Implicit methods for PDEs

November 22, 2016

Exercise 1. *Thomas algorithm for the heat equation*
Consider the heat equation

$$u_t = u_{xx} \tag{1}$$

on the domain $[x, t] \in [-1, 1] \times [0, 3]$ along with the initial conditions $u(x, 0) = \exp(-20x^2)$ and insulated boundary conditions ($u(\pm 1, t) = 0$). Apply the implicit algorithm described in the lecture notes, (6.39) to solve this problem. At each time-step, apply the Thomas algorithm. You can either write your own code for the Thomas algorithm or use the function `tridiag.m` (originally from the Matlab file exchange, now copied onto Moodle).

Plot your solution using `waterfall` or animate it.

Now try to apply zero-flux boundary conditions ($u_x(\pm 1, 0) = 0$). Why might you lose accuracy here and how could you avoid it? Do your solutions seem physically realistic?

Exercise 2. *Laplace's equation on a square*
We want to write a code to solve Laplace's equation

$$u_{xx} + u_{yy} = 0$$

on the unit square with boundary conditions $u(x, 0) = \sin \pi x$ and $u = 0$ on all other boundaries.

Use Jacobi point-iteration to compute the solution. Take an initial guess of $u = 0$. How many iterations does it take to get accuracy within 10^{-4}?

How could you write a code that builds the matrix explicitly? This is fiddlier than it would seem...