Assignment 6, PX390, 2019/2020: Reaction-diffusion.

January 24, 2020

An experiment is being conducted to understand chemical reaction kinetics in a cylindrical shell of fluid. These reactions have been chosen to mimic the waves of activation that occur in neurons, but similar behaviour leads to, for example, pattern formation in whorls of fingerprints and the stripes on zebras. The cylinder may be 'unwrapped' onto an 2D rectangular domain $x \in [0, L_x]$, $y \in [0, L_y]$. The equations to be solved are

$$\frac{\partial u}{\partial t} = \nabla^2 u + f(u) - \sigma v \tag{1}$$

$$\frac{\partial v}{\partial t} = \nabla^2 v + u - v \tag{2}$$

with $f = \lambda u - u^3 - \kappa$, and $\nabla^2 = (\partial/\partial x)^2 + (\partial/\partial y)^2$. Here u and v are related to the local concentrations of chemicals (but are allowed to take on negative values).

Ideally, your time-evolution code should implement an implicit solver, to allow long timesteps given a fine grid: it is only necessary to implicitly solve the Laplacian terms (second spatial derivative). Even with a partially implicit scheme, an appropriate timestep will need to be chosen to avoid stability issues.

Boundary conditions are

$$\frac{\partial u}{\partial y} = \frac{\partial v}{\partial y} = 0 \tag{3}$$

for $y = 0$ and $y = L_y$. The x domain is periodic, with $u(x, y) = u(x + L_x, y)$ and $v(x, y) = v(x + L_x, y)$.

The initial conditions are $u = (A_0 + A_1 \cos[\pi y/L_y]) \exp(-[A_2 x/L_x]^2)$, $v = 0$ (the initial condition is not smooth at $x = 0$ with periodicity accounted for).

1 Grids

The grid for input and output have N_x and N_y points in the x and y direction respectively. They are evenly spaced with $x_i = L_x i/N_x$ and $y_j = L_y j/(N_y - 1)$. You may assume $N_x, N_y \geq 2$.

2 Input

The input file, ‘input.txt’ will contain the following values, one on each line:

- N_x: number of x grid points.
- N_y: number of y grid points.
- L_x: Length of x domain.
- L_y: Length of y domain.
- t_f: final time for time evolution mode.
- λ: λ parameter in equations.
- σ: σ parameter in equations.
- κ: κ parameter in equations.
- A_0: first initialisation parameter.
- A_1: second initialisation parameter.
- A_2: third initialisation parameter.
- t_D: diagnostic timestep.

3 Compiling

You can (and probably should) use the LAPACK library to invert matrices. This library is installed on the lab computers and on daisy. Some instructions are given for installing it on a linux system on the course website; if you aren’t used to compiling/installing libraries on your own system, it might be quicker just to log in to daisy. Let us say your code file is named ‘prog3.c’ (if not, change the filename in the compile line). The code must compile and generate no warnings when compiled with:

```
gcc -Wall -std=c99 -g -L/home/phskak/Code/BLAS/ -L/home/phskak/Code/lapack-3.5.0/  \
    -I/home/phskak/Code/lapack-3.5.0/lapacke/include/ prog3.c -llapacke         \
    -llapack -lblas -lgfortran -lm
```

Be careful copying and pasting this command line: a script containing the compiler command is also on the website (for example if you add or remove spaces then this may prevent compilation). Please call the LAPACKE C interface and not LAPACK directly (because these are FORTRAN functions and the interface is not guaranteed to work the same way when compiled on different machines).

The lab computers require different libraries (the OS and hardware are different)

```
gcc -g band_utility.c -I/usr/include/lapacke -std=c99 \
     -Wall  -llapacke -llapack -lblas -lgfortran -lm
```

and this script is also on the moodle website next to the example. Note that you may also use the 'nenneke' unix machine to compile on as well as daisy: this has valgrind installed. You will need a CSC account, as described in my email.

4 Test cases

Useful test cases include the trivial cases where diffusion dominates. It would be useful to consider cases where the shell is much longer in one direction than the other. Since the number of grid points may be small, this allows 1D test cases to be run. The initial conditions are not particularly simple: you may wish to choose simpler conditions where analytical solutions exist for linearised versions of the problem.

5 Output

The code should the temperature $u(x, y)$ to a file 'output.txt' in four column format (ie four values per line, the x grid point, the y grid point, then the u value then the v value) for all the grid points (x_i, y_j), at time points $t = 0, t_D, 2t_D, ...$ etc. The order is first all the x grid points at $y = y_0$, then all the x grid points for $y = y_1$, and so on.