You've been slimed

Slime mold cells are able to spontaneously aggregate by secreting a chemical attractant in a process called *chemotaxis*. A nondimensional 1D model for the process is

$$\begin{aligned} u_t &= u_{xx} - \chi(u v_x)_x, \qquad -1 \le x \le 1, \quad t \ge 0, \\ v_t &= \epsilon v_{xx} + u - v, \end{aligned} \tag{1}$$

where $u(x,t)$ represents a scaled cell density function, $v(x,t)$ represents the attractant concentration, and χ and ϵ are nonnegative constants. We will use periodic end conditions.

In order to apply the method of lines to (1), first discretize in space using finite differences to get

$$\begin{aligned} \boldsymbol{u}' &= \boldsymbol{f}(\boldsymbol{u},\boldsymbol{v}) = \boldsymbol{D}_{xx}\boldsymbol{u} - \chi \boldsymbol{D}_x\big[\boldsymbol{u} \odot (\boldsymbol{D}_x \boldsymbol{v})\big] \\ \boldsymbol{v}' &= \boldsymbol{g}(\boldsymbol{u},\boldsymbol{v}) = \epsilon \boldsymbol{D}_{xx}\boldsymbol{v} + \boldsymbol{u} - \boldsymbol{v}, \end{aligned} \tag{2}$$

where $\odot$ is the elementwise product `.*` in MATLAB. Choosing the Euler time stepping method, for example, leads to this iteration in the time step j:

$$\begin{aligned} \boldsymbol{u}_{j+1} &= \boldsymbol{u}_j + \tau \boldsymbol{f}(\boldsymbol{u}_j,\boldsymbol{v}_j) \\ \boldsymbol{v}_{j+1} &= \boldsymbol{v}_j + \tau \boldsymbol{g}(\boldsymbol{u}_j,\boldsymbol{v}_j). \end{aligned} \tag{3}$$

To apply a built-in IVP solver, equation (2) has to be expressed as a single first-order vector equation. The two "variables" of the system are actually the two m-vectors $\boldsymbol{u}$ and $\boldsymbol{v}$, which are combined into a column vector $\boldsymbol{z} = [\boldsymbol{u}\,;\,\boldsymbol{v}]$, leading to a system in the standard form $\boldsymbol{z}' = \boldsymbol{F}(t,\boldsymbol{z})$.

Goals

You will simulate the chemotaxis of slime mold and observe concentration of cells around initially small increases in attractant. You will use both the Euler and `ode15s` timestepping methods.

Preparation

Read section 11.4.

Procedure

Throughout all of these steps, let $\chi = 6$ and $\epsilon = 0.1$ in all the equations above.

1. Let $m = 50$, $u(x,0) = 1$, and $v(x,0) = 1 + 0.1e^{-100x^2}$, representing a small drop of extra attractant. (Note: the initial v is not truly periodic, but the exponential is so small at $x = \pm 1$ that is practically so.) Use the Euler timestepping method as shown in (3) to solve over $0 \le t \le 0.05$ with $n = 100$ time steps. Make a waterfall plot of the solution.

2. Now you will repeat step 1 with $m = 200$ and for $0 \le t \le 0.6$. Due to absolute stability restrictions, there is a minimum acceptable value of n. Start with $n = 2000$ and increment n repeatedly by 500 until the solution at the final time is finite. (Use the syntax `all(isfinite(w))` to determine if a vector is finite.) Output the minimum stable n and plot $u(x,0.6)$.

3. The next goal is to use the built-in solver `ode15s` that works well for stiff problems. To do this, write a function `timederiv.m` with headline

   ```
   function dzdt = timederiv(u,v,chi,Dx,Dxx)
   ```

   and returning $\boldsymbol{z}' = \left[\boldsymbol{u}'\ ;\ \boldsymbol{v}'\right]$ for the given inputs.

4. Now solve (with the same m and initial conditions as step 2) using `ode15s` for $0 \le t \le 15$. You will need to write an anonymous function of t and $\boldsymbol{z}$ only that encapsulates a call to `timederiv`. Print out the number of time steps that were taken and make a waterfall plot of the solution.

5. You would expect that if there were two symmetrically placed drops of attractant, the solution would remain symmetric for all time. However, that solution is unstable in time and one of the drops will eventually "win." Repeat step 4 using

$$v(x,0) = 1 + 0.1e^{-100(x-0.5)^2} + 0.1e^{-100(x+0.5)^2}.$$