Mat1062: Introductory Numerical Methods for PDE

Problem Set 3

Friday April 1, 2016

due: Friay April 15, 2016

1. **Computing derivatives: Finite Difference versus Chebyshev versus Fourier**

   You already know a reasonable finite difference scheme for approximating the first derivative at a point:

   $$u'(x) \approx \frac{u(x+h) - u(x-h)}{2h}.$$

   And certainly you know how to approximate the derivative at a point for a periodic function using the Discrete Fourier Transform and the Discrete Inverse Fourier Transform.

   Now to introduce another spectral method based on Chebyshev Polynomials. This is for a function defined on $[-1, 1]$. (Given a function on a different interval, you'd just change coordinates as needed.) First step: choose the number of intervals: N. Second step, choose a nonuniform mesh in $[-1, 1]$ via

   $$x_i = \cos(ih) \qquad \text{where } j = 0, \ldots N \text{ and } h = \pi/N.$$

   These are "Chebyshev Extreme Points"; see page 262 of `https://people.maths.ox.ac.uk/trefethen/8all.pdf` You define weights

   $$c_i = \begin{cases} 2 & \text{if } i = 0 \text{ or } i = N \\ 1 & \text{otherwise;} \end{cases}$$

   see (8.2.2) on page 270. Using these weights, you define the $(N+1) \times (N+1)$ matrix D via

   $$D_{00} = \frac{2N^2+1}{6}, \qquad D_{NN} = -\frac{2N^2+1}{6}, \qquad D_{ij} = \frac{c_i}{c_j} \frac{(-1)^{i+j}}{x_i - x_j} \text{ if } i \neq j.$$

   (See Theorem 8.4 on page 270.) In principle, you define the diagonal entries via $D_{ii} = \frac{-x_i}{2(1-x_i^2)}$ for $1 \leq i \leq N-1$. In practice, you use the fact that the rows need to sum to zero and define the diagonal entry of the ith row to be the negative of the sum of all the other entries in the ith row.

   (a) Code up the Chebeshev Extreme Points and the differentiation matrix. Choose a function f on $[-1, 1]$, sample it at the Chebeshev Extreme Points, defining a vector $\vec{f}$. Compute $D\vec{f}$ and compare it to f' sampled at the Chebeshev Extreme Points. If your function was a reasonable function (like sine or cosine or a polynomial) then you should get excellent results.

(b) Choose a periodic function on $[-1, 1]$. You now have three ways of computing the derivative at $x = 0$ (assuming N is even). Approximate $f'(0)$ using all three methods, creating three errors for this value of N. Now double N and repeat this process. Present a log-log graph of errors versus N. Plot some sight lines to see if you can guess at any behaviours you observe of errors versus N.

2. **Pseudospectral computation of a semilinear heat equation**

Consider the semilinear heat equation

$$u_t = Du_{xx} + Bu^2 \tag{1}$$

with periodic boundary conditions on $(-\pi, \pi)$. The numbers D and B are assumed positive. This homework set is intended to be an exploration of "diffusion versus nonlinearity". For Burger's equation $u_t + uu_x = Du_{xx}$ the diffusion always "beats" the nonlinearity — there's never finite-time blow-up. We will see that things aren't so simple for equation (1).

(a) First look at the PDE without the nonlinearity: $u_t = Du_{xx}$. Can solutions blow up in finite time? Why or why not? If solutions don't blow up in finite time, what do they do as time goes to infinity? Now look at the equation without the diffusion: $u_t = Bu^2$. Can solutions of this ODE blow up in finite time? If yes, under what conditions can they blow up?

(b) Assume u is a solution of the above and define

$$v(x, t) = a\, u(bx, ct)$$

where a, b, and c are all positive numbers. What PDE does v satisfy? By choosing a, b, and c well, you should be able to find that v is a periodic solution of

$$v_t = v_{xx} + v^2 \tag{2}$$

on $(-\pi, \pi)$. What does this rescaling mean about the roles of D and B in whether or not solutions can blow up in finite time?

(c) Write a pseudo-spectral code that approximates solutions of (2) on $(-\pi, \pi)$. You choose the time-stepping. Choose some simple initial data, choose a (relatively) short final time, compute a sequence of solutions (v_1, v_2, v_3, ...) up to that time (where v_1 was computed with timestep k, v_2 with timestep $k/2$, v_3 with timestep $k/4$ and so on) and demonstrate that your code has the order of accuracy you expected.

(d) Find a two-parameter family of periodic initial data which satisfies: 1) there is one maximum, located at $x = 0$, 2) the maximum increases and decreases in height as you vary one of the parameters, and 3) the "width" of the central "bump" varies as you vary the other parameter. (For the last item you can equally well look at the first derivative and check that its maximum value increases and decreases as you vary the other parameter.)

(e) Do some explorations with your code on this class of initial data. Do you see what appears to be finite-time blow up sometimes? But not others? If yes, is what happens determined by only one parameter?

(f) Write a point-doubling code which takes a profile sampled at N points and returns the same profile centered at $2N$ points. As discussed in class, this is done as follows: given u sampled at $x_0 = -\pi, \ldots, x_{N-1} = \pi - h$ compute its FFT $\widehat{u}$. Create a vector $\widehat{v}$ of length $2N$ which uses the N values of $\widehat{u}$ but is padded out by zeroes elsewhere. (It's fine if you use only $N-1$ values of $\widehat{u}$.) Take the IFFT of $\widehat{v}$ to create a vector v which has $2N$ values. Taking a fairly simple function for u, demonstrate graphically that u and v agree where they should. Also, compute the maximum of $|u - v]$ (at the meshpoints they have in common).

(g) For one of the simulations in which you think you may be seeing finite-time blow-up, compute up to a time T_1 (of your choosing). Stop the computation, take the solution at that time, double the number of points, decrease the timestep by a factor of 2 and continue the compuation, computing up to a time T_2 (of your choosing). Stop the computation, take the solution at that time, double the number of points, decrease the timestep by a factor of 2 and continue the compuation, computing up to a time T_3 (of your choosing). Do this until you find that it's taking your computer "too long" or that you're using up too much disk space, whichever comes first. (I'm fine if you don't want to leave anything running for longer than an hour.) Now that you have these solutions, plot them at a sequence of times along with their spectra at a sequence of times. Find the maximum value from the solution at each time and plot $(t, \max u(\cdot, t))$. If you have any ideas for how to plot this so that it's a more compelling demonstration of finite-time blow-up, do so.

(h) Explain why a non-uniform mesh would be better than a uniform mesh for this particular exploration.