A FINITE ELEMENT BLOCK MODIFIED BACKWARD EULER METHOD FOR SOLVING A ONE-DIMENSIONAL POISSON-NERNST-PLANCK ION CHANNEL MODEL

by

Silas A. Winnemoeller

A Dissertation Submitted in

Partial Fulfillment of the

Requirements for the Degree of

Master of Science

in Mathematics

at

The University of Wisconsin–Milwaukee

May 2024

ABSTRACT

A FINITE ELEMENT BLOCK MODIFIED BACKWARD EULER METHOD FOR SOLVING A ONE-DIMENSIONAL POISSON-NERNST-PLANCK ION CHANNEL MODEL

by

Silas A. Winnemoeller

The University of Wisconsin–Milwaukee, 2024
Under the Supervision of Professor Dexuan Xie

In this thesis, a finite element block modified backward Euler method is introduced to solve a one-dimensional Poisson-Nernst-Planck ion channel (1D PNPic) model. This model is defined as a system of time-dependent nonlinear partial differential equations, called Poisson-Nernst equations and Poisson equation, describing the transport of charged ionic species across a cell membrane via an ion channel pore. For an electrolyte with n ionic species, its numerical solution gives a prediction to n ionic concentration functions and an electrostatic potential function. However, solving the 1DPNPic model numerically is challenging due to the model's strong nonlinearity and numerical stability issues. To address the numerical stability issues, the traditional backward Euler implicit time scheme is often selected to solve the 1DPNPic model but it may be too costly to be practical in application since it has to solve a system of $n + 1$ strongly nonlinear partial differential equations at each time step. Hence, its modification becomes necessary to reduce its computing cost while retaining its numerical stability properly. In this thesis, the new method is constructed by semi-discretization and finite element techniques such that its each time iteration only involves calculation within two blocks with each block only containing two linear differential equations. Consequently, the new method can reduce the computing cost of the Euler scheme sharply. In this thesis, the new method is implemented as a software package in Python based on the finite element library from the FEniCS project. Numerical tests are then done for an electrolyte with two ionic species, demonstrating the convergence and high performance of the new method.

Keywords: Poisson-Nernst-Planck equations, ion channel modeling, finite element method, Euler methods, FEniCS library.

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGMENTS

I want to express my heartfelt gratitude to the members of my committee, Prof. Lei Wang, Prof. Vincent Larson, and Prof. Dexuan Xie.

I want to thank my parents for all their support and encouragement. Without them, I would have not been possible to pursue my education abroad and gather valuable experiences. Special appreciation goes to Prof. Dexuan Xie, whose mentorship and dedication have helped me throughout the entire process. His expertise, encouragement and guidance have enriched my research journey, offering insights and support at every step.

I am sincerely grateful for the opportunity to work under Prof. Dexuan Xie's guidance and for the invaluable lessons gained through his mentorship.

1 INTRODUCTION

Certain cells in the body have the ability to generate an electrical signal. These cells include neurons, muscle cells, touch receptor cells and more. To convert a chemical or mechanical message in an electrical signal, they use ion channel receptors. Every cell contains a different concentration in its membrane than in the extracellular environment. Because of the concentration difference, a small electrical potential is created across the membrane. When the conditions are right, certain channels can open to allow a rapid transport of the ions and these channels are called ion channels. In the body, many different type of ion channels exist which transport different ions. For example a NA-channel transport many Na^+ ions. In many applications, it is important to model the channels by differential equations [OA10, Ch. 4.3].

The Poisson-Nernst-Planck, in the following referred to as PNP, equations, are widely used to describe concentration and electrostatical potential for ion interactions in ion channels since it can capture several important macroscopic properties such as electric current, current-voltage relations, conductance rectification, and membrane potentials. These equations provide qualitative explanation and increasingly quantitative predictions for ion transport in many areas, such as semiconductor devices, nanofluidic systems, and biological systems. A one-dimensional PNP ion channel (1DPNPic) model gives a good prediction of the ion transport phenomenon for the systems stated above but also has its limitations. The number of equations to be solved and the number of diffusion coefficient profiles depend on the number of the different ion species in the system. Each ion species corresponds to one Nernst-Planck equation and one diffusion coefficient profile. Therefore, in a complex system with multiple ion species, the numerical solution of the 1DPNPic model can be computationally expensive.

The PNP ion channel (PNPic) model is a system of partial differential equations for one electrostatic potential function u and n ionic concentration functions c_i when a solvent

contains n ionic species. The concentration functions are only defined in the solvent region while the potential function is defined on the whole domain. In order to avoid this difficulty, the PNP ion channel model often considers only a piece of an ion channel pore as its domain so that u and each c_i are defined in the same domain.

For the vast majority, these PNPic models cannot be solved by analytical methods. Therefore, an approximation of the models can be constructed, most of the times, by different discretization methods. These discretization methods model a PNPic model with a system of numerical equations, which can be solved by using numerical methods. Several PNP ion channel models have been solved numerically by finite difference [ZQ11, p. 1f.] and finite element schemes [XC20]. The **Finite Element Method** (FEM) is a natural choice for the discretization of the PNP model in ion channel simulations. Therefore, the ion channel simulation using FEM can be implemented effectively. The simulation has a high difficulty because of the complicated geometry of proteins along with singularity of permanent charges, which lead to rapid changes of the electrostatic potential [WLZL21].

In this thesis, we focus on a 1DPNPic model with two ion species to reduce the complexity of the model. We construct a FEM numerical scheme to calculate a solution of the 1DPNPic model. Furthermore, we implement the scheme as a Python package based on a finite element library from the FEniCS project [LL16].

The thesis starts with reviewing the 1DPNPic model. Then this model will be reformulated as a variational problem to yield a system of nonlinear finite element equations. We then introduce a numerical scheme for solving this nonlinear finite element system.

In the previous work [Kor23], a numerical iterative scheme using the forward and backward Euler Methods is presented and implemented. This scheme produces $n + 1$ nonlinear variational equations. Thus, the computational cost can be very high. In our new scheme, we linearize the nonlinear scheme by using a block modified backward Euler method, resulting in a more effective scheme for solving the 1DPNPic model.

2 THE 1DPNPIC MODEL

The one-dimensional Poisson-Nernst-Planck ion channel (1DPNPic) model is used to predict ionic concentrations and an electrostatic potential function within an ion channel pore that allows n ionic species to go through [LB10].

Let $u(x,t)$ denote the electrostatic potential density function and $c_i(x,t)$ denote the ionic concentration function of species i for $i = 1, 2, ..., n$ with the spatial variable x on an interval, $[0, L]$, and time variable t for $t \in [0, T]$. Here, n is the number of species in an ionic solution, L denotes the ion channel length, and T denotes the time at which the partial derivative of c_i with respect to t is zero to reach the steady state in which both u and c_i become independent of time. We define u and c_i for $i = 1, 2, ..., n$ by a 1D PNPic model as follows:

$$\begin{cases} \dfrac{\partial c_i(x,t)}{\partial t} = \dfrac{\partial}{\partial x} D_i \left[Z_i c_i(x,t) \dfrac{\partial u}{\partial x} + \dfrac{\partial c_i(x,t)}{\partial x} \right], \quad 0 < x < L, \quad t > 0, i = 1, 2, \ldots, n, & (2.1) \\ -\epsilon_s \dfrac{\partial^2 u(x,t)}{\partial x^2} = \beta \displaystyle\sum_{i=1}^{n} Z_i c_i(x,t) + p(x), \quad 0 < x < L, \quad t > 0, & (2.2) \end{cases}$$

subject to the Dirichlet boundary value conditions:

$$u(0,t) = u_0(t),\ u(L,t) = u_L(t),\ c_i(0,t) = c_{i,0}(t),\ c_i(L,t) = c_{i,L}(t),\ 0 \le t \le T, \tag{2.3}$$

for $i = 1, 2, ..., n$, and the initial value conditions

$$c_i(x,0) = g_i(x), \qquad 0 \le x \le L, \qquad i = 1, 2, ..., n, \tag{2.4}$$

where p denotes a permanent charge function; D_i is a diffusion constant of species i; Z_i is the charge of species i; ϵ_s is the water permittivity constant; $u_0, u_L, c_{i,0}$ and $c_{i,L}$ are the boundary values, and β is a physical constant. Each equation of (2.1) is called a Nernst-Planck equation and the equation of (2.2) is referred to as a Poisson equation.

Applying (2.4) to (2.2), we get the boundary value problem for determining an initial

value function, $u(x,0)$, of u as follows:

$$\begin{cases} -\epsilon_s \frac{\partial^2 u(x,0)}{\partial x^2} = \beta \sum_{i=1}^{n} Z_i g_i(x) + p(x), \quad 0 < x < L \\ u(0,0) = u_0(0), \quad u(L,0) = u_L(0) \end{cases} \tag{2.5}$$

In calculation we can simply select g_i by

$$g_i(x) = c_i^b, \quad i = 1, 2, ..., n, \tag{2.6}$$

where c_i^b denotes a balk concentration of species i satisfying the neutrality condition:

$$\sum_{i=1}^{n} Z_i c_i^b = 0. \tag{2.7}$$

With this selection, we can find $u(x,0)$ in the expression

$$u(x,0) = \frac{u_b - u_a}{L} x + u_b. \tag{2.8}$$

Applying the neutrality condition (2.7) to the system (2.5), we have to solve:

$$\begin{cases} -\epsilon_s \frac{\partial^2 u(x,0)}{\partial x^2} = 0, \quad 0 < x < L, \\ u(0,0) = u_0(0), \quad u(L,0) = u_L(0). \end{cases} \tag{2.9}$$

In fact, we get that $u(x,0)$ is a linear function

$$u(x,0) = ax + b \tag{2.10}$$

Using the boundary value conditions, we get

$$b = u_0(0), \qquad a = \frac{u_L(0) - u_0(0)}{L}.$$

Thus, $u(x,0)$ is found in the expression

$$u(x,0) = \frac{u_L(0) - u_0(0)}{L} x + u_0(0). \tag{2.11}$$

3 FORMULATION OF THE VARIATIONAL PROBLEM

We follow the method of lines to discretize the 1DPNPic model in two steps. In Step 1, we do a semi-discretization in time variable t to reduce the 1DPNPic model to an ordinary differential equation (ODE) system [Sch12, p.10]. The details of Step 1 are presented in Section 3.1. In Step 2, we do a semi-discretization in space to discretize the ODE system by the finite element method to yield a full discretization of the 1DPNPic model. The details of Step 2 are presented in Section 3.2.

3.1 A semi-discretization in time

We start by defining the n^{th} time number t_n by

$$t_n = n\tau, \quad n = 0, 1, 2, ...$$

where τ denotes the time step size, which we define by

$$\tau = T/m,$$

for m being the number of time steps. We integrate (2.1) over the time interval $[t_n, t_{n+1}]$ to get

$$\int_{t_n}^{t_{n+1}} \frac{\partial c_i(x,t)}{\partial t} dt = D_i \int_{t_n}^{t_{n+1}} \frac{\partial}{\partial x}\left[Z_i c_i(x,t)\frac{\partial u(x,t)}{\partial x} + \frac{\partial c_i(x,t)}{\partial x}\right] dt, \quad i = 1, 2, ..., n. \quad (3.1)$$

By the right hand rectangular quadrature method,

$$\int_{t_n}^{t_{n+1}} y(t)dt \approx (t_{n+1} - t_n)y(t_{n+1}) = \tau y(t_{n+1}),$$

where $t_{n+1} - t_n = \tau$, we approximate the right hand side of (3.1) by

$$\int_{t_n}^{t_{n+1}} \frac{\partial}{\partial x}\left[Z_i c_i(x,t)\frac{\partial u(x,t)}{\partial x} + \frac{\partial c_i(x,t)}{\partial x}\right] dt \approx \tau \frac{\partial}{\partial x}\left[Z_i c_i(x,t_{n+1})\frac{\partial u(x,t_{n+1})}{\partial x} + \frac{\partial c_i(x,t_{n+1})}{\partial x}\right]. \quad (3.2)$$

Clearly,

$$\int_{t_n}^{t_{n+1}} \frac{\partial c_i(x,t)}{\partial t} dt = c_i(x,t_{n+1}) - c_i(x,t_n). \tag{3.3}$$

For clarity, we introduce the following notation,

$$c_i^n(x) = c_i(x,t_n), \quad u^n(x) = u(x,t_n), \quad \frac{dc_i^n(x)}{dx} = \frac{\partial c_i(x,t_n)}{\partial x}, \quad \frac{du^n(x)}{dx} = \frac{\partial u(x,t_n)}{\partial x}.$$

Thus, with (3.2) and (3.3), we get the semi-discretization of (3.1) with respect to time variable t by

$$c_i^{n+1}(x) - c_i^n(x) = \tau D_i \frac{d}{dx}\left[Z_i c_i^{n+1}(x) \frac{du^{n+1}(x)}{dx} + \frac{dc_i^{n+1}(x)}{dx} \right], \quad i = 1, 2, \ldots, n, \tag{3.4}$$

where $u^{n+1}(x)$ is defined by the Poisson boundary value problem

$$\begin{cases} -\epsilon_s \frac{d^2 u^{n+1}(x)}{dx^2} = \beta \sum_{i=1}^{n} Z_i c_i^{n+1}(x) + p(x), \quad 0 < x < L, \\ u^{n+1}(0) = u_0(t_{n+1}), \quad u^{n+1}(0) = u_L(t_{n+1}). \end{cases} \tag{3.5}$$

3.2 Semi-discretization in space

To construct the finite element method, we need to reformulate (3.4) and (3.5) as variational forms. Let $\Omega = (0, L)$ and $H^1(\Omega)$ be a Sobolev Space on Ω. A function u of $H^1(\Omega)$ is in the Lebesgue Space $L^2(\Omega)$ and its weak partial derivative $\frac{\partial u}{\partial x}$ is again in $L^2(\Omega)$ space [Kor23, Joh12]. Additionally, we define $H_0^1(\Omega)$ by

$$H_0^1 = \{u \in H^1(\Omega) | u = 0 \text{ on the boundary } \partial\Omega \, of \, \Omega\}. \tag{3.6}$$

For $\Omega = (0, L)$ and $v \in H_0^1(\Omega)$, we have $v(0) = v(L) = 0$.

By integration by parts,

$$\int_a^b v du = vu|_a^b - \int_a^b u dv, \tag{3.7}$$

we can simplify the following equation

$$\int_0^L \left[c_i^{n+1}(x) - c_i^n(x)\right] v(x) dx = \tau D_i \int_0^L \frac{d}{dx}\left[Z_i c_i^{n+1}(x) \frac{du^{t^{n+1}}(x)}{dx} + \frac{dc_i^{n+1}(x)}{dx} \right] v(x) dx \tag{3.8}$$

as follows:

$$\int_0^L \left[c_i^{n+1}(x) - c_i^n(x)\right] v dx = -\tau D_i \int_0^L \left[Z_i c_i^{n+1}(x) \frac{\partial u^{n+1}(x)}{\partial x} + \frac{\partial c_i^{n+1}(x)}{\partial x}\right] \frac{dv}{dx} dx. \tag{3.9}$$

And moving all the terms containing unknown functions c_i^{n+1} and u^{n+1} to the left hand side and the term containing the given function c_i^n to the right hand side will lead us to the following equation

$$\int_0^L c_i^{n+1}(x) v dx + \tau D_i \int_0^L \left[Z_i c_i^{n+1}(x) \frac{du^{n+1}(x)}{dx} + \frac{dc_i^{n+1}(x)}{dx}\right] \frac{dv}{dx} dx = \int_0^L c_i^n(x) v dx \tag{3.10}$$

for $i = 1, 2, \ldots, n$. Similarly, we can get the variational formulation of (3.5) for the potential function u^{n+1} as follows:

$$\epsilon_s \int_0^L \frac{dv}{dx} \frac{\partial u^{n+1}(x)}{\partial x} - \beta \sum_{i=1}^{n} Z_i \int_0^L c_i^{n+1}(x) v dx = \int_0^L p(x) v dx. \tag{3.11}$$

Hence, we combine (3.9) with (3.11 to obtain a system of $n+1$ variational equations as follows:

Find $c_i^{n+1} \in H^1(\Omega)$ and $u^{n+1} \in H^1(\Omega)$ satisfying the boundary value conditions

$$\begin{aligned} u^{n+1}(0) = u_0(t_{n+1}), \quad u^{n+1}(L) = u_L(t_{n+1}), \\ c_i^{n+1}(0) = c_{i,0}(t_{n+1}), \quad c_i^n(L) = c_{i,L}(t_{n+1}), \quad i = 1, 2, ..., n, \end{aligned} \tag{3.12}$$

such that both (3.9) and (3.11) holds for all $v \in H_0^1(\Omega)$.

3.3 Finite Element Method for Solving Variational System

We construct a linear finite element function space, V_h, as a subspace of $H^1(\Omega)$ based on a mesh partition of Ω with the mesh point $x_j = jh$ with the mesh size $h = \frac{L}{N+1}$ for $j = 0, 1, 2, \ldots, N, N+1$ such that for each $u_h \in V_h$, u_h is a piecewise linear function, and is continuous in $\Omega = (0, L)$. We then define a subspace of V_h by

$$V_{h,0} = \{v \in V_h | v(0) = 0, v(L) = 0\}.$$

We now obtain a variational system of the finite element method as follows:

Find $u^{k+1} \in V_h$ *and* $c_i^{k+1} \in V_h$ *for* $i = 1, 2, ..., n$ *satisfying the boundary conditions*

$$u^{k+1}(0) = u_0(t_{n+1}), \quad u^{k+1}(L) = u_L(t_{n+1}),$$
$$c_i^{k+1}(0) = c_{i,0}(t_{n+1}), \quad c_i^{k+1}(L) = c_{i,L}(t_{n+1}),$$

such that for all $v \in V_{h,0}$,

$$\int_0^L c_i^{k+1} v dx + \tau D_i \int_0^L \frac{dv}{dx}\left(Z_i c_i^{k+1}\frac{du^{k+1}}{dx} + \frac{dc_i^{k+1}}{dx}\right)dx = \int_0^L c_i^k v dx, \quad i = 1, 2, ..., n, \tag{3.13}$$

$$\epsilon_s \int_0^L \frac{dv}{dx}\frac{du^{k+1}}{dx}dx - \beta \sum_{i=1}^{n} Z_i \int_0^L c_i^{k+1} v dx = \int_0^L p v dx, \tag{3.14}$$

where $k = 0, 1, 2, \ldots, m$.

Consequently, we have obtained the finite element backward Euler time scheme for solving the 1DPNPic model, which is defined by the system of (3.13) and (3.14) for time step $k = 0, 1, 2, \ldots, m$. From this system, it can be seen that we have to solve a system of $n + 1$ nonlinear variational equations for the $n + 1$ unknowns $c_1^{k+1}, c_2^{k+1}, \ldots, c_n^{k+1}$ and u^{k+1} at each time step k. Hence, the backward Euler time scheme can be very costly in calculation.

4 FINITE ELEMENT BLOCK MODIFIED BACKWARD EULER SCHEME

To reduce the cost of the backward Euler time scheme, we construct a finite element block modified backward Euler scheme in this chapter. For simplicity, we consider the case with two ionic species ($n = 2$). That is, for each step k, we find three unknown functions, c_1^{k+1}, c_2^{k+1} and u^{k+1} when c_1^k, c_2^k and u^k are given.

Let $V_h \times V_h$ and $V_{h,0} \times V_{h,0}$ denote the product function spaces. We define $w_1, w_2 \in V_h \times V_h$ and $\underset{\sim}{v} \in V_{h,0} \times V_{h,0}$ by

$$w_1 = (u^{k+1,1}, c_1^{k+1}), \quad w_2 = (u^{k+1}, c_2^{k+1}), \quad \underset{\sim}{v} = (v_1, v_2),$$

where $u^{k+1,1}, c_1^{k+1}, u^{k+1}$, and c_2^{k+1} are functions from V_h, v_1 and v_2 are functions from $V_{h,0}$, which are used in the formulation of our block variational problems. We then define the bilinear forms $a_i(\cdot,\cdot)$ and linear form $b_i(\cdot)$ for $i = 1, 2$ by adding (3.13) and (3.14) to get

$$\begin{aligned}
a_1(w_1, \underset{\sim}{v}) &= \epsilon_s \int_0^L \frac{dv_1}{dx}\frac{du^{k+1,1}}{dx}dx - \beta Z_1 \int_0^L c_1^{k+1} v_1 dx \\
&\quad + \int_0^L c_1^{k+1} v_2 dx + \tau D_1 \int_0^L \frac{dv_2}{dx}\left[Z_1 c_1^k \frac{du^{k+1,1}}{dx} + \frac{dc_1^{k+1}}{dx}\right] dx, \qquad (4.1)\\
b_1(\underset{\sim}{v}) &= \beta Z_2 \int_0^L c_2^k v_1 dx + \int_0^L p(x) v_1 dx + \int_0^L c_1^k v_2 dx, \qquad (4.2)\\
a_2(w_2, \underset{\sim}{v}) &= \epsilon_s \int_0^L \frac{dv_1}{dx}\frac{du^{k+1}}{dx}dx - \beta Z_2 \int_0^L c_2^{k+1} v_1 dx \\
&\quad + \int_0^L c_2^{k+1} v_2 dx + \tau D_2 \int_0^L \frac{dv_2}{dx}\left[Z_2 c_2^k \frac{du^{k+1}}{dx} + \frac{dc_2^{k+1}}{dx}\right] dx, \qquad (4.3)\\
b_2(\underset{\sim}{v}) &= \beta Z_1 \int_0^L c_1^{k+1} v_1 dx + \int_0^L p(x) v_1 dx + \int_0^L c_2^k v_2 dx, \qquad (4.4)
\end{aligned}$$

In Block 1, we define a linear variational problem as follows:

Find $w_1 = (u^{k+1,1}, c_1^{k+1}) \in V_h \times V_h$ satisfying the boundary value conditions:

$$c_1^{k+1}(0) = c_{1,0}, \quad c_1^{k+1}(L) = c_{1,L}, \quad u^{k+1,1} = u_0, \quad u^{k+1,1}(L) = u_L \qquad (4.5)$$

such that

$$a_1(w_1, \underset{\sim}{v}) = b_1(\underset{\sim}{v}) \quad \forall \underset{\sim}{v} \in V_{h,0} \times V_{h,0}. \tag{4.6}$$

In Block 2, we define a linear variational problem as follows: Find $w_2 = (u^{k+1}, c_2^{k+1}) \in V_h \times V_h$ satisfying the boundary value conditions:

$$c_2^{k+1}(0) = c_{1,0}, \quad c_2^{k+1}(L) = c_{2,L}, \quad u^{k+1} = u_0, \quad u^{k+1}(L) = u_L \tag{4.7}$$

such that

$$a_2(w_2, \underset{\sim}{v}) = b_2(\underset{\sim}{v}) \quad \forall \underset{\sim}{v} \in V_{h,0} \times V_{h,0}. \tag{4.8}$$

In implementation, we use the following termination rule for the time iterations: We stop the time iteration at the $k+1$th time step if

$$||c_i^k - c_i^{k+1}|| < \epsilon \quad , i = 1, 2, \quad ||u^k - u^{k+1}|| < \epsilon \tag{4.9}$$

We now present our time scheme in Algorithm 1.

Algorithm 1: Block Modified Backward Euler Scheme

```
Input the initial values c_1^0, c_1^0, u^0.
Set k = 0, m = 5000 and ε = 10^{-4} (by default).
repeat
    Find c_1^{k+1} and u^{k+1,1} by solving the linear problem in Block 1 given in (4.6).
    Find c_2^{k+1}, u^{k+1} by solving the linear problem in Block 2 given in (4.8).
    Increase k by 1
until ||c_i^k − c_i^{k+1}|| < ε, i = 1, 2 and ||u^k − u^{k+1}|| < ε or k > m;
```

4.1 Implementation with FEniCS

The scheme is implemented using DOLFIN, a popular open-source finite element library from the FEniCS project. DOLFIN serves as the computational high-performance C++ backend of FEniCS, facilitating the implementation of various data structures such as meshes, function spaces, and functions, as well as the computation of intensive algorithms like finite element assembly and interfaces to linear algebra solvers and data structures [LL16]

Due to the FEniCS project's reliance on a Linux environment, the Windows Subsystem for Linux (WSL) was installed for the installation process. Anaconda was then utilized on the WSL to set up a Python environment for installing the FEniCS project.

The implemented solver was executed for different test cases. Here we set the permanent charge function p to be a constant (1), $n = 2$ for a solution of table salt NaCl with a diffusion constant $D_1 = 0.133$ for Na^+ ions, $D_2 = 0.203$ for Cl^- ions, $\epsilon_s = 78$, $\beta \approx 4.2414$, and the spatial partition number N = 256.

We did initial tests on the Python program for the scheme. In the tests, we set the channel Length to $L = 40$ to get a mesh size of $h = 0.155642$ for a uniform interval mesh, where all adjacent mesh points have the same distance and the time step size $\tau = 0.5$. The boundary conditions were defined as follows: For $k = 0, 1, 2, ...$

$$u_0^k(0) = -1, \quad u_0^k(L) = 1,$$
$$c_1^k(0) = 0.1, \quad c_1^k(L) = 0.1$$
$$c_2^k(0) = 0.1, \quad c_2^k(L) = 0.1,$$

such that we get an expression for u_0 as follows

$$u_0(x) = 0.05x - 1.$$

We set the tolerance to 10^{-4} and run the algorithm for that test case. We reach a maximum error of 9.995873e-04 after a number of 645 iterations. Thus we get the time length to reach the steady state $T = 323$. The time we needed for the calculation was approximately 3 seconds.

We plot the error, which is defined by the difference of each function between two timesteps in *Figure 1.* In this figure, we applied a logarithmic scale to the error axis to accommodate all three function errors in one plot. Here, we see for the potential function u and the concentration function c_1, that the error is strictly decreasing. For the concentration function c_2 the error starts to slightly increase at the beginning and then decreases, but stays under the tolerance 10^{-4}. Therefore, the iteration stopped when the error of the potential function

u fell under the tolerance. We observed a significant reduction in the error of the potential function u after the initial time step, as this iteration depends the most on the quality of the initial value.

The finite element block modified backward Euler solution is presented in *Figure 2*. This figure shows a manually generated surface plot from the results in each timestep k,

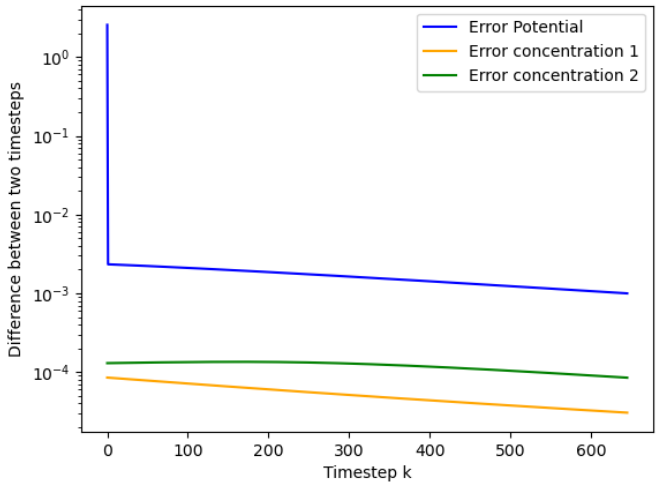
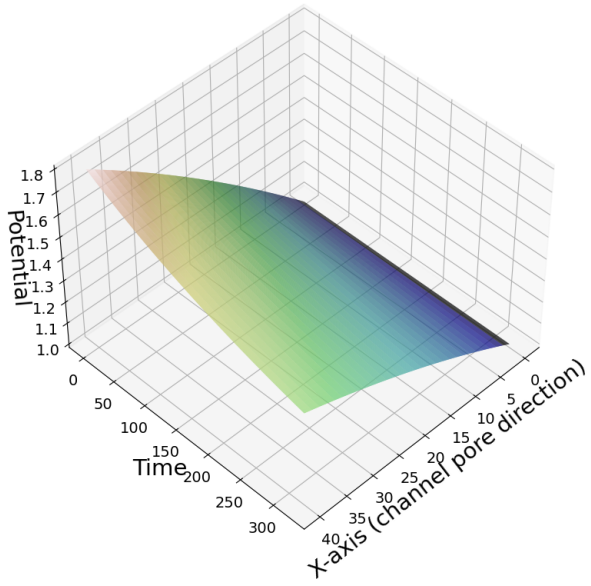
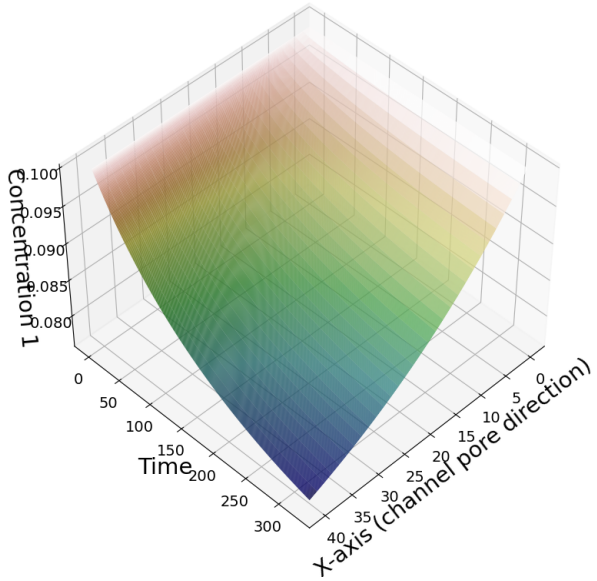


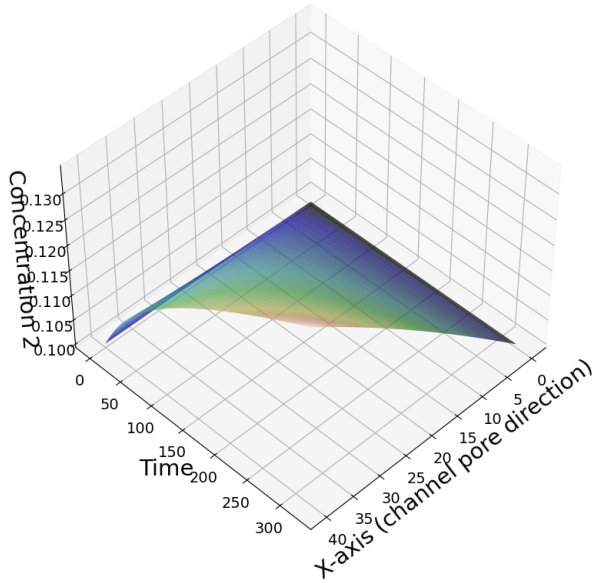
Figure 1: Solution difference between two timesteps as a function of timestep k.



(a) Electrostatic potential function in Test 1.



(b) Concentration function of the first ionic species in Test 1.



(c) Concentration function of the second ionic species in Test 1.

Figure 2: Finite element solution (u, c_1, c_2) of the 1DPNPic model in Test 1 with $\tau = 0.5$.

5 CONCLUSION

We introduced a new numerical scheme, the finite element block modified backward Euler method, to solve a one-dimensional Poisson-Nernst-Planck model. The scheme modifies the backward Euler method with $n + 1$ nonlinear variational equations to a linear scheme by using a block iterative method. We described the scheme for n ionic species.

In our test case, we considered the case with two ionic species (i.e., $n = 2$). We used the FEniCS project library for the implementation of the new scheme. To display the results, we constructed 3D surface plots and 2D curve plots for the numerical solutions.

In the future, we will continue to explore the new scheme by theoretically analyzing the convergence and stability of the new scheme and by doing more numerical tests. To validate the new scheme, we will create an analytical solution of the 1DPNPic model artificially. We then will do validation numerical tests by comparing the analytical solution with the numerical solution generated by the new scheme. Moreover, we will use the analytical solution to verify the convergence rate and numerical stability of the new scheme.

REFERENCES

[Joh12] C. Johnson. *Numerical Solution of Partial Differential Equations by the Finite Element Method*. Courier Corporation, 2012. URL: https://books.google.com/books?id=PYXjyoqy5qMC.

[Kor23] Michel Stanislas Korfhage. Numerical study of a one-dimensional Poisson-Nernst-Planck ion channel model by finite element backward and forward Euler methods. Master's Thesis, University of Wisconsin-Milwaukee, 2023.

[LB10] McCammon JA Zhou YC. Lu B, Holst MJ. *Poisson-Nernst-Planck Equations for Simulating Biomolecular Diffusion-Reaction Processes I: Finite Element Solutions.* J Comput Phys., 2010. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2922884/.

[LL16] H. Langtangen and A Logg. *Solving PDEs in Python, The FEniCS Tutorial I*. Springer International Publishing, 2016. URL: https://fenicsproject.org/, doi:10.1007/978-3-319-52462-7.

[OA10] C. O'Connor and J. Adams. *Essentials of Cell Biology*. Cambridge, MA: NPG Education,, 2010. URL: https://www.nature.com/scitable/topicpage/ion-channel-14047658/.

[Sch12] W.E. Schiesser. *The Numerical Method of Lines: Integration of Partial Differential Equations*. Elsevier Science, 2012. URL: https://books.google.com/books?id=2YDNCgAAQBAJ.

[WLZL21] Qin Wang, Hongliang Li, Linbo Zhang, and Benzhuo Lu. A stabilized finite element method for the Poisson–Nernst–Planck equations in three-dimensional ion channel simulations. *Applied Mathematics Letters*,

111:106652, 2021. URL: https://www.sciencedirect.com/science/article/pii/S0893965920303189, doi:10.1016/j.aml.2020.106652.

[XC20] D. Xie and Z Chao. A finite element iterative solver for a pnp ion channel model with neumann boundary condition and membrane surface charge. *Journal of Computational Physics*, 2020. URL: https://www.sciencedirect.com/science/article/abs/pii/S0021999120306896?via%3Dihub.

[ZQ11] Wei GW. Zheng Q. *Poisson–Boltzmann–Nernst–Planck model.* J Chem Phys., 2011. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3122111/.

Appendices

A Hardware

Acer Nitro 5 an515-57

Microprocessor	Intel® Core™ i5-11400H (2.7 GHz base frequency, up to 3.9 GHz with Intel® Turbo Boost Technology, 20 MB cache, 6 cores)
Memory, standard	16 GB DDR4-2400 SDRAM (2 x 8 GB)
Hard drive	512 GB PCIe® NVMe™ M.2 SSD

B Software

OS	Windows 11 — Windows Subsystem for Linux
OS-kernel	WSL (Ubuntu) Kernel Version 5.15.146.1-2
Python&Anaconda	Python 3.11.7 — Anaconda 24.1.2