Numerical Exploration of Electrical Signals in Neurons

TFY4235 - Exam

ID - -

February 3, 2025

Abstract*—A simple network model showed that electrical charges diffuse to nearby nodes, creating a uniform distribution of charges. The underlying mathematics of the transformation matrix gave insight into the behavior and properties of the networks in the system. Solving the emerging systems was most effectively executed by the NumPy linalg library while the iterative Lanczos method and other methods for large sparse matrices will scale better. The linear cable equation was solved with three different numerical schemes, and only with the addition of ion channels did the model mimic real action potential. The implicit Euler scheme was found to give effective and accurate results and the Crank-Nicolson scheme also performed well. The model gave behavior close to real electrical impulses and it was hypothesized that the addition of a chlorine ion channel would better model a real action potential.*

1. Introduction

Electrical signals are a foundational part of human physiology, all the way from making the heart beat to complex behavior in the brain. With the rise of artificial intelligence which tries to mimic learning with neural networks, understanding the complexity of the brain has become an even more important task. This paper seeks to explore some of the foundational physiological phenomena of neurons and how electrical signals propagate along cell membranes [5].

Firstly, a coarse-grained model of neurons where each neuron is modeled as a node connected to other nodes in a network is studied. Such a model can give insight into the high-level behavior of networks of neurons without the need to implement detailed biophysical models. How signals propagate and the equilibrium states of the networks will be investigated, in addition to a more mathematical approach to consider the underlying properties of the numerical systems. As always when utilizing numerical models, the efficiency and accuracy of the methods are important and this paper will analyze and compare multiple numerical solvers.

A more detailed approach to electrical impulses is to model the biophysics of a cell membrane. Such a membrane can be modeled as an electrical circuit giving rise to partial differential equations (PDE) [5], which can be solved with numerical schemes. The first approach uses the linear cable equation to model membrane depolarization, which models how electrical signals evolve along a cable. This model is later extended to include ion channels and it is explored how both passive- and voltage-gated ion channels influence the propagation of electrical impulses. The PDEs for the different models are solved numerically with three different schemes, Euler explicit, Euler implicit, and Crank-Nicolson. These schemes depend on discretization parameters to give correct approximations, thus the stability and accuracy of the numerical methods are studied.

2. Methodology

2.1. Systolic network model

As a simple model for a neural network, one can consider N nodes connected in a systolic network. These nodes can exchange information in the form of charges synchronously by links at each time step controlled by a clock. Each node i thus has a charge Q_i which can be exchanged with connected nodes. To describe the state of the whole network at a certain time t we use the state vector $V(t) = (Q_1, Q_2, ..., Q_N)$ [5]. At each time step dt, each node will transfer its charge equally to its connected nodes. This evolution can be described by the transformation matrix T [5], such that

$$V(t + dt) = T \cdot V(t). \tag{1}$$

A network of $N = 21$ where each node is connected to the two previous- and next nodes has the T-matrix represented in Figure 1. This matrix has periodic boundary conditions meaning that the nodes connect in a full circle and the resulting matrix is symmetric.

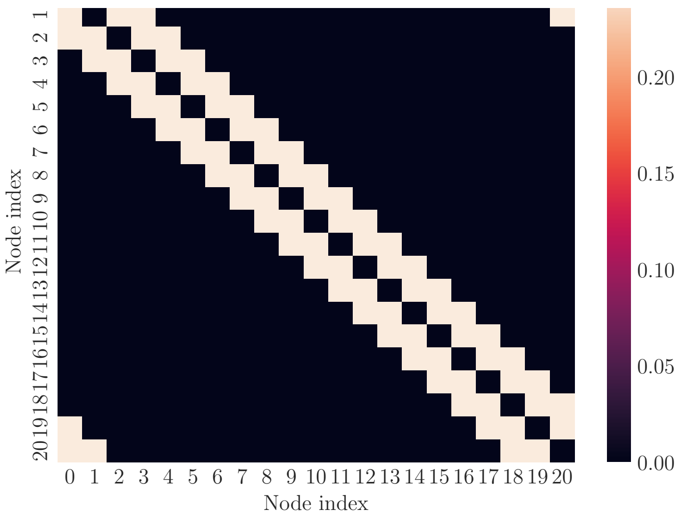


Figure 1. *Transformation matrix T for $N = 21$ nodes where each node is connected to the 4 nearest neighbors. The matrix is normalized such that each row sums to one, creating a stochastic matrix.*

One can also model disjoint networks, for example, a network with two separate and fully connected circles of $N = 11$ and $N = 10$, which has the T-matrix presented in Figure 11 in Appendix A. The eigenvectors and eigenvalues of such stochastic matrices (ergodic Markov chains) have as many eigenvalues equal to one as there are disjoint networks in the system. The remaining eigenvalues will be lower than one [5].

To find the lowest and highest eigenvalues in magnitude and corresponding eigenvectors, one can use the Lanczos method which is an iterative method for finding the extremal eigenvalues and eigenvectors of a matrix [6], [7]. This method works especially great for sparse matrices, which T is in our case, however, it is mostly used for larger matrices ($N \geq 10^5$). This project uses *scipy.sparse.linalg.eigsh* to find eigenvalues and vectors, which wraps *ARPACKs* Lanczos method [3]. One can also solve the system backward in time, which means $V(t)$ is the unknown variable in Eq. (1) and one thus has a linear system of equations. These equations are solved with three different solvers from the *NumPy* [2] and *SciPy* [3] python libraries; *np.linalg.solve* , *scipy.linalg.solve* and *scipy.sparse.linalg.spsolve*. In the normal, forward case the normal *NumPy* dot product is used to solve the right-hand side of Eq. (1) and evolve the system.

2.2. Modeling of electrical impulses

2.2.1. Membrane potentials. Neurons are cells that are specifically adapted for the transmission of electrical signals. One neuron can receive signals from many other neurons or cells, which are conducted inwards to the neuron cell body via dendrites. Both excitatory and inhibitory impulses change the membrane potential at the start of the cell axon, and when the membrane potential reaches a threshold value, an electrical impulse will be sent along the axon to the target cell, for example, other neurons or muscle cells [1]. These electrical impulses are transported along the cell membrane due to the membrane potential being depolarized and repolarized quickly. For this to happen, the potential across the membrane (V_{mem}) is normally at a resting potential (for example -60mV in a typical giant squid axon [5]) kept up by a mix between the concentration and charge of many ions, inside and outside the cell membrane, called the electrochemical gradient. The membrane is usually not permeable for ions, such that they need to go through specific channels to enter or leave the cell, and they can thus be regulated. Some channels are passive and let ions diffuse with the electrochemical gradient when they are open, and other channels are active, pumping the ions against the electrochemical gradient [1]. The Nernst potential describes the equilibrium between ion concentration and the voltage gradient across the membrane, such that when a passive ion channel is open, the ion will diffuse across the cell membrane toward the Nernst potential [5]. As an example, if the Nernst potential for Cl^--ions is $V_{Cl^-}^{Nernst} = -62$ mV with certain intra- and extracellular concentrations and $V_{mem} = -60$ mV, the chlorine ions will flow into the cell to make the membrane potential more negative, towards -62 mV.

Some ion channels are voltage regulated, meaning that they turn on when V_{mem} goes above a threshold value V^*. In this way, an incoming electrical impulse can activate an ion channel which in turn depolarizes the membrane and activates the next ion channel, sending the impulse further down the membrane. Ion channels will only stay activated for a certain amount of time before they deactivate, which together with other types of ion channels (including both active, passive, and leaky channels) repolarizes the membrane potential to its resting potential. This creates a propagating electrical wave called an action potential (AP) [1].

2.2.2. Modeling the membrane potential. The electrical properties of the cell membrane can be modeled as a collection of resistors, batteries, and capacitors [5]. A small patch of the cell membrane can be modeled as an outer and inner capacitor plate with capacitance C. The ion channels also connect the plates in parallel, where each channel is represented as a battery with driving force V_{ion}^{Nernst} followed in series by a resistance R_{ion}. Studying the properties of such a wire gives rise to the linear cable equation [5]:

$$\lambda^2 \frac{\partial^2 V(x,t)}{\partial x^2} - \tau \frac{\partial V(x,t)}{\partial t} = V(x,t). \tag{2}$$

This equation is thus used to represent the membrane depolarization $V(x,t)$ at position x and time t along a "wire" of cell membrane. In Eq. (2), λ and τ are the axon's space and time constants, which incorporate the wire's electrical properties.

The linear cable equation (2) only describes electrical flow along a cable and does not incorporate ion channels in the model. To incorporate Na^+ and K^- ion channels, Eq. (2) can be extended to

$$\begin{aligned}\lambda^2 \frac{\partial^2 V(x,t)}{\partial x^2} - \tau \frac{\partial V(x,t)}{\partial t} &= \frac{g_{Na}(V(x,t))}{g_K}[V(x,t) - V_{Na}^{Nernst}] \\ &\quad + [V(x,t) - V_K^{Nernst}],\end{aligned} \tag{3}$$

where g is the permeability of the specified ions [5]. The K^+ ion channel is represented as a leaky channel with constant permeability $g_K = 5.0\Omega^{-1}m^{-2}$, while the voltage-gated sodium ion channels permeability is modeled according to

$$g_{Na}(V) = \left(\frac{100}{1 + e^{\gamma(V^* - V)}} + \frac{1}{5}\right) \quad [\Omega^{-1}m^{-2}], \tag{4}$$

where γ is a constant.

2.2.3. Numerical schemes. To simulate the evolution of the membrane depolarization $V(x,t)$, Eq.(2) can be discretized and evolved using a finite difference numerical scheme. One such scheme is the forward Euler (explicit Euler) scheme, wich uses a forward difference approximation to approximate the 1st order derivatives in time and a central difference approximation for 2nd order derivatives in space [4], [7] which discretizes the linear cable equation (2) into

$$\lambda^2 \frac{V_{i+1}^n - 2V_i^n + V_{i-1}^n}{(\Delta x)^2} - \tau \frac{V_i^{n+1} - V_i^n}{\Delta t} = V_i^n. \tag{5}$$

In Eq. (5) i denotes the discretized spatial position and n the temporal component with interval lengths Δx and Δt respectively. By defining

$$\alpha = \frac{\lambda^2 \Delta t}{\tau (\Delta x)^2} \quad \text{and} \quad \beta = \frac{\Delta t}{\tau}, \tag{6}$$

Eq. (5) can be rearranged into

$$V_i^{n+1} = \alpha V_{i-1}^n + (1 - 2\alpha - \beta)V_i^n + \alpha V_{i+1}^n, \tag{7}$$

which is the forward Euler scheme. On the other hand, by using backward difference approximations one reaches the backward Euler (Implicit Euler) [4] scheme which reads

$$-\alpha V_{i-1}^{n+1} + (1 + 2\alpha)V_i^{n+1} - \alpha V_{i+1}^{n+1} = (1 - \beta)V_i^n. \tag{8}$$

One can combine the explicit and implicit schemes in Eq. (7) and (8), taking half of each spatial differential approximation, and thus obtain the Crank-Nicolson scheme [6]:

$$\begin{aligned}-\frac{\alpha}{2}V_{i-1}^{n+1} + (1+\alpha)V_i^{n+1} - \frac{\alpha}{2}V_{i+1}^{n+1} \\ = \frac{\alpha}{2}V_{i-1}^n + (1 - \alpha - \beta)V_i^n + \frac{\alpha}{2}V_{i+1}^n.\end{aligned} \tag{9}$$

All three schemes give rise to linear systems of equations on the form $\mathbf{A}\mathbf{V}^{n+1} = \mathbf{B}\mathbf{V}^{n}$, where $\mathbf{A}$ and $\mathbf{B}$ are tridiagonal matrices, and can be solved with the *numpy.linalg.solve* function.

A bounded system with reflective boundaries (Neumann boundary conditions) can be imposed on the discretized systems by introducing two fictive points $V_{-1}^n = V_1^n$ and $V_{N+2}^n = V_N^n$ [4]. The schemes written out at the boundary points $i = 0$ and $i = N+1$ will thus multiply V_{i+1} and V_{i-1} with 2 respectively, for both time steps n and $n+1$. For the unbounded problem the analytical solution is given by [5]

$$V(x,t) = \frac{\tilde{V}_0}{\sqrt{4\pi(\lambda^2/\tau)t}} \exp\left[\frac{-(x-x_0)^2}{4(\lambda^2/\tau)t} - \frac{t}{\tau}\right], \tag{10}$$

where the normalization term $\tilde{V}_0$ has units Vm.

To implement Eq. (3) with the Crank-Nicolson scheme, only the $-\beta V_i^n$ term on the right-hand side in Eq. (9) changes. With only one sodium ion channel in $x = x_0$, only activated when $v(x_0,t) > V^*$, g_{Na} is zero in all other points, thus giving the matrix equation $\mathbf{A}\mathbf{V}^{n+1} = \tilde{\mathbf{B}}\mathbf{V}^n - \beta(V^n - V_K^{Nernst})$ for all points, and additionally $-\beta$ times the whole g_{Na} term in Eq. (3) for $x = x_0$. Applying a potential difference V_{appl} at x_0 can be described by the initial condition [5]

$$V(x,0) = (V_{appl} - V_{mem}) \exp\left(-\frac{(x-x_0)^2}{2\lambda^2}\right) + V_{mem}. \tag{11}$$

3. Results

3.1. Networks

With a fully connected system with T as in Figure 1, starting from a random stochastic state vector $V(0)$, the system's evolution is plotted for multiple time steps in Figure 2, using Eq. (1) to evolve the system.

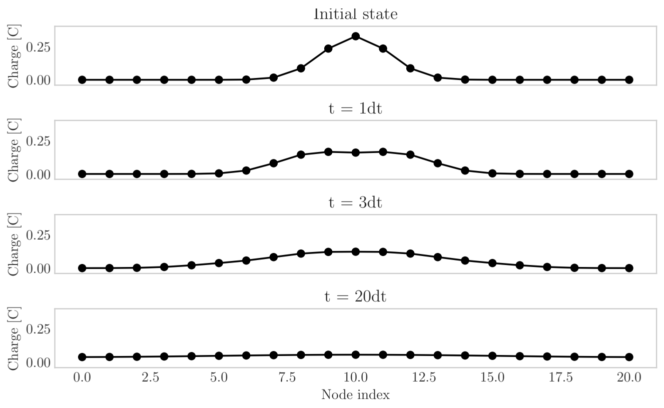


Figure 2. *The evolution of the initial state vector after 1, 3, and 20 time steps. Each dot represents one node with node index at the x-axis and the node charge on the y-axis.*

The largest in magnitude eigenvalue of the transformation matrix T was found to be 1.0000 with Lanczos iterative method, and the lowest being -0.0167 with degeneracy 2. The eigenvalues obtained with *numpy.linalg.eigh* gave the exact same answer for four decimals. The corresponding eigenvectors of these eigenvalues were compared between the Lanczos and Numpy methods by calculating the angle between them. The results were 0.00 radians for the unit eigenvectors and 2.69 radians for both the degenerate eigenvectors with the lowest eigenvalue. There was only one eigenvalue with value 1.0 and all the other eigenvalues and eigenvectors had degeneracy 2 (two of the same eigenvalue but different eigenvectors). The Lanczos method used 3.2807 ms to find the three lowest and highest eigenvalues and vectors while the NumPy method used 0.2430 ms to find all.

The same T matrix was used to evolve four different systems with 100 time steps, presented in Figure 3. The three random initial state vectors and the Gaussian wave state vector were normalized to sum to one, such that they are stochastic vectors. The resulting state vector in all cases after 100 time steps was parallel to the unit eigenvector of the transformation matrix T.

A $N = 21$ node system consisting of two disjoint networks, both connected in circles with 11 and 10 nodes each, give the transformation matrix presented in Figure 11 in Appendix A. Using the Lanczos method to calculate eigenvalues and vectors, there was found to be 2 eigenvalues of value 1. The Lanczos method used 3.0672 ms to find the three lowest and largest eigenvalues, while *numpy.linalg.eigh* used 0.8965 ms. An evolution of a random state vector with this transformation matrix can be found in Figure 3.

Finally the initial state vector give by $V(t) = (1, 1/2, ..., 1/i, ..., 1/N)$ were solved for $t - dt$ and $t - 5dt$, using the one-network transformation matrix in Figure 1. Using *np.linalg.solve* , *scipy.linalg.solve* and *scipy.sparse.linalg.spsolve* to solve the linear equations, the charge distributions are presented in Figure 4. The solvers were timed over 100 time steps to measure efficiency, and the average time and standard deviation (std) are presented in Table 1.

Solver	**Time per time step ($\pm$ std)**
np.linalg.solve	0.02515 ± 0.0066 ms
scipy.linalg.solve	0.12706 ± 0.0934 ms
scipy.sparse.linalg.spsolve	0.16388 ± 0.1057 ms

Table 1. *Average time per time step for different solvers.*

3.2. Electrical impulses

Using the analytical solution for an unbounded system in Eq. (10) at time $t = 1$ s as the initial condition for the numerical schemes, one can compare the solutions when using the same normalization constant $\tilde{V}_0 = 1$ Vm. Forward- and backward Euler, and Crank-Nicolson schemes were used to evolve the membrane depolarization with the linear cable equation, using $\lambda = 1.0$ m and $\tau = 1.0$ s. The results are presented in Figure 5. The error of the numerical schemes presented over the same timeframe is calculated as absolute-value norm $||V_{analytical} - V_{scheme}||$ at each time point and presented in Figure 6.

The unbounded analytical solution in Eq. (10) at the boundary point $x = a$ was found to be larger than a threshold $V(a,t) > 10^{-10}$ V after $t = 3.115$ s (for reference, $t = 2.115$ s in Figure 5). A further study of the numerical stability of the numerical schemes for different α values (Eq. (6) are presented in Appendix B, Figure 13. It was found that the explicit Euler scheme becomes unstable for $\alpha \geq 0.65$ values and the Crank-Nicolson scheme becomes unstable at $\alpha \geq 28.50$. The implicit Euler scheme remains stable for all α values but becomes very inaccurate at $\alpha \geq 233.33$.

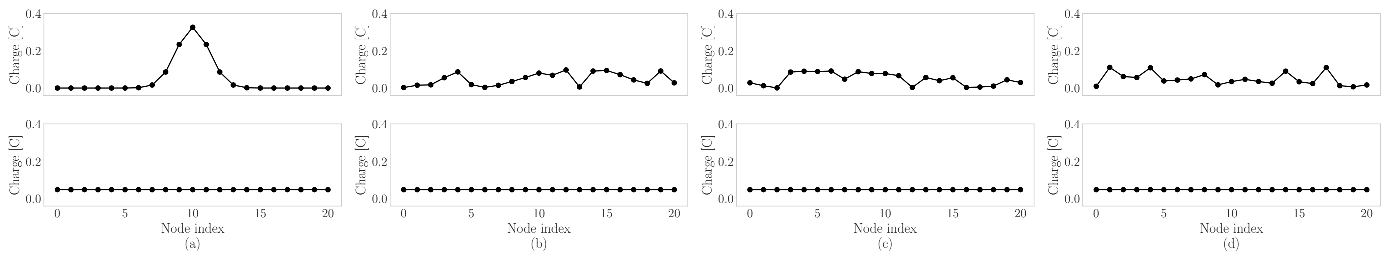


Figure 3. *The evolution of one initial Gaussian wave state vector (a) and three random states (b), (c), and (d), all normalized to be stochastic vectors and plotted after 100 time steps. Each dot represents one node with node index at the x-axis and the node charge on the y-axis.*

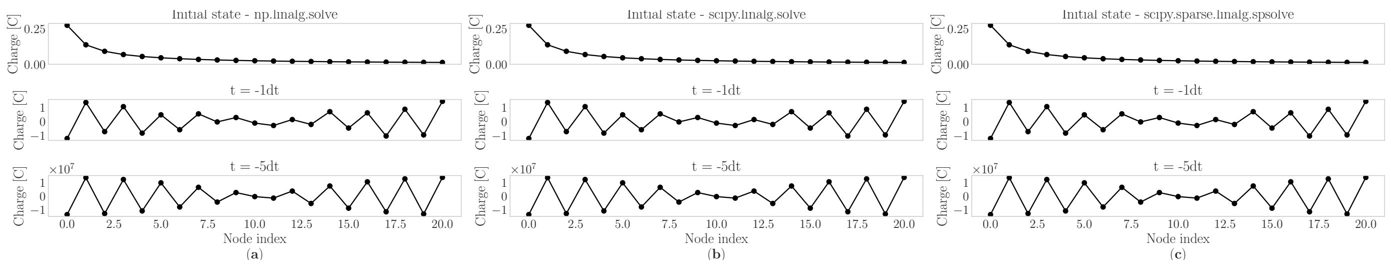


Figure 4. *The evolution of one initial state vector after $t - dt$ and $t - 5dt$, solved with (a) $np.linalg.solve$, (b) $scipy.linalg.solve$ and (c) $scipy.sparse.linalg.spsolve$. Each dot represents one node with node index at the x-axis and the node charge on the y-axis.*

Introducing the ion channels in Eq (3) gave the evolution in Figure 7 when using the initial condition in Eq. (11) and the following parameters: $\lambda = 0.18$ mm, $\tau = 2.0$ ms, $\gamma = 0.5$ mV^{-1}, $V^* = -40$ mV, $V_{Na}^{Nernst} = 56$ mV, $V_K^{Nernst} = -76$ mV, $V_{mem} = -70$ mV, and $V_{appl} = -50$ mV. The effect of the strength of V_{appl} is further investigated in Figure 8 for different $V_{appl} > V^*$.

An experiment is performed where the sodium ion channel is moved 0.25 mm to the right of the initial impulse at x_0 and the resulting evolution of the membrane depolarization is presented in Figure 9. For the sodium channel to be activated, the membrane potential must be greater than the threshold at the point where the sodium channel resides for the initially applied potential. On a discretized grid where the sodium channel is located at $x_{Na} = x_0 + 0.2467$ mm, the threshold is found analytically to be $V_{appl} > 6.72$ mV by solving Eq. (11). This result is further confirmed to be in the range between 6.70 mV and 6.80 mV in Figure 14 in Appendix B.

Following the experiment with a moved sodium channel (Figure 9), a patch clamp experiment [1] to measure the membrane potential over time was carried out. The resulting membrane potential over time is presented in Figure 10 for the position of the initially applied potential x_0 and the position of the sodium ion channel x_{Na}.

4. Discussion

4.1. Network model

The evolution of an initial Gaussian wave in a 21-node network can be seen in Figure 2. The charge can be described as diffusing outwards to the other nodes as time steps are taken, gradually distributing the charge evenly amongst all the nodes. The transformation matrix makes the nodes distribute their charge equally to their four nearest neighbors while receiving a quarter of the charge of each of the four neighbors. This means that if one node has a higher charge than its neighbors, the net flux of charge is negative and the node charge decreases, which is the behavior we see in the result. A higher difference in charge also gives a faster decrease, which is observed as the top peak goes down more quickly than the surrounding nodes.

The eigenvalues and vectors were found using the iterative method Lanczos and all eigenvalues were the same as the result from the $numpy.linalg.eigh$ library up to 8 decimals. The unit eigenvectors of both solvers were found to be parallel, which is expected as they are unit vectors. The lowest in magnitude eigenvalue were degenerate with two eigenvectors having the same eigenvalue. Between the two solvers, both angles between these eigenvectors were found to be 2.69 radians. However, both methods should give back parallel eigenvectors. This might be due to the Lanczos method not returning eigenvectors in the same order as NumPy or some other small error in the code implementation. This could however be further analyzed by testing the eigenvectors as they should give $TV = \lambda V$. The eigenvalues were further found to be degenerate with two states with the same eigenvalue for all eigenvectors except the unit eigenvector. In the disjoint network, it was found to have two eigenvalues equal to 1, one for each network, which is consistent with the literature [5]. The NumPy method uses less time than the Lanzos method to find the eigenvalues and vectors in both networks, but the Lanczos method seemed to be equally fast in both cases, while NumPy used almost four times as much time in the disjoint network. Additionally, a square matrix with 21 rows is a very small matrix and the Lanczos method is usually used for larger matrices and is expected to scale better than the NumPy method [7]. With this in mind, for larger networks the Lanczos method might work better but for small system the NumPy method is more efficient.

When initiating multiple random states in addition to a Gaussian initial state, all systems go to the same unit vector after 100 time steps, which is observed in Figure 3. Since the initial state vector and the transformation matrix are stochastic, the resulting state vector is also stochastic [5] and all the initial states end up with the same value at each node. The

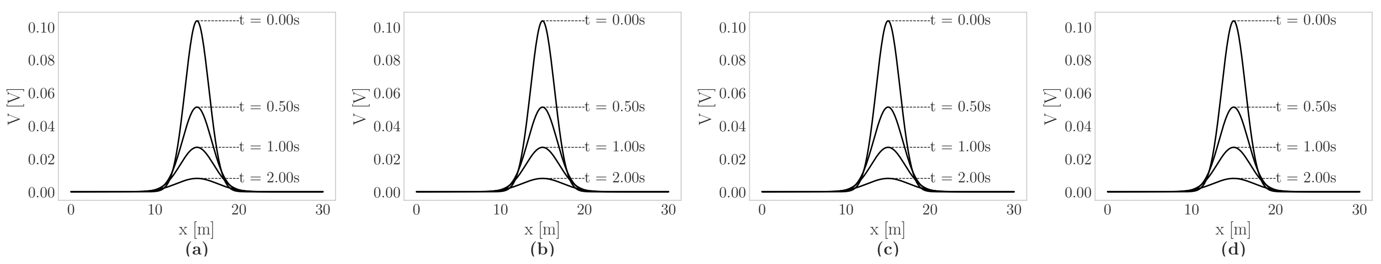


Figure 5. *The evolution of the membrane depolarization V with (a) the unbounded analytical solution, (b) implicit Euler scheme, (c) explicit Euler scheme, and (d) Crank-Nicolson scheme for different times t. $t = 0$ s is defined as the start time for the schemes which is $t' = 1$ s in the analytical solution in Eq. (10).*

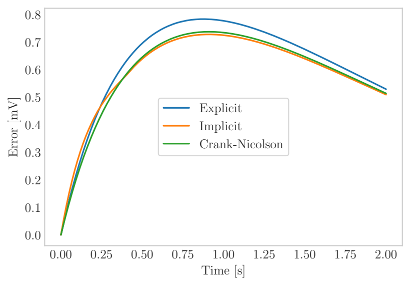


Figure 6. *The error for the numerical schemes as the absolute-value norm $||V_{analytical} - V_{scheme}||$ at each time point.*

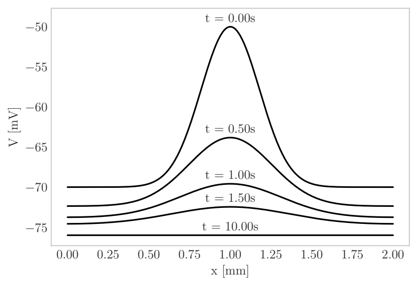


Figure 7. *Evolution of the membrane potential V over time for an initial applied potential at $t = 0$ s.*

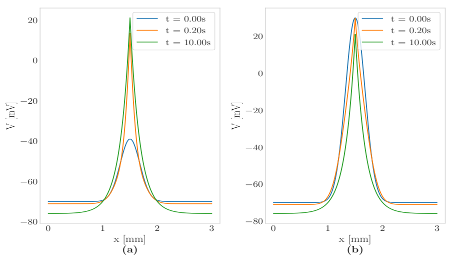


Figure 8. *Evolution of the membrane potential V over time for several initially applied potentials at $t = 0$ s.*

eigenvector of the transformation matrix with the corresponding largest eigenvalue is parallel to the final state vector in all four cases. The evolution of the system is calculated with $T \cdot V$, and when V is the eigenvector of T it will give back the same vector which we see as the equilibrium in Figure 3. Additionally, for the disjoint network, it's observed in Figure 12 that all nodes reach the same equilibrium value, but the value is different for the two networks. This makes sense as the total initial charge in each network is different, but is conserved within the network.

The initial state vector as presented in Figure 4 evolves back in time into a very irregular state which is not physically possible in our system considering that some nodes have negative charges. However, with our previous results in mind, it should not be possible to evolve into the initial condition presented, as one node has a very high charge compared to the closest neighbor, and one would need a nonphysical initial condition to reach this state in a forward evolution. The backward evolution was made with three different solvers which were timed and presented in Table 1. The NumPy solver is the fastest while the two SciPy methods are the slowest. The sparse SciPy method is the slowest, which might be expected as the matrix is small and thus not very sparse. However, this solver is expected to scale best with a larger system as the matrix becomes more and more sparse. The accuracy of the models is only visually confirmed by Figure 4, but in future works one could use a solver that uses an exact method and compare it with iterative approximate solvers.

4.2. Electrical impulses

An initial Gaussian wave was evolved in a bounded system with the linear cable equation using the forward- and backward Euler scheme, and the Crank-Nicolson scheme and compared with the analytical solution in an unbounded system in Figure 5. The Gaussian wave diffuses outwards over time and becomes flatter, and all methods give the same qualitative results. The accuracy of the numerical schemes is presented in Figure 6 and we see that the explicit Euler has the larger deviance from the exact solution while the implicit Euler scheme and Crank-Nicolson are marginally the same overall. After time $t = 1$ s the schemes error declines again which might be due to the membrane potential flattening out slowly and thus the schemes come closer. The explicit Euler method was found to be unstable for the lowest α values and is therefore not the preferable solver. Crank-Nicolson first becomes unstable for quite high α values, while the implicit Euler remains stable for all α values, which is expected [7]. However, the implicit Euler method becomes inaccurate for high α values. Considering these findings, the implicit Euler method might be the best choice, while the Crank-Nicolson scheme would also give good results without requiring too much computational resources.

Introducing the sodium and potassium ion channels and initiating the membrane potential with a Gaussian distribution gave the evolution seen in Figure 7. There is only one sodium ion channel in x_0 and the threshold potential for this voltage-gated channel to be activated is not reached with the initial potential strength. Considering that only the potassium channel is active and $V_K^{Nernst} = -76$ mV, the passive flow of K^+ will be outwards such that the membrane potential goes towards -76 mV. This is indeed what is observed in the figure as the membrane potential has an equilibrium at -76 mV after 10 s. When the V_{appl} is higher than the threshold potential, the sodium channel is activated as shown in Figure 8. Since V_{Na}^{Nernst} is 56 mV, the sodium ions will flow into the cell making the membrane potential more positive. The one ion channel creates a high peak with a membrane potential of around 20 mV as it reaches its equilibrium. The equilibrium potential does not reach the full Nernst potential of Na^+ because the potassium is active as well and works to make the membrane potential more negative. It is interesting to observe the locality

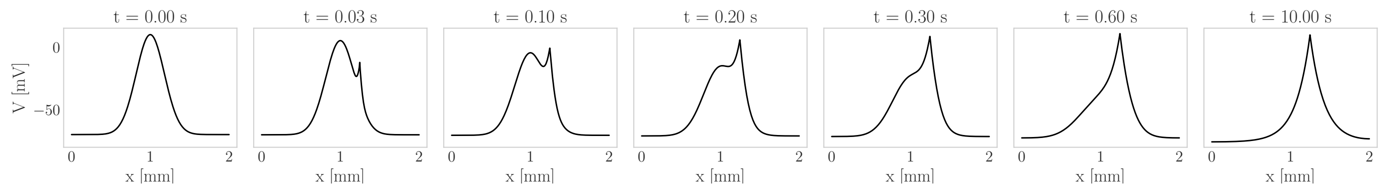


Figure 9. *Evolution of the membrane potential V over time for an initial applied potential at $t = 0$ s and a sodium ion channel at $x_{Na} = x_0 + 0.25$ mm.*

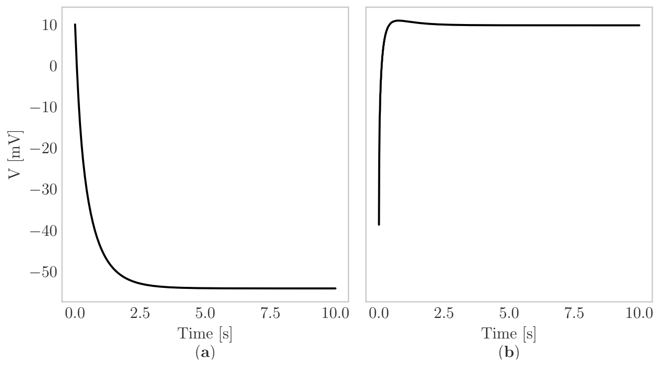


Figure 10. *Membrane depolarization over time at (a) the position of the initially applied potential $V(x_0, t)$ and (b) the position of the sodium ion channel $V(x_{Na}, t)$.*

of the sodium channel as the boundaries are going towards V_K^{Nernst}, which in a physical context is reasonable. Another physically correct behavior is that the membrane potential is depolarized around the ion channel, such that an open voltage-gated channel can activate nearby voltage-gated channels, and in this way spread the electrical impulse. However, a more accurate model of an action potential needs several ion channels such that the impulse can propagate and not just diffuse outwards from the active ion channel.

Moving the sodium channel 0.25 mm from the initially applied potential creates an exciting behavior presented in Figure 9. Both analytical and numerical analysis found that the applied voltage must be over 6.72 mV in this case, or inversely, the sodium channel has to be close enough for the membrane potential to reach the threshold. We can imagine the applied threshold is an action potential that has propagated along the membrane from the left side to the sodium channel and activated it. The sodium channel is then activated and the membrane potential increases on the right side, which in a real membrane could activate the next voltage-gated ion channel and continue to propagate the action potential. In this sense, the model does come closer to the real behavior of an action potential. To further investigate, we can look at the potential in x_0 and x_{Na} over time as presented in Figure 10, and compare it with a real action potential as the one given in [5]. The membrane potential at the sodium channel (b) can model an incoming action potential and closely mimics the initial depolarization of a real AP. The membrane potential in (a) model a closed ion channel that has already propagated the AP and we can compare it with the repolarization in a real AP. With the exception of the extra bump we see in a real AP, the modeled potential qualitatively has the same behavior. In a real cell, the ion channels would deactivate after some time [1] which we might compare by joining the two observed potentials and we thus come quite close to modeling a real AP. Modifications to the model could thus include several spaced ion channels that only stay activated on for a limited time and another type of channel to model the bump in the repolarization phase. With these implementations, one should be able to see a propagating AP.

Lastly, a chlorine ion channel would be an inhibitory process as the Nernst potential is -62 Mv and it would make the membrane potential decrease away from the threshold value for the sodium value. If one implements a voltage-gated chlorine channel with a higher potential than the sodium channel, creating a delayed opening, the chlorine channel repolarizes the membrane, deactivating the sodium channel and one would observe the sharp peak that is in a real AP. If V_{Cl}^{Nernst} is even lower than the membrane resting potential, one might even get the repolarization bump, and come even closer to a real AP.

5. Conclusion

The network model gave insights into how node charges are distributed in a connected system as time evolves. After reaching equilibrium, all initial states reached the same state vector which was parallel to the unit eigenvector of the transformation matrix. For a system of two disjoint networks, there was calculated two eigenvalues equal to 1 corresponding to the number of networks in the system. An initial state evolved backward revealed unphysical behavior of the system which makes sense given the diffusion of charges in the system could not create the initial state. Both the Lanczos iterative method for finding eigenvalues and eigenvectors and the sparse linear system solvers were found to be less efficient than the NumPy linalg approach, however, these methods should scale better and be more efficient for larger systems.

The linear cable equation is a simple model for the propagation of the membrane potential in a cell. Although giving insight into some electrical properties it did not fully explain the behaviour of an action potential. Introducing passive leaky- and voltage-gated ion channels gave the model more precise behavior that mimics a real AP. The voltage-gated sodium ion channel was found to be dependent on the strength of the initially applied (or incoming) depolarization of the membrane to be activated. Furthermore, when the applied potential was strong enough to activate the sodium channel it closely mimicked the behavior seen in a real AP. If the applied potential was not over the threshold, the membrane potential decreased towards the Nernst potential of the leaky potassium channel. The numerical schemes used to evolve this system gave accurate results when compared to an exact solution, and the numerical stability of the schemes found that the implicit Euler scheme was the most robust followed by the Crank-Nicolson scheme. Lastly, it was hypothesized that the introduction of a chlorine ion channel could account for the missing behavior compared to the real AP.

References

[1] W. M. Becker *et al.*, "Becker's world of the cell, 9th edition", pp. 682–697, 2017.

[2] NumPy, *Numpy documentation*, Accessed 01.05.24. [Online]. Available: https://numpy.org/doc/stable/index.html.

[3] SciPy, *Scipy documentation*, Accessed 01.05.24. [Online]. Available: https://docs.scipy.org/doc/scipy/index.html.

[4] R. Dias, "Introduction to assignment 1 - solving the diffusion equation", Spring 2024.

[5] "Exam in tfy4235/fy8904 - computational physics", Spring 2024.

[6] M. Hjorth-Jensen, "Computational physics lecture notes fall 2015", August 2015.

[7] I. Simonsen and R. Cabriolu, "Computational physics lectures", Spring 2024.

A. Additional theory

The transformation matrix T for two disjoint networks of $N = 11$ and $N = 10$ nodes each are presented in Figure 11. The separate networks are connected in a circle with connections to the four nearest neighbors.

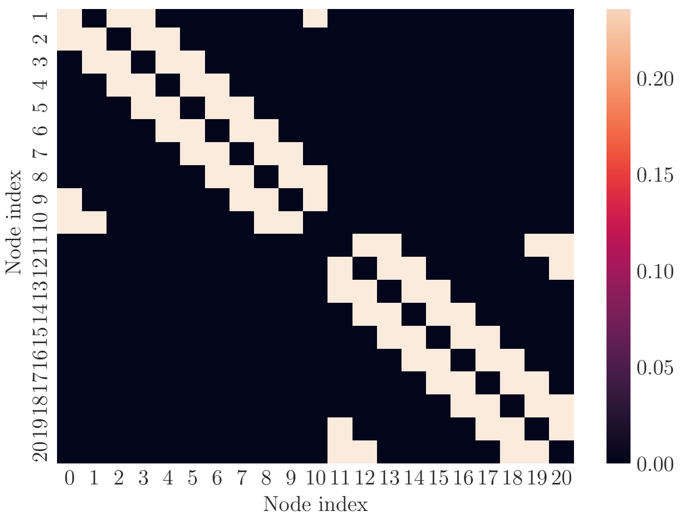


Figure 11. *Transformation matrix T for two disjoint networks of $N = 11$ and $N = 10$ nodes where each node is connected to the 4 nearest neighbors. The matrix is normalized such that each row sums to one, creating a stochastic matrix.*

B. Additional results

Figure 12 shows the evolution after 100 time steps for a random initial state vector, using the transformation matrix in Figure 11.

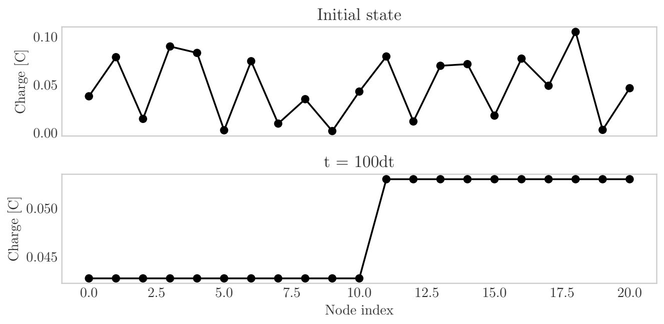


Figure 12. *The evolution of one initial random state, normalized to be a stochastic vector and plotted after 100 time steps. Each dot represents one node with node index at the x-axis and the node charge on the y-axis.*

A study of the stability of explicit- and implicit Euler, and Crank-Nicolson can be found in Figure 13. This shows how the different methods evolve for different α values.

The membrane potential is plotted for different initial applied potentials V_{appl} in Figure

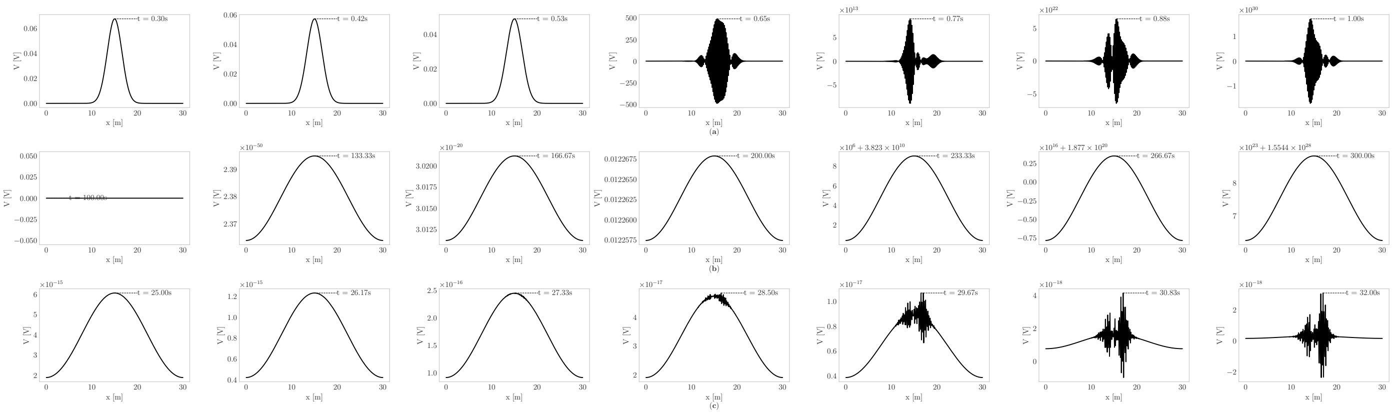
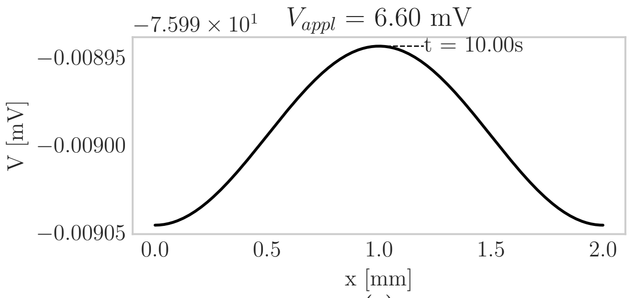
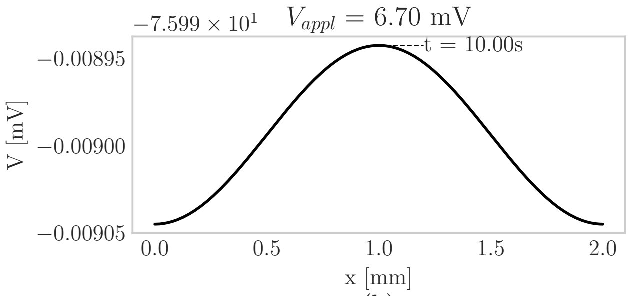


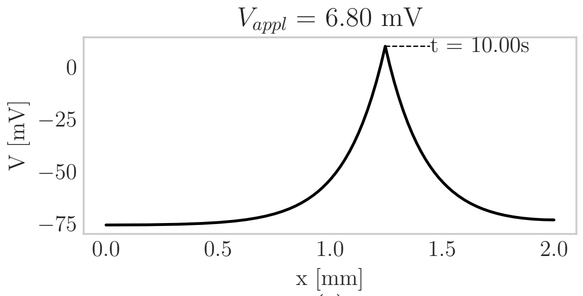
Figure 13. *Evolution of a Gaussian initial condition for different α values for the same amount of time steps, but varying Δt. The schemes presented are (a) (first row) explicit Euler, (b) implicit Euler, and (c) Crank-Nicolson.*



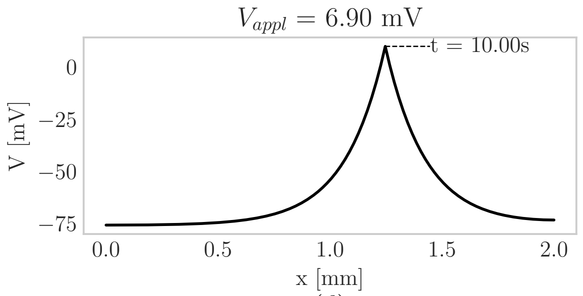
(a)



(b)



(c)



(d)

Figure 14. *Membrane potential V after 10 s for an initial applied potential at $t = 0$ s and a sodium ion channel located at $x = x_0 + 0.25$ mm. The strength of the initial applied potential is (a) $V_{appl} = 6.60$ mV, (b) $V_{appl} = 6.70$ mV, (c) $V_{appl} = 6.70$ mV, and (d) $V_{appl} = 6.80$ mV.*