Numerical Methods

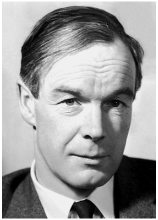
Accuracy, stability, speed

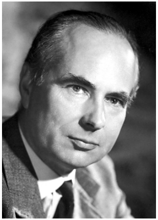
Robert A. McDougal

Yale School of Medicine

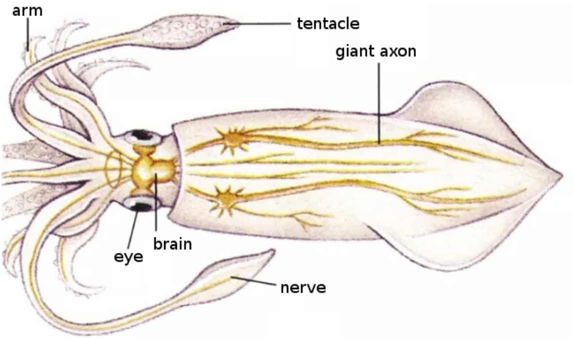
9 August 2018

Hodgkin and Huxley: squid giant axon experiments





Top: Alan Lloyd Hodgkin;
Bottom: Andrew Fielding
Huxley. Images from Wikipedia.



Adapted from Pearson Education 2009.

Hodgkin and Huxley equations

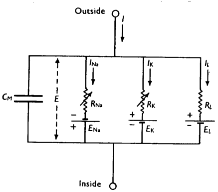
Top: Alan Lloyd Hodgkin; Bottom: Andrew Fielding Huxley. Images from Wikipedia.

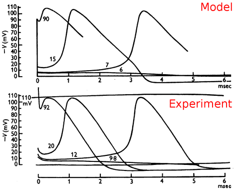
$$C\frac{dV}{dt} = -\left(g_{Na}m^3h(V - E_{Na}) + g_K n^4(V - E_K) + g_\ell(V - E_\ell)\right)$$

$$\frac{dm}{dt} = \alpha_m(V)(1 - m) - \beta_m(V)m$$

$$\frac{dh}{dt} = \alpha_h(V)(1 - h) - \beta_h(V)h$$

$$\frac{dn}{dt} = \alpha_n(V)(1 - n) - \beta_n(V)n$$





Adapted from Hodgkin and Huxley 1952.

What does it mean?

Electronics 101

Current

Current is the movement of charge. In electronics, current is carried by the movement of electrons. In neurons, current flows across the membrane by the movement of ions. These ions can be positively or negatively charged.

K^+ Na^+ Ca^{++} Cl^-

Resistors = Conductors

A **resistor** is a material that impedes current flow. This includes essentially all materials. For those materials obeying **Ohm's law**,

$$v = IR$$

where v is the voltage drop across the resistor, I is the current, and R is the **resistance** (this may be constant or a function of time).

This may alternatively be written as

$$I = gV$$

where $g = 1/R$ is the **conductance**.

Ion channels

Ion channels allow current to pass in the form of moving ions. They are therefore resistors. The resistance varies over time.

Capacitors

A **capacitor** accumulates charge according to

$$CV = Q$$

where Q is the charge, V is the potential, and C is the capacitance.

The capacitive current is the rate at which charge is being stored on the current, dQ/dt. Thus differentiating both sides of the above, we find

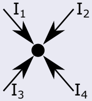
$$C\frac{dV}{dt} = \frac{dQ}{dt} = I.$$

Cell membrane

Charged ions accumulate along a neuron's membrane. It is therefore a capacitor.

Kirchhoff's Current Law

The algebraic sum of currents in a network of conductors meeting at a point is zero.

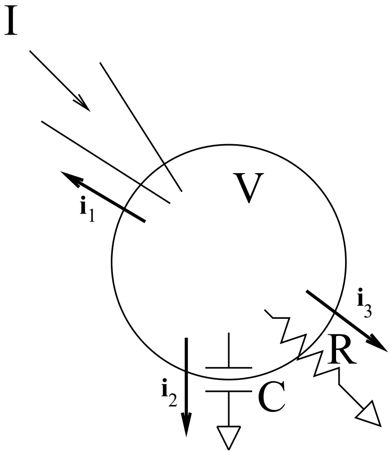


$$\sum_k I_k = 0.$$

Wording from https://en.wikipedia.org/wiki/Kirchhoff%27s_circuit_laws

Putting it together: the electronics of a neuron

Consider a simplified cell with three currents:



By Kirchoff,

$$\begin{aligned} 0 &= i_1 + i_2 + i_3 \\ &= -I + C\frac{dV}{dt} + gV \end{aligned}$$

Rearranging terms, we conclude:

$$C\frac{dV}{dt} = -gV + I.$$

The Hodgkin-Huxley equations account for a pull on ions due to the balance of chemical and electrical gradients. This approximately acts as a battery with potential E associated with each resistor and leads to terms of the form g(V - E).

Solving a differential equation

Consider the differential equation

$$C\frac{dV}{dt} = -gV + I, V(0) = V_0$$

We can solve this for $V(t)$ by **separation of variables**:

$$\frac{dV}{I - gV} = \frac{dt}{C}$$
$$\int \frac{dV}{I - gV} = \int \frac{dt}{C}$$
$$\frac{-1}{g}\ln|I - gV| = \frac{t}{C} + c_1$$
$$I - gV = c_2 e^{-gt/C}$$

Therefore,

$$V = \frac{1}{g}\left(I - c_2 e^{-gt/C}\right).$$

We can then solve for c_2 by plugging in $V(0) = V_0$:

$$V_0 = \frac{1}{g}(I - c_2)$$

so

$$c_2 = I - gV_0$$

and thus

$$V = \frac{1}{g}\left(I - (I - gV_0)e^{-gt/C}\right).$$

Note: This is a lot of work and is only possible because the equation is simple. This type of equation appears in leaky integrate and fire and is the basis of the `cnexp` solver.

To solve general differential equations, we must use numerical techniques.

Here we're assuming g is a constant. This is not true for voltage gated ion channels.

In the **Explicit Euler** method, we approximate

$$\frac{dy}{dt} \approx \frac{\Delta y}{\Delta t}$$

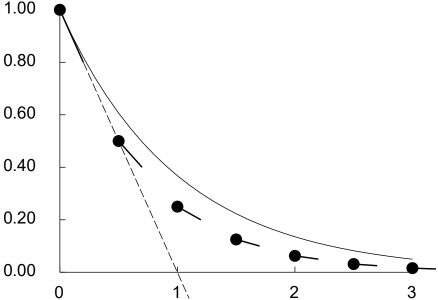
for some small time step Δt and estimate the function at a series of time points. Here $\Delta y_n = y_{n+1} - y_n$ and $\Delta t_n = t_{n+1} - t_n$.

Then starting from some initial point (t_0, y_0), we approximate $\frac{dy}{dt} = f(t, y)$ as $\frac{\Delta y_n}{\Delta t_n} = f(t_n, y_n)$ and thus

$$\Delta y_n = \Delta t_n f(t_n, y_n)$$

and therefore

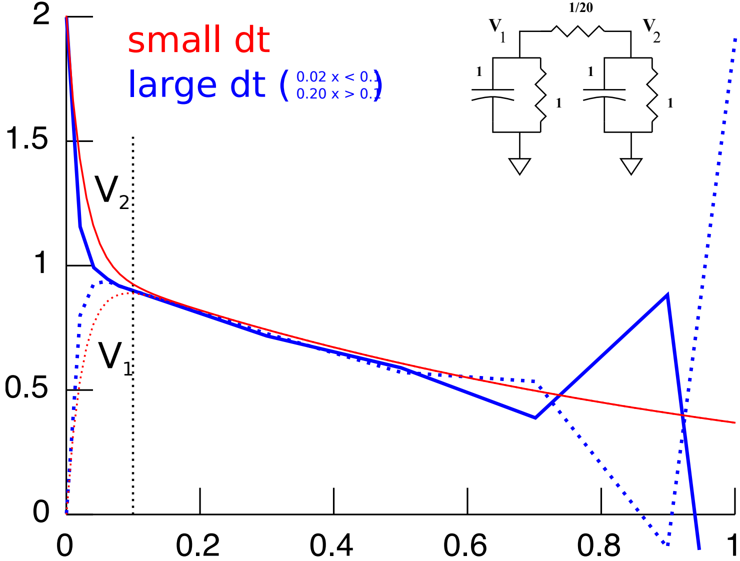
$$y_{n+1} = y_n + \Delta t_n f(t_n, y_n).$$



Explicit Euler starts at a point, moves in the direction of the tangent line (slope dy/dt) for a time Δt, then repeats.

Explicit Euler is numerically unstable

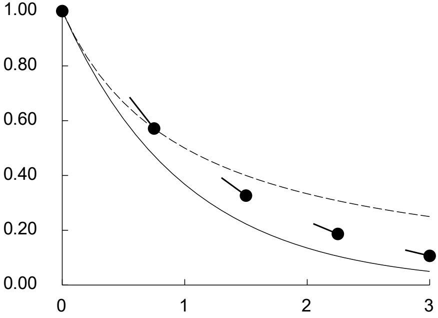
If the time step in Explicit Euler is too large, the solution will be unstable:



The **Implicit Euler** method is almost the same as the Explicit Euler method except instead of evaluating at $f(t_n, y_n)$, we evaluate at $f(t_{n+1}, y_{n+1})$. That is,

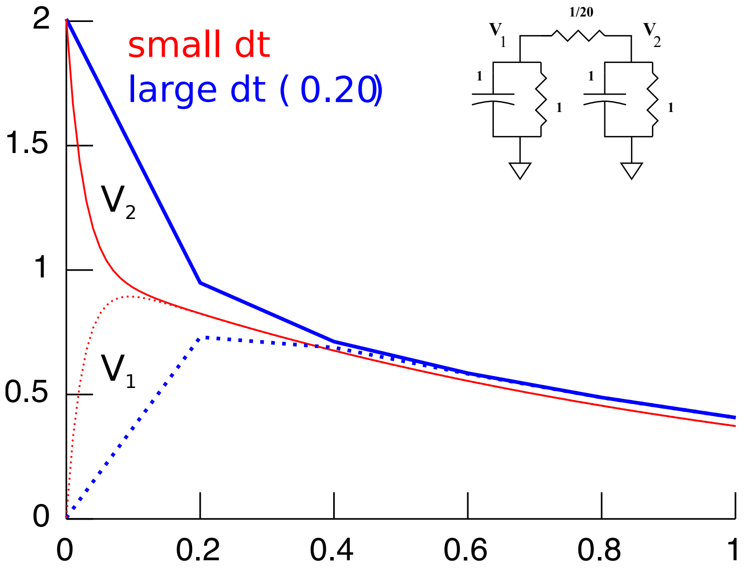
$$y_{n+1} = y_n + \Delta t_n f(t_{n+1}, y_{n+1}).$$

Note that y_{n+1} is on both sides, and thus we have an algebraic equation that must be solved to find y_{n+1}.



Implicit Euler finds a new point such that if we moved in the direction of the tangent line (slope dy/dt) backward in time by Δt, we would get where we started.

Implicit Euler is numerically stable



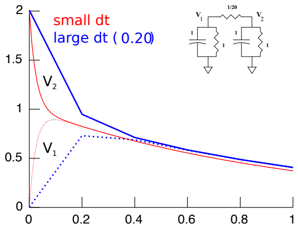
As Implicit Euler is numerically stable, it is NEURON's default integration method.

Accuracy of Implicit Euler

Note that the solutions found with a small *dt* and a large *dt* are different, even after the initial rapid change.

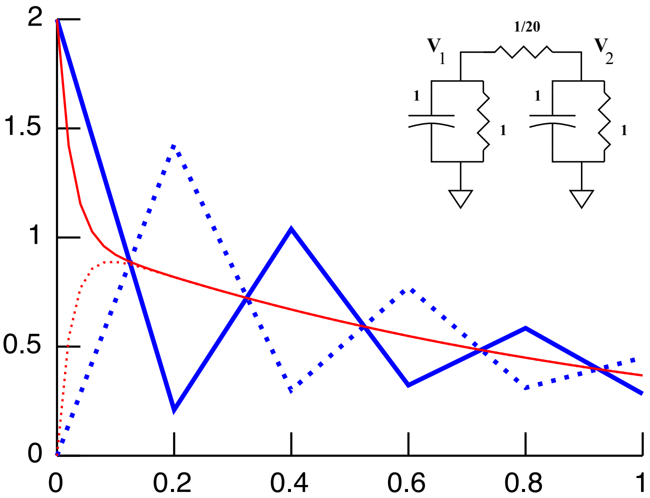
One can prove that halving *dt* will approximately halve the difference between the computed value and the true value.

Thus **Implicit Euler is a first order method**.



Error convergence estimates are true in the limit as $dt \rightarrow 0$.

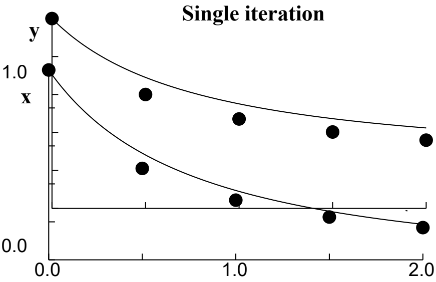
Crank-Nicolson is stable but can oscillate

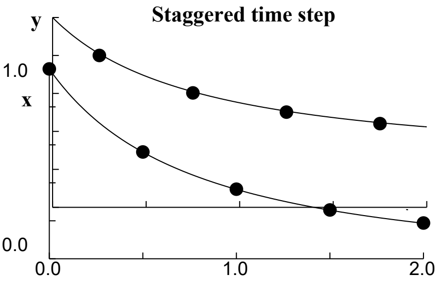


NEURON also supports the second-order Crank-Nicolson method (`h.secondorder=2`). The solution is stable and converges faster than Implicit or Explicit Euler, but it can exhibit oscillations.

If `h.secondorder=2`, then membrane potentials are second order correct at time t, currents at $t - dt/2$, and channel conductances at $t + dt/2$. To plot these correctly in NEURON, use a voltage axis, current axis, or state axis, respectively.

$$\dot{x} = -1.4xy, \quad \dot{y} = -xy$$

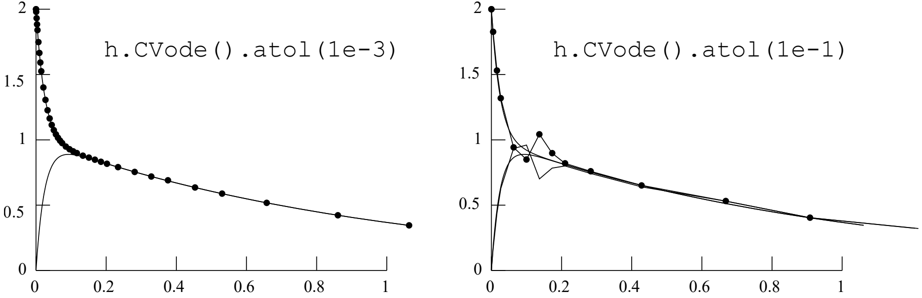




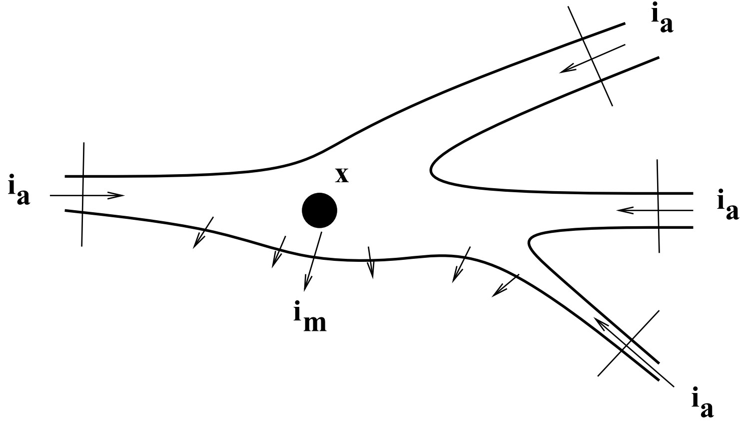
Variable time steps

So far, we have considered numerical error as a function of the time step *dt*. We can instead choose an error tolerance and use that to pick a new *dt* at each time step.

NEURON provides the `CVode` object for enabling variable step simulation.

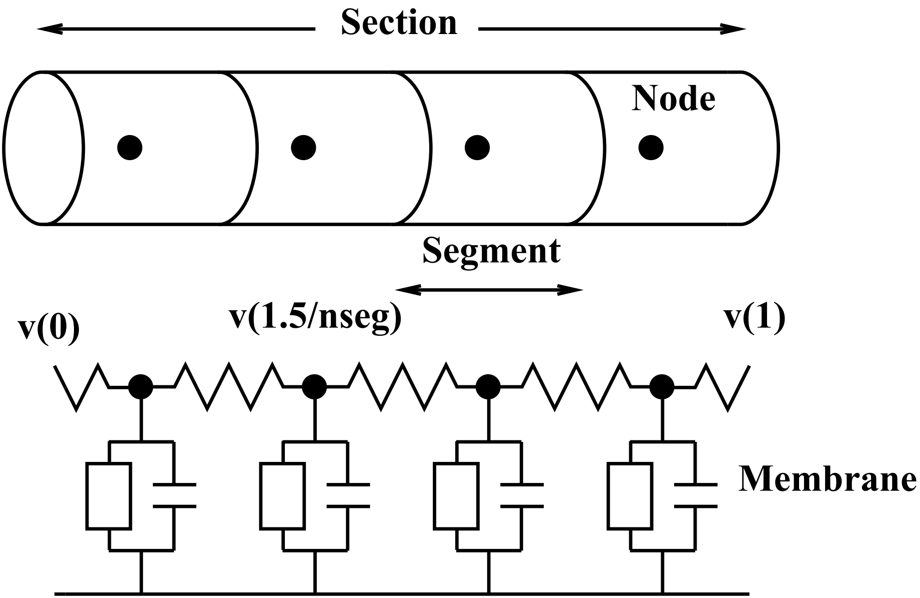


Incorporating space



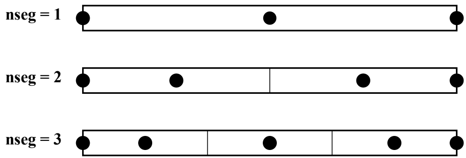
$$\int i_m = \sum i_a$$

$$c_j \frac{dv_j}{dt} + i_j = \sum_k \frac{v_k - v_j}{r_{jk}}$$


Section
Node
Segment
v(0)
v(1.5/nseg)
v(1)
Membrane

Improving accuracy by increasing nseg

Improve accuracy by reducing the size of spatial compartments. In NEURON, do this by increasing `nseg`, the number of segments:



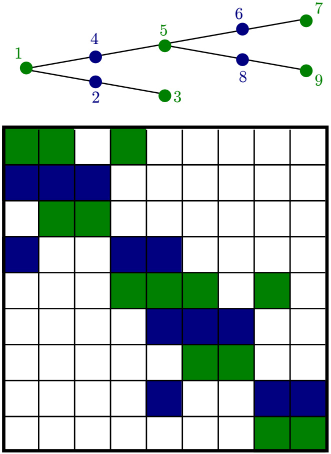

```
for sec in h.allsec():
    sec.nseg *= 3
```

Note that you must multiply `nseg` by an **odd** number to preserve the location of the computed values, which is essential to testing convergence.

Trees can be solved stably in $\mathcal{O}(n)$ Only unstable methods can solve arbitrary shapes in $\mathcal{O}(n)$

To solve $A\Delta y = b$ where y and b have n entries (e.g. if we want to solve for 4 variables at $n/4$ points) takes time proportional to:

- n^3 via Gaussian Elimination
- $n^{\log_2 7}$ via Strassen (1969)
- n if A corresponds to a "tree-matrix" (e.g. a neuron) discretized in a certain way (right).



McDougal et al 2013