Mat1062: Computational Methods for PDE

Problem Set 3

Tuesday, February 12, 2008

due: Thursday, February 28 in class

1. In this problem, you will do the von Neumann stability analysis of a variety of schemes on $h\mathbb{Z}$. For each scheme, find whether or not there is a stability constraint on the size of the timestep, k. You can do it either via the Fourier approach or the "quick and dirty" approach of page 7 of the February 5th notes. I trust that you have already convinced yourselves that the two are fully equivalent.

   (a) The theta scheme for $u_t = Du_{xx}$:

   $$\frac{u_i^{n+1} - u_i^n}{k} = \theta L u^{n+1} + (1-\theta) L u^n$$

   where L is the usual centered-difference approximation for $\partial^2/\partial x^2$.

   (b) The theta scheme for $u_t = Du_{xx} + \gamma u$, again with a centered-difference approximation for $\partial^2/\partial x^2$. How does your answer depend on the sign of γ?

   (c) The theta scheme for $u_t = \beta u_x$ with a centered difference approximation for $\partial/\partial x$. How does your answer depend on the sign of β?

   (d) The theta scheme for $u_t = \beta u_x$ with a right-difference approximation for $\partial/\partial x$. How does your answer depend on the sign of β?

2. Consider the theta scheme for $u_t = Du_{xx} + \beta u_x - \gamma u$ where you have used centered differences for both u_{xx} and u_x. Assume Dirichlet boundary conditions. To find u^{n+1} you will have to solve

   $$Au^{n+1} = Bu^n$$

   where A and B are tridiagonal matrices.

   (a) Assume $\beta = \gamma = 0$. Show that A is diagonally dominant, making assumptions on the size of the time step k, if needed.

(b) Assume $\beta = 0$. Show that A is diagonally dominant, making assumptions on the size of the time step k, if needed. How does your answer depend on the sign of γ?

(c) Assume $D = \gamma = 0$. Show that A is diagonally dominant, making assumptions on the size of the time step k, if needed.

(d) What can you say about the case when none of the coefficients are zero?

3. In the first homework set, you computed the solution of the heat equation on an interval $[-L/2, L/2]$ with initial data

$$u_0(x) = \begin{cases} \frac{1}{h} & x = 0 \\ 0 & \text{otherwise} \end{cases}$$

You did explicit ($\theta = 0$) time-stepping and considered both Dirichlet and Neumann boundary conditions. You expected to see something close to a spreading Gaussian of mass 1, at least for a little while:

$$u(x,t) \sim \frac{1}{\sqrt{4\pi D t}} e^{-\frac{x^2}{4Dt}}.$$

Parts a) and b) below are to help you learn how to present data in a manner that is scientifically compelling.

(a) If your solution were exactly equal to a spreading Gaussian then at each moment in time you could plot your solution in some way so that at each moment in time the profiles would look the same. *Hint: given a particular time t^n and profile u^n the first thing you do is create a new profile of height* 1. *You'll need to figure out some way to deal with space.* Do this for a sequence of times, showing a time or two which is "too early" for this collapse onto a Gaussian, some times which show pretty good collapses, and a time or two which is "too late" for this collapse onto a Gaussian. Make plots for both the Neumann boundary condition case and the Dirichlet boundary condition case. (In terms of how to choose h and k and the final time, do something reasonable. Choose the final time large enough that you've seen the real effect of the boundary conditions. Choose h small but not so small that you can't compute up to the final time in less than ten minutes. Recall that you'll have to choose k so that $\lambda = Dk/h^2 < 1/2$ holds.)

(b) Instead of showing profiles, for each time t^n you could find the height H^n (easily done) and the half-width (less easily done, see below). You could then plot the each of them independently of each other ($\log_{10}(t^n)$ versus $\log_{10}(H^n)$, say) and you can then plot them against each other is some way to see the time range in which you have reasonable collapse onto Gaussians. Whenever possible, do things in terms of logarithms. (For each plot, plot both the Neumann boundary condition case and the Dirichlet boundary condition case on the same plot.)

To find the half-width, assume U is a solution at some particular time. It's indexed from 1 to $N+1$. Let $Uhalf = max(U)/2$. Then $I = find(U > Uhalf)$ will give you the indices at which U is greater than $Uhalf$. If you plot $U(I)$ then you'll get the upper half of the profile. Let $i0 = min(I)$ and $i1 = max(I)$. On the graph of U, the left point of the half-width is somewhere between $(x(i0-1), U(i0-1))$ and $(x(i0), U(i0))$. You can approximate the exact location by linearly interpolating between these two points. Similarly, use $i1$ to find the right point and then find the half-width.

(c) For the second homework assignment, you wrote a code for the theta scheme for $u_t = Du_{xx}$ for both Dirichlet and Neumann boundary conditions. Set $\theta = 1/2$ so you're doing Crank-Nicolson. Choose either set of boundary conditions. Compute the solution for two sizes of the timestep: one so that $\lambda \geq 1/2$ and one so that $\lambda < 1/2$. What do you see in the first few profiles? (Recall that on page 16 of the first set of notes, I showed that the explicit ($\theta = 0$) scheme satisfies the comparison principle: that if $u_i^0 \geq M$ for all i then $u_i^1 \geq M$ for all i and therefore $u_i^n \geq M$ for all i.)

4. Code up the ADI scheme for the heat equation $u_t = D\Delta u$ on $[0, L] \times [0, L]$ with Dirichlet boundary conditions. Demonstrate that it's working properly by showing that it converges to the correct exact solution when you take initial data $u_0(x, y) = \sin(2\pi x/L)\sin(4\pi y/L)$.

   This should not be a big coding exercise — you've already coded up the 1-d heat equation for variable θ so you've got all the matrices you need handy.

5. In the ADI scheme, there are two steps to find u^{n+1} from u^n. First you solve

$$(I - \frac{\lambda}{2}\delta_x^2)u^* = (I + \frac{\lambda}{2}\delta_y^2)u^n$$

for u^* and then you solve

$$(I - \frac{\lambda}{2}\delta_y^2)u^{n+1} = (I + \frac{\lambda}{2}\delta_x^2)u^*$$

for u^{n+1}. That is, the directions alternated. What happens if you modify your code so that you don't alternate the directions? That is, first you solve

$$(I - \frac{\lambda}{2}\delta_x^2)u^* = (I + \frac{\lambda}{2}\delta_x^2)u^n$$

for u^* and then you solve

$$(I - \frac{\lambda}{2}\delta_y^2)u^{n+1} = (I + \frac{\lambda}{2}\delta_y^2)u^*$$

for u^{n+1}?