Chapter 3. Finite Difference Approximations

By a small sample we may judge of the whole piece.
— MIGUEL DE CERVANTES, *Don Quixote de la Mancha, Chap. 1 (1615)*

This chapter begins our study of time-dependent partial differential equations, whose solutions vary both in time, as in Chapter 1, and in space, as in Chapter 2. The simplest approach to solving partial differential equations numerically is to set up a regular grid in space and time and compute approximate solutions on this grid by marching forwards in time. The essential point is discretization.

Finite difference modeling of partial differential equations is one of several fields of science that are concerned with the analysis of regular discrete structures. Another is **digital signal processing,** already mentioned in Chapter 1, where continuous functions are discretized in a similar fashion but for quite different purposes. A third is **crystallography,** which investigates the behavior of physical structures that are themselves discrete. The analogies between these three fields are close, and we shall occasionally point them out. The reader who wishes to pursue them further is referred to *Discrete-Time Signal Processing,* by A. V. Oppenheim and R. V. Schafer, and to *An Introduction to Solid State Physics,* by C. Kittel.

This chapter will describe five different ways to look at finite difference formulas—as discrete approximations to derivatives, as convolution filters, as Toeplitz matrices, as Fourier multipliers, and as derivatives of polynomial interpolants. Each of these points of view has its advantages, and the reader should become comfortable with all of them.

The field of partial differential equations is broad and varied, as is inevitable because of the great diversity of physical phenomena that these equations model. Much of the variety is introduced by the fact that practical problems usually involve one or more of the following complications:

- multiple space dimensions,
- systems of equations,
- boundaries,
- variable coefficients,
- nonlinearity.

To begin with, however, we shall concentrate on a simple class of problems: "pure" finite difference models for linear, constant-coefficient equations on an infinite one-dimensional domain. The fascinating phenomena that emerge from this study turn out to be fundamental to an understanding of the more complicated problems too.

3.1. Scalar model equations

Partial differential equations fall roughly into three great classes, which can be loosely described as follows:

elliptic – time-independent,

parabolic – time-dependent and diffusive,

hyperbolic – time-dependent and wavelike; finite speed of propagation.

In some situations, this trichotomy can be made mathematically precise, but not always, and we shall not worry about the rigorous definitions. The reader is referred to various books on partial differential equations, such as those by John, Garabedian, or Courant and Hilbert. There is a particularly careful discussion of hyperbolicity in G. B. Whitham's book *Linear and Nonlinear Waves*. For linear partial differential equations in general, the state of the art among pure mathematicians is set forth in the four-volume work by L. Hörmander, *The Analysis of Linear Partial Differential Operators*.

Until Chapter 9, we shall consider only time-dependent equations.

The simplest example of a hyperbolic equation is

$$u_t = u_x, \tag{3.1.1}$$

the one-dimensional **first-order wave equation**, which describes **advection** of a quantity $u(x,t)$ at the constant velocity -1. Given sufficiently smooth initial data $u(x,0) = u_0(x)$, (3.1.1) has the solution

$$u(x,t) = u_0(x+t), \tag{3.1.2}$$

as can be verified by inserting (3.1.2) in (3.1.1); see Figure 3.1.1b. This solution is unique. The propagation of energy at a finite speed is characteristic of hyperbolic partial differential equations, but this example is atypical in having all of the energy propagate at exactly the *same* finite speed.

The simplest example of a parabolic equation is

$$u_t = u_{xx}, \tag{3.1.3}$$

the one-dimensional **heat equation**, which describes **diffusion** of a quantity such as heat or salinity. In this book u_{xx} denotes the partial derivative $\partial^2 u/\partial x^2$, and similarly with u_{xxx}, u_{xt}, and so on. For an initial-value problem

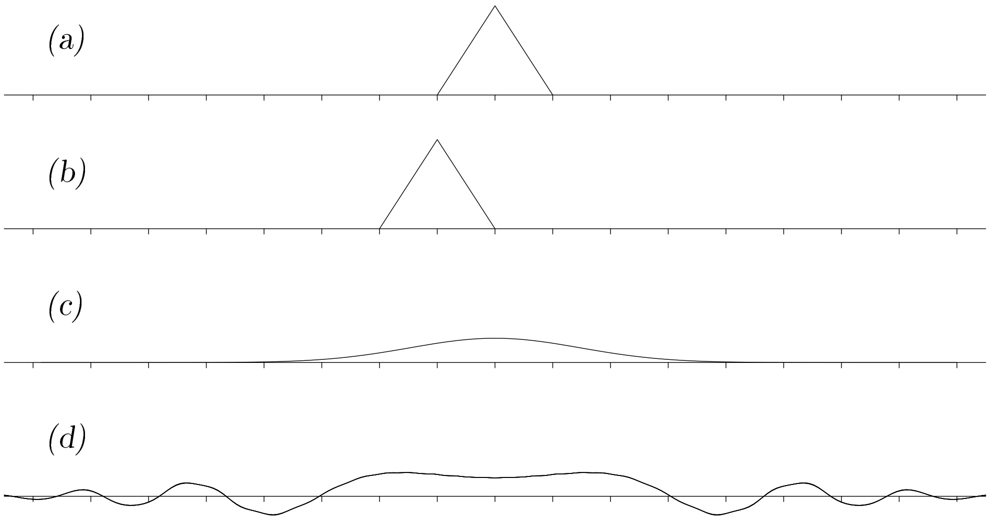


Figure 3.1.1. Evolution to $t=1$ of *(a)* hat-shaped initial data under *(b)* the wave equation (3.1.1), *(c)* the heat equation (3.1.3), and *(d)* the Schrödinger equation (3.1.5) (the real part is shown).

defined by (3.1.3) and sufficiently well-behaved initial data $u(x,0)=u_0(x)$, the solution

$$\begin{aligned} u(x,t) &= \frac{1}{2\pi}\int_{-\infty}^{\infty} e^{i\xi x-\xi^2 t}\,\hat{u}_0(\xi)\,d\xi \\ &= \frac{1}{\sqrt{4\pi t}}\int_{-\infty}^{\infty} e^{-(x-s)^2/4t} u_0(s)\,ds \end{aligned} \tag{3.1.4}$$

can be derived by Fourier analysis.* Physically, (3.1.4) asserts that the oscillatory component in the initial data of wave number ξ decays at the rate $e^{-\xi^2 t}$ because of diffusion, which is what one would expect from (3.1.3). See Figure 3.1.1c. Incidentally, (3.1.4) is not the only mathematically valid solution to the initial-value problem for (3.1.3). To make it unique, restrictions on $u(x,t)$ must be added such as a condition of boundedness as $|x|\to\infty$. This phenomenon of nonuniqueness is typical of parabolic partial differential equations, and results from the fact that (3.1.3) is of lower order with respect to t than x, so that $u_0(x)$ constitutes data on a "characteristic surface."

A third model equation that we shall consider from time to time is the one-dimensional **Schrödinger equation**,

$$u_t = iu_{xx}, \tag{3.1.5}$$

* In fact, it was Joseph Fourier who first derived the heat equation equation in 1807. He then invented Fourier analysis to solve it.

which describes the propagation of the complex state function in quantum mechanics and also arises in other fields such as underwater acoustics. Just as with the heat equation, the solution to the Schrödinger equation can be expressed by an integral,

$$u(x,t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} e^{i\xi x - i\xi^2 t}\,\hat{u}_0(\xi)\,d\xi$$
$$= \frac{1}{\sqrt{4\pi i t}}\int_{-\infty}^{\infty} e^{i(x-s)^2/4t}u_0(s)\,ds, \qquad (3.1.6)$$

but the behavior of solutions to this equation is very different. Schrödinger's equation is not diffusive but **dispersive**, which means that rather than decaying as t increases, solutions tend to break up into oscillatory wave packets. See Figure 3.1.1d.

Of course (3.1.1), (3.1.3), and (3.1.5) can all be modified to incorporate constant factors other than 1, so that they become $u_t = au_x$, $u_t = au_{xx}$, $u_t = iau_{xx}$. This affects the speed of advection, diffusion, or dispersion, but not the essential mathematics. The constant can be eliminated by a rescaling of x or t, so we omit it in the interests of simplicity (Exercise 3.1.1).

The behavior of our three model equations for a hat-shaped initial function is illustrated in Figure 3.1.1. The three waves shown there are obviously very different. In *(b)*, nothing has happened except advection. In *(c)*, strong dissipation or diffusion is evident: sharp corners have been smoothed. The Schrödinger result of *(d)* exhibits dispersion: oscillations have appeared in an initially non-oscillatory problem. These three mechanisms of advection, dissipation, and dispersion are central to the behavior of partial differential equations and their discrete models, and together account for most linear phenomena. We shall focus on them in Chapter 5.

Since many of the pages ahead are concerned with Fourier analysis of finite difference and spectral approximations to (3.1.1), (3.3.3), and (3.1.5), we should say a few words here about the Fourier analysis of the partial differential equations themselves. The fundamental idea is that when an equation is linear and has constant coefficients, it admits "plane wave" solutions of the form

$$u(x,t) = e^{i(\xi x + \omega t)}, \qquad \xi \in \mathbb{R}, \quad \omega \in \mathbb{C}, \qquad (3.1.7)$$

where ξ is again the **wave number** and ω is the **frequency**. Another way to put it is to say that if the initial data $u(x,0) = e^{i\xi x}$ are supplied to an equation of this kind, then there is a solution for $t > 0$ consisting of $u(x,0)$ multiplied by an oscillatory factor $e^{i\omega t}$. The difference between various equations lies in the different values of ω they assign to each wave number ξ, and this relationship,

$$\omega = \omega(\xi), \qquad (3.1.8)$$

is known as the **dispersion relation** for the equation. For first-order examples it might better be called a **dispersion function,** but higher-order equations typically provide multiple values of ω for each ξ, and so the more general term "relation" is needed. See Chapter 5.

It is easy to see what the dispersion relations are for our three model scalar equations. For example, substituting $e^{i(\xi x+\omega t)}$ into (3.1.1) gives $i\omega = i\xi$, or simply $\omega = \xi$. Here are the three results:

$$u_t = u_x: \quad \omega = \xi, \tag{3.1.9}$$

$$u_t = u_{xx}: \quad \omega = i\xi^2, \tag{3.1.10}$$

$$u_t = iu_{xx}: \quad \omega = -\xi^2. \tag{3.1.11}$$

Notice that for the wave and Schrödinger equations, $\omega \in \mathbb{R}$ for $x \in \mathbb{R}$; these equations conserve energy in the L^2 norm. For the heat equation, on the other hand, the frequencies are complex: every nonzero $\xi \in \mathbb{R}$ has $\operatorname{Im}\omega > 0$, which by (3.1.7) corresponds to an exponential decay, and the L^2 energy is not conserved.*

The solutions (3.1.4) and (3.1.6) can be derived by Fourier synthesis from the dispersion relations (3.1.10) and (3.1.11). For example, for the heat equation, (3.1.10) and (2.1.6) imply

$$\begin{aligned} u(x,t) &= \frac{1}{2\pi}\int_{-\infty}^{\infty} e^{i\xi x-\xi^2 t}\hat{u}_0(\xi)d\xi \\ &= \frac{1}{2\pi}\int_{-\infty}^{\infty} e^{i\xi x-\xi^2 t}\int_{-\infty}^{\infty} e^{-i\xi x'}u(x')\,dx'\,d\xi. \end{aligned} \tag{3.1.12}$$

From here to (3.1.4) is just a matter of algebra.

Equations (3.1.1), (3.1.3), and (3.1.5) will serve as our basic model equations for investigating the fundamentals of finite-difference and spectral methods. This may seem odd, since in all three cases the exact solutions are known, so that numerical methods are hardly called for. Yet the study of numerical methods for these equations will reveal many of the issues that come up repeatedly in more difficult problems. In some instances the reduction of complicated problems to simple models can be made quite precise. For example, a hyperbolic system of partial differential equations is defined to be one that can be locally diagonalized into a collection of problems of the form (3.1.1); see Chapter 6. In other instances the guidance given by the model equations is more heuristic.

*The **backwards heat equation** $u_t = -u_{xx}$ has the dispersion relation $\omega = -i\xi^2$, and its solutions blow up at an unbounded rate as as t increases unless the range of wave-numbers present is limited. The initial-value problem for this equation is ill-posed in L^2.

EXERCISES

▷ *3.1.1.* Show by rescaling x and/or t that the constants a and b can be eliminated in: *(a)* $u_t = au_x$, *(b)* $u_t = bu_{xx}$, *(c)* $u_t = au_x + bu_{xx}$.

▷ *3.1.2.* Consider the second-order wave equation $u_{tt} = u_{xx}$.

(a) What is the dispersion relation? Plot it in the real ξ–ω plane, and be sure to show all values of ω for each ξ.

(b) Verify that the function $u(x,t) = \frac{1}{2}[f(x+t)+f(x-t)]+\frac{1}{2}\int_{x-t}^{x+t} g(s)\,ds$ is the solution corresponding to initial data $u(x,0) = f(x)$, $u_t(x,0) = g(x)$.

▷ *3.1.3.*

(a) Verify that (3.1.4) and (3.1.6) represent solutions to (3.1.3) and (3.1.5)—both differential equation and initial conditions.

(b) Fill in the derivation of (3.1.4)—i.e., justify the second equals sign.

▷ *3.1.4.* Derive a Fourier integral representation of the solution (3.1.2) to the initial-value problem $u_t = u_x$, $u(x,0) = u_0(x)$.

▷ *3.1.5.*

(a) If (3.1.4) is written as a convolution $u(x,t) = u_0(x) * h_{[t]}(x)$, what is $h_{[t]}(x)$? (This function is called the **heat kernel**.)

(b) Prove that if $u_0(x)$ is a continuous function with compact support, then the resulting solution $u(x,t)$ to the heat equation is an entire function of x for each $t > 0$.

(c) Outline a proof of the **Weierstrass approximation theorem:** if f is a continuous function defined on an interval $[a,b]$, then for any $\epsilon > 0$, there exists a polynomial $p(x)$ such that $|f(x) - p(x)| < \epsilon$ for $x \in [a,b]$.

▶ *3.1.6.* *Method of characteristics.* Suppose $u_t = a(x,t)u_x$ and $u(x,0) = u_0(x)$ for $x \in \mathbb{R}$ and $t > 0$, where $a(x,t)$ is a smoothly varying positive function of x and t. Then $u(x,t)$ is constant along **characteristic curves** with slope $-1/a(x,t)$:

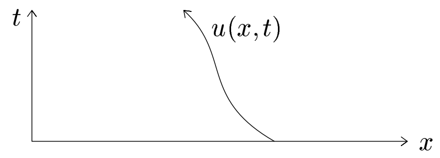


Figure 3.1.2

(a) Derive a representation for $u(0,1)$ as the solution to an ODE initial-value problem.

(b) Find $u(0,1)$ to five-digit accuracy for the problem $u_t = e^{(1+t)(1+\cos 3x)}u_x$, $u(x,0) = x$. Plot the appropriate characteristic curve.

(c) Find $u(0,1)$ to five-digit accuracy for the same equation defined on the interval $x \in [-1,1]$ with right-hand boundary condition $u(1,t) = 1+t$. Plot the appropriate characteristic curve.

3.2. Finite difference formulas

Let $h>0$ and $k>0$ be a fixed **space step** and **time step**, respectively, and set $x_j = jh$ and $t_n = nk$ for any integers j and n. The points (x_j, t_n) define a regular **grid** or **mesh** in two dimensions, as shown in Figure 3.2.1—formally, the subset $h\mathbb{Z}\times k\mathbb{Z}$ of $\mathbb{R}^2$. For the rest of this book our aim is to approximate continuous functions $u(x,t)$ by **grid functions** v_j^n,

$$v_j^n \approx u(x_j, t_n). \tag{3.2.1}$$

The notation $v(x_j, t_n) = v_j^n$ will also be convenient, and we shall sometimes write v^n or $v(t_n)$ to represent the spatial grid function $\{v_j^n\}$, $j\in\mathbb{Z}$, for a fixed value n.

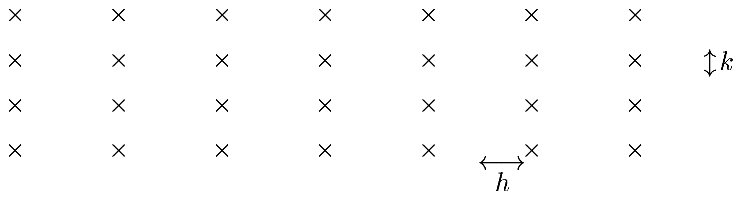


Figure 3.2.1. Regular finite difference grid in x and t.

The purpose of discretization is to obtain a problem that can be solved by a finite procedure. The simplest kind of finite procedure is an **s-step finite difference formula**, which is a fixed formula that prescribes v_j^{n+1} as a function of a finite number of other grid values at time steps $n+1-s$ through n (explicit case) or $n+1$ (implicit case). To compute an approximation $\{v_j^n\}$ to $u(x,t)$, we shall begin with initial data $v^0,\dots,v^{s-1}$, and then compute values $v^s, v^{s+1},\dots$ in succession by applying the finite difference formula. This process is sometimes known as **marching** with respect to t.

A familiar example of a finite difference model for the first-order wave equation (3.1.1) is the **leap frog** (LF) formula,

$$\text{LF}: \qquad \frac{1}{2k}(v_j^{n+1} - v_j^{n-1}) = \frac{1}{2h}(v_{j+1}^n - v_{j-1}^n). \tag{3.2.2}$$

This equation can be obtained from (3.1.1) by replacing the partial derivatives in x and t by centered finite differences. The analogous leap frog type approximation to the heat equation (3.1.3) is

$$\text{LF}_{xx}: \qquad \frac{1}{2k}(v_j^{n+1} - v_j^{n-1}) = \frac{1}{h^2}(v_{j+1}^n - 2v_j^n + v_{j-1}^n). \tag{3.2.3}$$

However, we shall see that this formula is unstable. A better approximation is the **Crank-Nicolson*** (CN) formula,

$$\text{CN}: \qquad \frac{1}{k}(v_j^{n+1} - v_j^n) = \frac{1}{2}\left[\frac{1}{h^2}(v_{j+1}^n - 2v_j^n + v_{j-1}^n) + \frac{1}{h^2}(v_{j+1}^{n+1} - 2v_j^{n+1} + v_{j-1}^{n+1})\right], \tag{3.2.4}$$

* Spelling note #1: the name is "Nicolson", not "Nicholson".

which is said to be **implicit**, since it couples together the values v_j^{n+1} at the new time step and therefore leads to a system of equations to be solved. In contrast, leap frog formulas are **explicit**. One can also define a CN formula for (3.1.1), namely

$$\mathrm{CN}_x: \qquad \frac{1}{k}(v_j^{n+1}-v_j^n) = \frac{1}{2}\left[\frac{1}{2h}(v_{j+1}^n - v_{j-1}^n) + \frac{1}{2h}(v_{j+1}^{n+1}-v_{j-1}^{n+1})\right], \tag{3.2.5}$$

but we shall see that since explicit formulas such as LF are stable for (3.1.1), and easier to implement, an implicit formula like (3.2.5) has little to recommend it in this case. Another famous and extremely important explicit approximation for $u_t = u_x$ is the **Lax-Wendroff** formula, discovered in 1960:

$$\mathrm{LW}: \qquad \frac{1}{k}(v_j^{n+1}-v_j^n) = \frac{1}{2h}(v_{j+1}^n - v_{j-1}^n) + \frac{k}{2h^2}(v_{j+1}^n - 2v_j^n + v_{j-1}^n). \tag{3.2.6}$$

The second term on the right is the first we have encountered whose function is not immediately obvious; we shall see later that it raises the order of accuracy from 1 to 2. We shall see also that although the leap frog formula may be suitable for linear hyperbolic problems such as arise in acoustics, the nonlinear hyperbolic problems of fluid mechanics generally require a formula like Lax-Wendroff that dissipates energy at high wave numbers.

We shall often use acronyms such as LF, CN, and LW to abbreviate the names of standard finite difference formulas, as above, and subscripts x or xx will be added sometimes to distinguish between a model of the wave equation and a model of the heat equation. For the formulas that are important in practice we shall usually manage to avoid the subscripts.

Of the examples above, as already mentioned, LF and CN are important in practice, while LF_{xx} and CN_x are not so important.

Before introducing further finite difference formulas, we need a more compact notation. Chapter 1 introduced the **time shift operator** Z,

$$Zv_j^n = v_j^{n+1}. \tag{3.2.7}$$

Similarly, let K denote the **space shift operator**

$$Kv_j^n = v_{j+1}^n, \tag{3.2.8}$$

and let I or 1 represent the **identity operator**,

$$Iv_j^n = 1v_j^n = v_j^n. \tag{3.2.9}$$

We shall make regular use of the following discrete operators acting in the space direction:

SPATIAL DIFFERENCE AND AVERAGING OPERATORS.

$$\mu_+ = \tfrac{1}{2}(I+K), \qquad \mu_- = \tfrac{1}{2}(K^{-1}+I) \qquad \mu_0 = \tfrac{1}{2}(K^{-1}+K), \tag{3.2.10}$$

$$\delta_+ = \frac{1}{h}(K-I), \qquad \delta_- = \frac{1}{h}(I-K^{-1}), \qquad \delta_0 = \frac{1}{2h}(K-K^{-1}), \tag{3.2.11}$$

$$\delta_\times = \frac{1}{h^2}(K-2I+K^{-1}). \tag{3.2.12}$$

μ_+, μ_-, and μ_0 are known as **forward**, **backward**, and **centered spatial averaging operators**, δ_+, δ_+, and δ_0 are the corresponding **spatial difference operators** of first order, and $\delta_\times$ is a centered spatial difference operator of the second order. For discretization in time we shall use exactly the same notation, but with superscripts instead of subscripts:*

TEMPORAL DIFFERENCE AND AVERAGING OPERATORS.

$$\mu^+ = \tfrac{1}{2}(I+Z), \qquad \mu^- = \tfrac{1}{2}(Z^{-1}+I) \qquad \mu^0 = \tfrac{1}{2}(Z^{-1}+Z), \tag{3.2.13}$$

$$\delta^+ = \frac{1}{k}(Z-I), \qquad \delta^- = \frac{1}{k}(I-Z^{-1}), \qquad \delta^0 = \frac{1}{2k}(Z-Z^{-1}), \tag{3.2.14}$$

$$\delta^\times = \frac{1}{k^2}(Z-2I+Z^{-1}). \tag{3.2.15}$$

In this notation, for example, the LF and CN formulas (3.2.2) and (3.2.4) can be rewritten

$$\text{LF: } \delta^0 v = \delta_0 v, \qquad \text{CN: } \delta^+ v = \mu^+ \delta_\times v.$$

Note that since Z and K commute, i.e., $ZK = KZ$, the order of the terms in any product of these discrete operators can be permuted at will. For example, we might have written $\delta_\times \mu^+$ above instead of $\mu^+\delta_\times$.

Since all of these operators depend on h or on k, a more complete notation would be $\delta_+(h)$, $\delta_-(h)$, $\delta_0(h)$, etc. For example, the symbol $\delta_0(2h)$ is defined by

$$\delta_0(2h)v_j = \frac{1}{4h}(K^2 - K^{-2})v_j = \frac{1}{4h}(v_{j+2} - v_{j-2}), \tag{3.2.16}$$

and similarly for $\delta_0(3h)$, etc. (Here and in subsequent formulas, subscripts or superscripts are omitted when they are irrelevant to the discrete process under consideration.)

In general there may be many ways to write a difference operator. For example,

$$\delta_0 = \tfrac{1}{2}(\delta_+ + \delta_-), \qquad \delta_\times = \delta_+\delta_- = \delta_-\delta_+ = [\delta_0(\tfrac{1}{2}h)]^2.$$

As indicated above, a finite difference formula is **explicit** if it contains only one nonzero term at time level $n+1$ (e.g. LF), and **implicit** if it contains several (e.g. CN). As in the ODE case, implicit formulas are typically more stable than explicit ones, but harder to implement. On an unbounded domain in space, in fact, an implicit formula would seem to require the solution of an infinite system of equations to get from v^n to v^{n+1}! This is essentially true, and in practice, a finite difference formula is usually applied on a bounded mesh, where a finite system of equations must be solved. Thus our discussion of unbounded meshes will be mainly a theoretical device—but an important one, for many of the stability and accuracy phenomena that need to be understood have nothing to do with boundaries.

In implementing implicit finite difference formulas, there is a wide gulf between one-dimensional problems, which lead to matrices whose nonzero entries are concentrated in a

*The notations δ_0, δ_+, δ_-, μ_0, μ_+, μ_- are reasonably common if not quite standard. The other notations of this section are not standard.

narrow band, and multidimensional problems, which do not. The problem of how to solve such systems of equations efficiently is one of great importance, to which we shall return in §3.4 and in Chapters 9.

We are now equipped to present a number of well-known finite difference formulas for the wave and heat equations. These are listed in Tables 3.2.1 (wave equation) and 3.2.2 (heat equation), and the reader should take the time to become familiar with them. The tables make use of the abbreviations

$$\lambda = \frac{k}{h}, \qquad \sigma = \frac{k}{h^2}, \tag{3.2.17}$$

which will appear throughout the book. The diagram to the right of each formula in the tables, whose meaning should be self-evident, is called the **stencil** of that formula. More extensive lists of formulas can be found in a number of books. For the heat equation, for example, see Chapter 8 of the book by Richtmyer and Morton.

Of the formulas mentioned in the tables, the ones most often used in practice are probably LF, UW (**upwind**), and LW (**Lax-Wendroff**) for hyperbolic equations, and CN and DF (**DuFort-Frankel**) for parabolic equations. However, computational problems vary enormously, and these judgments should not be taken too seriously.

As with linear multistep formulas for ordinary differential equations, it is useful to have a notation for an arbitrary finite difference formula for a partial differential equation. The following is an analog of equation (1.2.11):

An s-step **linear finite difference formula** *is a scalar formula*

$$\sum_{\nu=0}^{s} \sum_{\mu=-\ell}^{r} \alpha_{\mu\nu} v_{j+\mu}^{n+1-\nu} = 0 \tag{3.2.18}$$

for some constants $\{\alpha_{\mu\nu}\}$ *with* $\alpha_{00} \neq 0$, $\alpha_{-\ell,\nu_1} \neq 0$ *for some* ν_1, *and* $\alpha_{r,\nu_2} \neq 0$ *for some* ν_2. *If* $\alpha_{\mu 0} = 0$ *for all* $\mu \neq 0$ *the formula is* **explicit**, *whereas if* $\alpha_{\mu 0} \neq 0$ *for some* $\mu \neq 0$ *it is* **implicit**. *Equation (3.2.18) also describes a vector-valued finite difference formula; in this case each* v_j^n *is an* N*-vector, each* $\alpha_{\mu\nu}$ *is an* $N \times N$ *matrix, and the conditions* $\alpha_{\mu\nu} \neq 0$ *become* $\det \alpha_{\mu\nu} \neq 0$.

The analogy between (3.2.18) (linear finite difference formulas) and (1.2.11) (linear multistep formulas) is imperfect. What has become of the quantities $\{f^n\}$ in (1.2.11)? The answer is that (1.2.11) was a general formula that applied to any ODE defined by a function $f(u,t)$, possibly nonlinear; the word "linear" there referred to the way f enters into the formula, not to the nature of f itself. In (3.2.18), by contrast, we have assumed that the terms analogous to $f(u,t)$ in the partial differential equation are themselves linear and have been incorporated into the discretization. Thus (3.2.18) is more precisely analogous to (1.7.4).

EXERCISES

▶ *3.2.1. Computations for Figure 3.1.1.* The goal of this problem is to calculate the curves of Figure 3.1.1 by finite difference methods. In all parts below, your mesh should extend over an interval $[-M,M]$ large enough to be effectively infinite. At the boundaries it is simplest to impose the boundary conditions $u(-M,t)=u(M,t)=0$.

It will probably be easiest to program all parts together in a single collection of subroutines, which accepts various input parameters to control h, k, M, choice of finite difference formula, and so on. Note that parts *(c)*, *(f)*, and *(g)* involve complex arithmetic.

The initial function for all parts is $u_0(x)=\max\{0,1-|x|\}$, and the computation is carried to $t=1$.

Please make your output compact by combining plots and numbers on a page wherever appropriate.

(a) Lax-Wendroff for $u_t=u_x$. Write a program to solve $u_t=u_x$ by the LW formula with $k=2h/5$, $h=1/2,1/4,\dots,1/64$. Make a table of the computed values $v(-.5,1)$, and the error in these values, for each h. Make a plot showing the superposition of the results (i.e. $v(x,1)$) for various h, and comment.

(b) Euler for $u_t=u_{xx}$. Extend the program to solve $u_t=u_{xx}$ by the EU_{xx} formula with $k=2h^2/5$, $h=1/2,1/4,\dots,1/16$. Make a table listing $v(1,1)$ for each h. Plot the results and comment on them.

(c) Euler for $u_t=iu_{xx}$. Now solve $u_t=iu_{xx}$ by the EU_{xx} formula modified in the obvious way, with $k=2h^2/5$, $h=1/2,1/4$; can you go further? Make a table listing $v(1,1)$ for each h. Your results will be unstable. Explain why this has happened by drawing a sketch that compares the stability region of a linear multistep formula to the set of eigenvalues of a spatial difference operator. (This kind of analysis is discussed in the next section.)

(d) Tridiagonal system of equations. To compute the answer more efficiently for the heat equation, and to get any answer at all for Schrödinger's equation, it is necessary to use an implicit formula, which involves the solution of a tridiagonal system of equations at each time step. Write a subroutine *TRDIAG*(n,c,d,e,b,x) to solve the linear system of equations $Ax=b$, where A is the $n\times n$ tridiagonal matrix defined by $a_{i+1,i}=c_i$, $a_{ii}=d_i$, $a_{i,i+1}=e_i$. The method to use is Gaussian elimination without pivoting of rows or columns;* if you are in doubt about how to do this, you can find details in many books on numerical linear algebra or numerical solution of partial differential equations. Test *TRDIAG* carefully, and report the solution of the system

$$\begin{pmatrix}5&1&0&0\\1&5&2&0\\0&2&4&1\\0&0&1&4\end{pmatrix}\begin{pmatrix}x_1\\x_2\\x_3\\x_4\end{pmatrix}=\begin{pmatrix}7\\17\\20\\19\end{pmatrix}.$$

(e) Crank-Nicolson for $u_t=u_{xx}$. Write down carefully the tridiagonal matrix equation that is involved when $u_t=u_{xx}$ is solved by the formula CN. Apply *TRDIAG* to carry out this computation with $k=\frac{1}{2}h$, $h=1/4,1/8,\dots,1/64$. Make a table listing $v(1,1)$ for each h. Plot the results and comment on them.

(f) Crank-Nicolson for $u_t=iu_{xx}$. Now write down the natural modification of CN for solving $u_t=iu_{xx}$. Making use of *TRDIAG* again, solve this equation with $k=\frac{1}{2}h$, $h=$

*The avoidance of pivoting is justifiable provided that the matrix A is diagonally dominant, as it will be in the examples we consider. Otherwise Gaussian elimination may be unstable; see Golub & Van Loan, *Matrix Computations*, 2nd ed., Johns Hopkins, 1989.

$1/4, 1/8, \ldots, 1/32$. Make tables listing $v(1,1)$ and $v(6,1)$ for each h. Plot the results—both $\mathrm{Re}\, v(x,1)$ and $|v(x,1)|$, superimposed on a single graph—and comment on them. How far away does the boundary at $x = M$ have to be to yield reasonable answers?

(g) *Artificial dissipation.* In part *(f)* you may have observed spurious wiggles contaminating the solution. These can be blamed in part on unwanted reflections at the numerical boundaries at $x = \pm M$, and we shall have more to say about them in Chapters 5 and 6. To suppress these wiggles, try adding a **artificial dissipation** term to the right-hand side of the finite difference formula, such as

$$\frac{a}{h}\mu^+ \delta_\times v_j^n \approx h u_{xx}(x_j, t_n) \tag{3.2.19}$$

for some $a > 0$. What choices of M and a best reproduce Figure 3.1.1d? Does it help to apply the artificial dissipation only near the boundaries $x = \pm M$?

▶ *3.2.2. Model equations with nonlinear terms.* Our model equations develop some interesting solutions if nonlinear terms are added. Continuing the above exercise, modify your programs to compute solutions to the following partial differential equations, all defined in the interval $x \in [-1,1]$ and with boundary conditions $u(\pm 1) = 0$. Devise whatever strategies you can think of to handle the nonlinearities successfully; such problems are discussed more systematically in [??].

(a) *Burgers* equation:* $u_t = (\frac{1}{2}u^2)_x + \epsilon u_{xx}$, $\epsilon > 0$. Consider a Lax-Wendroff type of formula with, say, $\epsilon = 0.1$, and initial data the same as in Figure 3.1.1. How does the numerical solution behave as t increases? How do you think the exact mathematical solution should behave?

(b) *Nonlinear heat equation:* $u_t = u_{xx} + e^u$, $u(x,0) = 0$. For this problem you will need a variant of the Crank-Nicolson formula or perhaps the backward Euler formula. With the aid of a simple adaptive time-stepping strategy, generate a persuasive sequence of plots illustrating the "blow-up" of the solution that occurs. Make a plot of $\|u(\cdot,t)\|_\infty$—the maximum value of u—as a function of t. What is your best estimate, based on comparing results with several grid resolutions, of the time at which $\|u(\cdot,t)\|_\infty$ becomes infinite?

(c) *Nonlinear heat equation:* $u_t = u_{xx} + u^5$, $u(x,0) = 1 + \cos(\pi x)$. Repeat part *(b)* for this new nonlinearity. Again, with the aid of a simple adaptive time-stepping strategy, generate a persuasive sequence of plots illustrating the blow-up of the solution, and make a plot of $\|u(\cdot,t)\|_\infty$ as a function of t. What is your best estimate of the time at which $\|u(\cdot,t)\|_\infty$ becomes infinite?

(d) *Nonlinear Schrödinger equation:* $u_t = iu_{xx} + \alpha|u|^2 u$, $\alpha > 0$. Take the initial data from Figure 3.1.1 again. How does the solution behave as a function of t, and how does the behavior depend on α? Again, try to generate a good set of plots, and estimate the critical value of t if there is one.

---

* Spelling note #2: the name is "Burgers", so one may write "Burgers' equation" but never "Burger's equation".

(EU_x = Euler) $\delta^+ v = \delta_0 v$	$v_j^{n+1} = v_j^n + \frac{1}{2}\lambda(v_{j+1}^n - v_{j-1}^n)$
(BE_x = Backward Euler) $\delta^- v = \delta_0 v$	$v_j^{n+1} = v_j^n + \frac{1}{2}\lambda(v_{j+1}^{n+1} - v_{j-1}^{n+1})$
(CN_x = Crank-Nicolson) $\delta^+ v = \mu^+ \delta_0 v$	$v_j^{n+1} = v_j^n + \frac{1}{4}\lambda(v_{j+1}^n - v_{j-1}^n) + \frac{1}{4}\lambda(v_{j+1}^{n+1} - v_{j-1}^{n+1})$
LF = Leap frog $\delta^0 v = \delta_0 v$	$v_j^{n+1} = v_j^{n-1} + \lambda(v_{j+1}^n - v_{j-1}^n)$
BOX_x = Box $\mu_+ \delta^+ v = \mu^+ \delta_+ v$	$(1+\lambda)v_j^{n+1} + (1-\lambda)v_{j+1}^{n+1} = (1-\lambda)v_j^n + (1+\lambda)v_{j+1}^n$
LF4 = Fourth-order Leap frog $\delta^0 v = \frac{4}{3}\delta_0(h)v - \frac{1}{3}\delta_0(2h)v$	$v_j^{n+1} = v_j^{n-1} + \frac{4}{3}\lambda(v_{j+1}^n - v_{j-1}^n) - \frac{1}{6}\lambda(v_{j+2}^n - v_{j-2}^n)$
LXF = Lax-Friedrichs $\frac{1}{k}(Z - \mu_0)v = \delta_0 v$	$v_j^{n+1} = \frac{1}{2}(v_{j+1}^n + v_{j-1}^n) + \frac{1}{2}\lambda(v_{j+1}^n - v_{j-1}^n)$
UW = Upwind $\delta^+ v = \delta_+ v$	$v_j^{n+1} = v_j^n + \lambda(v_{j+1}^n - v_j^n)$
LW = Lax-Wendroff (1960) $\delta^+ v = \delta_0 v + \frac{1}{2}k\delta_\times v$	$v_j^{n+1} = v_j^n + \frac{1}{2}\lambda(v_{j+1}^n - v_{j-1}^n) + \frac{1}{2}\lambda^2(v_{j+1}^n - 2v_j^n + v_{j-1}^n)$

Table 3.2.1. Finite difference approximations for the first-order wave equation $u_t = u_x$, with $\lambda = k/h$. For the equation $u_t = au_x$, replace λ by λa in each formula. Names in parenthesis mark formulas that are not normally useful in practice. Orders of accuracy and stability restrictions are listed in Table 4.4.1.

EU_{xx} = Euler

$\delta^+ v = \delta_\times v$ $\qquad v_j^{n+1} = v_j^n + \sigma(v_{j+1}^n - 2v_j^n + v_{j-1}^n)$

BE_{xx} = Backward Euler (Laasonen, 1949)

$\delta^- v = \delta_\times v$ $\qquad v_j^{n+1} = v_j^n + \sigma(v_{j+1}^{n+1} - 2v_j^{n+1} + v_{j-1}^{n+1})$

CN = Crank-Nicolson (1947)

$\delta^+ v = \mu^+ \delta_\times v$ $\qquad v_j^{n+1} = v_j^n + \frac{1}{2}\sigma(v_{j+1}^n - 2v_j^n + v_{j-1}^n) + \frac{1}{2}\sigma(v_{j+1}^{n+1} - 2v_j^{n+1} + v_{j-1}^{n+1})$

(LF_{xx} = Leap frog)

$\delta^0 v = \delta_\times v$ $\qquad v_j^{n+1} = v_j^{n-1} + 2\sigma(v_{j+1}^n - 2v_j^n + v_{j-1}^n)$

BOX_{xx} = Box

$(\frac{5}{6}I + \frac{1}{6}\mu_0)\delta^+ v = \mu^+ \delta_\times v$

CN4 = Fourth-order Crank-Nicolson

$\delta^+ v = \mu^+ [\frac{4}{3}\delta_\times(h) - \frac{1}{3}\delta_\times(2h)]v$

DF = DuFort-Frankel (1953)

$\delta^0 v = h^{-2}(K - 2\mu^0 + K^{-1})v$ $\qquad v_j^{n+1} = v_j^{n-1} + 2\sigma(v_{j+1}^n - (v_j^{n-1} + v_j^{n+1}) + v_{j-1}^n)$

SA = Saul'ev (1957)

Table 3.2.2. Finite difference approximations for the heat equation $u_t = u_{xx}$, with $\sigma = k/h^2$. For the equation $u_t = au_{xx}$, replace σ by σa in each formula. Names in parenthesis mark formulas that are not normally useful in practice. Orders of accuracy and stability restrictions are listed in Table 4.4.2.

3.3. Spatial difference operators and the method of lines

In designing a finite difference method for a time-dependent partial differential equation, it is often useful to divide the process into two steps: first, discretize the problem with respect to space, thereby generating a system of ordinary differential equations in t; next, solve this system by some discrete method in time. Not all finite difference methods can be analyzed this way, but many can, and it is a point of view that becomes increasingly important as one considers more difficult problems.

EXAMPLE 3.3.1. For example, suppose $u_t = u_x$ is discretized in space by the approximation $\delta_0 \approx \partial/\partial x$. Then the PDE becomes

$$v_t = \delta_0 v, \tag{3.3.1}$$

where v represents an infinite sequence $\{v_j(t)\}$ of functions of t. This is an infinite system of ordinary differential equations, each one of the form

$$\frac{\partial v_j}{\partial t} = \frac{1}{2h}(v_{j+1} - v_{j-1}). \tag{3.3.2}$$

On a bounded domain the system would become finite, though possibly quite large.

A system of ordinary differential equations of this kind is known as a **semidiscrete** approximation to a partial differential equation. The idea of constructing a semidiscrete approximation and then solving it by a numerical method for ordinary differential equations is known as the **method of lines**. The explanation of the name is that one can think of $\{v_j(t)\}$ as an approximation to $u(x,t)$ defined on an array of parallel lines in the x-t plane, as suggested in Figure 3.3.1:

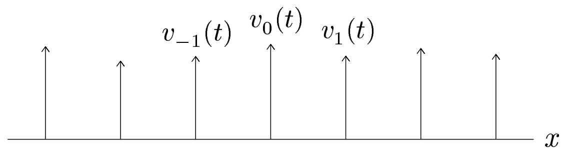


Figure 3.3.1. The "method of lines"—semidiscretization of a time-dependent PDE.

EXAMPLE 3.3.1, CONTINUED. Several of the formulas of the last section can be interpreted as time-discretizations of (3.3.1) by linear multistep formulas. The Euler and Backward Euler discretizations (1.2.3) and (1.2.4) give the Euler and Backward Euler formulas listed in Table 3.2.1. The trapezoid rule (1.2.5) gives the Crank-Nicolson formula of (3.2.5), and the midpoint rule (1.2.6) gives the leap frog formula of (3.2.2). On the other hand the upwind formula comes from the Euler discretization, like the Euler formula, but with the spatial difference operator δ_+ instead of δ_0. The Lax-Wendroff and Lax-Friedrichs formulas do not fit the semidiscretization framework.

The examples just considered were first- and second-order accurate approximations with respect to t (the definition of order of accuracy will come in §4.2). Higher-order time discretizations for partial differential equations have also become popular in recent years, although one would rarely go so far as the sixth- or eighth-order formulas that appear in many adaptive ODE codes. The advantage of higher-order methods is, of course, accuracy. One disadvantage is complexity, both of analysis and of implementation, and another is computer storage. For an ODE involving a few dozen variables, there is no great difficulty if three or four time levels of data must be stored, but for a large-scale PDE—for example, a system of five equations defined on a $200\times200\times200$ mesh in three dimensions—the storage requirements become quite large.

The idea of semidiscretization focuses attention on spatial difference operators as approximations of spatial differential operators. It happens that just as in Chapter 1, many of the approximations of practical interest can be derived by a process of interpolation. Given data on a discrete mesh, the idea is as follows:

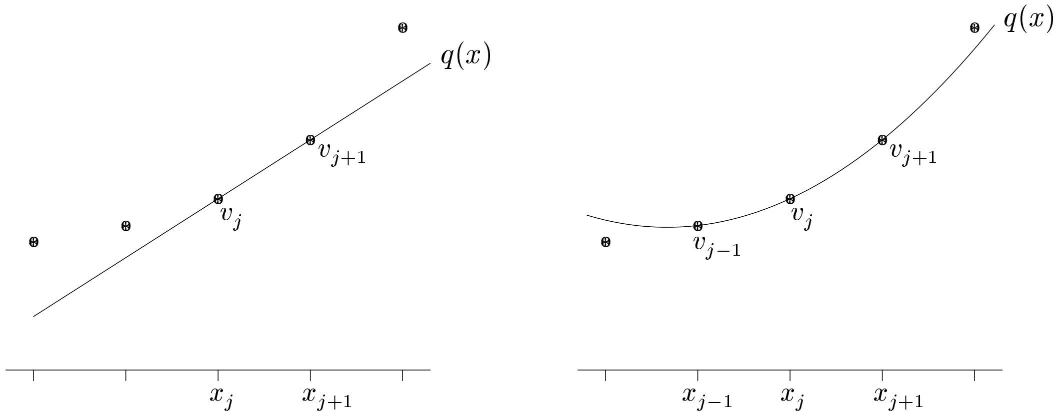
(1) Interpolate the data;
(2) Differentiate the interpolant at the mesh points. (3.3.3)

In step *(2)* one differentiates once for a first-order difference operator, twice for a second-order difference operator, and so on. The spatial discretizations of many finite difference and spectral methods fit the scheme (3.3.3); the variations among them lie in in the nature of the grid, the choice of interpolating functions, and the order of differentiation.

EXAMPLE 3.3.2.

*First order of accuracy.** For example, suppose data v_j, v_{j+1} are interpolated by a polynomial $q(x)$ of degree 1. Then $\delta_+ v_j = q_x(x_j)$. See Figure 3.3.2a.

*Unfortunately, the word "order" customarily refers both to the order of a differential or difference operator, and to the order of accuracy of the latter as an approximation to the former. The reader is advised to be careful.



(a) δ_+ (b) δ_0

Figure 3.3.2. Derivation of spatial difference operators via polynomial interpolation.

Second order of accuracy. Let data v_{j-1}, v_j, v_{j+1} be interpolated by a polynomial $q(x)$ of degree 2. Then $\delta_0 v_j = q_x(x_j)$ and $\delta_\times v_j = q_{xx}(x_j)$. See Figure 3.3.2b.

Fourth order of accuracy. Let $v_{j-2}, v_{j-1}, v_j, v_{j+1}, v_{j+2}$ be interpolated by a polynomial $q(x)$ of degree 4. Then $q_x(x_j) = \frac{4}{3}\delta_0(h)v_j - \frac{1}{3}\delta_0(2h)v_j$, the fourth-order approximation listed in Table 3.2.1.

To proceed systematically, let $x_0, x_1, \dots, x_{n_{\max}}$ be a set of arbitrary distinct points of $\mathbb{R}$, not necessarily uniformly spaced. Suppose we wish to derive the coefficients c_{nj}^m of all of the spatial difference operators centered at $x=0$ of orders $0 \le m \le m_{\max}$ based on any initial subset $x_0, x_1, \dots, x_n$ of these points. That is, we want to derive all of the approximations

$$\frac{d^m f}{dx^m}(0) \approx \sum_{j=0}^{n} c_{nj}^m f(x_j) \qquad 0 \le m \le m_{\max}, \quad m_{\max} \le n \le n_{\max} \tag{3.3.4}$$

in a systematic fashion. The following surprisingly simple algorithm for this purpose was published by Bengt Fornberg in "Generation of finite difference formulas on arbitrarily spaced grids," *Math. Comp. 51* (1988), 699–706.

FINITE DIFFERENCE APPROXIMATIONS ON AN ARBITRARY GRID

Theorem 3.1. *Given $m_{\max} \ge 0$ and $n_{\max} \ge m_{\max}$, the following algorithm computes coefficients of finite difference approximations in arbitrary distinct points $x_0, \dots, x_{n_{\max}}$ to $\partial^m/\partial x^m$ $(0 \le m \le m_{\max})$ at $x = 0$, as described above:*

$c_{00}^0 := 1$, $\alpha := 1$
for $n := 1$ **to** $n_{\max}$
  $\beta := 1$
  for $j := 0$ **to** $n-1$
    $\beta := \beta(x_n - x_j)$
    if $n \le m_{\max}$ **then** $c_{n-1,j}^n := 0$
    for $m := 0$ **to** $\min(n, m_{\max})$
      $c_{nj}^m := (x_n c_{n-1,j}^m - m\, c_{n-1,j}^{m-1})/(x_n - x_j)$
  for $m := 0$ **to** $\min(n, m_{\max})$
    $c_{n,n}^m := \alpha(m\, c_{n-1,n-1}^{m-1} - x_{n-1} c_{n-1,n-1}^m)/\beta$
  $\alpha := \beta$

(The undefined quantities $c_{n-1,j}^{-1}$ appearing for $m = 0$ may be taken to be 0.)

Proof. [Not yet written] ∎

From this single algorithm one can derive coefficients for centered, one-sided, and much more general approximations to all kinds of derivatives. A number are listed in Tables 3.3.1–3.3.3 at the end of this section; see also Exercise 3.3.2.

If the grid is regular, then simple formulas can be derived for these finite difference approximations. In particular, let D_{2p} denote the discrete first-order spatial difference operator obtained by interpolating $v_{j-p}, \dots, v_{j+p}$ by a polynomial $q(x)$ of degree $2p$ and then differentiating q once at x_j, and let $D_{2p}^{(m)}$ be the analogous higher-order difference operator obtained by differentiating m times. Then we have, for example,

$$D_2 = \delta_0(h), \qquad D_2^{(2)} = \delta_\times(h), \tag{3.3.5}$$

and

$$D_4 := \tfrac{4}{3}\delta_0(h) - \tfrac{1}{3}\delta_0(2h), \qquad D_4^{(2)} := \tfrac{4}{3}\delta_\times(h) - \tfrac{1}{3}\delta_\times(2h), \tag{3.3.6}$$

and

$$D_6 := \tfrac{3}{2}\delta_0(h) - \tfrac{3}{5}\delta_0(2h) + \tfrac{1}{10}\delta_0(3h), \tag{3.3.7}$$

$$D_6^{(2)} := \tfrac{3}{2}\delta_\times(h) - \tfrac{3}{5}\delta_\times(2h) + \tfrac{1}{10}\delta_\times(3h). \tag{3.3.8}$$

The corresponding coefficients appear explicitly in Table 3.3.1.

The following theorem gives the coefficients for first- and second-order formulas of arbitrary order of accuracy:

CENTERED FINITE DIFFERENCE APPROXIMATIONS ON A REGULAR GRID

Theorem 3.2. *For each integer $p \ge 1$, there exist unique first-order and second-order difference operators D_{2p} and $D_{2p}^{(2)}$ of order of accuracy $2p$ that utilize the points $v_{j-p}, \dots, v_{j+p}$, namely:*

$$D_{2p} := \sum_{j=1}^{p} \alpha_j \, \delta_0(jh), \qquad D_{2p}^{(2)} := \sum_{j=1}^{p} \alpha_j \, \delta_\times(jh), \tag{3.3.9}$$

where

$$\alpha_j := 2(-1)^{j+1} \binom{p}{p-j} \Big/ \binom{p+j}{p} := \frac{2(-1)^{j+1}(p!)^2}{(p-j)!\,(p+j)!}. \tag{3.3.10}$$

Proof. [Not yet written] ∎

As $p \to \infty$, (3.3.9) and (3.3.10) have the following formal limits:

$$D_\infty := 2\delta_0(h) - 2\delta_0(2h) + 2\delta_0(3h) - \cdots. \tag{3.3.11}$$

$$D_\infty^{(2)} := 2\delta_\times(h) - 2\delta_\times(2h) + 2\delta_\times(3h) - \cdots. \tag{3.3.12}$$

These series look both infinite and nonconvergent—unimplementable and possibly even meaningless! However, that is far from the case. In fact they are precisely the first-order and second-order **spectral differentiation operators** for data defined on the infinite grid $h\mathbb{Z}$. The corresponding interpolation processes involve trigonometric or sinc functions rather than polynomials:

(1) Interpolate the data by sinc functions as in §2.3;
(2) Differentiate the interpolant at the mesh points. (3.3.13)

As was described in §2.3, such a procedure can be implemented by a semidiscrete Fourier transform, and it is well defined for all data $v \in \ell_h^2$. The uses of these operators will be the subject of Chapter 7.

One useful way to interpret spatial differencing operators such as D_{2p} is in terms of convolutions. From (2.2.3), it is easy to verify that the first-order operators mentioned above can be expressed as

$$D_2 v := \frac{1}{h^2}(\cdots 0 \quad 0 \quad 0 \quad \tfrac{1}{2} \quad 0 \quad -\tfrac{1}{2} \quad 0 \quad 0 \quad 0 \cdots) * v, \qquad (3.3.14)$$

$$D_4 v := \frac{1}{h^2}(\cdots 0 \quad 0 \quad -\tfrac{1}{12} \quad \tfrac{2}{3} \quad 0 \quad -\tfrac{2}{3} \quad \tfrac{1}{12} \quad 0 \quad 0 \cdots) * v, \qquad (3.3.15)$$

$$\vdots$$

$$D_\infty v := \frac{1}{h^2}(\cdots -\tfrac{1}{4} \quad \tfrac{1}{3} \quad -\tfrac{1}{2} \quad 1 \quad 0 \quad -1 \quad \tfrac{1}{2} \quad -\tfrac{1}{3} \quad \tfrac{1}{4} \cdots) * v \qquad (3.3.16)$$

(recall Exercise 2.2.1). In each of these formulas the sequence in parentheses indicates a grid function $w = \{w_j\}$, with the zero in the middle representing w_0. Since w has compact support, except in the case D_∞, there is no problem of convergence associated with the convolution.

Any convolution can also be thought of as multiplication by a **Toeplitz matrix**—that is, a matrix with constant entries along each diagonal ($a_{ij} = a_{i-j}$). For example, if v is interpreted as an infinite column vector $(\dots, v_{-1}, v_0, v_1, , \dots,)^T$, then $\delta_0 v$ becomes the left-multiplication of v by the *infinite* matrix of the form

$$\delta_0 := \frac{1}{h}\begin{pmatrix} 0 & \tfrac{1}{2} & & & \\ -\tfrac{1}{2} & 0 & \tfrac{1}{2} & & \\ & -\tfrac{1}{2} & 0 & \tfrac{1}{2} & \\ & & -\tfrac{1}{2} & 0 & \tfrac{1}{2} \\ & & & -\tfrac{1}{2} & 0 \end{pmatrix}. \qquad (3.3.17)$$

All together, there are at least five distinct ways to interpret the construction of spatial difference operators on a regular grid—all equivalent, but each having its own advantages:

1. *Approximation of differential operators.* To the classical numerical analyst, a spatial difference operator is a finite difference approximation to a differential operator.

2. *Interpolation.* To the data fitter, a spatial difference operator is an exact differential operator applied to an interpolatory approximant, as described above. This point of view is basic to spectral methods, which are based on global rather than local interpolants.

3. *Convolution.* To the signal processor, a difference operator is a convolution filter whose coefficients happen to be chosen so that it has the effect of differentiation.

4. *Toeplitz matrix multiplication.* To the linear algebraist, it is multiplication by a Toeplitz matrix. This point of view becomes central when

problems of implementation of implicit formulas come up, where the matrix defines a system of equations that must be solved.

5. *Fourier multiplier.* Finally, to the Fourier analyst, a spatial difference operator is the multiplication of the semidiscrete Fourier transform by a trigonometric function of ξ—which happens to approximate the polynomial corresponding to a differential operator.

* * *

Going back to the method of lines idea of the beginning of this section, if we view a finite difference model of a partial differential equation as a system of ordinary differential equations which is solved numerically, what can we say about the stability of this system? This viewpoint amounts to taking $h>0$ fixed but letting k vary. From the results of Chapter 1, we would expect meaningful answers in the limit $k\to 0$ so long as the discrete ODE formula is stable. On the other hand if k is fixed as well as h, the question of **absolute stability** comes up, as in §1.7. Provided that the infinite size of the system of ODEs can safely be ignored, we expect time-stability whenever the eigenvalues of the spatial difference operator lie in the stability region of the ODE method. In subsequent sections we shall determine these eigenvalues by Fourier analysis, and show that their location often leads to restrictions on k as a function of h.

EXERCISES

▷ *3.3.1. Nonuniform grids.* Consider an exponentially graded mesh on $(0,\infty)$ with $x_j = h\,s^j$, $s>1$. Apply (3.3.3) to derive a 3-point centered approximation on this grid to the first-order differentiation operator ∂_x.

▶ *3.3.2. Fornberg's algorithm.* Write a brief program (either numerical or, better, symbolic) to implement Fornberg's algorithm of Theorem 3.1. Run the program in such a way as to reproduce the coefficients of backwards differentiation formulas in Table 1.4.3 and equivalently Table 3.3.3. What are the coefficients for "one-sided half-way point" approximation of zeroth, first, and second derivatives in the points $-1/2$, $1/2$, $3/2$, $5/2$?

▷ *3.3.3. Lax-Wendroff formula.* Derive the Lax-Wendroff formula (3.2.6) via interpolation of v_{j-1}^n, v_j^n and v_{j+1}^n by a polynomial $q(x)$ followed by evaluation of $q(x)$ at an appropriate point.

Table 3.3.1. Coefficients for centered finite difference approximations (from Fornberg).

Table 3.3.2. Coefficients for centered finite difference approximations at a "half-way" point (from Fornberg).

Table 3.3.3. Coefficients for one-sided finite difference approximations (from Fornberg).

3.4. Implicit formulas and linear algebra

[This section is not yet written, but here is a sketch.]

Implicit finite difference formula lead to systems of equations to solve. If the PDE is linear this becomes a linear algebra problem $Ax = b$, while if it is nonlinear an iteration will possibly be required that involves a linear algebra problem at each step. Thus it is hard to overstate the importance of linear algebra in the numerical solution of partial differential equations.

For a finite difference grid involving N points in each of d space dimensions, A will have dimension $\Theta(N^d)$ and thus $\Theta(N^{2d})$ entries. Most of these are zero; the matrix is sparse. If there is just one space dimension, A will have a narrow band-width and $Ax = b$ can be solved in $\Theta(N)$ operations by Gaussian elimination or related algorithms. Just a few remarks about solutions of this kind.... First, if A is symmetric and positive definite, one normally preserves this form by using the Cholesky decomposition. Second, unless the matrix is positive definite or diagonally dominant, pivoting of the rows is usually essential to ensure stability.

When there are two or more space dimensions the band-width is larger and the number of operations goes up, so algorithms other than Gaussian elimination become important. Here are some typical operation counts (orders of magnitude) for the canonical problem of solving the standard five-point Laplacian finite-difference operator on a rectangular domain. For the iterative methods, ϵ denotes the accuracy of the solution; typically $\log\epsilon = \Theta(\log N)$, and we have assumed this in the last line of the table.

Algorithm	1D	2D	3D
Gaussian elimination	N^3	N^6	N^9
banded Gaussian elimination	N	N^4	N^7
Jacobi or Gauss-Seidel iteration	$N^3\log\epsilon$	$N^4\log\epsilon$	$N^5\log\epsilon$
SOR iteration	$N^2\log\epsilon$	$N^3\log\epsilon$	$N^4\log\epsilon$
conjugate gradient iteration	$N^2\log\epsilon$	$N^3\log\epsilon$	$N^4\log\epsilon$
preconditioned CG iteration	$N\log\epsilon$	$N^{2.5}\log\epsilon$	$N^{??}\log\epsilon$
nested dissection	N	N^3	$N^{??}\log\epsilon$
fast Poisson solver	$N\log N$	$N^2\log N$	$N^3\log N$
multigrid iteration	$N\log\epsilon$	$N^2\log\epsilon$	$N^3\log\epsilon$
"full" multigrid iteration	N	N^2	N^3

These algorithms vary greatly in how well they can be generalized to variable coefficients, different PDEs, and irregular grids. Fast Poisson solvers are the most narrowly applicable and multigrid methods, despite their remarkable speed, the most general. Quite a bit of programming effort may be involved in multigrid calculations, however.

Two observations may be made about the state of linear algebra in scientific computing nowadays. First, multigrid methods are extremely important and becoming ever more so. Second, preconditioned conjugate gradient (CG) methods are also extremely important, as well as other preconditioned iterations such as GMRES, BCG, and QMR for nonsymmetric matrices. These are often very easy to implement, once one finds a good preconditioner, and can be spectacularly fast. See Chapter 9.

3.5. Fourier analysis of finite difference formulas

In §3.3 we noted that a spatial difference operator D can be interpreted as a convolution: $Dv = a*v$ for some a with compact support. By Theorem 2.5, it follows that if $v \in \ell_h^2$, then $\widehat{Dv}(\xi) = \widehat{a*v}(\xi) = \hat{a}(\xi)\hat{v}(\xi)$. This fact is the basis of Fourier analysis of finite difference methods. In this section we shall work out the details for scalar one-step finite difference formulas ($s=1$ in (3.2.18)), treating first explicit formulas and then implicit ones. The next section will extend these developments to vector and multistep formulas.*

To begin in the simplest setting, consider an explicit, scalar, one-step finite difference formula

$$v_j^{n+1} := Sv_j^n := \sum_{\mu=-\ell}^{r} \alpha_\mu v_{j+\mu}^n, \tag{3.5.1}$$

where $\{\alpha_\mu\}$ are fixed constants. The symbol S denotes the operator that maps v^n to v^{n+1}. In this case of an explicit formula, S is defined for arbitrary sequences v, and by (2.2.3) we have

$$Sv := a*v, \qquad a_\mu := \frac{1}{h}(\alpha_{-\mu}). \tag{3.5.2}$$

To be able to apply Fourier analysis, however, let us assume $v \in \ell_h^2$, which implies $Sv \in \ell_h^2$ also since S is a finite sum. Then Theorem 2.5 gives

$$\widehat{Sv}(\xi) := \widehat{a*v}(\xi) := \hat{a}(\xi)\hat{v}(\xi). \tag{3.5.3}$$

EXAMPLE 3.5.1. *Upwind formula for $u_t = u_x$.* The upwind formula (Table 3.2.1) is defined by

$$v_j^{n+1} := Sv_j^n := v_j^n + \lambda(v_{j+1}^n - v_j^n), \tag{3.5.4}$$

where $\lambda = k/h$. By (2.2.3) or (3.5.2), this is equivalent to $Sv = a*v$ with

$$a_j := \begin{cases} \dfrac{1}{h}\lambda & \text{if } j=-1, \\ \dfrac{1}{h}(1-\lambda) & \text{if } j=0, \\ 0 & \text{otherwise.} \end{cases}$$

*A good reference on the material of this section is the classic monograph by R. D. Richtmyer and K. W. Morton, *Difference Methods for Initial-Value Problems,* 1967, Chapters 4 and 7.

By (2.2.6), the Fourier transform is

$$\hat{a}(\xi) := h(e^{-i\xi x_{-1}}a_{-1} + e^{-i\xi x_0}a_0) := \lambda e^{i\xi h} + (1-\lambda).$$

The interpretation of $\hat{a}(\xi)$ is simple: it is the **amplification factor** by which the component in v of wave number ξ is amplified when the finite difference operator S is applied. Often we shall denote this amplification factor simply by $g(\xi)$, a notation that will prove especially convenient for later generalizations.

To determine $g(\xi)$ for a particular finite difference formula, one can always apply the definition above: find the sequence a, then compute its Fourier transform. This process is unnecessarily complicated, however, because of a factor h that is divided out and then multiplied in again, and also a pair of factors -1 in the exponent and the subscripts that cancel. Instead, as a practical matter, it is simpler (and easier to remember) to insert the trial solution $v_j^n = g^n e^{i\xi jh}$ in the finite difference formula and see what expression for $g = g(\xi)$ results.

EXAMPLE 3.5.1, CONTINUED. To derive the amplification factor for the upwind formula more quickly, insert $v_j^n = g^n e^{i\xi jh}$ in (3.5.4) to get

$$g^{n+1}e^{i\xi jh} := g^n\left(e^{i\xi jh} + \lambda(e^{i\xi(j+1)h} - e^{i\xi jh})\right),$$

or after factoring out $g^n e^{i\xi jh}$,

$$g(\xi) := 1 + \lambda(e^{i\xi h} - 1). \tag{3.5.5}$$

EXAMPLE 3.5.2. *Lax-Wendroff formula for $u_t = u_x$.* The Lax-Wendroff formula (Table 3.2.1) is defined by

$$v_j^{n+1} := Sv_j^n := v_j^n + \tfrac{1}{2}\lambda(v_{j+1}^n - v_{j-1}^n) + \tfrac{1}{2}\lambda^2(v_{j+1}^n - 2v_j^n + v_{j-1}^n). \tag{3.5.6}$$

Inserting $v_j^n = g^n e^{i\xi jh}$ and dividing by the common factor $g^n e^{i\xi jh}$ gives

$$g(\xi) := 1 + \tfrac{1}{2}\lambda(e^{i\xi h} - e^{-i\xi h}) + \tfrac{1}{2}\lambda^2(e^{i\xi h} - 2 + e^{-i\xi h}).$$

The two expressions in parentheses come up so often in Fourier analysis of finite difference formulas that it is worth recording what they are equivalent to:

$$e^{i\xi h} - e^{-i\xi h} := 2i\sin\xi h, \tag{3.5.7}$$

$$e^{i\xi h} - 2 + e^{-i\xi h} = 2\cos\xi h - 2 := -4\sin^2\frac{\xi h}{2}. \tag{3.5.8}$$

The amplification factor function for the Lax-Wendroff formula is therefore

$$g(\xi) := 1 + i\lambda\sin\xi h - 2\lambda^2\sin^2\frac{\xi h}{2}. \tag{3.5.9}$$

EXAMPLE 3.5.3. *Euler formula for* $u_t = u_{xx}$. As an example involving the heat equation, consider the Euler formula of Table 3.2.2,

$$v_j^{n+1} := Sv_j^n := v_j^n + \sigma(v_{j+1}^n - 2v_j^n + v_{j-1}^n), \tag{3.5.10}$$

where $\sigma = k/h^2$. By (3.5.8), insertion of $v_j^n = g^n e^{i\xi jh}$ gives

$$g(\xi) := 1 - 4\sigma \sin^2 \frac{\xi h}{2}. \tag{3.5.11}$$

Now let us return to generalities. Since $g(\xi) = \hat{a}(\xi)$ is bounded as a function of ξ, (3.5.3) implies the bound $\|\widehat{Sv}\| = \|\hat{a}\hat{v}\| \le \|\hat{a}\|_\infty \|\hat{v}\|$ (by (2.2.4)), hence $\|Sv\| \le \|\hat{a}\|_\infty \|v\|$ (by (2.2.8)), where $\|\hat{a}\|_\infty$ denotes the "sup-norm"

$$\|\hat{a}\|_\infty := \max_{\xi \in [-\pi/h, \pi/h]} |\hat{a}(\xi)|. \tag{3.5.12}$$

Thus S is a **bounded linear operator** from ℓ_h^2 to ℓ_h^2. Moreover, since $\hat{v}(\xi)$ could be chosen to be a function arbitrarily narrowly peaked near a wave number ξ_0 with $|\hat{a}(\xi_0)| = \|\hat{a}\|_\infty$, this inequality cannot be improved. Therefore

$$\|S\| := \|\hat{a}\|_\infty. \tag{3.5.13}$$

The symbol $\|S\|$ denotes the **operator ℓ_h^2-2-norm** of S, that is, the norm on the operator $S : \ell_h^2 \to \ell_h^2$ induced by the usual norm (2.2.1) on ℓ_h^2 (see Appendix B):

$$\|S\| := \sup_{v \in \ell_h^2} \frac{\|Sv\|}{\|v\|}. \tag{3.5.14}$$

Repeated applications of the finite difference formula are defined by $v^n = S^n v^0$, and if $v_0 \in \ell_h^2$, then $\widehat{v^n}(\xi) = (\hat{a}(\xi))^n \widehat{v^0}(\xi)$. Since $\hat{a}(\xi)$ is just a scalar function of ξ, we have

$$\|(\hat{a}(\xi))^n\|_\infty := \max_\xi (|\hat{a}(\xi)|^n) := (\max_\xi |\hat{a}(\xi)|)^n := (\|\hat{a}\|_\infty)^n,$$

and therefore by the same argument as above,

$$\|v^n\| \le (\|\hat{a}\|_\infty)^n \|v^0\|$$

and

$$\|S^n\| := (\|\hat{a}\|_\infty)^n. \tag{3.5.15}$$

A comparison of (3.5.13) and (3.5.15) reveals that if S is the finite difference operator for an explicit scalar one-step finite difference formula, then $\|S^n\| = \|S\|^n$. In general, however, bounded linear operators satisfy only the inequality $\|S^n\| \le \|S\|^n$, and this will be the best we can do when we turn to multistep finite difference formulas or to systems of equations in the next section.

The results above can be summarized in the following theorem.

FOURIER ANALYSIS OF
EXPLICIT SCALAR ONE-STEP FINITE DIFFERENCE FORMULAS

Theorem 3.3. *The scalar finite difference formula (3.5.1) defines a bounded linear operator* $S: \ell_h^2 \to \ell_h^2$, *with*

$$\|S^n\| := \|\hat{a}^n\|_\infty := (\|\hat{a}\|_\infty)^n \quad \text{for } n \ge 0. \tag{3.5.16}$$

If $v^0 \in \ell_h^2$ *and* $v^n = S^n v^0$, *then*

$$\widehat{v^n}(\xi) := (\hat{a}(\xi))^n \widehat{v^0}(\xi), \tag{3.5.17}$$

$$v_j^n := \frac{1}{2\pi} \int_{-\pi/h}^{\pi/h} e^{i\xi x_j} (\hat{a}(\xi))^n \widehat{v^0}(\xi) d\xi, \tag{3.5.18}$$

and

$$\|v^n\| \le (\|\hat{a}\|_\infty)^n \|v^0\|. \tag{3.5.19}$$

Proof. The discussion above together with Theorem 2.5. ∎

Now let us generalize this discussion by considering an arbitrary one-step scalar finite difference formula, which may be explicit or implicit. This is the special case of (3.2.18) with $s = 1$, defined as follows:

A **one-step linear finite difference formula** *is a formula*

$$\sum_{\mu=-\ell}^{r} \beta_\mu v_{j+\mu}^{n+1} := \sum_{\mu=-\ell}^{r} \alpha_\mu v_{j+\mu}^n \tag{3.5.20}$$

for some constants $\{\alpha_\mu\}$ *and* $\{\beta_\mu\}$ *with* $\beta_0 \ne 0$. *If* $\beta_\mu = 0$ *for* $\mu \ne 0$ *the formula is* **explicit**, *while if* $\beta_\mu \ne 0$ *for some* $\mu \ne 0$ *it is* **implicit**.

Equation (3.5.1) is the special case of (3.5.20) with $\beta_0 = 1$, $\beta_\mu = 0$ for $\mu \ne 0$. Again we wish to use (3.5.20) to define an operator $S: v^n \mapsto v^{n+1}$, but now we have to be more careful. Given any sequence v^n, (3.5.20) amounts to an infinite system of linear equations for the unknown sequence v^{n+1}. In the terms of the last section, we must solve

$$Bv^{n+1} := Av^n \tag{3.5.21}$$

for v^{n+1}, where A and B are infinite Toeplitz matrices. But before the operator S will be well-defined, we must make precise what it means to do this.

The first observation is easy. Given any sequence v^n, the right-hand side of (3.5.20) is unambiguously defined since only finitely many numbers α_μ are nonzero. Likewise, given any sequence v^{n+1}, the left-hand side is unambiguously defined. Thus for any pair v^n, v^{n+1}, there is no ambiguity about whether or not they satisfy (3.5.20): it is just a matter of whether the two sides are equal for all j. We can write (3.5.20) equivalently as

$$b*v^{n+1} := a*v^n \tag{3.5.22}$$

for sequences $a_\mu = \dfrac{1}{h}\alpha_{-\mu}$, $b_\mu = \dfrac{1}{h}\beta_{-\mu}$; there is no ambiguity about what it means for a pair v^n, v^{n+1} to satisfy (3.5.22).

The difficulty comes when we ask: given a sequence v^n, does there exist a unique sequence v^{n+1} satisfying (3.5.22)? In general the answer is *no*, as is shown by the following example.

EXAMPLE 3.5.4. *Crank-Nicolson for $u_t = u_{xx}$.* The Crank-Nicolson formula (3.2.4) is

$$v_j^{n+1} - v_j^n := \tfrac{1}{2}\sigma(v_{j+1}^n - 2v_j^n + v_{j-1}^n) + \tfrac{1}{2}\sigma(v_{j+1}^{n+1} - 2v_j^{n+1} + v_{j-1}^{n+1}), \tag{3.5.23}$$

where $\sigma = k/h^2$. Suppose v^n is identically zero. Then the formula reduces to

$$v_{j+1}^{n+1} - 2(1+\frac{1}{\sigma})v_j^{n+1} + v_{j-1}^{n+1} := 0. \tag{3.5.24}$$

One solution of this equation is $v_j^{n+1} = 0$ for all j, and this is the "right" solution as far as applications are concerned. But (3.5.24) is a second-order recurrence relation with respect to j, and therefore it has two linearly independent nonzero solutions too, namely $v_j^{n+1} = \kappa^j$, where κ is either root of the characteristic equation

$$\kappa^2 - 2(1+\frac{1}{\sigma})\kappa + 1 := 0. \tag{3.5.25}$$

Thus solutions to implicit finite difference formulas on an infinite grid may be nonunique. In general, if the nonzero coefficients at level $n+1$ extend over a range of $J+1$ grid points, there will be a J-dimensional space of possible solutions at each step. In a practical computation, the grid will be truncated by boundaries, and the nonuniqueness will usually disappear. However, from a conceptual point of view there is a more basic observation to be made, which has relevance even to finite grids: the finite difference formula has a unique solution *in the space* ℓ_h^2.

To make this precise we need the following assumption, which is satisfied for the finite difference formulas used in practice. Let $\hat{b}(\xi)$ denote, as usual, the Fourier transform of the sequence b.

Solvability Assumption *for implicit scalar one-step finite difference formulas:*

$$\hat{b}(\xi) \neq 0 \quad \text{for } \xi \in [-\pi/h, \pi/h]. \tag{3.5.26}$$

Since $\hat{b}(\xi)$ is $2\pi/h$-periodic, this is equivalent to the assumption that $\hat{b}(\xi) \neq 0$ for all $\xi \in \mathbb{R}$. It is also equivalent to the statement that no root κ of the characteristic equation analogous to (3.5.25) lies on the unit circle in the complex plane.

Now suppose v^n and v^{n+1} are two sequences in ℓ_h^2 that satisfy (3.5.22). Then Theorem 2.5 implies

$$\hat{b}(\xi)\widehat{v^{n+1}}(\xi) := \hat{a}(\xi)\widehat{v^n}(\xi), \tag{3.5.27}$$

or by the solvability assumption,

$$\widehat{v^{n+1}}(\xi) := g(\xi)\widehat{v^n}(\xi), \tag{3.5.28}$$

where

$$g(\xi) := \frac{\hat{a}(\xi)}{\hat{b}(\xi)}. \tag{3.5.29}$$

Since $g(\xi)$ is a continuous function on the compact set $[-\pi/h, \pi/h]$, it has a finite maximum

$$\|g\|_\infty := \max_{\xi \in [-\pi/h, \pi/h]} \left| \frac{\hat{a}(\xi)}{\hat{b}(\xi)} \right| < \infty. \tag{3.5.30}$$

Now a function in ℓ_h^2 is uniquely determined by its Fourier transform. Therefore (3.5.28) implies that for any $v^n \in \ell_h^2$, there is at most one solution $v^{n+1} \in \ell_h^2$ to (3.5.22). On the other hand obviously such a solution exists, since (3.5.28) tells how to construct it.

We have proved that if $\hat{b}$ satisfies the solvability assumption (3.5.26), then for any $v^n \in \ell_h^2$, there exists a unique $v^{n+1} \in \ell_h^2$ satisfying (3.5.22). In other words, (3.5.22) defines a bounded linear operator $S: v^n \mapsto v^{n+1}$.

This and related conclusions are summarized in the following generalization of Theorem 3.3:

FOURIER ANALYSIS OF
IMPLICIT SCALAR ONE-STEP FINITE DIFFERENCE FORMULAS

Theorem 3.4. *If the solvability assumption (3.5.26) holds, then the implicit finite difference formula (3.5.20) defines a bounded linear operator* $S:\ell_h^2 \to \ell_h^2$, *with*

$$\|S^n\| := \|g^n\|_\infty := (\|g\|_\infty)^n \quad \text{for } n \ge 0, \tag{3.5.31}$$

where $g(\xi) = \hat{a}(\xi)/\hat{b}(\xi)$. *If* $v^0 \in \ell_h^2$ *and* $v^n = S^n v^0$, *then*

$$\widehat{v^n}(\xi) := (g(\xi))^n \widehat{v^0}(\xi), \tag{3.5.32}$$

$$v_j^n := \frac{1}{2\pi}\int_{-\pi/h}^{\pi/h} e^{i\xi x_j}(g(\xi))^n \hat{v}(\xi)d\xi, \tag{3.5.33}$$

and

$$\|v^n\| \le (\|g\|_\infty)^n \|v^0\|. \tag{3.5.34}$$

In principle, the operator S might be implemented by computing a semidiscrete Fourier transform, multiplying by $\hat{a}(\xi)/\hat{b}(\xi)$, and computing the inverse transform. In practice, an implicit formula will be applied on a finite grid and its implementation will usually be based on solving a finite linear system of equations. But as will be seen in later sections, sometimes the best methods for solving this system are again based on Fourier analysis.

EXAMPLE 3.5.4, CONTINUED. As with explicit formulas, the easiest way to calculate amplification factors of implicit formulas is by insertion of the trial solution $v_j^n = g^n e^{i\xi jh}$. For the implicit Crank-Nicolson model of (3.5.23), by (3.5.8), this leads to

$$g(\xi) := 1 - 2\sigma \sin^2\frac{\xi h}{2} - 2\sigma g(\xi)\sin^2\frac{\xi h}{2},$$

that is,

$$g(\xi) = \frac{\hat{a}(\xi)}{\hat{b}(\xi)} = \frac{1-2\sigma\sin^2\frac{\xi h}{2}}{1+2\sigma\sin^2\frac{\xi h}{2}}, \tag{3.5.35}$$

where again $\sigma = k/h^2$. Since the denominator $\hat{b}(\xi)$ is positive for all ξ, the Crank-Nicolson formula satisfies the solvability assumption (3.5.26), regardless of the value of σ.

EXERCISES

▷ *3.5.1. Amplification factors.* Calculate the amplification factors for the *(a)* Euler, *(b)* Crank-Nicolson, *(c)* Box, and *(d)* Lax-Friedrichs models of $u_t = u_x$. For the implicit formulas, verify that the solvability assumption is satisfied.

3.6. Fourier analysis of vector and multistep formulas

It is not hard to extend the developments of the last section to vector or multistep finite difference formulas. Both extensions are essentially the same, for we shall reduce a multistep scalar formula to a one-step vector formula, in analogy to the reduction of higher-order ordinary differential equations in §1.1 and of linear multistep formulas in §1.5.

It is easiest to begin with an example and then describe the general situation.

EXAMPLE 3.6.1. *Leap frog for* $u_t = u_x$. The leap frog formula (3.2.2) is

$$v_j^{n+1} := v_j^{n-1} + \lambda(v_{j+1}^n - v_{j-1}^n). \tag{3.6.1}$$

Let $w^n = \{w_j^n\}$ be the vector-valued grid function defined by

$$w_j^n := \begin{pmatrix} v_j^n \\ v_j^{n-1} \end{pmatrix}. \tag{3.6.2}$$

Then the leap frog formula can be rewritten as

$$\begin{pmatrix} v_j^{n+1} \\ v_j^n \end{pmatrix} = \begin{pmatrix} -\lambda & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} v_{j-1}^n \\ v_{j-1}^{n-1} \end{pmatrix} + \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} v_j^n \\ v_j^{n-1} \end{pmatrix} + \begin{pmatrix} \lambda & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} v_{j+1}^n \\ v_{j+1}^{n-1} \end{pmatrix},$$

that is,

$$w_j^{n+1} := \alpha_{-1} w_{j-1}^n + \alpha_0 w_j^n + \alpha_1 w_{j+1}^n,$$

where

$$\alpha_{-1} := \begin{pmatrix} -\lambda & 0 \\ 0 & 0 \end{pmatrix}, \qquad \alpha_0 := \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \qquad \alpha_1 := \begin{pmatrix} \lambda & 0 \\ 0 & 0 \end{pmatrix}. \tag{3.6.3}$$

Equivalently,

$$w^{n+1} := a * w^n,$$

where a is the infinite sequence of 2×2 matrices with $a_\mu = h^{-1}\alpha_{-\mu}$ (§2.5). For $w^n \in (\ell_h^2)^N$ (the set of N-vector sequences with each component sequence in ℓ_h^2), we then have

$$\widehat{w^{n+1}}(\xi) := \hat{a}(\xi)\widehat{w^n}(\xi).$$

As described in §2.6, all of these transforms are defined componentwise. From (3.6.3) we get

$$\hat{a}(\xi) := \begin{pmatrix} 2i\lambda\sin\xi h & 1 \\ 1 & 0 \end{pmatrix}. \tag{3.6.4}$$

This is the **amplification matrix** $\hat{a}(\xi)$ or $G(\xi)$ for leap frog, and values at later time steps are given by

$$\underline{\widehat{w^n} := [\hat{a}(\xi)]^n \widehat{w_0}.}$$

In general, an arbitrary explicit or implicit, scalar or vector multistep formula (3.2.15) can be reduced to the one-step form (3.5.20), where each v_j is an N-vector and each β_μ or α_μ is an $N\times N$ matrix, for a suitable value N. The same formula can also be written in the form (3.5.21), if B and A are infinite Toeplitz matrices whose elements are $N\times N$ matrices (i.e., A and B are tensors), or in the form (3.5.22), if a and b are sequences of $N\times N$ matrices. The condition (3.5.26) becomes

Solvability Assumption *for implicit vector one-step finite difference formulas:*

$$\det \hat{b}(\xi) \neq 0 \qquad \text{for } \xi \in [-\pi/h, \pi/h]. \tag{3.6.5}$$

The **amplification matrix** for the finite difference formula is

$$G(\xi) := [\hat{b}(\xi)]^{-1}[\hat{a}(\xi)], \tag{3.6.6}$$

and as before, the finite difference formula defines a bounded linear operator on sequences of N-vectors in $(\ell_h^2)^N$:

FOURIER ANALYSIS OF
IMPLICIT VECTOR ONE-STEP FINITE DIFFERENCE FORMULAS

Theorem 3.5. *If the solvability assumption (3.6.5) holds, the implicit vector finite difference formula (3.5.20) defines a bounded linear operator* $S:(\ell_h^2)^N \to (\ell_h^2)^N$, *with*

$$\|S^n\| := \|G^n\|_\infty \le (\|G\|_\infty)^n \quad \text{for } n \ge 0, \tag{3.6.6}$$

where $G(\xi) = [\hat{b}(\xi)]^{-1}\hat{a}(\xi)$. *If* $v^0 \in (\ell_h^2)^N$ *and* $v^n = S^n v^0$, *then*

$$\widehat{v^n}(\xi) := (G(\xi))^n \widehat{v^0}(\xi), \tag{3.6.7}$$

$$v_j^n := \frac{1}{2\pi}\int_{-\pi/h}^{\pi/h} e^{i\xi x_j}(G(\xi))^n \hat{v}(\xi)\,d\xi, \tag{3.6.8}$$

and

$$\|v^n\| \le (\|G\|_\infty)^n \|v^0\|. \tag{3.6.9}$$

In (3.6.6) and (3.6.9), $\|G\|_\infty$ is the natural extension of (3.5.4) to the matrix-valued case: it denotes the maximum

$$\|G\|_\infty := \max_{\xi \in [-\pi/h, \pi/h]} \|G(\xi)\|,$$

where the norm on the right is the matrix 2-norm (largest singular value) of an $N \times N$ matrix. The formula $\|S^n\| = \|S\|^n$ is no longer valid in general for vector finite difference formulas.

EXERCISES

▷ *3.6.1. Amplification matrices.* Calculate the amplification matrices for the *(a)* DuFort-Frankel and *(b)* fourth-order leap frog formulas of §3.2. Be sure to describe precisely what one-step vector finite difference formula your amplification matrix is based upon.