7 Finite Differences: Partial Differential Equations

The world is defined by structure in space and time, and it is forever changing in complex ways that can't be solved exactly. Therefore the numerical solution of partial differential equations leads to some of the most important, and computationally intensive, tasks in all of numerical analysis (such as forecasting the weather). This chapter introduces *finite difference* techniques; the next two will look at other ways to discretize partial differential equations (finite elements and cellular automata). Just as we used a Taylor expansion to derive a numerical approximation for ordinary differential equations, the same procedure can be applied to partial differential equations. Because the discretization must be done in space as well as time, there are many more possible strategies for finding good (and bad) approximations.

We will start with two degrees of freedom, say one spatial variable x and a time t. Given a function $u(x,t)$, its spatial derivatives are found from the Taylor expansion

$$u(x+\Delta x, t) = u(x,t) + \Delta x \left.\frac{\partial u}{\partial x}\right|_{x,t} + \frac{(\Delta x)^2}{2!} \left.\frac{\partial^2 u}{\partial x^2}\right|_{x,t} + \mathcal{O}[(\Delta x)^3] \quad . \tag{7.1}$$

The first partial derivative can be approximated by the *forward difference*

$$\frac{u(x+\Delta x,t) - u(x,t)}{\Delta x} = \left.\frac{\partial u}{\partial x}\right|_{x,t} + \mathcal{O}[\Delta x] \quad . \tag{7.2}$$

If we replace Δx with $-\Delta x$, this becomes the equally reasonable *backwards difference* approximation

$$\frac{u(x,t) - u(x-\Delta x,t)}{\Delta x} = \left.\frac{\partial u}{\partial x}\right|_{x,t} + \mathcal{O}[\Delta x] \quad . \tag{7.3}$$

Both have first-order errors. The order of the approximation can be raised to second order by taking the difference between two time steps, which subtracts out the quadratic term:

$$\frac{u(x+\Delta x,t) - u(x-\Delta x,t)}{2\Delta x} = \left.\frac{\partial u}{\partial x}\right|_{x,t} + \mathcal{O}[(\Delta x)^2] \quad . \tag{7.4}$$

Although this might appear always to be preferable, we will see that it can have surprising undesirable stability properties.

The straightforward finite difference approximation to the second partial derivative

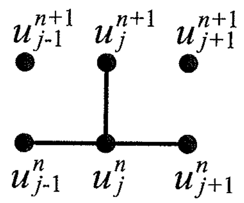

Figure 7.1. A computational cluster.

also has a second-order error:

$$\frac{1}{\Delta x}\left[\frac{u(x+\Delta x,t)-u(x,t)}{\Delta x}-\frac{u(x,t)-u(x-\Delta x,t)}{\Delta x}\right]=$$

$$\frac{u(x+\Delta x,t)-2u(x,t)+u(x-\Delta x,t)}{(\Delta x)^2}=\left.\frac{\partial^2 u}{\partial x^2}\right|_{x,t}+\mathcal{O}[(\Delta x)^2] \quad . \tag{7.5}$$

Numerical methods for partial differential equations are usually classified by the characteristics for the equation that they apply to (Chapter 3), which measure how information from the boundary conditions influences the solution. Characteristics can even be used as the basis for numerical solvers [Ames, 1992], but here we will simply use them as convenient labels for the most common cases: a wave equation (hyperbolic characteristics), diffusive processes (parabolic), and boundary value problems (elliptic). More complex systems can have some or all of these elements.

7.1 HYPERBOLIC EQUATIONS: WAVES

To see how the stability of the solution depends on the finite difference scheme, let's start with a simple first-order hyperbolic PDE for a conserved quantity in one dimension

$$\frac{\partial u}{\partial t}=-v\frac{\partial u}{\partial x} \quad . \tag{7.6}$$

Substitution readily shows that this is solved by any function of the form

$$u=f(x-vt) \quad . \tag{7.7}$$

Writing $u(j\Delta x, n\Delta t) = u_j^n$ to make the notation clearer, a simple discretization is first-order in time and second-order in space:

$$\frac{u_j^{n+1}-u_j^n}{\Delta t}=-v\left(\frac{u_{j+1}^n-u_{j-1}^n}{2\Delta x}\right)$$

$$u_j^{n+1}=u_j^n-\frac{v\Delta t}{2\Delta x}(u_{j+1}^n-u_{j-1}^n) \tag{7.8}$$

(using a first-order spatial approximation would make it asymmetrical). It can be convenient to represent such approximations by drawing the cluster of values used in the update rule (Figure 7.1). Given an initial distribution u_j^n, it is straightforward to iterate this rule forward in time.

To analyze the stability of a finite difference scheme, the *von Neumann stability* *analysis* locally linearizes the equations (if they are not linear) and then looks at the growth of the linear modes

$$u_j^n = A(k)^n e^{ikj} \tag{7.9}$$

which have an oscillatory dependence on space and an exponential dependence on time. Plugging in this ansatz gives a solution to the finite difference equation for $A(k)$. If $|A(k)| > 1$ for some k, then these modes will diverge and the scheme will be unstable (remember that the exact solution (7.7) does not diverge). For equation (7.1) this gives

$$\begin{aligned} A^{n+1}e^{ikj} &= A^n e^{ikj} - \frac{v\Delta t}{2\Delta x}\left(A^n e^{ik(j+1)} - A^n e^{ik(j-1)}\right) \\ A &= 1 - \frac{v\Delta t}{2\Delta x}\left(e^{ik} - e^{-ik}\right) \\ &= 1 - i\frac{v\Delta t}{\Delta x}\sin k \quad . \end{aligned} \tag{7.10}$$

The absolute magnitude of this is always greater than 1, and so this scheme is always unstable. Any initial condition will diverge!

This disturbing behavior in such a sensible approximation is easily corrected with the *Lax method*, which averages the neighbors for the time derivative:

$$u_j^{n+1} = \frac{1}{2}(u_{j+1}^n + u_{j-1}^n) - \frac{v\Delta t}{2\Delta x}(u_{j+1}^n - u_{j-1}^n) \quad . \tag{7.11}$$

Repeating the stability analysis shows that the amplitude of a solution is

$$A = \cos k - i\frac{v\Delta t}{\Delta x}\sin k \quad . \tag{7.12}$$

Requiring that the magnitude be less than 1,

$$|A|^2 = \cos^2 k + \left(\frac{v\Delta t}{\Delta x}\right)^2 \sin^2 k \le 1$$

$$\Rightarrow \frac{|v|\Delta t}{\Delta x} \le 1 \quad . \tag{7.13}$$

This is the *Courant–Friedrichs–Levy* stability criterion, and it will recur for a number of other schemes. It says that the velocity at which information propogates within the numerical algorithm ($\Delta x/\Delta t$) must be faster than the velocity of the solution v. For space and time steps that satisfy this condition, the Lax method will be stable. Otherwise, there is a "numerical boom" as the real solution tries to out-run the rate at which the numerical solution can advance. The lateral averaging for the time derivative in the Lax method helps the numerical information propagate, compared to the unstable approximation that we started with (equation (7.1)). The origin of this stability becomes clearer if the Lax method is rewritten by subtracting u_j^n from both sides:

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} = -v\left(\frac{u_{j+1}^n - u_{j-1}^n}{2\Delta x}\right) + \frac{1}{2\Delta t}(u_{j+1}^n - 2u_j^n + u_{j-1}^n) \quad . \tag{7.14}$$

This is just our
depends on the

This is an exa
be added inten
damp out the s
wavelength sol
long-term beha
The Lax me
but it is only fi
Δx to have th
will be stable).

The stability
amplitude, giv

If $|v|\Delta t/\Delta x \le$
condition app
wavelength k
the *leapfrog* n
do not influen
can lead to a
artificial coup
Problem 7.

We will next

and will assu

The methods

This is just our original equation (7.1), with an extra fictitious diffusion term added that depends on the discretization:

$$\frac{\partial u}{\partial t} = -v\frac{\partial u}{\partial x} + \frac{(\Delta x)^2}{2\Delta t}\frac{\partial^2 u}{\partial x^2} \quad . \tag{7.15}$$

This is an example of an artificial *numerical dissipation*, which can occur (and even be added intentionally) in stable schemes. In this case it is good, because it serves to damp out the spurious high-frequency modes ($k \sim 1$) while preserving the desired long wavelength solutions. In other cases it might be a problem if the goal is to look at the long-term behavior of a nondissipative system.

The Lax method cures the stability problem and is accurate to second order in space, but it is only first-order in time. This means that $v\Delta t$ will need to be much smaller than Δx to have the same accuracy in time and space (even though a much larger time step will be stable). A natural improvement is to go to second order in time:

$$u_j^{n+1} = u_j^{n-1} - \frac{v\Delta t}{\Delta x}\left(u_{j+1}^n - u_{j-1}^n\right) \quad . \tag{7.16}$$

The stability analysis for this equation now leads to a quadratic polynomial for the amplitude, giving two solutions

$$A = -i\frac{v\Delta t}{\Delta x}\sin(k) \pm \sqrt{1 - \left[\frac{v\Delta t}{\Delta x}\sin(k)\right]^2} \quad . \tag{7.17}$$

If $|v|\Delta t/\Delta x \leq 1$ then the radical will be real, and $|A|^2 = 1$ independent of k. The Courant condition applies again, but now there is no dependence of the amplitude on the spatial wavelength k and so there is no artificial damping (unlike the Lax method). This is called the *leapfrog method* because it separates the space into two interpenetrating lattices that do not influence each other (u_j^{n+1} does not depend on u_j^n). Numerical round-off errors can lead to a divergence of the sublattices over long times, requiring the addition of an artificial coupling term.

Problem 7.1 considers the finite difference approximaton to the wave equation.

7.2 PARABOLIC EQUATIONS: DIFFUSION

We will next look for finite difference approximations for the 1D diffusion equation

$$\frac{\partial u}{\partial t} = \frac{\partial}{\partial x}\left(D\frac{\partial u}{\partial x}\right) \quad , \tag{7.18}$$

and will assume that the diffusion coefficient is constant

$$\frac{\partial u}{\partial t} = D\frac{\partial^2 u}{\partial x^2} \quad . \tag{7.19}$$

The methods to be described will have natural generalizations when D is not constant.

The straightforward discretization is

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} = D\left[\frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{(\Delta x)^2}\right]$$

$$u_j^{n+1} = u_j^n + \frac{D\Delta t}{(\Delta x)^2}[u_{j+1}^n - 2u_j^n + u_{j-1}^n] \quad . \tag{7.20}$$

Solving the stability analysis,

$$\begin{aligned} A &= 1 + \frac{D\Delta t}{(\Delta x)^2} \underbrace{\underbrace{[e^{ik} - 2 + e^{-ik}]}_{2\cos k - 2}}_{2\left(2\cos^2\frac{k}{2} - 1\right) - 2} \\ &= 1 - \frac{4D\Delta t}{(\Delta x)^2}\sin^2\frac{k}{2} \\ |A| \le 1 &\Rightarrow \frac{4D\Delta t}{(\Delta x)^2} \le 2 \Rightarrow \frac{2D\Delta t}{(\Delta x)^2} \le 1 \quad . \end{aligned} \tag{7.21}$$

The method is stable for small step sizes, but since for a diffusive process the time t to expand a distance L is roughly $t \sim L^2/D$, the number of time steps required to model this will be $\sim L^2/(\Delta x)^2$ (i.e., a *very* large number).

The stability can be improved by evaluating the space derivative forwards in time:

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} = D\left(\frac{u_{j+1}^{n+1} - 2u_j^{n+1} + u_{j-1}^{n+1}}{(\Delta x)^2}\right)$$

$$u_j^{n+1} - \frac{D\Delta t}{(\Delta x)^2}[u_{j+1}^{n+1} - 2u_j^{n+1} + u_{j-1}^{n+1}] = u_j^n \quad . \tag{7.22}$$

The stability analysis for this is

$$\begin{aligned} A - \frac{D\Delta t}{(\Delta x)^2}[Ae^{ik} - 2A + Ae^{-ik}] &= 1 \\ A\left[1 + \frac{4D\Delta t}{(\Delta x)^2}\sin^2\frac{k}{2}\right] &= 1 \\ A = \frac{1}{1 + \frac{4D\Delta t}{(\Delta x)^2}\sin^2\frac{k}{2}} &\le 1 \ . \end{aligned} \tag{7.23}$$

This scheme is stable for all step sizes, but might appear to be useless: how can we implement it since we don't know the forward values used in the space derivative? These future values are implicitly determined by the past values, and the trick is to recognize that the full set of equations can be inverted. The stability follows because peeking into the future in this way helps move information through the solution more quickly.

The boundary conditions are typically given as either *fixed* (u_1 and u_N are specified) or *periodic* ($u_1 = u_{N+1}$, so that the system does not have edges). If we assume fixed boundary conditions and define $\alpha = D\Delta t/(\Delta x)^2$, then equation (7.22) can be written as