Notes on numerical methods for EMBO Course

Calina Copos (University of California, Davis)

October 15-22, 2016

FitzHugh-Nagumo equations

$$\begin{aligned} \frac{dv}{dt} &= v - \frac{v^3}{3} - w + I \\ \frac{dw}{dt} &= \epsilon(v + a + bw) \end{aligned}$$

is a two-dimensional simplification of the Hodgkin-Huxley model of spike generation in axons. Here $v(t)$ si the membrane potential (mV), $w(t)$ is the recovery potential, and I is the magnitude of the current stimulus. The system of ODEs is solved using ode45, a Matlab numerical integrating package that uses a variable step size fourth or fifth order Runge-Kutta method to approximate the solution of the system of first-order ODEs.

Short note on Runge-Kutta integration method. Instead of depending on the derivatives of the ODE, Runge-Kutta methods just use additional evaluations of the ODE, i.e. use a weighted average of the function to integrate the function at different time instances. For example, Runge-Kutta 4-5 method uses the 4th order approximation of the equation you want to integrate and limits the propagation of numerical errors on the order of the 5th time step. Consider $x'(t) = f(t, x)$ with the initial condition $x(t_0) = x_0$. Let h denote the time step size and $t_i = t_0 + ih$. Then, the following formula:

$$\begin{aligned} w_0 &= x_0 \\ k_1 &= hf(t_i, w_i) \\ k_2 &= hf(t_i + \frac{h}{2}, w_i + \frac{k_1}{2}) \\ k_3 &= hf(t_i + \frac{h}{2}, w_i + \frac{k_2}{2}) \\ k_4 &= hf(t_i + h, w_i + k_3) \\ w_{i+1} &= w_i + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4) \end{aligned}$$

computes an approximate solution, that is $w_i \approx x(t_i)$.

Non-stiff vs. stiff ODEs. Shortly, "stiff" equations are equations where the derivative, $\dot{x}(t)$, and the solution, $x(t)$, grow rapidly in time. This forces the numerical integrating package to take a lot of very small time steps and result in simulations with long execution time. In such instances, it is useful to know that there are other numerical integrating packages that can solve stiff systems at the cost of lower accuracy (e.g. ode23s, ode15s, etc.).

Reaction-diffusion advection equation

Let $a = a(x, y, t)$, $b = b(x, y, t)$ be two-dimensional quantities at a time instant, e.g. concentration of a signaling protein on a two-dimensional cell domain, Ω. The active form of the signaling protein, a, and the inactive form, b, diffuse in the domain with diffusion coefficients D_a and D_b, respectively. The two forms exchange at a rate $g(a, b)$. The dynamics of the signaling model are thus governed by a pair of reaction-diffusion advection equations:

$$a_t + \boldsymbol{u} \cdot \nabla a = D_a \Delta a + g(a, b) \quad (1)$$

$$b_t + \boldsymbol{u} \cdot \nabla b = D_b \Delta b - g(a, b) \quad (2)$$

where $\boldsymbol{u}$ is the constant flow velocity. The advection term esentially counts for the fact that the domain Ω moves with respect to the lab coordinated, and carries the biochemistry along. For $g(a, b)$, we have assumed a positive feedback enhancing conversion of the inaction form b to the active form a, but a constant rate ν of converting active into inactive form,

$$g(a, b) = \mu \left(1 + \frac{a^2}{K + a^2}\right) b - \nu a$$

Discretization for a rectangular domain

- **Domain.** Let us consider the domain of a unit square, $\Omega = \{(x, y) : 0 \leq x, y \leq 1\}$. We choose a grid for which the grid points are integered in the x, y-directions, that is,

  $$x_i = i\Delta x, y_j = j\Delta y$$

  where $\Delta x = \Delta y = 1/N$, and $i = 1, 2, ..., N, j = 1, 2, ..., N$. Similarly, time is discretized with $t_n = n\Delta t$, where $\Delta t = 0.0001$. Let the discrete values be denotes by $a_{ij}^n \approx a(x_i, y_j, t_n)$, $b_{ij}^n \approx b(x_i, y_j, t_n)$, and $g_{ij}^n \approx g(x_i, y_j, t_n)$.

- **Laplacian operator.** Using a centered finite difference to discretize the two-dimensional Laplacian operator, Δa, we have

  $$(\Delta a)_{ij}^n = \frac{a_{i+1,j}^n - 2a_{ij}^n + a_{i-1,j}^n}{\Delta x^2} + \frac{a_{i,j+1}^n - 2a_{ij}^n + a_{i,j-1}^n}{\Delta y^2} + \mathcal{O}(\Delta x^2, \Delta x \Delta y, \Delta y^2)$$

- **First-order time derivative.** Using a forward finite difference (Forward Euler) to discretize the operator a_t, we have

  $$(a_t)_{ij}^n = \frac{a_{ij}^{n+1} - a_{ij}^n}{\Delta t} + \mathcal{O}(\Delta t^2)$$

- **Advection equation.** Consider the linear advection equation,

  $$a_t + \boldsymbol{u} \cdot \nabla a = 0$$

  for a wave propagating along the x-axis with a constant velocity $\boldsymbol{u} = (u_1, u_2)$. For $\boldsymbol{u} \geq 0$, the advection equation is discretized using the first-order upwind scheme:

  $$(a_t)_{ij}^n + (\boldsymbol{u} \cdot \nabla a)_{ij}^n = \frac{a_{ij}^{n+1} - a_{ij}^n}{\Delta t} + u_1 \frac{a_{ij}^n - a_{i-1,j}^n}{\Delta x} + u_2 \frac{a_{ij}^n - a_{i,j-1}^n}{\Delta y} + \mathcal{O}(\Delta t^2, \Delta x^2, \Delta x \Delta y, \Delta y^2)$$

  Conditionally stable if the CFL condition is met, $\left|\frac{a_1 \Delta t}{\Delta x}\right| + \left|\frac{a_2 \Delta t}{\Delta y}\right| \leq 1$.

With the further assumption that $\Delta x = \Delta y$. the system of reaction-diffusion advection equations can be discretized to give a first-order approximation in space and time of the continuous equations:

$$\begin{aligned} a_{ij}^{n+1} &= a_{ij}^n + \frac{\Delta t}{\Delta x}\Big(u_1 a_{i-1,j}^n - (u_1+u_2)a_{ij}^n + u_2 a_{i,j-1}^n\Big) + \\ &+ \frac{D_a \Delta t}{\Delta x^2}\Big(a_{i+1,j}^n + a_{i,j+1}^n - 4a_{ij}^n + a_{i-1,j}^n + a_{i,j-1}^n\Big) + \Delta t\, g(a_{ij}^n, b_{ij}^n) \\ b_{ij}^{n+1} &= b_{ij}^n + \frac{\Delta t}{\Delta x}\Big(u_1 b_{i-1,j}^n - (u_1+u_2)b_{ij}^n + u_2 b_{i,j-1}^n\Big) + \\ &+ \frac{D_b \Delta t}{\Delta x^2}\Big(b_{i+1,j}^n + b_{i,j+1}^n - 4b_{ij}^n + b_{i-1,j}^n + b_{i,j-1}^n\Big) + \Delta t\, g(a_{ij}^n, b_{ij}^n) \end{aligned}$$

Discretization for a disk domain

- **Domain.** Let us consider the domain of a unit square, $\Omega = \{(x,y) : x^2 + y^2 \leq 1\}$. We choose a grid for which the grid points are half-integered in the radial-direction and integered in the azimuthal direction, that is,

  $$r_i = \Big(i - \frac{1}{2}\Big)\Delta r, \theta_j = (j-1)\Delta\theta$$

  where $\Delta r = \frac{2}{2N+1}$, $\Delta\theta = \frac{2\pi}{M}$, and $i = 1, 2, ..., N+1$, $j = 1, 2, ..., M+1$. Similarly, time is discretized with $t_n = n\Delta t$, where $\Delta t = 0.0001$. Let the discrete values be denotes by $a_{ij}^n \approx a(r_i, \theta_j, t_n)$, $b_{ij}^n \approx b(r_i, \theta_j, t_n)$, and $g_{ij}^n \approx g(x_i, \theta_j, t_n)$.

- **Laplacian operator.** In polar coordinates, $\Delta a = \frac{1}{r}\Big(\frac{\partial}{\partial r}(r\, a_r)\Big) + \frac{1}{r^2}a_{\theta\theta} = A_r a + D_{rr} a A_\theta$ where the operators are $A_r = \frac{1}{r}\Big(\frac{\partial}{\partial r}(r\,\frac{\partial}{\partial r})\Big)$, $D_{rr} = \frac{1}{r^2}$, and $A_\theta = \frac{\partial^2}{\partial\theta^2}$. Both A_r and A_θ are sparse diagonal matrices with a 3-point stencil while D_{rr} is a diagonal matrix.

  Using a centered finite difference, we have

  $$(A_r a)_{ij}^n = \frac{1}{2(\Delta r)^2}\Big[\frac{r_i + \Delta r/2}{r_i}a_{i+1,j}^n - 2a_{i,j}^n + \frac{r_i - \Delta r/2}{r_i}a_{i-1,j}^n\Big]$$

  $$(A_\theta a)_{ij}^n = \frac{1}{\Delta\theta^2}(a_{i,j+1}^n - 2a_{ij}^n + a_{i,j-1}^n)$$

  Must enforce that the Laplacian has periodic boundary conditions in the azimuthal direction.

- **Advection equation.** Consider the linear advection equation,

  $$a_t + \boldsymbol{u}\cdot\nabla a = 0$$

  for a wave propagating along the radial direction with a constant velocity $\boldsymbol{u} = (u_1, 0)$. For $\boldsymbol{u} \geq 0$, the advection equation is discretized using the first-order upwind scheme:

  $$(a_t)_{ij}^n + (\boldsymbol{u}\cdot\nabla a)_{ij}^n = \frac{a_{ij}^{n+1} - a_{ij}^n}{\Delta t} + u_1\frac{a_{ij}^n - a_{i-1,j}^n}{\Delta r} + \mathcal{O}(\Delta t^2, \Delta r^2)$$

Elastic representation of a cell[1]

[1] C code has been validated on both Mac OSX and Ubuntu systems.