Numerical Methods II, Spring 2019
Assignment V: Finite Difference Methods for the Diffusion Equation

Aleksandar Donev
Courant Institute, NYU, donev@courant.nyu.edu

February 25th 2023
Due: **Tuesday 5pm April ? 2023**

1 [160 pts] Finite Difference Discretizations in 1D

In this homework you will consider constructing a finite difference method for solving the non-constant coefficient diffusion equation

$$u_t = (d(x)u_x)_x + f(x,t), \tag{1}$$

on the domain $0 < x < 1$ with (potentially inhomogeneous) Dirichlet BCs at $x = 0$ and Neumann BCs at $x = 1$,

$$u(0,t) = \gamma(t)$$
$$u_x(1,t) = g(t).$$

Here $d(x) > 0$ is a diffusion coefficient which can in principle depend on position; more generally, the diffusion coefficient can also depend on time or even on u itself (leading to a nonlinear diffusion equation).

Discretize the right hand side of the PDE in space using a finite-difference discretization that is at least second-order accurate (or at least *try* to achieve second-order accuracy), and write down the discretization in the report and then justify the choices you made. Make sure to develop a method and a code that can work for any $d(x)$, $\gamma(t)$ and $g(t)$ that are given as input to the problem (e.g., in the form of function handles in Matlab). Write down a discretization and develop code for a grid where the first and last point on the grid are separated from the endpoints by $h/2$, where h is the grid spacing, that is, the first point is at $x = h/2$ and the last point at $1 - h/2$. This is the "staggered grid" discretization we mentioned in class, and would arise if a finite-volume method were used, but for the purposes of this homework (second-order accuracy) it does not matter whether we think of the method as finite difference or finite volume.

The book by LeVeque uses a non-staggered grid, i.e., a grid where the first and last point are on the boundary, and does not discuss BCs for parabolic. For the staggered grid, follow the idea of introducing ghost cells as described in class (and in the "second approach" in Section 2.12 in LeVeque) for elliptic PDEs in order to discretize the operator $(d(x)u_x)_x$ with the imposed boundary conditions. Then simply use this spatial discretization for the parabolic PDE to obtain a system of ODEs to solve, and then solve it and see what you get.

If you are unable to handle the boundary conditions, do as much of this assignment as you can by replacing the domain with $-1 \le x < 1$ and putting homogeneous Dirichlet conditions on both ends with a grid where the first point is at $x = -1$ and the last point is at $x = 1$ (this is trivial so everyone should be able to do this properly); think about why this is equivalent to the case of a homogeneous Neumann BC at $x = 0$ and homogeneous Dirichlet at $x = 1$ and report your explanation if you end up doing this.

1.1 [110 pts] Order of Accuracy

In order to test the order of accuracy of our method we will use the analytical *manufactured solution*

$$u_{\text{sol}}(x,t) = \exp^{-\pi^2 t/4} \cos\left(\frac{\pi x}{2}\right) \tag{2}$$

to obtain the specific forms of the initial condition $u(x,0)$, the forcing function $f(x,t)$ and the boundary conditions $h(t)$ and $g(t)$, so that the solution is (2), for a given

$$d(x) = 2 + \cos(\pi x).$$

Note: Observe that if you chose instead $u_{sol}(x,t) = \exp^{-\pi^2 t/4}\sin(\pi x/2)$ *then the boundary conditions will become homogeneous. This is sometimes easier to get right first, so you may want to start from there while debugging the code. Also observe that if* $d(x) = 1$ *is a constant, then* $f(x,t) = 0$ *so there is no forcing term, which also simplifies things and may be worth trying out first as a debugging tool.*

1.1.1 [50 pts] Elliptic PDE

The limit $u(x, t \to \infty) \to v(x)$ of the parabolic PDE (1) is given by the solution of a corresponding elliptic PDE. To make that elliptic PDE be simple but nontrivial, let's add a constant-in-time forcing term on the right hand side and consider

$$0 = (d(x)u_x)_x + \tilde{f}(x). \tag{3}$$

Figure out what $\tilde{f}(x)$ needs to be so that the solution of this PDE is $u_{\text{sol}}(x, t = 0)$.

[20 pts] First, confirm that your discretization of the operator $(d(x)u_x)_x$ is second-order accurate in space. This is always the first step in debugging codes involving solvers: test the forward operator before constructing the inverse operator! To do this, there are two options. In the first option, you should include also the cells right next to the boundary. Then, fill the ghost cells with values computed from the *exact* solution (not one obtained by extrapolating from the interior as you do in the actual numerical method). Or, in the second option, exclude the cells right next to the boundary. At those two points, we are enforcing the boundary conditions to second order but we are *not* enforcing the PDE (you cannot both enforce the PDE and the boundary condition at the boundary because there will be more unknowns than there are equations). See the section titled "Second approach" in 2.12 in LeVeque to understand this.

[30 pts] Now solve the elliptic PDE (3) using your spatial discretizations. Make sure to explain how you solved the linear system and how expensive your method is in terms of the number of grid intervals $N = 1/h$. Compute and report the local truncation error of your scheme, including also the boundary points (don't be alarmed if it seems to not be second-order at the boundary!). Find numerically the order of accuracy of the *global* error *for* your method; now you should include all points including those near the boundary.

1.1.2 [60 pts] Parabolic PDE

[30 pts] Now solve the parabolic PDE up to time $T = 1/4$. Write your code so that it is efficient, reusing quantities instead of recomputing them, and reusing as much of the code between different runs/methods. Use one explicit and one implicit method to integrate the ODEs in time, and report what method you chose. For the implicit method, write down what linear system(s) you needed to solve, and how you solved them. For each method, estimate the stability limit on the time step size, if any. Confirm numerically for at least two different grid sizes that this estimate is a good one.

[30 pts] Compute the solution using several grid sizes that are powers of two, and explain how and why you chose the specific value of the time step size for each grid size. Confirm the overall spatio-temporal order of accuracy of your method. Plot the error in the solution (as a function of x) and explain how large the grid size needs to be in order for you to be in the asymptotic error regime. Comment on whether you think different function norms of the error will behave differently (and of course test your hypothesis numerically).

1.2 [50 pts] Robustness for large time step sizes

Now consider the diffusion equation with a constant $d(x) \equiv d = 1$,

$$u_t = u_{xx}, \tag{4}$$

on the domain $0 < x < 1$ with Dirichlet BCs at $x = 0$ and Neumann BCs at $x = 1$,

$$\begin{aligned} u(0,t) &= 0 \\ u_x(1,t) &= 0, \end{aligned}$$

and with a discontinuous initial condition

$$u(x,0) = \begin{cases} 0 & \text{if } x \le 1/2 \\ 1 & \text{otherwise.} \end{cases} \tag{5}$$

Note that the initial conditions satisfies the BCs (this is *not* required for parabolic PDEs but we want the only sharp jumps to be in the middle at $x = 1/2$ and not at the boundaries). If you are unable to handle Neumann BCs properly, it is OK to use Dirichlet BCs on both ends for this part.

[20 pts] Figure out a way (e.g., pen and paper) to obtain the exact solution of the PDE to a high accuracy (say 6 digits of accuracy).

[Optional] Figure out a way to do this numerically using the FFT in time $O(N \log N)$ for *any initial condition*.

Reuse the spatial discretization you already developed. Write code to integrate the resulting semi-discrete system forward in time using the backward Euler (BE) and the implicit trapezoidal method (Crank-Nicolson, CN) for a specified time step size. Explain how you solved the linear system arising in the implicit temporal discretization. Then:

1. [15 pts] Using a time step size corresponding to a diffusive CFL number of $\mu = d\tau/h^2 = 1$, 10, 100, compare the numerical solutions at time $t = 0.01$ for the two methods (BE vs CN) to the exact solution, and comment on your observations. Is one of the two methods clearly more robust? Figure out numerically how small μ needs to be so that you do not get oscillations (spurious minima or maxima) in the numerical solution. Try several different grid resolutions N to get a better feeling for what is going on.
2. [15 pts] Compare the solutions obtained by the two temporal integrators at a much longer time $t = 10$ for a time step size corresponding to a diffusive CFL number of $\mu = 1000$ (choose one grid resolution) and comment on your observations. Compare the numerical solution to the steady state solution $u(x, \infty)$ of the PDE and discuss your observations of the differences between CN and BE, and explain your observations theoretically.

Note: In order to understand what you see and explain it will be helpful if you look at section 8.3.2 on L-stability in the book of LeVeque and in particular Fig. 8.4.

[**Optional**] Can you find a method that is second-order in time but is more robust (explain in what way) than Crank-Nicolson?

1.2.1 [Optional] Discrete Green's Function

In the continuum setting, we use Green's functions to solve parabolic problems. A key property of the Green's function for the heat equation is that it is positive and smooth. We can define a discrete Green's function over a time step as the solution at the end of the time step if we start with an initial condition that is a "discrete delta function" (as discussed in class for elliptic PDEs). By looking at this Green's function for different methods one can see how big the time step can be before some key *physical property* of the continuum Green's function is violated by the discrete one. Note that this is a qualitative rather than a quantitative distinction – the method can be low order but physically robust so it is not necessarily more accurate but rather more physical.

Compute analytically or numerically the discrete Green's function for BE and CN for the diffusion equation with constant coefficients in periodic conditions (or on the unbounded real line) and plot it. Discuss what happens as the time step size increases, and try to relate it to what you observed numerically for initial condition (5). Try to prove analytically that the solution remains positive (more precisely, monotone) for sufficiently small $\mu < \mu_0$ and figure out μ_0 for the two methods.

1.3 [optional] Fun with nonlinearities

Now consider the nonlinear heat equation

$$u_t = (d(x)u_x)_x + u^2 \tag{6}$$

on the domain $-1 \le x < 1$ with homogeneous Dirichlet boundary conditions $u(-1) = u(1) = 0$, with

$$d(x) = 2 + \cos(2\pi x),$$

and initial condition

$$u(x, 0) = u_0 \left[\cos\left(\pi x/2\right)\right]^{100}.$$

This PDE is a simplified model for, say, ignition of a flame when a substance burns – the variable u is the temperature, the diffusion models heat conduction, and the nonlinear term models the heat released by the chemical reactions during burning.

Describe a numerical method to solve this equation; you can choose which grid and spatial discretization you want to use and what method in time you employ. There is no right or wrong answer, use the best method you can given your abilities and prior knowledge and available time and interest. Summarize with equations and words the method you used in your report. Find the smallest value of u_0 (only to within 1-2 digits, no need for a precise number) for which the solution of (6) blows up in finite time, and make a movie of the blow up. Are you convinced that the blow-up solution you computed is sufficiently accurate and not a numerical error (explain your reasoning and what you tried to test this out)?