sep),  # Separator
261         ps_p
262     ])
263     return ps_new
```

In der Anode wird das Potential durch die Erdung des linken Randes $\phi_s(0) = 0$ eindeutig festgelegt. Dadurch entfällt hier die zusätzliche Nebenbedingung. Das resultierende Gleichungssystem (13) wird direkt mit dem linearen Gleichungslöser berechnet. Vor dem RETURN werden dann sowohl $\phi_{s,p}$ als auch $\phi_{s,n}$ zusammengeführt.

5.5 Hauptsimulationsschleife

Die Hauptsimulationsschleife ist das Kernelement der Simulation. Hier werden die in Kapitel 4.4 beschriebenen Gleichungen gelöst. Die Konzentration im Elektrolyt wird durch einen Gleichungssystemlöser der scipy-Bibliothek gelöst. Dies gilt auch für die Konzentration im Aktivpartikel, hier wird aber in einem Unterprogramm jeder Partikel einzeln berechnet. Die Potentiale ϕ_s und ϕ_e werden durch die in 5.3 und 5.4 benannten Verfahren bestimmt.

```
553     # ce berechnen
554     b_c = tau*F_Konzentration(ce, pe, cs, ps) + tau * G_Konzentration
555     ce = A_c_matrix_lu_zerlegung.solve(ce+b_c)
556     for i in range(N):
557         if ce[i] < 0:
558             ce[i] = 1e-10  # negative Konzentrationen vermeiden
559             print("error")
560 
561 
562     # pe berechnen
563     pe = meansol(A_u_matrix, f_u, pe)
564     # ps berechnen
```

```
565
566     ps = berechne_ps(ce, pe, cs, ps)
567
568     # cs berechnen
569     matrix_cs = aktivpartikel(ce, pe, cs, ps, matrix_cs)
```

5.6 Validierung

Das Programm wurde teilweise validiert, um zu überprüfen, ob die Prozesse korrekt simuliert werden. Diese Validierung beschränkt sich auf den Diffusionsprozess im Elektrolyt. Die numerische Lösung wurde mit einer konstruierten analytischen Lösung verglichen und der maximale Fehler für unterschiedliche Zeitschrittweiten τ und Berechnungspunkte N analysiert. Es gilt $h = L/(N-1)$. Bei einem funktionierenden System gilt für die Abhängigkeit des Fehlers für eine konstante Zeitschrittweite und variierende Berechnungspunkte:

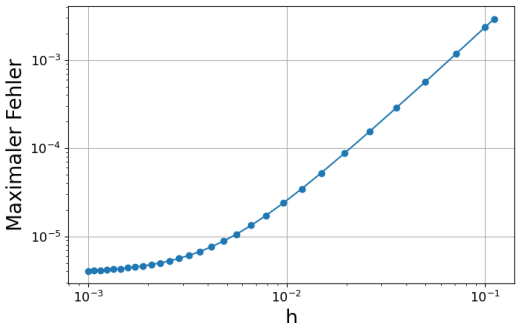
$$err(h) = C_1 h^2.$$

Die Größe h steht im quadratischen Verhältnis zum Fehler. Man kann diese Gleichung auf beiden Seiten logarithmieren. Dann entsteht:

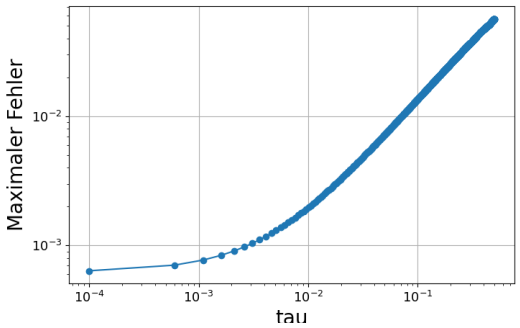
$$\log(err(h)) = \log(C_1) + 2\log(h).$$

Daraus geht hervor, dass bei einem funktionierenden Programm die Steigung der doppelt logarithmischen Fehlerkurve bei der Validierung annähernd 2 ist. Für eine variierende Zeitschrittweite τ und für festgesetzte Berechnungspunkte gilt

$$err(\tau) = C_2\tau \longrightarrow \log(err(\tau)) = \log(C_2) + 1\log(\tau).$$



(a) h-Fehler Diagramm, doppelte logarithmische Skala



(b) τ-Fehler Diagramm, doppelte logarithmische Skala

Abb. 1: Validierung des Modells: Fehlerverhalten in Abhängigkeit von Diskretisierungsschritt h und Zeitschrittweite τ

Hier ergibt sich bei einem korrekt funktionierenden Programm, bei einer doppelt logarithmischen Fehlerkurve, eine Steigung von ungefähr 1. Der Zusammenhang zwischen Fehler und h beziehungsweise τ kann dargestellt werden, indem die Diffusionssimulation mit unterschiedlichen Werten der variablen Größe durchlaufen wird und mit der

analytischen Lösung verglichen wird. Hierbei wird zuerst in jedem Zeitschritt eines einzelnen Programmdurchlaufs ein Differenzvektor zwischen analytischer und numerischer Lösung gebildet und der Betrag des Vektors genommen. Dieser wird mit h multipliziert, um Vergleichbarkeit zwischen den Werten herzustellen. Für den Fehler in einem einzelnen Zeitschritt gilt:

$$err = \left(h \sum_{j=1}^{N} |c(x_j) - C_j|^2 \right)^{1/2}$$

Für jeden Programmdurchlauf wird der maximale Fehler ausgegeben und das Wertepaar $(h \,|\, err_{max})$ oder $(\tau \,|\, err_{max})$ auf einer doppelt logarithmischen Skala aufgetragen. Für die Validierung mit der variablen Größe h ergibt sich dann im Optimalfall eine Steigung von 2, für die variable Größe τ eine Steigung von 1.
Die Steigung des h-Fehler-Diagramms liegt bei 2.04, die des τ-Fehler-Diagramms bei 0.94. Das Modell entspricht damit den Erwartungen an ein korrekt implementiertes numerisches Programm.

6 Auswertung

6.1 Ergebnisse

Die Ergebnisse zeigen, dass das Modell es ermöglicht, den Ladevorgang zu simulieren. Die Abb. 2 stellt mit verschiedenen Farben den zeitlichen Verlauf der Verteilung der Konzentration dar. Hierbei wird die Konzentration in entdimensionierten Werten dargestellt. Es ist erkennbar, dass sich die Lithium-Ionen von der Anode, welche im Diagramm die linke Seite darstellt, in die Kathode, rechte Seite, bewegen. Es wird ein Entladevorgang dargestellt. Hierbei ist die Konzentrationszunahme in der Kathode und die Konzentrationsabnahme in der Anode, über ihre jeweiligen Gebiete gemittelt, konstant. Dies geht aus den Nebenbedingungen (11) und (12) hervor.

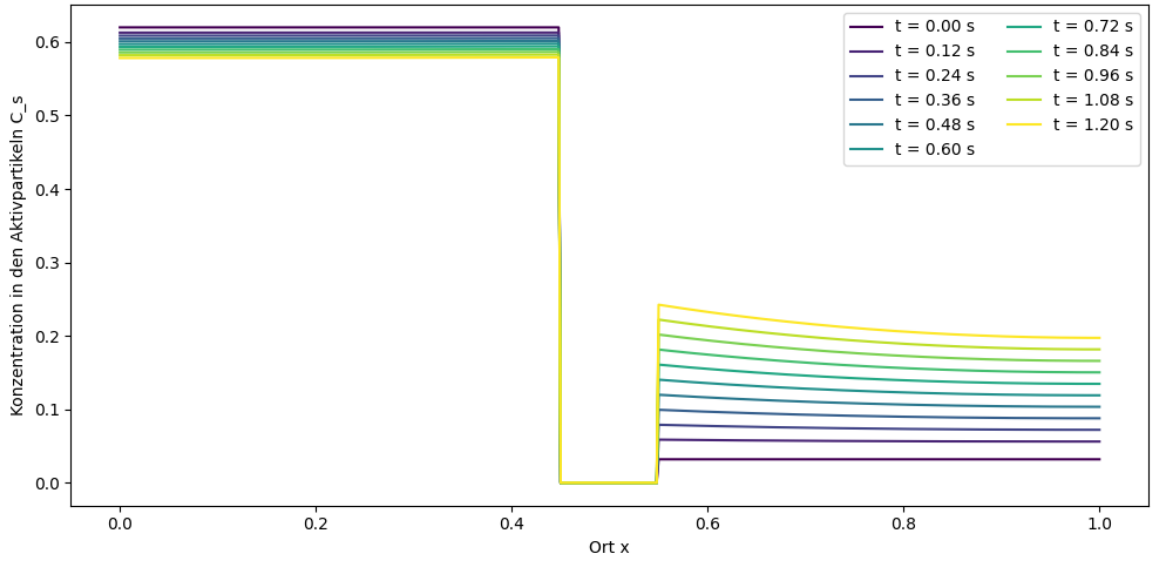


Abb. 2: Zeitprofil für die Konzentrationen im Aktivpartikel mit N=501 und τ=0.08s

Auch der Konzentrationsverlauf (Abb. 3) im Elektrolyt zur Endberechnungszeit zeigt, dass sich die Lithium-Ionen an der Anode aus dem Aktivpartikel hinausbewegen und an der Kathode in den Aktivpartikel hineinbewegen, da die Lithium-Ionen im Bereich der Anode deutlich konzentrierter sind. Hier ist auch zu erkennen, dass das System in sich abgeschlossen ist, da es an den Rändern keinen Konzentrationsfluss gibt.

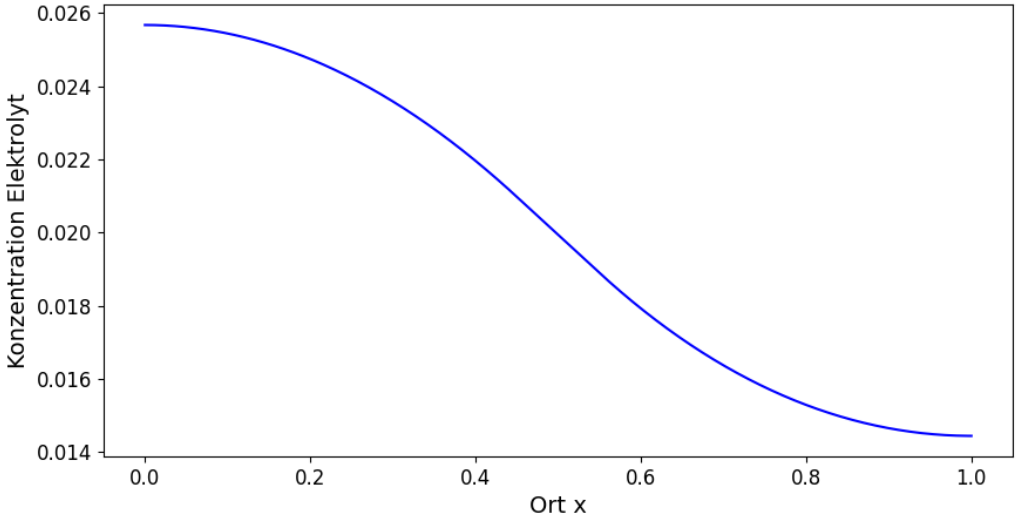
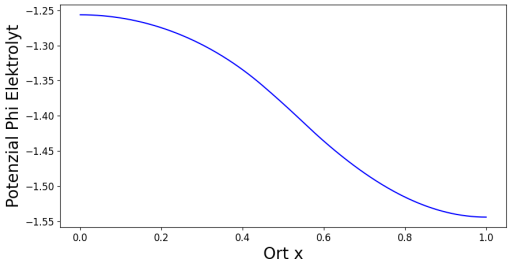
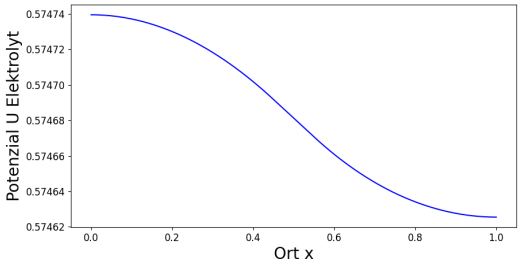


Abb. 3: Profil für die Konzentration im Elektrolyt für N=501 und τ=0.08s

Die Bewegung der Lithium-Ionen wird durch das elektrochemische Potential im Elektrolyt und am Aktivpartikel bestimmt. Für das elektrochemische Potential ϕ_e und das elektrische Potential U_e gilt auch wieder, dass kein Randfluss besteht und bestehen darf. Dies ist auch in den Diagrammen (4a) und (4b) erkennbar.



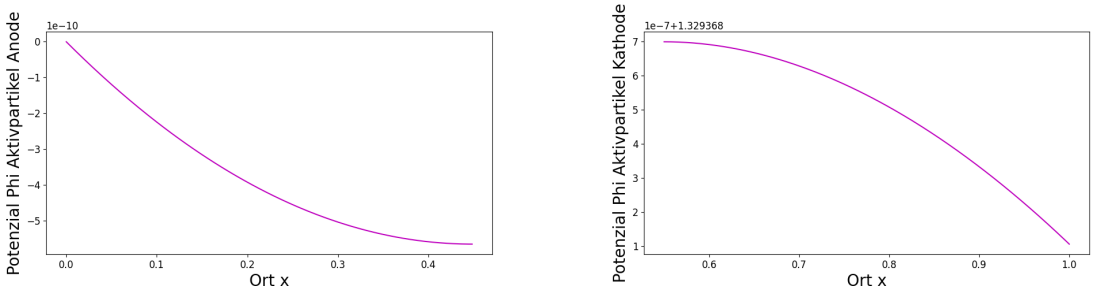
(a) ϕ_e Profil für N=501 und τ=0.08s



(b) U_e Profil für N=501 und τ=0.08s

Abb. 4: Profilverlauf vom elektrochemischen Potential ϕ_e und dem elektrischen Potential U_e

Das Potential am Aktivpartikel wird in zwei Bereiche eingeteilt, in Anode und Kathode. Hier ist relevant, dass der Randfluss der Größe J entspricht. Dies wird, wie in Kapitel 4.3 bereits erklärt, durch die beiden Nebenbedingungen festgelegt. In der berechneten Simulation (Abb. 5) liegt ein Fluss von $9,95$ an der Anode und ein Fluss von $-9,95$ an der Kathode vor, was zu den zu erwartenden Ergebnissen passt. Zur Überprüfung



(a) $\phi_{s,n}$ Profil für N=501 und τ=0.08s (b) $\phi_{s,p}$ Profil für N=501 und τ=0.08s

Abb. 5: Profilverlauf von ϕ_s in der Anode und Kathode

der Ergebnisse müssen zum Endzeitpunkt die Gleichungen (11) und (12) überprüft werden. Die Nichterfüllung dieser Gleichungen deutet auf Fehler im Programm hin. Für die Simulationseinstellungen $N = 501$ und $\tau = 0.08s$ gilt

$$\int_{\Omega} a \cdot i_{bv}\, dx \approx 3.43^{-4}$$

und

$$\int_{\Omega_p} a \cdot i_{bv}\, dx \approx -10.00 \approx J.$$

Diese Ergebnisse deuten darauf hin, dass das Programm korrekt funktioniert.

6.2 Diskussion

Ziel des Projektes war ein einfaches und kompaktes Python-basiertes Programm für die Simulation des Ladevorgangs bei einem Lithium-Ionen-Akkumulator zu programmieren. Hierbei werden auf Basis des DFN-Modells sowohl Ionenbewegungen als auch der Potentialverlauf dargestellt und analysiert. Aufgrund der eindimensionalen Implementierung fällt auch die Berechnungsdauer gering aus. Trotzdem gibt es innerhalb des Programms einige mathematische Idealisierungen, welche die Genauigkeit beeinflussen. Maßgeblich für die Qualität der Simulation ist die gelingende Berechnung der Aktivpartikel [TL21]. Hierfür gibt es verschiedene Möglichkeiten.
Zum einen sind Aktivpartikel in der Realität nicht perfekt rund, wie in dieser Arbeit angenommen wird. Dies führt zu messbaren Feldfluktuationen und Inhomogenitäten in der Konzentrationsverteilung, die durch das Modell nicht erklärt werden können. Ein Ansatz besteht in der Simulation ellipsoidaler Partikel. Das beeinflusst die spezifische Oberfläche a, welches zu einer veränderten Butler-Volmer-Gleichung führt. Aufgrund einer unregelmäßigen Krümmung am Rand besteht so die Möglichkeit eines praxisnäheren Ergebnisses. Die Auswirkungen dieses Prozesses könnten in nachfolgenden Untersuchungen behandelt werden.
Zum anderen kann man auch die Ausdehnung der Aktivpartikel während des Ladevorgangs simulieren. Auch hier sorgt die verändernde Form für ein anderes Ladeverhalten. Ein weiterer Ansatz wäre, die Diffusion der Lithium-Ionen in den Aktivpartikeln mit zwei Phasen zu simulieren.
Möglich ist auch die Präzision durch Einbeziehung der Temperatur zu erhöhen. Diese ändert sich während der Simulation und beeinflusst annähernd alle physikalischen Prozesse.

Danksagung

Wir bedanken uns bei Herrn Prof. Dr. Willy Dörfler und Herrn Gruber, die uns die Möglichkeit gegeben haben, an diesem Projekt zu arbeiten und uns fachlich sowie organisatorisch unterstützt haben. Außerdem bedanken wir uns bei der Hector-Seminar Stiftung und bei Herrn Dr. hc. Hans-Werner Hector sowie Frau Josephine Hector, welche uns diese Forschungsmöglichkeiten erst ermöglicht haben.

Selbständigkeitserklärung

Hiermit versichern wir, dass wir die vorliegende Arbeit unter der Beratung von Herrn Prof. Dr. Willy Dörfler und Herrn Dietmar Gruber selbstständig verfasst haben und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt haben.

Karlsruhe, 6. Oktober 2025

Arthur Messerschmidt	Eric Frommherz

A Anhang

A.1 Gleichungssysteme

A.1.1 Konzentration Elektrolyt

$$A_{ce}C_e^{t+1} = C_e + \tau F_{ce} + \tau G_{ce}$$

$$\begin{pmatrix} 1+2\alpha & -2\alpha & 0 & \cdots & 0 \\ -\alpha & 1+2\alpha & -\alpha & \ddots & \vdots \\ 0 & -\alpha & 1+2\alpha & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & -\alpha \\ 0 & \cdots & 0 & -2\alpha & 1+2\alpha \end{pmatrix} \begin{pmatrix} C_{e,1}^{t+1} \\ C_{e,2}^{t+1} \\ \vdots \\ C_{e,N-1}^{t+1} \\ C_{e,N}^{t+1} \end{pmatrix} = \begin{pmatrix} C_{e,1}^{t} \\ C_{e,2}^{t} \\ \vdots \\ C_{e,N-1}^{t} \\ C_{e,N}^{t} \end{pmatrix} + \tau \begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_{N-1} \\ f_N \end{pmatrix} + \tau \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix}$$

$$\alpha = \frac{D_e\,\tau}{h^2}$$

A.1.2 Konzentration Aktivpartikel

$$A_{cs}C_{s,j}^{k+1} = C_{s,j}^{k} + \tau G_{cs,j}$$

$$\begin{pmatrix} 1 & -1 & 0 & \cdots & 0 \\ -\beta & 1+2\beta+\delta_2 & -(\beta+\delta_2) & \ddots & \vdots \\ 0 & -\beta & 1+2\beta+\delta_3 & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & -(\beta+\delta_{N-1}) \\ 0 & \cdots & 0 & -\psi & \psi \end{pmatrix} \begin{pmatrix} C_{s,j,1}^{k+1} \\ C_{s,j,2}^{k+1} \\ \vdots \\ C_{s,j,N-1}^{k+1} \\ C_{s,j,N}^{k+1} \end{pmatrix} = \begin{pmatrix} C_{s,j,1}^{k} \\ C_{s,j,2}^{k} \\ \vdots \\ C_{s,j,N-1}^{k} \\ C_{s,j,N}^{k} \end{pmatrix} + \tau \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ -\dfrac{i_{\mathrm{Bv}}}{F} \end{pmatrix}.$$

$$\text{mit } N = N_{\text{solid}}, \qquad \beta = \frac{\tau D_s}{h_{\text{solid}}^2}, \qquad \delta_m = \frac{2\tau D_s}{r_m\, h_{\text{solid}}}, \qquad \psi = \frac{D_s}{h_{\text{solid}}}.$$

A.1.3 Elektrisches Potential Elektrolyt

$$A_{ue}U_e = F_{ue} + G_{ue}$$

$$\begin{pmatrix} \dfrac{2\kappa_e}{h^2} & -\dfrac{2\kappa_e}{h^2} & 0 & \cdots & 0 \\ -\dfrac{\kappa_e}{h^2} & \dfrac{2\kappa_e}{h^2} & -\dfrac{\kappa_e}{h^2} & \ddots & \vdots \\ 0 & -\dfrac{\kappa_e}{h^2} & \dfrac{2\kappa_e}{h^2} & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & -\dfrac{\kappa_e}{h^2} \\ 0 & \cdots & 0 & -\dfrac{2\kappa_e}{h^2} & \dfrac{2\kappa_e}{h^2} \end{pmatrix} \begin{pmatrix} U_{e,1} \\ U_{e,2} \\ \vdots \\ U_{e,N-1} \\ U_{e,N} \end{pmatrix} = \begin{pmatrix} F_{ue,1} \\ F_{ue,2} \\ \vdots \\ F_{ue,N-1} \\ F_{ue,N} \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix}.$$

A.1.4 Elektrochemisches Potential Aktivpartikel Anode

$$A_{ps,n}\phi_{s,n} = F_{\phi_{s,n}} + G_{\phi_{s,n}}$$

$$\begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ -\frac{\sigma_{s,n}}{h^2} & \frac{2\sigma_{s,n}}{h^2} & -\frac{\sigma_{s,n}}{h^2} & \ddots & \vdots \\ 0 & -\frac{\sigma_{s,n}}{h^2} & \frac{2\sigma_{s,n}}{h^2} & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & -\frac{\sigma_{s,n}}{h^2} \\ 0 & \cdots & 0 & -\frac{2\sigma_{s,n}}{h^2} & \frac{2\sigma_{s,n}}{h^2} \end{pmatrix} \begin{pmatrix} \phi_{s,1} \\ \phi_{s,2} \\ \vdots \\ \phi_{s,L_n-1} \\ \phi_{s,L_n} \end{pmatrix} = \begin{pmatrix} 0 \\ F_{\phi_s,2} \\ \vdots \\ F_{\phi_s,L_n-1} \\ F_{\phi_s,L_n} \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix}.$$

A.1.5 Elektrochemisches Potential Aktivpartikel Kathode

$$A_{ps,p}\phi_{s,p} = F_{\phi_{s,p}} + G_{\phi_{s,p}}$$

$$\begin{pmatrix} \frac{2\sigma_{s,p}}{h^2} & -\frac{2\sigma_{s,p}}{h^2} & 0 & \cdots & 0 \\ -\frac{\sigma_{s,p}}{h^2} & \frac{2\sigma_{s,p}}{h^2} & -\frac{\sigma_{s,p}}{h^2} & \ddots & \vdots \\ 0 & -\frac{\sigma_{s,p}}{h^2} & \frac{2\sigma_{s,p}}{h^2} & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & -\frac{\sigma_{s,p}}{h^2} \\ 0 & \cdots & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \phi_{s,p,1} \\ \phi_{s,p,2} \\ \vdots \\ \phi_{s,p,L-1} \\ \phi_{s,p,L} \end{pmatrix} = \begin{pmatrix} F_{\phi_{s,p},1} \\ F_{\phi_{s,p},2} \\ \vdots \\ F_{\phi_{s,p},L-1} \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ \hat{\phi}_s \end{pmatrix}.$$

A.2 Database

database.py:

```
# database.py
# ---------------- Referenzwerte aus Traskunov & Latz (2021) ----------------
# Basiert auf Traskunov & Latz (2021) - Electrochimica Acta 379 (2021) 138144

Dimensionierte_Werte = {
    #Universelle Konstanten
    "FarK": 9.64853321e4, #C/mol
    "GasK": 8.3145, #J/(mol K)

    #spezifische Werte
    "L_ges": 1e-4, #m
    "L_p": 4.5e-5, #m
    "L_n": 4.5e-5, #m

    "R_p": 7.5e-6, #m
    "R_n": 7.5e-6, #m
    "asn": 6.0e5, #1/m
    "asp": 6.0e5, #1/m
    "T_sim": 1200, #s

}

```

```python
def calculate_dimensionless_parameters():
    dim = Dimensionierte_Werte

    L_ref = dim["L_ges"]
    C_ref = 5e4
    T_ref = 1000
    K_ref = 293 #K
    U_ref= dim["FarK"] / (dim["GasK"] * K_ref)
    J_ref= 1

    faraday=dim["FarK"]*L_ref*C_ref/(T_ref*J_ref)

    # Dimensionslose Parameter
    parameter = {

        "L": dim["L_ges"] / L_ref,
        "L_p": dim["L_p"] / L_ref,
        "L_n": dim["L_n"] / L_ref,
        "L_s": 1-dim["L_p"] / L_ref - dim["L_n"] / L_ref,
        "R_p": dim["R_p"] / L_ref,
        "R_n": dim["R_n"] / L_ref,
        "asn": dim["asn"] * L_ref,
        "asp": dim["asp"] * L_ref,
        "T_ref": T_ref,
        "Fara": faraday,
        "kappa_e": 0.12*U_ref/(L_ref*J_ref),
        "sigma_n": 1e4*U_ref/(L_ref*J_ref),
        "sigma_p": 1e4*U_ref/(L_ref*J_ref),
        "difen":1e-11*T_ref/L_ref**2,
        "difsn": 3.4e-14*T_ref/L_ref**2,
        "difsp": 3.4e-14*T_ref/L_ref**2,
        "ce0": 1e3/C_ref,
        "cs0p": 3.1e4/C_ref,
        "cs0n": 3.1e4/C_ref,
        "i00": 10/J_ref,
        "cmaxn": 3.2e4/C_ref,
        "cmaxp": 3.2e4/C_ref,
        "T_end": dim["T_sim"] / T_ref,
        "J_ext": -10 / J_ref,
        "Xi" : faraday/(dim["GasK"]*293),
    }

    return parameter


case_1= calculate_dimensionless_parameters()
```

A.3 Programmcode

MathBattery.py

```python
# ---------------- Imports ----------------
import time
import math
import numpy as np
import matplotlib.pyplot as plt
from scipy.sparse import diags, lil_matrix
from database import case_1 as database
from scipy.sparse import lil_matrix, csc_matrix
import numpy as np
from scipy.optimize import fsolve
from scipy.sparse.linalg import splu

# ---------------- Physikalische Konstanten ----------------
D_e = database["difen"]
kappa_e = database["kappa_e"]
kappa_s_n = database["sigma_n"]
kappa_s_p = database["sigma_p"]
```

```
asn = database["asn"]
asp = database["asp"]
d_s_n = database["difsn"]
d_s_p = database["difsp"]
T= database["T_end"]
Faraday = database["Fara"]
J_ext = database["J_ext"]
T_ref = database["T_ref"]
i00 = database["i00"]
Xi = database["Xi"]

T_ref = database["T_ref"]
L_p_l = database["L_p"]
L_n_l = database["L_n"]
R_p = database["R_p"]
R_n = database["R_n"]
asn = database["asn"]
asp = database["asp"]

cmaxn = database["cmaxn"]
cmaxp = database["cmaxp"]
asp = database["asp"]

sigma_n = database['sigma_n']
sigma_p = database['sigma_p']

# ---------------- Simulationseinstellungen ----------------
info = 4 # Info kann einen Wert zwischen 0-4 annehmen
N = 501 # Anzahl Intervallpunkte
tau_dimensioniert = 0.08
tau = tau_dimensioniert/T_ref # Zeitschrittweite
N_solid = 30 # Anzahl der Unterteilungen im Aktivpartikel

steps = int(T / tau)
h = 1 / (N - 1)
L_n=round(L_n_l/h)
L_p=round((1-L_p_l)/h)
x = np.linspace(0, 1, N)
plot_frequency = round((T/tau)/10)
print_frequency = plot_frequency
if info > 1:
   print("Program potential_solid_15.8")
   print("  N = ",N,", N-solid = ",N_solid)
   print("  T = ",T,", tau = ",tau)
h_solid_n= R_n / (N_solid - 1)
h_solid_p= R_p / (N_solid - 1)
y = np.linspace(0, 1, N_solid)

r_n= np.ones(N_solid)
for i in range(N_solid):
   r_n[i] = (i+0.5 )* h_solid_n

r_p= np.ones(N_solid)
for i in range(N_solid):
   r_p[i] = (i+0.5 )* h_solid_p

# ---------------- Anfangswerte ----------------
C_values = []
ce = np.full(N,database["ce0"])
cs = np.zeros(N)

# 1) Anoden-Region [0 .. L_n]:
cs[:L_n] = database["cs0n"]

# 2) Kathoden-Region [L_p .. N-1]:
cs[L_p:] = database["cs0p"]
pe = np.zeros(N)

matrix_cs = [np.full(N_solid, cs[i]) for i in range(N)]

```

```
ps= np.zeros(N)
pe= 0.5* np.log(ce)
testlaenge= np.zeros(N)
N_L_p = testlaenge[L_p:]
length = len(N_L_p)


# Lokale Gr   en
idx = np.arange(len(cs))
mask_n = idx <= L_n        # Anodenregion
mask_p = idx >= L_p        # Kathodenregion
mask_sep = (idx > L_n) & (idx < L_p)

# lokale c_max je nachdem, ob Anode oder Kathode
cmax = np.empty_like(cs)
cmax[mask_n] = cmaxn
cmax[mask_p] = cmaxp
cmax[mask_sep] = 0  # im Separator keine Reaktion     keine Begrenzung n tig

# Fl chenfaktor a lokal
a_lokal = np.zeros_like(cs)
a_lokal[mask_n]  = asn
a_lokal[mask_p]  = asp
# im Separator bleibt a_lokal=0

# ---------------- Funktionen ----------------

# Butler Volmer Funktion
def i_BV(ce, ue, cs, us):
    """
     B u t l e r Volmer -Stromdichte mit separaten Ubsn/Ubsp und cmaxn/cmaxp.
    """
    # Austauschstrom  i
    i0 = np.sqrt(ce * cs * (cmax - cs))
    i0 = np.where(cs > cmax, 0.0, i0)

    #   berpotential   je nach Region
    ubs = np.zeros_like(cs)
    ubs[mask_n] = Ubsn(cs[mask_n])
    ubs[mask_p] = Ubsp(cs[mask_p])
    # Separator bleibt ubs=0

    #  B u t l e r Volmer
    i_BV = 2.0 * i00 * i0 * np.sinh((1/2.0) * (us - ue - ubs))



    return i_BV

def Ubsn(z):
    """
      berpotential  -Funktion f  r die Anode.

    """
    return (
        0.6379
        + 0.5416 * np.exp(-100 * z)
        + 0.044  * np.tanh(-(z - 0.18) / 0.05)
        + 0.2    * np.tanh(-(z - 1.035) / 0.055)
        + 0.6875 * np.tanh(-(z + 0.02) / 0.038)
        + 0.0175 * np.tanh(-(z - 0.511) / 0.02)
    )

def Ubsp(z):
    """
      berpotential  -Funktion f  r die Kathode.
    """
    return (
        1091.60809
        + 0.5416         * np.exp(-100 * z)
```

```
158         + 0.721524957     * np.tanh(-(z - 0.984753725) / 0.0422847481)
159         + 1089.28504      * np.tanh(-(z + 5.39175183)   / 1.49617161)
160     )
161
162 # Matrizen
163 def A_u():
164     A = np.zeros((N, N))
165     main_diag_u = 2 * kappa_e / h**2 * np.ones(N)
166     off_diag_u = -(kappa_e / h**2) * np.ones(N - 1)
167     A = diags([off_diag_u, main_diag_u, off_diag_u], [-1, 0, 1]).tocsc()
168     A[0, 1] = -2 * kappa_e / h**2
169     A[-1, -2] = -2 * kappa_e / h**2
170     return A
171
172 def A_ps_n():
173     kap_n = sigma_n
174     A = lil_matrix((L_n, L_n))
175
176     for i in range(1, L_n-1):
177         A[i, i-1] = -kap_n/h**2
178         A[i, i]   =  2*kap_n/h**2
179         A[i, i+1] = -kap_n/h**2
180     # Randbedingungen:
181     A[0, 1] = 0
182     A[0,0] = 1
183     A[-1, -2] = -2 * kap_n / h**2
184     A[-1, -1] = 2 * kap_n / h**2
185     return csc_matrix(A)
186
187 def A_ps_p():
188     kap_p = sigma_p
189     A = lil_matrix((length, length))
190
191     for i in range(1, length-1):
192         A[i, i-1] = -kap_p/h**2
193         A[i, i]   =  2*kap_p/h**2
194         A[i, i+1] = -kap_p/h**2
195     # Randbedingungen:
196     A[0, 1] = -2 * kap_p / h**2
197     A[0,0] = 2*kap_p / h**2
198     A[-1, -2] = 0
199     A[-1, -1] = 1
200     return csc_matrix(A)
201
202 def A_c():
203     A = np.zeros((N, N))
204     haupt_diag_c = (1 + 2 * D_e * tau / h**2) * np.ones(N)
205     neben_diag_c = -(D_e * tau / h**2) * np.ones(N - 1)
206     A = diags([neben_diag_c, haupt_diag_c, neben_diag_c], [-1, 0, 1]).tocsc()
207     A[0, 1] = -2 * D_e * tau / h**2
208     A[-1, -2] = -2 * D_e * tau / h**2
209     return A
210
211 def A_aktivpartikel_n(r_n, h_solid_n, D_s_n, N_solid, tau):
212     """
213     Matrix f r n-Seite mit Diffusionskoeffizient D_s_n.
214     r_n: Radialvektor, h_solid_n Gitterabstand, N_solid Anzahl Punkte.
215     """
216     main_diag = (1 + tau*(2*D_s_n/h_solid_n**2 + 2*D_s_n/(r_n*h_solid_n))) * np.ones(
            N_solid)
217     off_dm1   = -tau*(D_s_n/h_solid_n**2) * np.ones(N_solid-1)
218     off_dp1   = -tau*(D_s_n/h_solid_n**2 + 2*D_s_n/(r_n[:-1]*h_solid_n)) * np.ones(
            N_solid-1)
219     A = diags([off_dm1, main_diag, off_dp1], offsets=[-1,0,1]).tocsc()
220     # Randbedingungen:
221     A[0,0] = 1;     A[0,1] = -1
222     A[-1,-1] = D_s_n/h_solid_n;     A[-1,-2] = -D_s_n/h_solid_n
223     return A
224
225 def A_aktivpartikel_p(r_p, h_solid_p, D_s_p, N_solid, tau):
```

```
226     """
227     Matrix f r p-Seite mit Diffusionskoeffizient D_s_p.
228     """
229     main_diag = (1 + tau*(2*D_s_p/h_solid_p**2 + 2*D_s_p/(r_p*h_solid_p))) * np.ones(
            N_solid)
230     off_dm1   = -tau*(D_s_p/h_solid_p**2) * np.ones(N_solid-1)
231     off_dp1   = -tau*(D_s_p/h_solid_p**2 + 2*D_s_p/(r_p[:-1]*h_solid_p)) * np.ones(
            N_solid-1)
232     A = diags([off_dm1, main_diag, off_dp1], offsets=[-1,0,1]).tocsc()
233     # Randbedingungen:
234     A[0,0] = 1;    A[0,1] = -1
235     A[-1,-1] = D_s_p/h_solid_p;    A[-1,-2] = -D_s_p/h_solid_p
236     return A
237
238 # Potenzialberechnung
239
240 def berechne_ps(ce, pe, cs, ps):
241     rhs_g = G_Potenzial_Solid_t_0 + F_Potenzial_Solid(ce, pe, cs, ps)
242     rhs_g[0]=0
243     f_ps_n = np.zeros(L_n)
244     for j in range(L_n):
245         f_ps_n[j] = rhs_g[j]
246     ps_n = A_ps_n_splu.solve(f_ps_n)
247     ps_p_0 = ps[L_p:]
248     ce_p = ce[L_p:]
249     cs_p = cs[L_p:]
250     pe_p = pe[L_p:]
251
252     def f_ps_p(ps_p):
253         g_ps = F_potenzial_solid_p(ce_p, pe_p, cs_p, ps_p)
254         return g_ps
255
256     ps_p = bisektionsverfahren(A_ps_p_matrix, f_ps_p, ps_p_0, h, J_ext)
257     N_sep = L_p - L_n
258     ps_new = np.concatenate([
259         ps_n,
260         np.zeros(N_sep),  # Separator
261         ps_p
262     ])
263     return ps_new
264
265 def bisektionsverfahren(A, f, u0, h, I0):
266     Startwert = u0[-1]
267     bisektionswerte = np.full(2, Startwert)
268
269     def Integral_wert(phi_rand):
270         phi_g = Matrixloeser(phi_rand, f)
271         Funktionswert = np.trapezoid(-f(phi_g), dx=h) - I0
272         return Funktionswert
273
274     def Matrixloeser(u_rand, f):
275         u = np.full(length, u_rand)
276         for i in range(100):
277             rhs = f(u)
278             rhs[-1] = u_rand
279             u_neu = A_ps_p_splu.solve(rhs)
280             if np.linalg.norm(u_neu - u) < 1e-10:
281                 break
282             u = u_neu
283         return u
284
285     # Startintervall ermitteln
286     for i in range(100):
287         bisektionswerte[0] = Startwert - 2**(i-2)
288         bisektionswerte[1] = Startwert + 2**(i-2)
289         if Integral_wert(bisektionswerte[0]) * Integral_wert(bisektionswerte[1]) < 0:
290             break
291
292     # Bisektion
293     for i in range(100):
```

```
294             Mittelwert = (bisektionswerte[0] + bisektionswerte[1]) / 2
295             if abs(Integral_wert(Mittelwert)) < 1e-10:
296                 break
297             if Integral_wert(Mittelwert) * Integral_wert(bisektionswerte[0]) < 0:
298                 bisektionswerte[1] = Mittelwert
299             elif Integral_wert(Mittelwert) * Integral_wert(bisektionswerte[1]) < 0:
300                 bisektionswerte[0] = Mittelwert
301             else:
302                 print("Fehler")
303 
304     phi_p = Matrixloeser(Mittelwert, f)
305     return phi_p
306 
307 def f_u(u):
308     phi_e = u + (1/2) * np.log(ce)
309     g_u = G_Potenzial_t_0 + F_Potenzial(ce, phi_e, cs, ps)
310     return g_u
311 
312 def meansol(A: np.ndarray, f, u0: np.ndarray) -> np.ndarray:
313 
314     u0 = u0 - 1/2 * np.log(ce)
315     if hasattr(A, "toarray"):
316         A = A.toarray()
317 
318     u0 = np.asarray(u0)
319     N = u0.size
320     ev = np.ones(N)
321     # 1) Mittelwert herausnehmen
322     u0m = np.mean(u0)
323     u0c = u0 - u0m
324 
325     # 2) Kompatibilit tsbedingung:
326     def compat(s):
327         return np.trapezoid(f(u0c + s * ev),dx= h)
328 
329     # 3) Nullstelle of compat(s) finden
330     u0m_root, = fsolve(compat, u0m)
331 
332     # 4)
333     rhs = f(u0c + u0m_root * ev)
334 
335 
336 
337     # 5) Blocksystem
338     BM = np.block([
339         [A,          ev.reshape(-1,1)],
340         [ev.reshape(1,-1), np.zeros((1,1))]
341     ])
342     FM = np.concatenate([rhs, [0]])
343 
344     # 6) System l sen und u extrahieren
345     UM = np.linalg.solve(BM, FM)
346     u = UM[:N]
347 
348     # 7) Verschiebung zur ckaddieren
349     u_return = u + u0m_root
350     return u_return +0.5* np.log(ce)
351 
352 # Interface
353 
354 def F_potenzial_solid_p(ce_p, pe_p, cs_p, us):
355     """
356      B u t l e r Volmer -Funktion mit separaten Ubsp und cmaxp.
357     """
358 
359     # Austauschstrom  i
360     i0 = np.sqrt(ce_p * cs_p * (cmaxp - cs_p))
361     i0 = np.where(cs_p > cmaxp, 0.0, i0)
362 
363     #  B u t l e r Volmer
```

```
364     i_BV = 2.0 * i00 * i0 * np.sinh((1/2.0) * (us - pe_p - Ubsp(cs_p)))
365
366
367     return -asp* i_BV
368
369 def F_Konzentration(ce, pe, cs, us):
370
371     #  B u t l e r Volmer -Strom
372     ibv = i_BV(ce, pe, cs, us)
373
374     # R ckgabe der Reaktionsrate
375     return (a_lokal / Faraday) * ibv
376
377 def F_Potenzial(ce, pe, cs, us):
378     """
379     Potenzial-Quellenfunktion mit getrennten spezifischen Fl chen
380     f r Anode (asn) und Kathode (asp).
381     """
382     #  B u t l e r Volmer -Strom
383     ibv = i_BV(ce, pe, cs, us)
384
385     # Potenzialquelle
386     f = a_lokal * ibv
387     return f
388
389 def F_Potenzial_Solid(ce, pe, cs, ps):
390     """
391     Solid-Potenzial-Quellenfunktion mit getrennten spezifischen Fl chen
392     f r Anode (asn) und Kathode (asp).
393     """
394
395     #  B u t l e r Volmer -Strom im Solid
396     ibv = i_BV(ce, pe, cs, ps)
397
398     # Solid-Potenzialquelle (negatives Vorzeichen)
399     f = - a_lokal * ibv
400     return f
401
402
403 # Randbedingungen-teilweise momentan ungenutzt
404 def G_Konzentration():
405    G = np.zeros(N)
406    return G
407
408 def G_Potenzial():
409    G = np.zeros(N)
410    G[0] = 0
411    G[-1] = 0
412    return G
413
414 def G_Potenzial_t_0():
415    G = np.zeros(N)
416    G[0] = 0
417    G[-1] = 0
418    return G
419
420 def G_Potenzial_Solid():
421     G = np.zeros(N)
422     return G
423
424 def G_Potenzial_Solid_t_0():
425     G = np.zeros(N)
426     return G
427
428
429 # Berechnung f r cs
430 def Flussrandbedingung(ce_s, ue_s, cs_s, ps_s, i):
431     """
432     Flussrandbedingung am    uersten    Punkt des Aktivpartikels.
433     """
```

```
434     return - (1.0 / Faraday) * i_BV_scalar(ce_s, ue_s, cs_s, ps_s, i)
435
436 def i_BV_scalar(ce, ue, cs, us, i):
437     """
438     Berechnet den  B u t l e r Volmer -Strom an Gitterstelle i.
439     """
440     ce_s= np.full(N, ce)
441     ue_s= np.full(N, ue)
442     cs_s= np.full(N, cs)
443     us_s= np.full(N, us)
444
445     ibv_all = i_BV(ce_s, ue_s, cs_s, us_s)
446     return float(ibv_all[i])
447
448 def aktivpartikel(ce, pe, cs, ps, matrix_cs):
449     """
450     L  s e  f  r  jedes Gitter i:
451     - i <= L_n   : Anoden-Partikel mit D_s_n
452     - i >= L_p   : Kathoden-Partikel mit D_s_p
453     - sonst      : Separator     Konzentration = 0
454     """
455
456     for i in range(N):
457         if i <= L_n:
458             lu = Aktiv_mat_lu_n                 # Anode
459         elif i >= L_p:
460             lu = Aktiv_mat_lu_p                 # Kathode
461         else:
462             matrix_cs[i] = np.zeros(N_solid)    # Separator: keine Partikel, alles
                Null
463             continue
464
465         # rechter Vektor b_cs
466         b_cs = matrix_cs[i].copy()
467         b_cs[0]   = 0
468         b_cs[-1]  = Flussrandbedingung(ce[i], pe[i], cs[i], ps[i],i)
469
470         # L s en des linearen Systems
471         matrix_cs[i] = lu.solve(b_cs)
472
473     return matrix_cs
474
475
476 # Rest
477
478 def Abbruchkriterium(pe, ps) :
479     Integral_gesamt = np.trapezoid(F_Potenzial(ce, pe, cs, ps),dx=h)
480     Integral_p = np.trapezoid(F_Potenzial(ce, pe, cs, ps)[L_p:],dx=h)
481
482     if abs(Integral_gesamt) < 1e-8 and abs(Integral_p-J_ext) < 1e-8:
483         return True
484     else:
485         return False
486
487 def mag(x):
488    return math.sqrt(h * sum(z ** 2 for z in x))
489
490 # ---------------- Startpotenzial berechnen ----------------
491
492 def kombistartwert(ps_old, pe_old):
493     global ps
494     ps = ps_old
495     for k in range(1000):
496         for i in range(100):
497
498             pe= meansol(A_u_matrix, f_u, pe_old)
499             if mag(pe - pe_old) < 1e-20:
500                 if info > 1:
501                     #print("  Anfangspotential")
502                     #print("    Konvergenz in ", i, "Schritten zu |Pe-Pe_old| = ",mag(
```

```
                        pe - pe_old))
503                     break
504                 pe_old = pe
505             for i in range(100):
506                 ps= berechne_ps(ce, pe, cs, ps_old)
507                 if mag(ps - ps_old) < 1e-12:
508                     if info > 1:
509                         #print("  Anfangspotential")
510                         #print("    Konvergenz in ", i, "Schritten zu |Ps-Ps_old| = ",mag(
                                ps - ps_old))
511                     break
512                 ps_old = ps
513             if Abbruchkriterium(pe, ps) :
514                 Integral_gesamt = np.trapezoid(F_Potenzial(ce, pe, cs, ps),dx=h)
515                 Integral_p = np.trapezoid(F_Potenzial(ce, pe, cs, ps)[L_p:],dx=h)
516                 print("␣Anfangspotenzial:␣")
517                 print("␣Konvergenz␣in␣", k,"Schritten")
518                 print("I_ges:␣", Integral_gesamt)
519                 print("Integral_k:␣", Integral_p)
520                 break
521
522     return ps, pe
523
524 # Vektoren
525 G_Potenzial_Solid_t_0 = G_Potenzial_Solid_t_0()
526 G_Potenzial_t_0 = G_Potenzial_t_0()
527 G_Konzentration = G_Konzentration()
528
529 # Matrizen
530 A_c_matrix_lu_zerlegung = splu(A_c().tocsc())
531 A_u_matrix = A_u()
532 A_ps_n_matrix = A_ps_n()
533 A_ps_p_matrix = A_ps_p()
534 Aktiv_mat_lu_n = splu(A_aktivpartikel_n(r_n, h_solid_n, d_s_n, N_solid, tau).tocsc())
535 Aktiv_mat_lu_p = splu(A_aktivpartikel_p(r_p, h_solid_p, d_s_p, N_solid, tau).tocsc())
536 A_ps_n_splu = splu(A_ps_n().tocsc())
537 A_ps_p_splu = splu(A_ps_p().tocsc())
538 ps_p = np.zeros(length)
539 ps, pe = kombistartwert(ps, pe)
540 cs_speicher= []
541 cs_zeit = []
542 # ---------------- Haupt-Simulationsschleife ----------------
543 z = 0
544 start_time = time.time()
545 for t in range(steps):
546     z += tau
547     if t % print_frequency == 0 and info > 0:
548         print(f"␣␣Zeit:␣{z:.2f}s")
549         cs_speicher.append(cs.copy())
550         cs_zeit.append(z)
551
552
553     # ce berechnen
554     b_c = tau*F_Konzentration(ce, pe, cs, ps) + tau * G_Konzentration
555     ce = A_c_matrix_lu_zerlegung.solve(ce+b_c)
556     for i in range(N):
557         if ce[i] < 0:
558             ce[i] = 1e-10  # negative Konzentrationen vermeiden
559             print("error")
560
561
562     # pe berechnen
563     pe = meansol(A_u_matrix, f_u, pe)
564     # ps berechnen
565
566     ps = berechne_ps(ce, pe, cs, ps)
567
568     # cs berechnen
569     matrix_cs = aktivpartikel(ce, pe, cs, ps, matrix_cs)
570     for i in range(N):
```

```
    # Bestimme cmax f r Anode/Kathode, berspringe Separator
        if i <= L_n:
            cmax_i = cmaxn
        elif i >= L_p:
            cmax_i = cmaxp
        else:
            continue

    # clippe auf h chstens cmax_i - 1e-8
        matrix_cs[i] = np.clip(matrix_cs[i], None, cmax_i - 1e-8)


    cs = np.array([reihe[-1] for reihe in matrix_cs])

cs_speicher.append(cs.copy())
cs_zeit.append(z)
ue = pe -0.5 * np.log(ce)
end_time = time.time()

# Ausgabe
Partikel_werte = matrix_cs[1]
if info > 0:
   print(f"  Stop zur Zeit {z:.2f}")
   print(f"  Ce_min: {min(ce):4.2e} ",f"   Ce_max: {max(ce):4.2e} ")
   print(f"  Cs_min: {min(cs):4.2e} ",f"   Cs_max: {max(cs):4.2e} ")
   print(f"  Ue_min: {min(ue):4.2e} ",f"   Ue_max: {max(ue):4.2e} ")
   print(f"  Pe_min: {min(pe):4.2e} ",f"   Pe_max: {max(pe):4.2e} ")
   print(f"  Ps_min: {min(ps):4.2e} ",f"   Ps_max: {max(ps):4.2e} ")
   print(f"Ben tigte Rechenzeit: {end_time - start_time:.2f} Sekunden")
   Integral_gesamt = np.trapezoid(F_Potenzial(ce, pe, cs, ps),dx=h)
   Integral_p = np.trapezoid(F_Potenzial(ce, pe, cs, ps)[L_p:],dx=h)
   print("I_ges: ", Integral_gesamt)
   print("Integral_k: ", Integral_p)
if info > 2:
   print(f"  2F*Fluss ce (Rand  links): {2*Faraday*D_e*(ce[ 0] - ce[ 1]) / h:4.2e}")
   print(f"  2F*Fluss ce (Rand rechts): {2*Faraday*D_e*(ce[-1] - ce[-2]) / h:4.2e}")
   print(f"  2F*Fluss cs (Rand  links): {2*Faraday*d_s_n*(cs[ 0] - cs[ 1]) / h:4.2e}")
   print(f"  2F*Fluss cs (Rand rechts): {2*Faraday*d_s_p*(cs[-1] - cs[-2]) / h:4.2e}")
   print(f"  Fluss ue (Rand  links): {kappa_e*(ue[ 0] - ue[ 1]) / h:4.2e}")
   print(f"  Fluss ue (Rand rechts): {kappa_e*(ue[-1] - ue[-2]) / h:4.2e}")
   print(f"  Fluss ps (Rand  links): {kappa_s_n*(ps[ 0] - ps[ 1]) / h:4.2e}")
   print(f"  Fluss ps (Rand rechts): {kappa_s_p*(ps[-1] - ps[-2]) / h:4.2e}")

# ---------------- Visualisierung ----------------
plt.figure(figsize=(10, 5))
plt.subplot(1, 3, 1)
plt.plot(x, ce, label="Konzentration C_e", color='b')
plt.xlabel("Ort x")
plt.ylabel("Konzentration Elektrolyt")
plt.legend()
plt.subplot(1, 3, 2)
plt.plot(x, pe, label="Potenzial Phi_e", color='g')
plt.xlabel("Ort x")
plt.ylabel("Potenzial Phi Elektrolyt")
plt.legend()
plt.subplot(1, 3, 3)
plt.plot(x, ue, label="Potenzial U Elektrolyt", color='b')
plt.xlabel("Ort x")
plt.ylabel("Konzentration Elektrolyt")
plt.legend()
plt.tight_layout()
plt.show()


plt.figure(figsize=(10, 5))
plt.subplot(1, 3, 1)
plt.plot(x, ps, label="Potenzial_solid Phi_s", color='m')
plt.xlabel("Ort x")
plt.ylabel("Potenzial ")
plt.legend()
```

```
641 plt.subplot(1, 3, 2)
642 plt.plot(x, cs, label="Konzentration_solid␣C_s", color='m')
643 plt.xlabel("Ort␣x")
644 plt.ylabel("Konzentration␣C_s")
645 plt.legend()
646 plt.subplot(1, 3, 3)
647 plt.plot(y, Partikel_werte, label="Konzentration_solid␣Partikel", color='m')
648 plt.xlabel("Ort␣x")
649 plt.ylabel("Konzentration_solid␣Partikel")
650 plt.legend()
651 plt.tight_layout()
652 plt.show()
653
654 if info > 0:
655     print(f"Fertig")
```

Literatur

[ALS09] Gerold Adam, Peter Läuger, and Günther Stark, *Physikalische Chemie und Biophysik*, 5 ed., Springer, 2009, Kapitel 8.4: Nernst-Planck-Gleichung und Diffusionspotential.

[AMWWW09] Gerhard Angerer, Frank Marscheider-Weidemann, Matthias Wendl, and Martin Wietschel, *Lithium für Zukunftstechnologien*, Nachfrage und Angebot unter besonderer Berücksichtigung der Elektromobilität. Karlsruhe: Fraunhofer ISI **69** (2009).

[Arr89] Svante Arrhenius, *Über die Reaktionsgeschwindigkeit bei der Inversion von Rohrzucker durch Säuren*, Zeitschrift für Physikalische Chemie **4U** (1889), no. 1, 226–248.

[Cas16] Giuseppe Fabian Castelli, *Die Numerische Simulation eines Mikroskalenmodells für Li-Ionen-Batterien*, Master's thesis, Karlsruher Institut für Technologie, 2016.

[Ein05] Albert Einstein, *Über die von der molekularkinetischen Theorie der Wärme geforderte Bewegung von in ruhenden Flüssigkeiten suspendierten Teilchen*, Annalen der Physik **322** (1905), no. 8, 549–560, Freier Volltext.

[ES+19] Arbeitsgruppe Erneuerbare Energien-Statistik et al., *Zeitreihen zur Entwicklung der erneuerbaren Energien in Deutschland*, Bundesministerium für Wirtschaft und Energie (BMWi) (2019).

[FM94] A.-S. Feiner and A. J. McEvoy, *The nernst equation*, Journal of Chemical Education **71** (1994), no. 6, 493.

[fwZuEB15] Bundesministerium für wirtschaftliche Zusammenarbeit und Entwicklung (BMZ), *Klimaabkommen von Paris*, Bundesministerium für wirtschaftliche Zusammenarbeit und Entwicklung (BMZ) (2015).

[GCF+14] Rolando Guidelli, Richard G Compton, Juan M Feliu, Eliezer Gileadi, Jacek Lipkowski, Wolfgang Schmickler, and Sergio Trasatti, *Definition of the transfer coefficient in electrochemistry (iupac recommendations 2014)*, Pure and Applied Chemistry **86** (2014), no. 2, 259–262.

[NB21] John Newman and Nitash P Balsara, *Electrochemical systems*, John Wiley & Sons, 2021.

[New75] John Newman, *Porous-Electrode Theory with Battery Applications*, AIChE Journal (1975).

[RJ25] Romain de Loubens Richard Joly, Grégoire Allaire, *Geometric optimization of a lithium-ion battery with the doyle-fuller-newman model*.

[Son23] Boheng Song, *Erneuerbare Energien: Statistik der Energiespeicher*, Ph.D. thesis, Masterarbeit, Dresden, Technische Universität Dresden, 2023.

[TL21] Igor Traskunov and Arnulf Latz, *New Reduced-Order Lithium-Ion Battery Model to Account for the Local Fluctuations in the Porous Electrodes*, Energy Technology **9** (2021), no. 6, 2000861.

[ZGL+24] Jihao Zhang, Huding Guo, Libin Lei, Shuanglin Shen, Keqing Zheng, and Minfang Han, *A critical review of the Butler-Volmer equation for the activation polarization of solid oxide fuel cells*, Journal of Power Sources **613** (2024), 234871.

Abbildungsverzeichnis