A Brief Review of Practical Techniques for Single-Cell Biophysical Computation Within the Framework of a Neuronal Network Model for the Rat Hippocampal Formation

Nan Xiao

August 9, 2005

Motivation

From a biophysical standpoint, the investigation of network behavior in the rat hippocampal formation necessitates the use of spatially detailed single-cell models. Pyramidal cells and interneurons of the hippocampal region are morphologically complex, not only superficially, in the sense that microscopy reveals the presence of elaborate dendritic trees, but also because evidence strongly points to the nonuniform distribution of ion channels across these cells. With regard to synaptic connectivity, the function of a dendritic tree is apparent when it is observed that the axonal projections of the presynaptic neighbors of a given hippocampal neuron attach to many different locations on the dendritic tree. As such, it is not unwise to hypothesize that the morphology of single cells plays an important role in neuronal network function, justifying the use of computational models that take into account morphological complexity.

Nonlinear Cable Equation

The use of spatially detailed models of single hippocampal cells invokes cable theory. First used to describe transmission along underwater telegraph lines, cable theory is applied to neurons to describe the propagation of membrane potential along axonal and dendritic processes. For models of hippocampal cells, it is necessary to use the nonlinear cable equation. Assuming that dendritic segments are represented on a one-dimensional domain and that the dendrite contains non-ohmic active conductances, the balance of axial and longitudinal currents along every point of the dendritic segment gives rise to the nonlinear cable equation:

$$\frac{1}{2R_a a(x)}\frac{\partial}{\partial x}\left[a(x)^2\frac{\partial V}{\partial x}(x,t)\right] = C_m\frac{\partial V}{\partial t}(x,t) + I_{\text{membrane}}(V) + I_{\text{synaptic}}(V) + I_{\text{stimulus}} \quad (1)$$

The two ends of the dendritic segment are sealed. There is no current flow at either end.

$$\frac{\partial V}{\partial x}(x_0,t) = \frac{\partial V}{\partial x}(x_\ell,t) = 0$$

At time $t = t_0$ the membrane potential distribution is described by $\psi(x)$.

$$\frac{\partial V}{\partial t}(x, t_0) = \psi(x)$$

The use of a linear spatial domain applies to arbitrarily branched neurons as well: intuitively, each branch in the neuron is assigned a specific interval. Specifically, a set of points x_i $(i = 0, 1, 2, 3, \ldots, \ell)$ is defined such that $x_0 = 0 < x_1 < x_2 < x_3 < \ldots < x_\ell$. The spatial domain for branch i is the set of points on the interval $[x_i, x_{i+1})$.

Branching

Branch Types

For strictly computational purposes, there exist only two types of branches. A non-terminal branch is connected to other branches at both ends, whereas a terminal branch is connected to other branches at less than two ends. Furthermore, a branch is either an axonal segment or a dendritic segment depending on the ionic conductances that are defined on its spatial domain. The soma, though represented by a point, counts as one branch. Thus, every branch except for the soma has one parent branch.

Branch Ordering

Branch ordering is done in the manner of the Hines method [4]. First, the soma is represented by the point x_ℓ. The branches are then ordered by decreasing "depth" in the branch tree relative to the soma. Specifically, the sets of points representing the branches directly physiologically conected to the soma are assigned depth '1'. The sets of points representing the branches that connect to branches of depth k are assigned depth $k + 1$ $(k = 1, \ldots, maxd)$, where $maxd$ is the maximum possible depth in the branching trees of a neuron. Naturally, the terminal branches will be assigned with the largest depth numbers. The branches are now ordered by their assigned depth numbers.

$$\text{domain for branch } i := [x_i, x_{i+1})$$

$$\text{where}$$

$$i = \begin{cases} 0, \ldots, n_{maxd} - 1 & \text{for the } n_{maxd} \text{ branches of depth } maxd \\ \sum_{q=0}^{k-1} n_{maxd-q}, \ldots, \sum_{q=0}^{k} n_{maxd-q} & \text{for the } n_{maxd-k} \text{ branches of depth } maxd - k \end{cases}$$

$$\text{and } k = 1, \ldots, maxd - 1$$

The specific numbering of branches within a given depth is unimportant, only that the branches of higher depth are indexed ahead (smaller index number) of branches of lower depth. The purpose of this ordering method is made clear when the spatial domains are discretized.

The boundary conditions are now such that if branch j is a terminal branch in a neuron with more than one branch, and if the spatial domain of the terminal branch is defined by $[x_j, x_{j,\text{start}}]$, then

$$\frac{\partial v}{\partial x}(x_j, t) = 0 \quad j = 1, \ldots, n_{\text{terminal branch}}$$

In the general, the soma is sealed

$$\frac{\partial v}{\partial x}(x_\ell, t) = 0 \quad j = 1, \ldots, n_{\text{terminal branch}}$$

For all branches, $x_{j,\text{start}}$ represents the end of the branch that is connected to the parent branch.

Ionic Conductances

Neurons of the hippocampal formation contain many different voltage-gated ion channels aside. Apart from the standard leak current, these active currents are modeled in an ensemble fashion through the use of voltage-dependent terms in the cable equation describing current flow elicited by the specific ion channel types. The inclusion of these currents gives the cable equation nonlinearity. Active currents involve the voltage-gated diffusion of Na^+, K^+, and Ca^{2+} ions.

$$I_{\text{membrane}} = I_{\text{leak}} + I_{\text{active}}$$
$$I_{\text{active}} = \sum_i I_{\{\text{Na}^+\}_i} + \sum_j I_{\{\text{K}^+\}_j} + \sum_k I_{\{\text{Ca}^{2+}\}_k}$$

The leakage conductance is represented by an ohmic conductance whereas the active currents use the Hodgkin and Huxley formalism.

$$I_{\text{leak}} = G_{\text{leak}}[V(x,t) - E_{\text{leak}}]$$
$$I_{\{X\}_x} = \overline{G}_{\{X\}_x} M_{\{X\}_x}(t)^p N_{\{X\}_x}(t)^q [V(x,t) - E_{\{X\}_x}]$$

$I_{\{X\}_x}$ denotes an active current of type x involving ions of element X. E represents the reversal potential of the particular channel. M and N respectively are the activation and inactivation variables for the particular active current. They represent a lumped probability of the activation and inactivation of the particular ion channel. In accordance with the specific biophysical properties of the channel in question, the particular active current may lack an inactivation variable. The activation and inactivation variables are voltage-dependent in the sense that

$$\frac{dM_{\{X\}_x}(t)}{dt} = \text{Rate of Opening} - \text{Rate of Closing}$$
$$\frac{dM_{\{X\}_x}(t)}{dt} = \alpha_{M_{\{X\}_x}}(V)[1 - M_{\{X\}_x}(t)] - \beta_{M_{\{X\}_x}}(V) M_{\{X\}_x}(t)$$

The rate functions $\alpha(V)$ and $\beta(V)$ are voltage-dependent functions governing the rate of change of the activation variable.

Synaptic Conductances

Synaptic currents at specific locations on the simulated neuron model the effect of synaptic connections from neighboring cells. The synaptic current takes the form

$$I_{\text{synaptic}} = \sum_i \sum_j g_{syn_i} \delta(x - x_{syn_{i,j}})[V(x,t) - E_{syn_i}]$$

A synaptic current of type i is present at spatial location j on the neuron. The function g_{syn} depends on the complexity of the synaptic model.

Alpha function conductance

A simple 'α function' can be used for $g_{syn}(t)$ to represent the non-NMDA synaptic inputs mediated by the mossy fibers (koch).

$$g_{syn}(t) = C\ t\ e^{-t/t_{peak}}$$

Where C is a constant $= \frac{g_{peak}e}{t_{peak}}$ so that $g_{syn}(t_{peak}) = g_{peak}$.

Graded synapse

In reality, release of synaptic vessicles is related to the potential in the presynaptic membrane. The simplest approach to incorporated dependency of the presynaptic potential results in a 'graded' synaptic conductance that obeys the relationship

$$\frac{dg_{syn}}{dt}(t) = \frac{g_{syn_\infty}(v) - g_{syn}(t)}{\tau}$$

General biophysical synapse

More complicated synaptic models may include differential equations describing neurotransmitter release. Characteristics of short-term potentiation can be captured in a general three-step model

- Depolarization of the presynaptic membrane induces a Ca^{2+} current in the presynaptic region, causing a rise in the concentration of Ca^{2+}.
- Ca binds to receptors and neurotransmitter is released at a rate determined by the opening and closing rates of the bound receptors.

- Neurotransmitter, in turn, binds to receptors (ion channels) on the postsynaptic region; the synaptic conductance g_{syn} is therefore given by

$$g_{syn}(t) = \gamma o(t)$$

where γ is the maximum single-postsynaptic-channel conductance and $o(t)$ is the number of open channels. A straightfoward system of the differential equations describes the Ca^{2+}, neurotransmitter, presynaptic receptors and postsynaptic channels.

Spatial Discretization

It is most feasible to solve the nonlinear cable equation with nonuniform coefficients numerically. Linearization of the active conductances and discretization of the spatial domain gives rise to a system of linear ordinary differential equations, which may be solved numerically through an appropriate time-stepping method. Spatial discretization creates a finite-dimensional spatial mesh; solutions to the discretized cable equation are therefore finite-dimensional vectors.

Finite differences: central space method

The popular NEURON neurosimulator uses the method of finite differences for spatial discretization, where a second difference matrix is substituted for the Laplacian operator. The cable equation becomes

$$\frac{1}{2R_a}\mathbf{B}\vec{V} = C_m\frac{d\vec{V}}{dt}(t) + \vec{I}_{\text{membrane}} + \vec{I}_{\text{synaptic}} + \vec{I}_{\text{stimulus}} \tag{2}$$

$$t > t_0$$

$$\frac{d\vec{V}}{dt}(t_0) = \vec{\psi} \quad t = t_0$$

This finite difference method (central-space) has second-order accuracy ($O(\Delta x^2)$) and imposes a regular mesh upon the spatial domain.

Finite elements: Galerkin method

On the other hand, the finite elements method naturally handles irregular spatial meshes. Specifically, the Galerkin method is used, with a finite-dimensional subspace of continuous linear piecewise polynomials spanned by 'hat' functions [3] (Appendix 1). The finite elements method also handles the nonuniform coefficients in the cable equation in a more mathematically rigorous fashion. This method is second-order accurate. The discretized cable equations becomes

$$\vec{I}_{\text{stimulus}} + \vec{c} = C_m\mathbf{M}\frac{d\vec{V}}{dt}(t) + \left[\mathbf{C} + \mathbf{L} + \frac{1}{2R_a}\mathbf{K}\right]\vec{V}(t) \tag{3}$$

$$t > t_0$$

$$\frac{d\vec{V}}{dt}(t_0) = \vec{\psi} \quad t = t_0$$

Efficiency

An efficient implementation of either method would have a computation time that is practically independent of the number of the branches in the simulated neuron. Computation time would vary on the order of $1/\Delta x$ per time step. However, in the case of the finite elements method, the assembly of the matrices for the ODE system includes many more numerical evaluations, making the finite elements method slower but still on the order of $1/\Delta x$ per time step.

Time-stepping

To numerically integrate the stiff system of ODEs, an unconditionally stable time-stepping method should be used. In the case of a passive neuron, the linear cable equation is

$$\frac{1}{2R_a}\mathbf{B}\vec{V} = C_m\frac{d\vec{V}}{dt}(x,t) + \vec{G}_l(\vec{V}(t) - E_l)$$

Integrating gives

$$\int_t^{t_f} \frac{d\vec{V}}{dt}dt = \int_t^{t_f} \frac{1}{C_m}\left[\frac{1}{2R_a}\mathbf{B}\vec{V}(t) - \vec{G}_l(\vec{V}(t) - E_l)\right]dt$$

$$\vec{V}(t_f) = \vec{V}(t) + \int_t^{t_f} \frac{1}{C_m}\left[\frac{1}{2R_a}\mathbf{B}\vec{V}(t) - \vec{G}_l(\vec{V}(t) - E_l)\right]dt$$

Backward Euler

The most simple appropriate time-stepping method is the fully implicit backward Euler scheme, in which

$$\vec{V}(t+\Delta t) = \vec{V}(t) + \frac{\Delta t}{C_m}\left[\frac{1}{2R_a}\mathbf{B}\vec{V}(t+\Delta t) - \vec{G}_l(\vec{V}(t+\Delta t) - E_l)\right]$$

This method is first-order accurate ($O(\Delta t)$) and stable for all Δt.

Crank-Nicolson

For second-order accuracy ($O(\Delta t)^2$) as well as unconditional stability, the Crank-Nicolson method (half-implicit Euler) is very efficient.

$$\vec{V}(t+\Delta t) = \vec{V}(t) + \frac{\Delta t}{2C_m}\left[\left(\frac{1}{2R_a}\mathbf{B}\vec{V}(t+\Delta t) - \vec{G}_l(\vec{V}(t+\Delta t) - E_l)\right) + \left(\frac{1}{2R_a}\mathbf{B}\vec{V}(t) - \vec{G}_l(\vec{V}(t) - E_l)\right)\right]$$

Since this method is essentially the average of the forward and backward Euler methods, a shortcut to reduce the number of numerical evaluations is often used. The result is a two-step process that uses half-time steps. Essentially, backward Euler is performed first and forward Euler is performed second (Appendix 2).

Linearization

The nonlinear terms (active currents) in the cable equation depend upon membrane potential through the voltage-dependent rate functions. As such, it suffices to allow the rate functions to depend upon the membrane potential calculated at the previous time step. The ODE describing an active variable becomes

$$\frac{dM(t)}{dt} = \alpha_M(V(t))[1 - M(t)] - \beta_M(V(t))M(t)$$

Numerical Integration through backward Euler gives

$$M(t+\Delta t) = M(t) + \int_t^{t+\Delta t}\Big[\alpha_M(V(t))[1 - M(t)] - \beta_M(V(t))M(t)\Big]dt$$

$$M(t+\Delta t) = M(t) + \Delta t\Big[\alpha_M(V(t))[1 - M(t+\Delta t)] - \beta_M(V(t))M(t+\Delta t)\Big]$$

Essentially, the activation variables are computed prior to solving for the membrane potential for the current point in time.

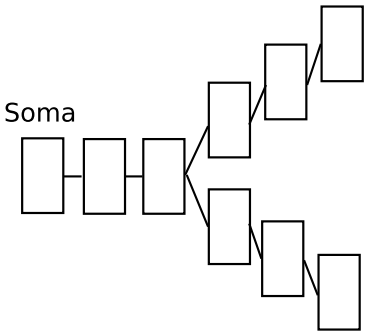
Branch Ordering Revisited

The numerical methods for spatial discretization essentially produce a matrix equation

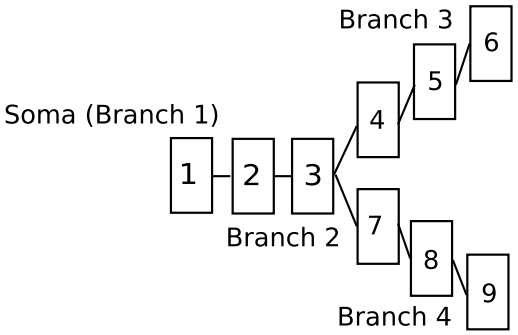
$$\mathbf{A}\vec{x} = \vec{b}$$

to be solved for $\vec{x}$ at every time step. $\mathbf{A}$ not only contains numerical information; the neuronal morphology is encoded in the sparsity pattern. For the simulated single neurons, $\mathbf{A}$ is always symmetric and nearly tridiagonal. If the neuron consists of a soma and a straight fiber $\mathbf{A}$ will be tridiagonal. On the other hand, if each discretized compartment is a branch itself, $\mathbf{A}$ will hardly be tridiagonal. Though, if the Hines reordering method is used, $\mathbf{A}$ for the latter case will be essentially tridiagonal. Given the special nature of

A, it is inefficient to use an expensive general Guassian elimination algorithm to solve for $\vec{x}$, especially if the Hines ordering is used, in which case a barely modified tridiagonal solver, whose computation time scales proportional to the size of **A**, is sufficient. This example, the discretized three-branch (technically four, including the soma) fork neuron, illustrates the importance of the Hines ordering method.



An intuitive method of numbering the nodes assigns increasing branch numbers moving away from the soma like so



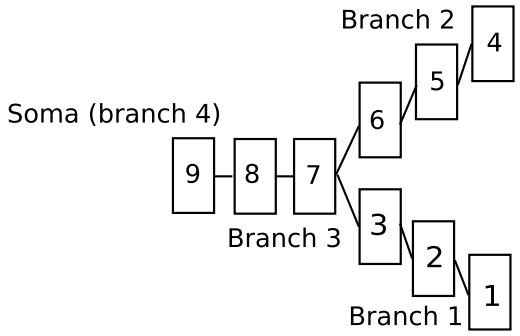
The left-hand-side matrix that corresponds to this numbering scheme is

$$\mathbf{A} = \begin{bmatrix} A_{1,1} & A_{1,2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ A_{2,1} & A_{2,2} & A_{2,3} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & A_{3,2} & A_{3,3} & A_{3,4} & 0 & 0 & A_{3,7} & 0 & 0 \\ 0 & 0 & A_{4,3} & A_{4,4} & A_{4,5} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & A_{5,4} & A_{5,5} & A_{5,6} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & A_{6,5} & A_{6,6} & 0 & 0 & 0 \\ 0 & 0 & A_{7,3} & 0 & 0 & 0 & A_{7,7} & A_{7,8} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & A_{8,7} & A_{8,8} & A_{8,9} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & A_{9,8} & A_{9,9} \end{bmatrix}$$

A one-step-per-node diagonalizing algorithm used to eliminate the lower diagonal elements will run into problems at $A_{7,7}$ where unwanted fill-in of entry $A_{7,4}$ will occur, such that a more general elimination algorithm must be used. Consider a rather common pyramidal neuron with two large dendritic trees, ℓ branches in total, and solely bifurcating branches. Any numbering scheme that assigns higher branch numbers to branches that are "further away" (in terms of depth as defined earlier) will encounter at least $\ell/2 - 1$ entries in matrix **A** where a general elimination algorithm must be used. This dramatically lowers

the overall efficiency of solving for $\vec{x}$.

The Hines method assigns larger branch indices to branches "closer" to the soma. Furthermore, the nodes within each branch end up numbered in increasing order moving closer to the branch/parent node.



This ensures that any branch in the neuron is only coupled to one other branch, which has a larger branch index. As such, $\mathbf{A}$ looks like

$$\mathbf{A} = \begin{bmatrix} A_{1,1} & A_{1,2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ A_{2,1} & A_{2,2} & A_{2,3} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & A_{3,2} & A_{3,3} & 0 & 0 & 0 & A_{3,7} & 0 & 0 \\ 0 & 0 & 0 & A_{4,4} & A_{4,5} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & A_{5,4} & A_{5,5} & A_{5,6} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & A_{6,5} & A_{6,6} & A_{6,7} & 0 & 0 \\ 0 & 0 & A_{7,3} & 0 & 0 & A_{7,6} & A_{7,7} & A_{7,8} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & A_{8,7} & A_{8,8} & A_{8,9} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & A_{9,8} & A_{9,9} \end{bmatrix}$$

A slightly modified tridiagonal solver is sufficient for the Hines matrix, which is essentially a tridiagonal matrix. Fill-in problems can now be avoided.

Neuronal Networks

In modelling networks of neurons, it is only necessary to change the spatial domain to reflect the presence of two cells. The Hines ordering is kept for individual cells. When descritized and simplified, again the matrix equation $\mathbf{A}\vec{x} = \vec{b}$ must be solved at each time step. However

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_1 & & & \\ & \mathbf{A}_2 & & \\ & & \ddots & \\ & & & \mathbf{A}_{NC} \end{bmatrix}$$

where $\mathbf{A}_i$ represents the left-hand-side matrix at the current time step for cell i and NC is the number of cells. Similarly for the right-hand-side vector at the current time step,

$$\vec{b} = \begin{bmatrix} \vec{b}_1 \\ \vec{b}_2 \\ \vdots \\ \vec{b}_{NC} \end{bmatrix}$$

This simple method is valid when no neurons in the network communicate through eletrical synapses, where current directly passes from cell to cell.

Appendix 1: Finite Elements Method for Branched Neurons

Simple Tapered Passive Fiber

The tapered cable equation [2] on a linear domain ($x \in [0, \ell]$)

$$\frac{1}{2Ra}\frac{\partial}{\partial x}\left(a(x)^2\frac{\partial v}{\partial x}(x,t)\right) = a(x)\sqrt{1+\left(\frac{da}{dx}(x)\right)^2}\left[C_m\frac{\partial v}{\partial t}(x,t) + G_m(v(x,t)-E_l)\right] \tag{4}$$

$$v(x,t_o) = \psi(x)$$

$$\frac{\partial v}{\partial x}(0,t) = \frac{R_a}{\pi a^2(0)}i_o(t) \quad t > t_o$$

$$\frac{\partial v}{\partial x}(\ell,t) = 0 \quad t > t_o$$

Multiply by a test function in $\tilde{V} = C^2[0,\ell]$ and integrate over the domain

$$\int_0^\ell \frac{1}{2Ra}\frac{\partial}{\partial x}\left(a(x)^2\frac{\partial v}{\partial x}(x,t)\right)u(x)\ dx$$

$$= \int_0^\ell a(x)\sqrt{1+\left(\frac{da}{dx}(x)\right)^2}\left[C_m\frac{\partial v}{\partial t}(x,t) + G_m(v(x,t)-E_l)\right]u(x)\ dx$$

Integrate by parts to get the weak form:

$$\frac{1}{2Ra}\left[a(x)^2u(x)\frac{\partial v}{\partial x}(x,t)\right]_0^\ell - \frac{1}{2Ra}\int_0^\ell a(x)^2\frac{\partial v}{\partial x}(x,t)\frac{du}{dx}(x)\ dx$$

$$= \int_0^\ell a(x)\sqrt{1+\left(\frac{da}{dx}(x)\right)^2}\left[C_m\frac{\partial v}{\partial t}(x,t) + G_m(v(x,t)-E_l)\right]u(x)\ dx$$

Apply boundary conditions to get:

$$\frac{u(o)}{2\pi}i_o(t) - \frac{1}{2Ra}\int_0^\ell a(x)^2\frac{\partial v}{\partial x}(x,t)\frac{du}{dx}(x)\ dx$$

$$= \int_0^\ell a(x)\sqrt{1+\left(\frac{da}{dx}(x)\right)^2}\left[C_m\frac{\partial v}{\partial t}(x,t) + G_m(v(x,t)-E_l)\right]u(x)\ dx$$

Rearrange:

$$\frac{u(o)}{2\pi}i_o(t) - \frac{1}{2Ra}\int_0^\ell a(x)^2\frac{\partial v}{\partial x}(x,t)\frac{du}{dx}(x)\ dx + E_l\int_0^\ell G_m(x)a(x)u(x)\sqrt{1+\left(\frac{da}{dx}(x)\right)^2}dx$$

$$= C_m\int_0^\ell a(x)u(x)\frac{\partial v}{\partial t}(x,t)\sqrt{1+\left(\frac{da}{dx}(x)\right)^2}\ dx + \int_0^\ell G_m(x)a(x)u(x)v(x,t)\sqrt{1+\left(\frac{da}{dx}(x)\right)^2}\ dx$$

A solution $v(x,t)$ that satisfies the weak form for every $u(x,t) \in \tilde{V}$ satisfies the PDE. To find an approximation to the solution, a finite dimensional subspace $\tilde{S_n} = \{p : [0,\ell] \to \mathbb{R} \ : \ \text{p is piecewise linear/continuous}\} = \text{span}\{\phi_0, \phi_1, ..., \phi_N\}$ is defined. A regular mesh $(0 < x_0 < x_1 < x_2 < ... < x_N = \ell)$ with $N+1$ nodes separated by intervals of length h is used to spatially discretize the domain.

The ϕ_i are 'hat functions' defined by

$$\phi(x) = \begin{cases} 1/h(x-(i-1)h) & \text{if } x_{i-1} < x < x_i, \\ -1/h(x-(i+1)h) & \text{if } x_i \leq x < x_{i+1}, \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

Membrane potential and dendritic radius are linear combinations of the hat functions

$$v_n(t,x) = \sum_{i=0}^{N} v_i(t)\phi_i(x)$$

$$a_n(x) = \sum_{i=0}^{N} a_i\phi_i(x)$$

Membrane leakage conductance is represented by piecewise constants

$$G_m(x) = \sum_{i=0}^{N-1} G_{m_j}\chi_j(x)$$

It now suffices to find $v_n(t)$ that satisfies the weak form for every $u(x,t) \in \tilde{S}_n$. Specifically, $v_n(t)$ has to satisfy:

$$\frac{\phi_j(o)}{2\pi} i_o(t) - \frac{1}{2Ra}\int_0^\ell a(x)^2 \Big[\sum_{i=0}^{N} v_i(t)\frac{d\phi}{dx}(x)\Big]\frac{d\phi_j}{dx}(x)\ dx$$

$$+E_l \int_0^\ell G_m(x)a(x)\phi_j(x)\sqrt{1+\left(\frac{da}{dx}(x)\right)^2}\ dx$$

$$= C_m \int_0^\ell a(x)\phi_j(x)\Big[\sum_{i=0}^{N}\frac{dv_i}{dt}(t)\phi_i(x)\Big]\sqrt{1+\left(\frac{da}{dx}(x)\right)^2}\ dx$$

$$+\int_0^\ell G_m(x)a(x)\phi_j(x)\Big[\sum_{i=0}^{N} v_i(t)\phi_i(x)\Big]\sqrt{1+\left(\frac{da}{dx}(x)\right)^2}\ dx$$

$$j = 0, 1, 2, ..., N$$

Rearranging:

$$\frac{\phi_j(o)}{2\pi} i_o(t) - \sum_{i=0}^{N} v_i(t)\frac{1}{2Ra}\int_0^\ell \Big[\sum_{i=0}^{N} a_i\phi_i(x)\Big]^2 \frac{d\phi_i}{dx}(x)\frac{d\phi_j}{dx}(x)\ dx$$

$$+E_l \int_0^\ell G_m(x)\Big[\sum_{i=0}^{N} a_i\phi_i(x)\Big]\phi_j(x)\sqrt{1+\Big[\sum_{i=0}^{N} a_i\frac{d\phi_i}{dx}(x)\Big]^2}\ dx$$

$$= C_m \sum_{i=0}^{N}\frac{dv_i}{dt}(t)\int_0^\ell \Big[\sum_{i=0}^{N} a_i\phi_i(x)\Big]\phi_j(x)\phi_i(x)\sqrt{1+\Big[\sum_{i=0}^{N} a_i\frac{d\phi_i}{dx}(x)\Big]^2}\ dx$$

$$+\sum_{i=0}^{N} v_i(t) \int_0^{\ell} \left[\sum_{i=0}^{N-1} G_{m_j} \chi_j \right] \left[\sum_{i=0}^{N} a_i \phi_i(x) \right] \phi_j(x) \phi_i(x) \sqrt{1 + \left[\sum_{i=0}^{N} a_i \frac{d\phi_i}{dx}(x) \right]^2} \, dx$$

$$j = 0, 1, 2, ..., N$$

Define the matrices **M**,**K**,**L**:

$$K_{ij} = \int_0^{\ell} \left[\sum_{i=0}^{N} a_i \phi_i(x) \right]^2 \frac{d\phi_i}{dx}(x) \frac{d\phi_j}{dx}(x)$$

$$M_{ij} = \int_0^{\ell} \left[\sum_{i=0}^{N} a_i \phi_i(x) \right] \phi_j(x) \phi_i(x) \sqrt{1 + \left[\sum_{i=0}^{N} a_i \frac{d\phi_i}{dx}(x) \right]^2} \, dx$$

$$L_{ij} = \int_0^{\ell} \left[\sum_{i=0}^{N-1} G_{m_i} \chi_i \right] \left[\sum_{i=0}^{N} a_i \phi_i(x) \right] \phi_j(x) \phi_i(x) \sqrt{1 + \left[\sum_{i=0}^{N} a_i \frac{d\phi_i}{dx}(x) \right]^2} \, dx$$

and the battery vector $\vec{b}$

$$b_j = E_l \int_0^{\ell} \left[\sum_{i=0}^{N-1} G_{m_i} \chi_i \right] \left[\sum_{i=0}^{N} a_i \phi_i(x) \right] \phi_j(x) \sqrt{1 + \left[\sum_{i=0}^{N} a_i \frac{d\phi_i}{dx}(x) \right]^2} \, dx$$

Evaluating the integrals we get tridiagonal matrices:

Main diagonal:

$$j = i = 0$$

$$K_{ij} = \frac{1}{3h}(a_0^2 + a_0 a_1 + a_1^2)$$

$$M_{ij} = \frac{1}{12} \sqrt{h^2 + (a_0 - a_1)^2} (3a_0 + a_1)$$

$$L_{ij} = \frac{G_{m_i}}{12} \sqrt{h^2 + (a_0 - a_1)^2} (3a_0 + a_1)$$

$$b_i = \frac{E_l G_{m_i}}{12} \sqrt{h^2 + (a_0 - a_1)^2} (4a_0 + 2a_1)$$

$$0 < j = i < N$$

$$K_{ij} = \frac{1}{3h}(2a_i^2 + a_{i-1}^2 + a_{i+1}^2 + a_i a_{i-1} + a_i a_{i+1})$$

$$M_{ij} = \frac{1}{12} \sqrt{h^2 + (a_{i-1} - a_i)^2} (a_{i-1} + 3a_i) + \frac{1}{12} \sqrt{h^2 + (a_i - a_{i+1})^2} (3a_i + a_{i+1})$$

$$L_{ij} = \frac{G_{m_{i-1}}}{12} \sqrt{h^2 + (a_{i-1} - a_i)^2} (a_{i-1} + 3a_i) + \frac{G_m}{12} \sqrt{h^2 + (a_i - a_{i+1})^2} (3a_i + a_{i+1})$$

$$b_i = \frac{E_l G_{m_{i-1}}}{12} \sqrt{h^2 + (a_{i-1} - a_i)^2} (2a_{i-1} + 4a_i) + \frac{E_l G_m}{12} \sqrt{h^2 + (a_i - a_{i+1})^2} (4a_i + 2a_{i+1})$$

$$j = i = N$$

$$K_{ij} = \frac{1}{3h}(a_i^2 + a_i a_{i-1} + a_{i-1}^2)$$

$$M_{ij} = \frac{1}{12}\sqrt{h^2 + (a_i - a_{i-1})^2}(3a_i + a_{i-1})$$

$$L_{ij} = \frac{G_{m_{i-1}}}{12}\sqrt{h^2 + (a_i - a_{i-1})^2}(3a_i + a_{i-1})$$

$$b_i = \frac{E_l G_{m_{i-1}}}{12}\sqrt{h^2 + (a_i - a_{i-1})^2}(4a_i + 2a_{i-1})$$

Off diagonal:

$$j = i + 1$$

$$K_{ij} = -\frac{1}{3h}(a_i^2 + a_i a_{i+1} + a_{i+1}^2)$$

$$M_{ij} = \frac{1}{12}(a_i + a_{i+1})\sqrt{h^2 + (a_i - a_{i+1})^2}$$

$$L_{ij} = \frac{G_{m_i}}{12}(a_i + a_{i+1})\sqrt{h^2 + (a_i - a_{i+1})^2}$$

Solve the linear system numerically

$$\frac{i_o}{2\pi}\vec{e_o} + \vec{b} = C_m \mathbf{M}\frac{d\vec{v}}{dt}(t) + \left[\mathbf{L} + \frac{1}{2R_a}\mathbf{K}\right]\vec{v}(t)$$

with backward Euler time-stepping:

$$\left[C_m \mathbf{M} + \Delta t(\mathbf{L} + \frac{1}{2R_a}\mathbf{K})\right]\vec{v}_{j+1} = C_m \mathbf{M}\vec{v}_j + \left[\vec{b} + \frac{i_o}{2\pi}e_o\right]\Delta t$$

Standard units:

C_m is in units of $\frac{\mu\text{F}}{cm^2}$

G_m is in units of $\frac{\text{mS}}{\text{cm}^2}$

R_a is in units of kΩ.cm

The elements of the matrices M and K are in m

The elements of the matrix L are in m.mS

Tapered fiber with soma

The 'left' boundary condition is changed to accomodate a soma of surface area A_s

$$\frac{\partial v}{\partial x}(0,t) = \frac{R_a A_s}{\pi a^2}\left[C_m \frac{\partial v}{\partial t}(0,t) + G_s(v(0,t) - E_l)\right] - \frac{R_a i_o(t)}{\pi a^2}$$

The weak form becomes

$$u(o)\left[\frac{i_o(t)}{2\pi} - A_s\left(C_m \frac{\partial v}{\partial t}(0,t) + G_s(v(0,t) - E_l)\right)\right] - \frac{1}{2Ra}\int_0^\ell a(x)^2 \frac{\partial v}{\partial x}(x,t)\frac{du}{dx}(x)\ dx$$

$$+E_l \int_0^\ell G_m(x)a(x)u(x)\sqrt{1 + \left(\frac{da}{dx}(x)\right)^2}\,dx$$

$$= C_m \int_0^\ell a(x)u(x)\frac{dv}{dt}(x,t)\sqrt{1 + \left(\frac{da}{dx}(x)\right)^2}\,dx + \int_0^\ell G_m(x)a(x)u(x)v(x,t)\sqrt{1 + \left(\frac{da}{dx}(x)\right)^2}\,dx$$

The finite element method yields the system

$$\phi_j(o)\left[\frac{i_o(t)}{2\pi} - A_s\left(C_m \frac{dv_0}{dt}(t) + G_s(v_0 - E_l)\right)\right] - \sum_{i=0}^N v_i(t)\frac{1}{2Ra}\int_0^\ell \left[\sum_{i=0}^N a_i\phi_i(x)\right]^2 \frac{d\phi_i}{dx}(x)\frac{d\phi_j}{dx}(x)\ dx$$

$$+E_l \int_0^\ell \left[\sum_{i=0}^{N-1} G_{m_i}\chi_i\right]\left[\sum_{i=0}^N a_i\phi_i(x)\right]\phi_j(x)\sqrt{1 + \left[\sum_{i=0}^N a_i \frac{d\phi_i}{dx}(x)\right]^2}\,dx$$

$$= C_m \sum_{i=0}^N \frac{dv_i}{dt}(t)\int_0^\ell \left[\sum_{i=0}^N a_i\phi_i(x)\right]\phi_j(x)\phi_i(x)\sqrt{1 + \left[\sum_{i=0}^N a_i \frac{d\phi_i}{dx}(x)\right]^2}\,dx$$

$$+\sum_{i=0}^N v_i(t)\int_0^\ell \left[\sum_{i=0}^{N-1} G_{m_i}\chi_i\right]\left[\sum_{i=0}^N a_i\phi_i(x)\right]\phi_j(x)\phi_i 0(x)\sqrt{1 + \left[\sum_{i=0}^N a_i \frac{d\phi_i}{dx}(x)\right]^2}\,dx$$

$$j = 0, 1, 2, ..., N$$

Rearranging, we have

$$\phi_j(o)\frac{i_o(t)}{2\pi} - \sum_{i=0}^N v_i(t)\frac{1}{2Ra}\int_0^\ell \left[\sum_{i=0}^N a_i\phi_i(x)\right]^2 \frac{d\phi_i}{dx}(x)\frac{d\phi_j}{dx}(x)\ dx$$

$$\phi_j(0)A_s G_s E_l + E_l \int_0^\ell \left[\sum_{i=0}^{N-1} G_{m_i}\chi_i\right]\left[\sum_{i=0}^N a_i\phi_i(x)\right]\phi_j(x)\sqrt{1 + \left[\sum_{i=0}^N a_i \frac{d\phi_i}{dx}(x)\right]^2}\,dx$$

$$= A_s C_m \frac{dv_0}{dt}(t) + C_m \sum_{i=0}^N \frac{dv_i}{dt}(t)\int_0^\ell \left[\sum_{i=0}^N a_i\phi_i(x)\right]\phi_j(x)\phi_i(x)\sqrt{1 + \left[\sum_{i=0}^N a_i \frac{d\phi_i}{dx}(x)\right]^2}\,dx$$

$$+A_s G_s v_0 + \sum_{i=0}^N v_i(t)\int_0^\ell \left[\sum_{i=0}^{N-1} G_{m_i}\chi_i\right]\left[\sum_{i=0}^N a_i\phi_i(x)\right]\phi_j(x)\phi_i(x)\sqrt{1 + \left[\sum_{i=0}^N a_i \frac{d\phi_i}{dx}(x)\right]^2}\,dx$$

$$j = 0, 1, 2, ..., N$$

M and L are altered to reflect the addition of the soma

$$\mathbf{M}_{0,0} = \mathbf{M}_{0,0} + A_s$$
$$\mathbf{L}_{0,0} = \mathbf{L}_{0,0} + A_s G_s$$
$$\vec{b}_0 = \vec{b}_0 + A_s G_s E_l$$

Branched neuron

Consider the case of the forked branch which consists of a root and two daughter branches. The finite element discretization requires the definition of the branched hat function. N_0 denotes the final node on the mesh of the root branch and b denotes the index of the daughter branch; there is a regular mesh for each branch:

$$\text{Root branch} : x_{0,0} < x_{0,1} < \ldots < x_{0,N_0}$$

$$\text{Daughter branch} : x_{b,0} = x_{0,N_0} < x_{b,1} < \ldots < x_{b,N_1}$$

The branched hat function is therefore:

$$\phi_{\text{branch}}(x) = \begin{cases} \phi_{0,N_0} = 1/h(x - (N_0 - 1)h) & \text{if } x_{0,N_0-1} < x < x_{0,N_0} \text{ on root branch,} \\ \phi_{b,0} = -1/h(x - (N_0 + 1)h) & \text{if } x_{0,N_0} \leq x < x_{b,1} \text{ on daughter branches,} \\ 0 & \text{otherwise.} \end{cases} \tag{6}$$

The rest of the hat functions remain the same

$$\begin{cases} \phi_{0,i}(x) = \phi_i(x) & i = 0, \ldots, N_0 - 1, \\ \phi_{b,i}(x) = \phi_i(x) & i = 1, \ldots, N_b. \end{cases}$$

At the branch point, $\chi(x)$ is

$$\chi_{\text{branch}}(x) \begin{cases} \chi_{0,N_0-1}(x) = 1 & \text{if } x_{0,N_0-1} < x < x_{0,N_0} \text{ on root branch,} \\ \chi_{b,0}(x) = 1 & \text{if } x_{0,N_0} \leq x < x_{b,1} \text{ on daughter branches,} \\ 0 & \text{otherwise} \end{cases}$$

The rest of the $\chi(x)$ remain the same

$$\begin{cases} \chi_{0,i}(x) = \chi_i(x) & i = 0, \ldots, N_0 - 2, \\ \chi_{b,i}(x) = \chi_i(x) & i = 1, \ldots, N_b - 1. \end{cases}$$

At the branch node on the discretized mesh, we now utilize the branched hat function

$$\frac{v_{0,N_0}(t)}{2Ra}\int_{(N_0-1)h}^{N_0h}\left(a_{0,N_0-1}\phi_{0,N_0-1}(x)+a_{0,N_0}\phi_{0,N_0}(x)\right)^2\frac{d\phi_{0,N_0}}{dx}(x)\frac{d\phi_j}{dx}(x)\,dx$$
$$+\frac{v_{0,N_0}(t)}{2Ra}\int_0^h\left(a_{0,N_0}\phi_{1,0}(x)+a_{1,1}\phi_{1,1}(x)\right)^2\frac{d\phi_{1,0}}{dx}(x)\frac{d\phi_j}{dx}(x)\,dx$$
$$+\frac{v_{0,N_0}(t)}{2Ra}\int_0^h\left(a_{0,N_0}\phi_{2,0}(x)+a_{2,1}\phi_{2,1}(x)\right)^2\frac{d\phi_{2,0}}{dx}(x)\frac{d\phi_j}{dx}(x)\,dx$$
$$+E_l\int_{(N_0-1)h}^{N_0h}G_{m_{0,N_0-1}}\chi_{0,N_0-1}\phi_j(x)\left(a_{0,N_0-1}\phi_{0,N_0-1}(x)+a_{0,N_0}\phi_{0,N_0}(x)\right)\ldots$$
$$\sqrt{1+\left(a_{0,N_0-1}\frac{d\phi_{0,N_0-1}}{dx}(x)+a_{0,N_0}\frac{d\phi_{0,N_0}}{dx}(x)\right)^2}\,dx$$
$$E_l\int_0^h G_{m_{1,0}}\chi_{1,0}\phi_j(x)\left(a_{0,N_0}\phi_{1,0}(x)+a_{1,1}\phi_{1,1}(x)\right)\ldots$$
$$\ldots\sqrt{1+\left(a_{0,N_0}\frac{d\phi_{1,0}}{dx}(x)+a_{1,1}\frac{d\phi_{1,1}}{dx}(x)\right)^2}\,dx$$
$$+E_l\int_0^h G_{m_{2,0}}\chi_{2,0}\phi_j(x)\left(a_{0,N_0}\phi_{2,0}(x)+a_{2,1}\phi_{2,1}(x)\right)\ldots$$
$$\ldots\sqrt{1+\left(a_{0,N_0}\frac{d\phi_{2,0}}{dx}(x)+a_{2,1}\frac{d\phi_{2,1}}{dx}(x)\right)^2}\,dx$$
$$=C_m\frac{dv_{0,N_0}(t)}{dt}(t)\int_{(N_0-1)h}^{N_0h}\phi_j(x)\phi_{0,N_0}(x)\left(a_{0,N_0-1}\phi_{0,N_0-1}(x)+a_{0,N_0}\phi_{0,N_0}(x)\right)\ldots$$
$$\ldots\sqrt{1+\left(a_{0,N_0-1}\frac{d\phi_{0,N_0-1}}{dx}(x)+a_{0,N_0}\frac{d\phi_{0,N_0}}{dx}(x)\right)^2}\,dx$$
$$+C_m\frac{dv_{0,N_0}(t)}{dt}(t)\int_0^h\phi_j(x)\phi_{0,N_0}(x)\left(a_{0,N_0}\phi_{1,0}(x)+a_{1,1}\phi_{1,1}(x)\right)\ldots$$
$$\ldots\sqrt{1+\left(a_{0,N_0}\frac{d\phi_{1,0}}{dx}(x)+a_{1,1}\frac{d\phi_{1,1}}{dx}(x)\right)^2}\,dx$$
$$+C_m\frac{dv_{0,N_0}(t)}{dt}(t)\int_0^h\phi_j(x)\phi_{0,N_0}(x)\left(a_{0,N_0}\phi_{2,0}(x)+a_{2,1}\phi_{2,1}(x)\right)\ldots$$
$$\ldots\sqrt{1+\left(a_{0,N_0}\frac{d\phi_{2,0}}{dx}(x)+a_{2,1}\frac{d\phi_{2,1}}{dx}(x)\right)^2}\,dx$$
$$+v_{0,N_0}(t)\int_{(N_0-1)h}^{N_0h}G_{m_{0,N_0-1}}\chi_{0,N_0-1}\phi_j(x)\phi_{0,N_0}(x)\left(a_{0,N_0-1}\phi_{0,N_0-1}(x)+a_{0,N_0}\phi_{0,N_0}(x)\right)\ldots$$
$$\ldots\sqrt{1+\left(a_{0,N_0-1}\frac{d\phi_{0,N_0-1}}{dx}(x)+a_{0,N_0}\frac{d\phi_{0,N_0}}{dx}(x)\right)^2}\,dx$$
$$+v_{0,N_0}(t)\int_0^h G_{m_{1,0}}\chi_{1,0}\phi_j(x)\phi_{1,0}(x)\left(a_{0,N_0}\phi_{1,0}(x)+a_{1,1}\phi_{1,1}(x)\right)\ldots$$
$$\ldots\sqrt{1+\left(a_{0,N_0}\frac{d\phi_{1,0}}{dx}(x)+a_{1,1}\frac{d\phi_{1,1}}{dx}(x)\right)^2}\,dx$$
$$+v_{0,N_0}(t)\int_0^h G_{m_{2,0}}\chi_{2,0}\phi_j(x)\phi_{2,0}(x)\left(a_{0,N_0}\phi_{2,0}(x)+a_{2,1}\phi_{2,1}(x)\right)\ldots$$
$$\ldots\sqrt{1+\left(a_{0,N_0}\frac{d\phi_{2,0}}{dx}(x)+a_{2,1}\frac{d\phi_{2,1}}{dx}(x)\right)^2}\,dx$$
$$j=0,1,2,...,N$$

Redefine $\mathbf{M}$,$\mathbf{K}$,$\mathbf{L}$ to account for the presence of the branched fiber

$$\mathbf{K}_{ij} = \int_{(N_0-1)h}^{N_0 h} \left(a_{0,N_0-1}\phi_{0,N_0-1}(x) + a_{0,N_0}\phi_{0,N_0}(x) \right)^2 \frac{d\phi_{0,N_0}}{dx}(x)\frac{d\phi_j}{dx}(x)\ dx$$
$$+ \int_0^h \left(a_{0,N_0}\phi_{1,0}(x) + a_{1,1}\phi_{1,1}(x) \right)^2 \frac{d\phi_{1,0}}{dx}(x)\frac{d\phi_j}{dx}(x)\ dx$$
$$+ \int_0^h \left(a_{0,N_0}\phi_{2,0}(x) + a_{2,1}\phi_{2,1}(x) \right)^2 \frac{d\phi_{2,0}}{dx}(x)\frac{d\phi_j}{dx}(x)\ dx$$

$$\mathbf{M}_{ij} = \int_{(N_0-1)h}^{N_0 h} \phi_j(x)\phi_{0,N_0}(x)\left(a_{0,N_0-1}\phi_{0,N_0-1}(x) + a_{0,N_0}\phi_{0,N_0}(x) \right)\ldots$$
$$\ldots\sqrt{1+\left(a_{0,N_0-1}\frac{d\phi_{0,N_0-1}}{dx}(x) + a_{0,N_0}\frac{d\phi_{0,N_0}}{dx}(x) \right)^2}\ dx$$
$$+ \int_0^h \phi_j(x)\phi_{0,N_0}(x)\left(a_{0,N_0}\phi_{1,0}(x) + a_{1,1}\phi_{1,1}(x) \right)\sqrt{1+\left(a_{0,N_0}\frac{d\phi_{1,0}}{dx}(x) + a_{1,1}\frac{d\phi_{1,1}}{dx}(x) \right)^2}\ dx$$
$$+ \int_0^h \phi_j(x)\phi_{0,N_0}(x)\left(a_{0,N_0}\phi_{2,0}(x) + a_{2,1}\phi_{2,1}(x) \right)\sqrt{1+\left(a_{0,N_0}\frac{d\phi_{2,0}}{dx}(x) + a_{2,1}\frac{d\phi_{2,1}}{dx}(x) \right)^2}\ dx$$

$$\mathbf{L}_{ij} = \int_{(N_0-1)h}^{N_0 h} G_{m_{0,N_0-1}}\chi_{0,N_0-1}\phi_j(x)\phi_{0,N_0}(x)\left(a_{0,N_0-1}\phi_{0,N_0-1}(x) + a_{0,N_0}\phi_{0,N_0}(x) \right)\ldots$$
$$\ldots\sqrt{1+\left(a_{0,N_0-1}\frac{d\phi_{0,N_0-1}}{dx}(x) + a_{0,N_0}\frac{d\phi_{0,N_0}}{dx}(x) \right)^2}\ dx$$
$$+ \int_0^h G_{m_{1,0}}\chi_{1,0}\phi_j(x)\phi_{1,0}(x)\left(a_{0,N_0}\phi_{1,0}(x) + a_{1,1}\phi_{1,1}(x) \right)\ldots$$
$$\ldots\sqrt{1+\left(a_{0,N_0}\frac{d\phi_{1,0}}{dx}(x) + a_{1,1}\frac{d\phi_{1,1}}{dx}(x) \right)^2}\ dx$$
$$+ \int_0^h G_{m_{2,0}}\chi_{2,0}\phi_j(x)\phi_{2,0}(x)\left(a_{0,N_0}\phi_{2,0}(x) + a_{2,1}\phi_{2,1}(x) \right)\ldots$$
$$\ldots\sqrt{1+\left(a_{0,N_0}\frac{d\phi_{2,0}}{dx}(x) + a_{2,1}\frac{d\phi_{2,1}}{dx}(x) \right)^2}\ dx$$

$$\vec{b}_j = E_l \int_{(N_0-1)h}^{N_0 h} G_{m_{0,N_0-1}}\chi_{0,N_0-1}\phi_j(x)\left(a_{0,N_0-1}\phi_{0,N_0-1}(x) + a_{0,N_0}\phi_{0,N_0}(x) \right)\ldots$$
$$\ldots\sqrt{1+\left(a_{0,N_0-1}\frac{d\phi_{0,N_0-1}}{dx}(x) + a_{0,N_0}\frac{d\phi_{0,N_0}}{dx}(x) \right)^2}\ dx$$

$$+E_l \int_0^h G_{m_{1,0}\chi_{1,0}\phi_j}(x)\Big(a_{0,N_0}\phi_{1,0}(x)+a_{1,1}\phi_{1,1}(x)\Big)\dots$$
$$\dots\sqrt{1+\left(a_{0,N_0}\frac{d\phi_{1,0}}{dx}(x)+a_{1,1}\frac{d\phi_{1,1}}{dx}(x)\right)^2}\,dx$$
$$+E_l \int_0^h G_{m_{2,0}\chi_{2,0}\phi_j}(x)\Big(a_{0,N_0}\phi_{2,0}(x)+a_{2,1}\phi_{2,1}(x)\Big)\dots$$
$$\dots\sqrt{1+\left(a_{0,N_0}\frac{d\phi_{2,0}}{dx}(x)+a_{2,1}\frac{d\phi_{2,1}}{dx}(x)\right)^2}\,dx$$

Evaluate the integrals

$$i=j=N_0 \text{ at the branch point}$$
$$\mathbf{K}_{ij}=\frac{1}{3h}\Big[(3a_{0,N_0}^2+a_{0,N_0-1}^2+a_{1,1}^2+a_{2,1}^2)+a_{0,N_0}(a_{0,N_0-1}+a_{1,1}+a_{2,1})\Big]$$
$$\mathbf{M}_{ij}=\frac{1}{12}\Big[(a_{0,N_0-1}+3a_{0,N_0})\sqrt{h^2+(a_{0,N_0-1}-a_{0,N_0})^2}+(3a_{0,N_0}+a_{1,1})\sqrt{h^2+(a_{0,N_0}-a_{1,1})^2}$$
$$+(3a_{0,N_0}+a_{2,1})\sqrt{h^2+(a_{0,N_0}-a_{2,1})^2}\Big]$$

$$\mathbf{L}_{ij}=\frac{1}{12}\Big[G_{m_{0,N_0-1}}(a_{0,N_0-1}+3a_{0,N_0})\sqrt{h^2+(a_{0,N_0-1}-a_{0,N_0})^2}+G_{m_{1,1}}(3a_{0,N_0}+a_{1,1})\sqrt{h^2+(a_{0,N_0}-a_{1,1})^2}$$
$$+G_{m_{2,1}}(3a_{0,N_0}+a_{2,1})\sqrt{h^2+(a_{0,N_0}-a_{2,1})^2}\Big]$$

$$\vec{b}_i=\frac{E_l}{12}\Big[G_{m_{0,N_0-1}}(2a_{0,N_0-1}+4a_{0,N_0})\sqrt{h^2+(a_{0,N_0-1}-a_{0,N_0})^2}+G_{m_{1,1}}(4a_{0,N_0}+2a_{1,1})\sqrt{h^2+(a_{0,N_0}-a_{1,1})^2}$$
$$+G_{m_{2,1}}(4a_{0,N_0}+2a_{2,1})\sqrt{h^2+(a_{0,N_0}-a_{2,1})^2}\Big]$$

$$j=1 \text{ on daughter branch 1}$$
$$\mathbf{K}_{ij}=-\frac{1}{3h}(a_{N_0}^2+a_{N_0}a_{1,1}+a_{1,1}^2)$$
$$\mathbf{M}_{ij}=\frac{1}{12}(a_{N_0}+a_{1,1})\sqrt{h^2+(a_{N_0}-a_{1,1})^2}$$
$$\mathbf{L}_{ij}=\frac{G_{m_{1,1}}}{12}(a_{N_0}+a_{1,1})\sqrt{h^2+(a_{N_0}-a_{1,1})^2}$$

$$j=1 \text{ on daughter branch 2}$$
$$\mathbf{K}_{ij}=-\frac{1}{3h}(a_{N_0}^2+a_{N_0}a_{2,1}+a_{2,1}^2)$$

$$\mathbf{M}_{ij} = \frac{1}{12}(a_{N_0} + a_{2,1})\sqrt{h^2 + (a_{N_0} - a_{2,1})^2}$$

$$\mathbf{L}_{ij} = \frac{G_{m_{1,1}}}{12}(a_{N_0} + a_{2,1})\sqrt{h^2 + (a_{N_0} - a_{2,1})^2}$$

Appendix B: Time Stepping

Efficient half-implicit Euler method

The Crank-Nicolson method is equivalent to performing two Euler integral approximations using a 'half' time step.

Implicit step from t to $t + \Delta t/2$

$$\vec{V}(t + \Delta t/2) = \vec{V}(t) + \frac{\Delta t/2}{C_m}\left[\frac{1}{2R_a}\mathbf{B}\vec{V}(t + \Delta t/2) - \vec{G}_l(\vec{V}(t + \Delta t/2) - E_l)\right]$$

Explicit step from $t + \Delta t/2$ to $t + \Delta t$

$$\vec{V}(t + \Delta t) = \vec{V}(t + \Delta t/2) + \frac{\Delta t/2}{C_m}\left[\frac{1}{2R_a}\mathbf{B}\vec{V}(t + \Delta t/2) - \vec{G}_l(\vec{V}(t + \Delta t/2) - E_l)\right]$$

Combining the previous two equations gives

$$\vec{V}(t + \Delta t) = \vec{V}(t) + \frac{\Delta t}{2C_m}\left[\frac{1}{R_a}\mathbf{B}\vec{V}(t + \Delta t/2) - \vec{G}_l(2\vec{V}(t + \Delta t/2) - 2E_l)\right]$$

The full Crank-Nicolson rewritten is

$$\vec{V}(t + \Delta t) = \vec{V}(t) + \frac{\Delta t}{2C_m}\left[\frac{1}{2R_a}\mathbf{B}\left(\vec{V}(t + \Delta t) + \vec{V}(t)\right) - \vec{G}_l\left(\vec{V}(t + \Delta t) + \vec{V}(t) - 2E_l\right)\right]$$

Comparing the previous two equations gives a simple explicit expression to step from $t + \Delta t/2$ to $t + \Delta t$

$$\begin{aligned} 2V(t + \Delta t/2) &= V(t + \Delta t) + V(t) \\ \Rightarrow V(t + \Delta t) &= V(t) + 2V(t + \Delta t/2) \end{aligned}$$

This is also obvious from realizing Crank-Nicolson takes the average of the two integral approximations.

References

[1] S. Cox. The Synapse. (www.caam.rice.edu/~caam415) 2005.

[2] S. Cox; J. Raol. Recovering the passive properties of tapered dendrites from single and dual potential recordings. Mathematical Biosciences 190 (2004) 9-37.

[3] Gockenbach, M. Partial Differential Equations: Analytical and Numerical Methods. SIAM, 2002.

[4] Hines, M. Efficient Computation of Branched Nerve Equations. Int. J. Bio-Medical Computing (15) (1984) 69-76.

[5] Koch, C. Biophysics of Computation: Information Processing in Single Neurons. Oxford University Press, 1999.

[6] Traub, R.; Miles, R. Neuronal Networks of the Hippocampus. Cambridge University Press, 1991.