Yasmeen Hussain
REU Report
Fall 2007

Diffusion, Flow, and Partial Differential Equations: An exploration in advanced biochemical reaction systems

Goals and Purpose

In biochemical reaction systems where reaction rates are governed by the Law of Mass Action, the Goldbeter-Koshland function, and Michaelis-Menten analysis, ordinary differential equations will suffice to build working models of these systems. However, not all biochemical functions are so simple. This semester, Professor Aaron Fogelson and I looked at reactions involving diffusion and blood flow, where models are built using partial differential equations.

We first studied the biological systems involved in diffusion and such similar reactions. Then I looked at the construction of partial differential equations describing these reactions and at the manual methods of solving systems of diffusion equations, including Forward (Explicit) Euler Method, Backward (Implicit) Euler Method, and the combinatory Crank-Nicholson (Implicit) Scheme.

The focus of my work then shifted to computational solutions. I used MATLAB software to solve and manipulate the numerical solutions of partial differential equations, specifically diffusion-modeling equations. Applying numerical solution methods to partial differential equations converts the equations into ODEs, thus making them workable and solvable. Doing so, I was able to solve diffusion equations and find at which points the algorithm for explicit solutions is stable.

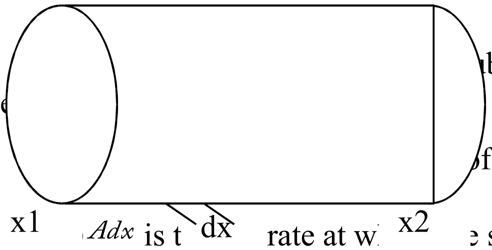
Background

When the inner wall of a blood vessel is injured, the body tries to fix the injury by inducing coagulation. Coagulation is where proteins join and gel to fill the hole by using a series of enzymatic reactions which control the portion of the enzyme thrombin, which causes gelling. These enzymatic reactions are regulated by both positive and inhibitory feedback loops. The final enzyme in these reactions, thrombin, acts by cutting off pieces of a material called fribrinogen into fibrin monomers. This exposes a binding site so that the fibrin monomers can bind in different arrangements and make strands of fibrin polymer. These fibrin polymers bind side by side to form the gel designed to cover inner blood vessel wall injuries.

The reactions in this system deal with diffusion and blood flow (advection) and respond with respect to time and concentration gradient of various enzymes and chemicals. An enzyme activates its precursor, called zymogen, to make more of itself, in a process called autocatalysis. When analyzing this system, it is assumed that all enzyme activation activity takes place on membrane patches as opposed to in the surrounding solution.

Diffusion Equation-Example

The diffusion equation is often used as a simple model built using partial differential equations. Concentration distribution can be shown in a segmented rod or box model:



If bstance at a point x and a time t, then
is th and time t. Following this line of logic,
f substance over the length of the segment and
$\frac{d}{dt}\int_{x1} Adx$ is t rate at w e same substance crosses into the segment from the outside. Flux is de as the amount of a substance crossing from the left of some point x to the right of that same location x per unit area per unit time. Here, flux will be defined as J, where $\frac{d}{dt}\int_{x1}^{x2} c(x,t)Adx = J(x1,t)A - J(x2,t)A$ and where J>0 denotes the substance moving to the right and J<0 denotes flow in the opposite direction. By rearranging this equation, it is found that $\int_{x1}^{x2}\left(\frac{\partial c}{\partial t}(x,t) + \frac{\partial J}{\partial x}(x,t)\right)dx = 0$ is true for all arbitrary x1 and x2. Integrating thereafter, we find the equation $\frac{\partial c}{\partial t}(x,t) + \frac{\partial J}{\partial x}(x,t) = 0$, supported by the Conservation Law. Combining this result with Fick's Law $J(x,t) = -D\left[\frac{\partial c}{\partial x}\right]$, we obtain the ideal diffusion equation, $\frac{\partial c}{\partial t} = D\frac{\partial^2 c}{\partial x^2}$.

Forward Euler (Explicit)

Given a differential equation such as the diffusion equation shown above, we can discretize the solution's domain as the first step of a finite-difference approximation.

$$\frac{\partial c}{\partial t}(x,t) \approx \frac{c(x,t+\Delta t) - c(x,t)}{\Delta t} = D\left[\frac{c(x-\Delta x,t) - 2c(x,t) + c(x+\Delta x,t)}{(\Delta x)^2}\right] \approx \frac{\partial c^2}{\partial^2 x}$$

Thus, if we know

$$\begin{gathered}c(x,0) = f(x)\\ c(x,t)\\ c(x,t+\Delta t)\\ c(x,t+2\Delta t)\\ x_j = j \cdot \Delta x\\ t_n = n \cdot \Delta t\\ c(x_j,t_{n+1}) = D\frac{\Delta t}{(\Delta x)^2}c(x_{j-1},t_n) - D\frac{\Delta t}{(\Delta x)^2}2c(x_j,t_n) + D\frac{\Delta t}{(\Delta x)^2}c(x_{j+1},t_n) + c(x_j,t_n)\\ k = \Delta t\\ h = \Delta x\\ s = \frac{k}{h^2}\end{gathered}$$

, we can use this to find

$$\begin{gathered}c(x,0) = f(x)\\ c(x,t)\\ c(x,t+\Delta t)\\ c(x,t+2\Delta t)\\ x_j = j \cdot \Delta x\\ t_n = n \cdot \Delta t\\ c(x_j,t_{n+1}) = D\frac{\Delta t}{(\Delta x)^2}c(x_{j-1},t_n) - D\frac{\Delta t}{(\Delta x)^2}2c(x_j,t_n) + D\frac{\Delta t}{(\Delta x)^2}c(x_{j+1},t_n) + c(x_j,t_n)\\ k = \Delta t\\ h = \Delta x\\ s = \frac{k}{h^2}\end{gathered}$$

and so on for certain t's and x's.

More specifically, these discrete variables are fitted as so,

$$\begin{gathered}c(x,0) = f(x)\\ c(x,t)\\ c(x,t+\Delta t)\\ c(x,t+2\Delta t)\\ x_j = j \cdot \Delta x\\ t_n = n \cdot \Delta t\\ c(x_j,t_{n+1}) = D\frac{\Delta t}{(\Delta x)^2}c(x_{j-1},t_n) - D\frac{\Delta t}{(\Delta x)^2}2c(x_j,t_n) + D\frac{\Delta t}{(\Delta x)^2}c(x_{j+1},t_n) + c(x_j,t_n)\\ k = \Delta t\\ h = \Delta x\\ s = \frac{k}{h^2}\end{gathered}$$

, where j is any integer and t is any positive integer.
Using these new definitions, our function can be reworked as:

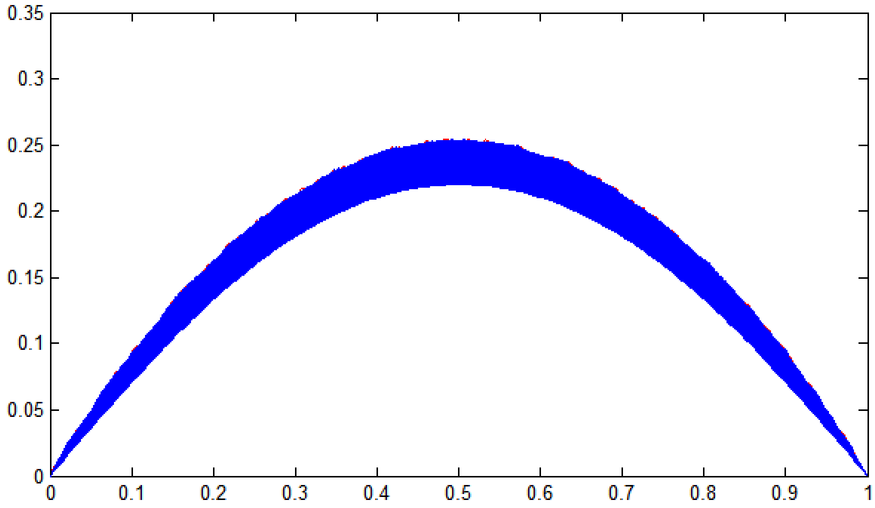
$$\begin{gathered}c(x,0) = f(x)\\ c(x,t)\\ c(x,t+\Delta t)\\ c(x,t+2\Delta t)\\ x_j = j \cdot \Delta x\\ t_n = n \cdot \Delta t\\ c(x_j,t_{n+1}) = D\frac{\Delta t}{(\Delta x)^2}c(x_{j-1},t_n) - D\frac{\Delta t}{(\Delta x)^2}2c(x_j,t_n) + D\frac{\Delta t}{(\Delta x)^2}c(x_{j+1},t_n) + c(x_j,t_n)\\ k = \Delta t\\ h = \Delta x\\ s = \frac{k}{h^2}\end{gathered}$$

This method of solution is explicit, as shown by the fact that the new function value found is set in regard to already-known terms. For the sake of ease in MATLAB code, three variables will be redefined:

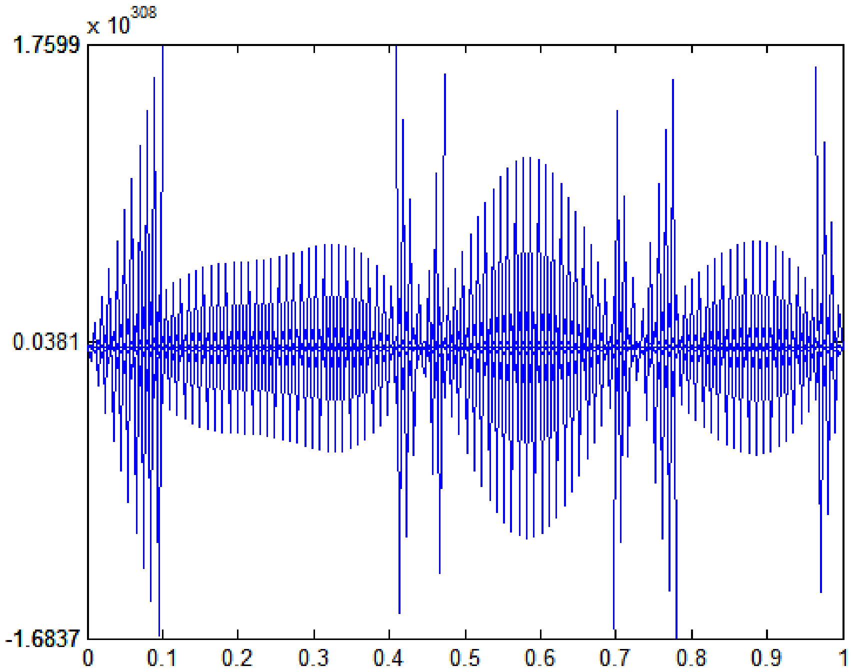
$$\begin{gathered}c(x,0) = f(x)\\ c(x,t)\\ c(x,t+\Delta t)\\ c(x,t+2\Delta t)\\ x_j = j \cdot \Delta x\\ t_n = n \cdot \Delta t\\ c(x_j,t_{n+1}) = D\frac{\Delta t}{(\Delta x)^2}c(x_{j-1},t_n) - D\frac{\Delta t}{(\Delta x)^2}2c(x_j,t_n) + D\frac{\Delta t}{(\Delta x)^2}c(x_{j+1},t_n) + c(x_j,t_n)\\ k = \Delta t\\ h = \Delta x\\ s = \frac{k}{h^2}\end{gathered}$$

By varying k and h (time step sizes) to collectively change s, the solution's stability critical value is found to be s= ½, where the system is stable only when s is less than or equal to ½.

s= ½



s> ½



Backward Euler (Implicit)

In this method, a different approximation of the partial differential equation is utilized.

$$\frac{\mathrm{d}c}{\mathrm{d}t}(x,t)\ .\ \frac{c(x,t)-c(x,t-\mathrm{D}t)}{\mathrm{D}t}$$

$$c(x_j,t_n)=-D\frac{\mathrm{D}t}{(\mathrm{D}x)^2}c(x_{j-1},t_{n+1})+D\frac{\mathrm{D}t}{(\mathrm{D}x)^2}2c(x_j,t_{n+1})-D\frac{\mathrm{D}t}{(\mathrm{D}x)^2}c(x_{j+1},t_{n+1})+c(x_j,t_{n+1})$$

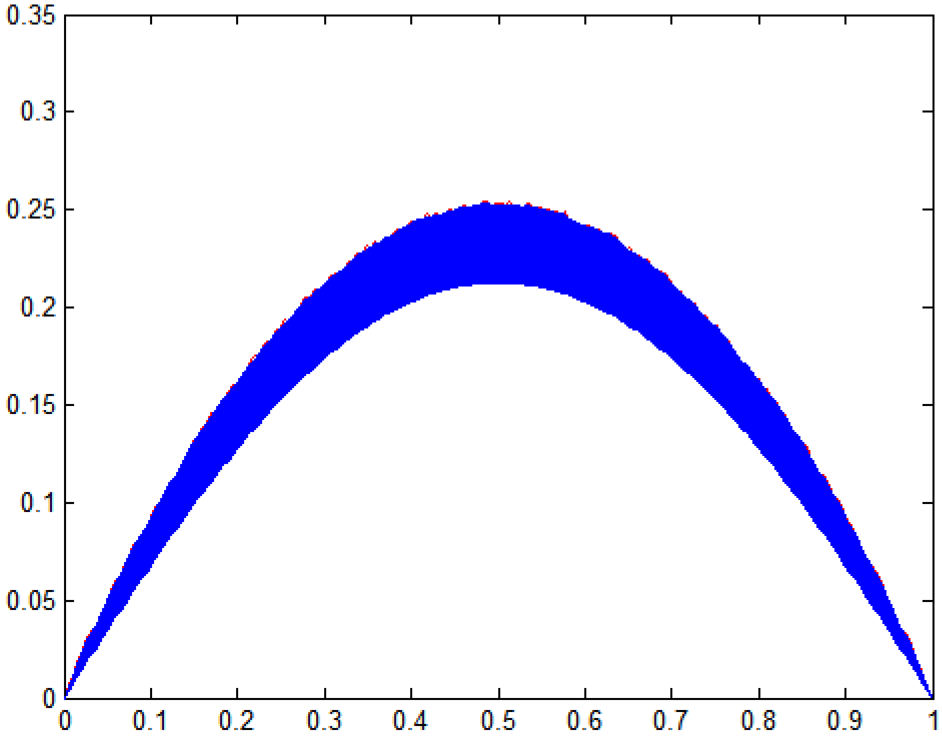
The result of a different beginning approximation is a differing end function.

$$\frac{dc}{dt}(x,t) \,.\, \frac{c(x,t) - c(x,t - Dt)}{Dt}$$

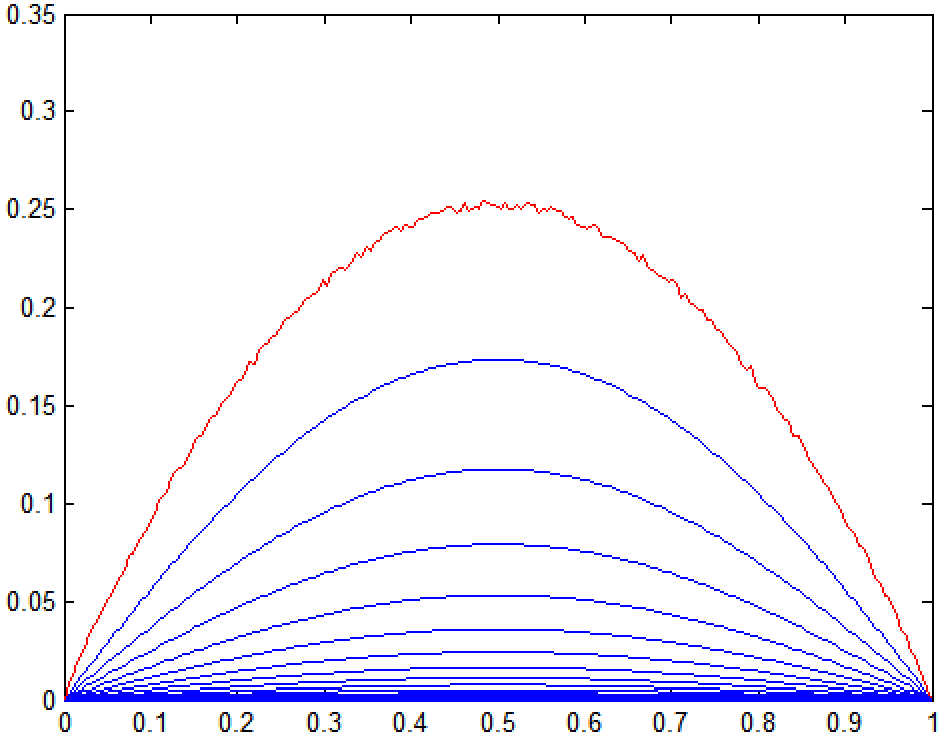
$$c(x_j, t_n) = -D\frac{Dt}{(Dx)^2}c(x_{j-1}, t_{n+1}) + D\frac{Dt}{(Dx)^2}2c(x_j, t_{n+1}) - D\frac{Dt}{(Dx)^2}c(x_{j+1}, t_{n+1}) + c(x_j, t_{n+1})$$

This method is implicit, meaning that another equation needs to be solved before solving for c. This is usually done using a fixed point iteration, here left up to MATLAB. This method takes more time and effort, but it is more effective in solving stiff equations, with larger step sizes (here denoted as Δx or h). However, like the Forward Euler method, stability breaks at s greater than ½.

s< ½



s> ½



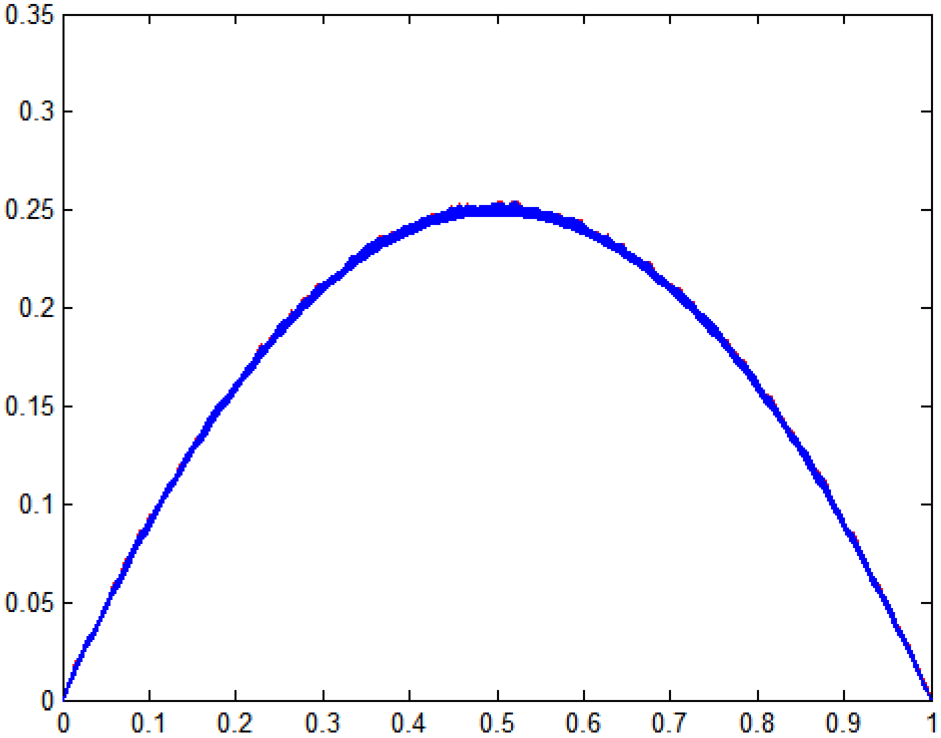
Crank-Nicholson Scheme

The Crank-Nicholson Scheme is an implicit solution method that, generally, averages the functions found using Forward and Backward Euler. The approximation found follows suit:

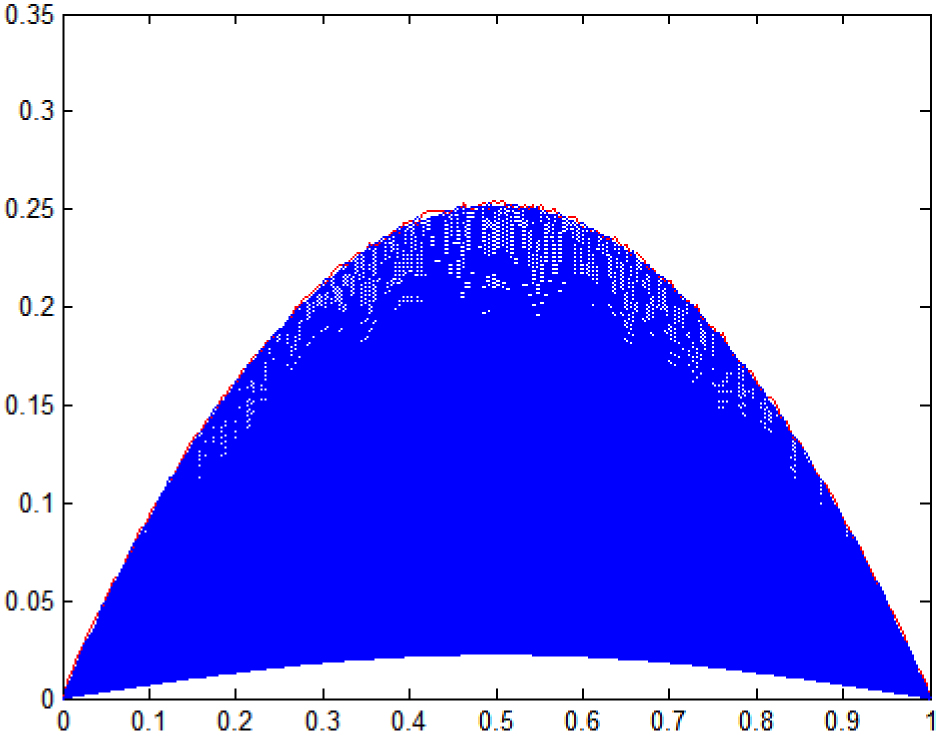
$$\frac{dc}{dt}(x,t) \approx \frac{1}{2}\left[\frac{c(x,t+\Delta t)-c(x,t)}{\Delta t}+\frac{c(x,t)-c(x,t-\Delta t)}{\Delta t}\right]$$

The stability trend of the solution also follows that of the methods by which Crank-Nicholson is composed. As shown by the following graphs, system stability breaks at s greater than ½.

s< ½



s> ½



Conclusion and Future Plans

This fall's adventure into the world of partial differential equations is a stepping stone to their uses in modeling hydrobiological systems, where flow (specifically blood flow) is a countable factor in a system's actions. Using the above research, I was able to understand and derive the basis of two papers dealing with blood vessel injuries and their resulting fibrin-based healing system. Continuing this research, I will work with Dr. Aaron Fogelson to look further into these biological systems and model enzyme concentration and interaction in clot formation. This work promises to be challenging and fascinating-applicable to truly "real life" situations.

References

Beltrami E, Jesty J: The role of membrane patch size and flow in regulating a proteolytic feedback threshold on a membrane: possible application in blood coagulation. Mathematical Biosciences 2001;172:1-13.

Fogelson, Aaron, Kuharsky, Andrew. Membrane Binding-Site Density Can Modulate Activation Thresholds in Enzyme Systems. Journal of Theoretical Biology; 1998, 193.

Kincaid, David, and Ward Cheney. Numerical Analysis. 3rd ed. N.p.: Brooks/Cole, 2002.

Strogatz, Steven. Nonlinear Dynamics and Chaos. USA: Addison-Wesley Publishing Company; 1994.

All graphical figures produced in MATLAB
All equations produced in MathMagic