FINITE DIFFERENCE AND SPECTRAL METHODS FOR ORDINARY AND PARTIAL DIFFERENTIAL EQUATIONS

Lloyd N. Trefethen
Cornell University

Note

A trap that academics sometimes fall into is to begin a book and fail to finish it. This has happened to me. This book was intended to be published several years ago. Alas, two other projects jumped the queue (*Numerical Linear Algebra,* with David Bau, to be published in 1997 by SIAM, and *Spectra and Pseudospectra,* to be completed thereafter). At this point I do not know whether this numerical ODE/PDE book will ever be properly finished.

This is a shame—as people keep telling me. Electronically by email, and in person at conferences, I am often asked "when that PDE book is going to get done." Anyone who has been in such a situation knows how flattering such questions feel... at first.

The book is based on graduate courses taught to mathematicians and engineers since 1982 at the Courant Institute, MIT, and Cornell. Successively more complete versions have been used by me in these courses half a dozen times, and by a handful of colleagues at these universities and elsewhere. Much of the book is quite polished, especially the first half, and there are numerous exercises that have been tested in the classroom. But there are many gaps too, and undoubtedly a few errors. I have reproduced here the version of the text I used most recently in class, in the spring of 1994.

I am proud of Chapter 1, which I consider as clean an introduction as any to the classical theory of the numerical solution of ODE. The other main parts of the book are Chapter 2, which emphasizes the crucial relationship between smoothness of a function and decay of its Fourier transform; Chapters 3 and 4, which present the classical theory of numerical stability for finite difference approximations to linear PDE; and Chapters 7 and 8, which offer a hands-on introduction to Fourier and Chebyshev spectral methods.

This book is largely linear, and for that reason and others, there is much more to the numerical solution of differential equations than you will find here. On the other hand, Chapters 1–4 represent material that every numerical analyst should know. These chapters alone can be the basis of an appealing course at the graduate level.

 Copies can be obtained for \$22 or £15 each (including shipping) by contacting me directly. Professors who are using the book in class may contact me to obtain solutions to most of the problems. [This is no longer true – the book is freely available online at
`http://web.comlab.ox.ac.uk/oucl/work/nick.trefethen/pdetext.html`.]

Nick Trefethen
July 1996

Department of Computer Science and Center for Applied Mathematics
Upson Hall, Cornell University, Ithaca, NY 14853, USA
(607) 255-4222, fax (607) 255-4428, LNT@cs.cornell.edu

Contents

Preface

Many books have been written on numerical methods for partial differential equations, ranging from the mathematical classic by Richtmyer and Morton to recent texts on computational fluid dynamics. But this is not an easy field to set forth in a book. It is too active and too large, touching a bewildering variety of issues in mathematics, physics, engineering, and computer science.

My goal has been to write an advanced textbook focused on basic principles. Two remarkably fruitful ideas pervade this field: numerical stability and Fourier analysis. I want the reader to think at length about these and other fundamentals, beginning with simple examples, and build in the process a solid foundation for subsequent work on more realistic problems. Numerical methods for partial differential equations are a subject of subtlety and beauty, but the appreciation of them may be lost if one views them too narrowly as tools to be hurried to application.

This is not a book of pure mathematics. A number of theorems are proved, but my purpose is to present a set of ideas rather than a sequence of theorems. The book should be accessible to mathematically inclined graduate students and practitioners in various fields of science and engineering, so long as they are comfortable with Fourier analysis, basic partial differential equations, some numerical methods, the elements of complex variables, and linear algebra including vector and matrix norms. At MIT and Cornell, successive drafts of this text have formed the basis of a large graduate course taught annually since 1985.

One unusual feature of the book is that I have attempted to discuss the numerical solution of ordinary and partial differential equations in parallel. There are numerous ideas in common here, well known to professionals, but usually not stressed in textbooks. In particular I have made extensive use of the elegant idea of *stability regions* in the complex plane.

Another unusual feature is that in a single volume, this book treats spectral as well as the more traditional finite difference methods for discretization of partial differential equations. The solution of discrete equations by multigrid iterations, the other most conspicuous development in this field in the past twenty years, is also introduced more briefly.

The unifying theme of this book is Fourier analysis. To be sure, it is well

known that Fourier analysis is connected with numerical methods, but here we shall consider at least four of these connections in detail, and point out analogies between them: stability analysis, spectral methods, iterative solutions of elliptic equations, and multigrid methods. Indeed, a reasonably accurate title would have been "Fourier Analysis and Numerical Methods for Partial Differential Equations." Partly because of this emphasis, the ideas covered here are primarily linear; this is by no means a textbook on computational fluid dynamics. But it should prepare the reader to read such books at high speed.

Besides these larger themes, a number of smaller topics appear here that are not so often found outside research papers, including:

- *Order stars* for stability and accuracy analysis,
- *Modified equations* for estimating discretization errors,
- Stability in ℓ^p *norms,*
- *Absorbing boundary conditions* for wave calculations,
- Stability analysis of non-normal discretizations via *pseudospectra,*
- *Group velocity* effects and their connection with the stability of boundary conditions.

Like every author, I have inevitably given greatest emphasis to those topics I know through my own work.

The reader will come to know my biases soon enough; I hope he or she shares them. One is that exercises are essential in any textbook, even if the treatment is advanced. (The exercises here range from cookbook to philosophical; those marked with solid triangles require computer programming.) Another is that almost every idea can and should be illustrated graphically. A third is that numerical methods should be studied at all levels, from the most formal to the most practical. This book emphasizes the formal aspects, saying relatively little about practical details, but the reader should forget at his or her peril that a vast amount of practical experience in numerical computation has accumulated over the years. This experience is magnificently represented in a wealth of high-quality mathematical software available around the world, such as EISPACK, LINPACK, LAPACK, ODEPACK, and the IMSL and NAG Boeing Fortran libraries, not to mention such interactive systems as Matlab and Mathematica.* Half of every applied problem can and should be handled by off-the-shelf programs nowadays. This book will tell you something about what those programs are doing, and help you to think productively about the other half.

*For on-line acquisition of numerical software, every reader of this book should be sure to be familiar with the "Netlib" facility maintained at Bell Laboratories and Oak Ridge National Laboratory. For information, send the e-mail message "help" to netlib@research.att.com or netlib@ornl.gov.

Notation

$\mathbb{R}, \mathbb{C}, \mathbb{Z}$	real numbers, complex numbers, integers
$\star$	marks a number contaminated by rounding errors
O, o, Ω, Θ	order relations analogous to $\leq$, $<$, $\geq$, $=$
x, y, t	space and time variables
ξ, η, ω	x wave number, y wave number, frequency
T	initial-value problem is posed for $t \in [0, T]$
N	dimension of system of equations
d	number of space dimensions
u	dependent variable (scalar or N-vector)
h, k	space and time step sizes $h = \Delta x$, $k = \Delta t$
i, j, n	space and time indices
ℓ, r, s	integers defining size of finite difference stencil
v_{ij}^n	numerical approximation to $u(ih, jh, nk)$
$\hat{u}, \hat{v}$	Fourier transforms
λ, σ	mesh ratios $\lambda = k/h$, $\sigma = k^2/h$
K, Z	space and time shift operators K: $v_j \mapsto v_{j+1}$, Z: $v^n \mapsto v^{n+1}$
κ, z	space and time amplification factors $\kappa = e^{i\xi h}$, $z = e^{i\omega k}$
Δ, ∇	forward and backward difference operators $\Delta = K - 1$, $\nabla = 1 - K^{-1}$
$\rho(z), \sigma(z)$	characteristic polynomials for a linear multistep formula
c, c_g	phase and group velocities $c = -\omega/\xi$, $c_g = -d\omega/d\xi$
$\Lambda(A), \Lambda_\epsilon(A)$	spectrum and ϵ-pseudospectrum of A

The following little sections α and β contain essential background material for the remainder of the text. They are not chapters, just tidbits. Think of them as appetizers.

α. Big-O and related symbols

How fast is an algorithm? How accurate are the results it produces? Answering questions like these requires estimating operation counts and errors, and we need a convenient notation for doing so. This book will use the six symbols o, O, Ω, Θ, $\sim$, and $\approx$. Four of these are standard throughout the mathematical sciences, while Ω and Θ have recently become standard at least in theoretical computer science.*

Here are the first four definitions. They are written for functions $f(x)$ and $g(x)$ as $x \to \infty$, but analogous definitions apply to other limits such as $x \to 0$.

O $f(x) = O(g(x))$ as $x \to \infty$ if there exist constants $C > 0$ and $x_0 > 0$ such that $|f(x)| \le Cg(x)$ for all $x \ge x_0$. If $g(x) \ne 0$ this means $|f(x)|/g(x)$ is bounded from above for all sufficiently large x. In words: "f is O of g."

o $f(x) = o(g(x))$ as $x \to \infty$ if for *any* constant $C > 0$ there exists a constant $x_0 > 0$ such that $|f(x)| \le Cg(x)$ for all $x \ge x_0$. If $g(x) \ne 0$ this means $|f(x)|/g(x)$ approaches 0 as $x \to \infty$. In words: "f is little-O of g."

Ω $f(x) = \Omega(g(x))$ as $x \to \infty$ if there exist constants $C > 0$ and $x_0 > 0$ such that $f(x) \ge Cg(x)$ for all $x \ge x_0$. If $g(x) \ne 0$ this means $f(x)/g(x)$ is bounded from below for all sufficiently large x. In words: "f is omega of g."

Θ $f(x) = \Theta(g(x))$ as $x \to \infty$ if there exist constants $C, C' > 0$ and $x_0 > 0$ such that $Cg(x) \le f(x) \le C'g(x)$ for all $x \ge x_0$. If $g(x) \ne 0$ this means $f(x)/g(x)$ is bounded from above and below. In words: "f is theta of g."

The use of the symbols O, o, Ω, and Θ is a bit illogical, for properly speaking, $O(g(x))$ for example is the *set* of all functions $f(x)$ with a certain property; but why then don't we write $f(x) \in O(g(x))$? The answer is simply that the traditional usage, illogical or not, is well established and very convenient.

The final two definitions are not so irregular:

*Thanks to the delightful note "Big omicron and big omega and big theta" by D. E. Knuth, *ACM SIGACT News 8*, 1976.

$\sim$ $f(x) \sim g(x)$ as $x \to \infty$ if $f(x) - g(x) = o(|g(x)|)$. If $g(x) \neq 0$ this is equivalent to $\lim_{x\to\infty} f(x)/g(x) = 1$. In words: "$f$ is asymptotic to g."

$\approx$ This symbol is easy: it has no precise definition! The statement $f(x) \approx g(x)$ is qualitative only, and the argument x plays no role. Often in this book, but not always, we shall use $\approx$ to indicate that a real number has been rounded or chopped to a finite number of digits, e.g., $\pi \approx 3$ or $\pi \approx 3.14159265358979323846$.

EXAMPLE α.1. True statements as $x \to \infty$: $\sin x = O(1)$, $\log x = o(x^{1/100})$, $e^x = \Omega(x^{100})$, $2 + \sin^2 x = \Theta(1)$, $\cosh x = e^x + o(1) \sim e^x$. False statements as $x \to \infty$: $x = o(100x)$, $\sin x = \Omega(1)$, $e^{-x} \sim 0$.

The relationship between these definitions can be summarized by the Venn diagram of Figure α.1. Let X denote the set of all pairs of functions $\{f, g\}$. The diagram illustrates, for example, that the subset of X consisting of pairs $\{f, g\}$ with $f(x) = o(g(x))$ is contained in the subset of pairs with $f(x) = O(g(x))$. In other words, $f(x) = o(g(x))$ implies $f(x) = O(g(x))$.

Figure α.1. Venn diagram showing the relationship between o, O, Ω, and Θ.

EXERCISES

▷ *α.1. True or False?*

(a) $x^2 \sin(1/x) \sim x^2$ as $x \to 0$.

(b) $x^2 \sin(1/x) \sim x$ as $x \to \infty$.

(c) $(\sin x)^2 \sim \sin(x^2)$ as $x \to 0$.

(d) $(\log x)^2 = o(\log(x^2))$ as $x \to \infty$.

(e) $n! = \Theta(n/e)^n$ as $n \to \infty$.

▷ *α.2.* Suppose $g(x) \sim 1$ as $x \to \infty$. Find interesting examples of functions $f \neq g$ occupying all five distinct positions in the Venn diagram of Figure α.1.

▷ *α.3. True or False?*

(a) $A = \Theta(V^{2/3})$ as $V \to \infty$, where A and V are the surface area and volume of a sphere measured in square miles and cubic microns, respectively.

(b) $\lim_{n\to\infty} a_n = b \iff a_n = b + o(1)$ as $n \to \infty$.

(c) A set of N words can be sorted into alphabetical order by an algorithm that makes $O(N \log N)$ pairwise comparisons.

(d) Solving an $N \times N$ matrix problem $Ax = b$ by Gaussian elimination on a serial computer takes $\Omega(N^3)$ operations.

β. Rounding and truncation errors

Throughout this book we shall be concerned with computed, inexact approximations to ideal quantities. Two things keep them from being exact: rounding errors and truncation errors.

Rounding errors are the errors introduced by computers in simulating real or complex arithmetic by floating-point arithmetic.* For most purposes they can be treated as ubiquitous and random, and it is possible to be quite precise about how they behave. Given a particular machine with a particular system of floating-point arithmetic, let **machine epsilon** be defined as the smallest positive floating-point number ϵ such that $1 \oplus \epsilon > 1$. Here $\oplus$ denotes floating-point addition of floating-point numbers; analogous notations apply for $-$, $\times$, and $\div$. Then on well-designed computers, if a and b are arbitrary floating-point numbers, the following identities hold:

$$\left.\begin{aligned} a \oplus b &= (a+b)(1+\delta) \\ a \ominus b &= (a-b)(1+\delta) \\ a \otimes b &= (a \times b)(1+\delta) \\ a \oslash b &= (a \div b)(1+\delta) \end{aligned}\right\} \text{ for some } \delta \text{ with } |\delta| < \epsilon; \qquad (\beta.1)$$

the quantity δ is of course different from one appearance to the next. In other words, each floating-point operation introduces a *relative error* of magnitude less than ϵ.†

Truncation errors, or **discretization errors**, are the errors introduced by algorithms in replacing an infinite or infinitesimal quantity by something finite; they have nothing to do with floating-point arithmetic. For example, the error introduced in truncating an infinite series at some finite term N is a truncation error, and so is the error introduced in replacing a derivative du/dt by the finite difference $[u(t+\Delta t) - u(t-\Delta t)]/2\Delta t$. Truncation errors do not appear in all areas of numerical computation; they are absent, for example, in solving a linear system of equations by Gaussian elimination. They are usually unavoidable, however, in the numerical solution of differential equations. Unlike rounding errors, truncation errors are generally not at all random, but may have a great deal of regularity.

Rounding errors would vanish if computers were infinitely accurate. Truncation errors would vanish if computers were infinitely fast and had infinitely large memories, so that there were no need to settle for finite approximations. Further discussion of this distinction can

---

*__Floating-point numbers__ are real numbers of the form $x = \pm f \times \beta^e$, where β is the **base** (usually 2, 10, or 16), $f < 1$ is the **fraction** represented to t digits in base β, and e is an adjustable **exponent**. **Floating-point arithmetic** refers to the use of floating-point numbers together with approximate arithmetic operations defined upon them. Early computers sometimes used "fixed-point arithmetic" instead, but this is unheard-of nowadays except in special-purpose machines. For an introduction to floating-point arithmetic see G. B. Forsythe, M. A. Malcolm & C. B. Moler, *Computer Methods for Mathematical Computations*, Prentice-Hall, 1977, and for a table of the floating-point characteristics of existing computers as of 1988, see W. J. Cody, *ACM Trans. Math. Softw. 14* (1988), 303–311.

†This description ignores the possibility of underflow or overflow.

be found in many books on numerical analysis, such as P. Henrici, *Essentials of Numerical Analysis,* Wiley, 19??.

In this book we shall frequently ask how a particular computation may be affected by rounding or truncation errors. Which source of error dominates? Most often, rounding errors are negligible and the accuracy is determined by truncation errors. When an instability is present, however, the situation is sometimes reversed. We shall pay careful attention to such matters, for there is no other way to understand mathematical computations fully.

Truncation errors depend only on the algorithm, but rounding errors depend also on the machine. We shall use the following notation throughout the book: a number preceded by a star * is significantly contaminated by rounding errors and hence machine-dependent.

EXAMPLE β.1. For example, suppose we approximate e^x by the 101-term series

$$f(x) = 1 + x + \frac{x^2}{2} + \cdots + \frac{x^{100}}{100!},$$

evaluated on the computer from left to right as written. On a Sun Workstation in double precision with $\epsilon = 2^{-52} \approx 2.22 \times 10^{-16}$, we obtain

$$\begin{aligned} e^{20} &\approx 4.8517 \times 10^8, & f(20) &\approx 4.8517 \times 10^8, \\ e^{100} &\approx 2.6881 \times 10^{43}, & f(100) &\approx 1.4155 \times 10^{43}, \\ e^{-20} &\approx 2.0612 \times 10^{-9}, & f(-20) &\approx {}^\star 5.6219 \times 10^{-9}. \end{aligned}$$

Thus $f(20)$ is accurate, but $f(100)$ is inaccurate because of truncation errors, and $f(-20)$ is inaccurate because of rounding errors. (Why? See Exercise β.2.) This straightforward summation of the Taylor series for e^x is a classic example of a bad algorithm: it is sometimes unstable and always inefficient, unless $|x|$ is small.

Some people consider the study of rounding errors a tedious business, and there are those who assume that all of numerical analysis must be tedious, for isn't it mainly the study of rounding errors? Here are three reasons why these views are unjustified.

First, thanks to the simple formulas (β.1), the modeling of rounding errors is easy and even elegant. Rounding errors are by no means capricious or unpredictable, and one can understand them well without worrying about machine-dependent engineering details.

Second, the existence of inexact arithmetic should be blamed not on computers, but on mathematics itself. It has been known for a long time that many well-defined quantities cannot be calculated exactly by any finite sequence of elementary operations; the classic example is the "unsolvability" of the roots of polynomials of degree ≥ 5. Thus approximations are unavoidable even in principle.*

---

* Mathematics also rests on approximations at a more fundamental level. The set of real numbers is uncountably infinite, but as was first pointed out by Turing in his famous paper on "Computable

Finally and most important, numerical analysis is not just the study of rounding errors! Here is a better definition:

Numerical analysis is the study of algorithms
for mathematical problems involving continuous variables.

Thus it is a field devoted to *algorithms,* in which the basic questions take the form, "What is the best algorithm for computing such-and-such?" Insensitivity to rounding errors is only a part of what makes a numerical algorithm good; speed, for example, is certainly as important.

EXERCISES

▶ *β.1.* Figure out machine epsilon exactly on your calculator or computer. What are the base β and the precision t? Explain your reasoning carefully.

▷ *β.2. (Continuation of Example β.1.)*
(a) Explain the mechanism by which rounding errors affected the computation of $f(-20)$.
(b) If we compute $f(-100)$ in the same way, will there be a star in front of the result?
(c) Suggest an algorithm for calculating e^x more quickly and accurately in floating-point arithmetic.

Numbers" in 1937, only a countable subset of them can ever be specified by any finite description. The vast majority of real numbers can only be described in effect by listing their infinite decimal expansions, which requires an infinite amount of time. Thus most real numbers are not merely unrepresentable on computers; they are unmentionable even by pure mathematicians, and exist only in a most shadowy sense.

Chapter 1.
Ordinary Differential Equations

Just as the twig is bent, the tree's inclined.
— ALEXANDER POPE, Moral Essays (1734)

The central topic of this book is time-dependent partial differential equations (PDEs). Even when we come to discuss elliptic PDEs, which are not time-dependent, we shall find that some of the best numerical methods are iterations that behave very much as if there were a time variable. That is why elliptic equations come at the end of the book rather than the beginning.

Ordinary differential equations (ODEs) embody one of the two essential ingredients of this central topic: variation in time, but not in space. Because the state at any moment is determined by a finite set of numbers, ODEs are far easier to solve and far more fully understood than PDEs. In fact, an ODE usually has to be nonlinear before its behavior becomes very interesting—which is not the case at all for a PDE.

The solution of ODEs by discrete methods is one of the oldest and most successful areas of numerical computation. A dramatic example of the need for such computations is the problem of predicting the motions of planets or other bodies in space. The governing differential equations have been known since Newton, and for the case of only two bodies, such as a sun and a planet, Newton showed how to solve them exactly. In the three centuries since then, however, no one has ever found an exact solution for the case of three or more bodies.* Yet the numerical calculation of such orbits is almost effortless by modern standards, and is a routine component of the control of spacecraft.

The most important families of numerical methods for ODEs are

- linear multistep methods,
- Runge-Kutta methods.

This chapter presents the essential features of linear multistep methods, with emphasis on the fundamental ideas of accuracy, stability, and convergence. An important distinction is made between "classical" stability and "eigenvalue stability". All of these notions will prove indispensable when we move on to discuss PDEs in Chapter 3. Though we do not describe Runge-Kutta methods except briefly in §1.8, most of the analytical ideas described here apply with minor changes to Runge-Kutta methods too.

*In fact there are theorems to the effect that such an exact solution can never be found. This unpredictability of many-body motions is connected with the phenomenon of chaos.

1.1. nitial alue prole s

An, or , is an equation of the form

$$() \quad (()) \tag{111}$$

where is the time variable, is a real or complex scalar or vector function of (() , 1), and is a function that takes values in .* e also say that (1.1.1) is a . The symbol denotes , and if 1, it should be interpreted componentwise: ( ) ( ) . Similarly, denotes , and so on.

here clarity permits, we shall leave out the arguments in equations like (1.1.1), which then becomes simply .

The study of ODEs goes back to Newton and Leibni in the 167s, and like so much of mathematics, it owes a great deal also to the work of Euler in the 18th century. Systems of ODEs were first considered by Lagrange in the 175s, but the use of vector notation did not become standard until around 189.

If ( ) () () for some functions () and (), the ODE is , and if () it is linear and. (In the vector case () is an matrix and () is an -vector.) Otherwise it is I f ( ) is independent of, the ODE is. I f ( ) is independent of , the ODE reduces to an indefinite integral.

To make (1.1.1) into a fully specified problem, we shall provide at and look for solutions on some interval, . The choice of as a starting point introduces no loss of generality, since any other could be treated by the change of variables.

$$\begin{aligned} &() \\ &( ) \\ &() (\quad ()) \end{aligned} \tag{112}$$

* is the space of complex column vectors of lenth . In practical problems the variables are usually real so that for many purposes we could writeR instead.When we come to Fourier analysis of linear partial differential quations howev er the use of complex variables will be very convenient.

Numerical methods for solving ODE initial-value problems are the subject of this chapter. e shall not discuss boundary- value problems, in which various components of are specified at two or more distinct points of time; see Keller (1978) and Ascher, et al. (1988).

EXAMPLE 1.1.1. The scalar initial-value problem $u_t = Au$, $u(0) = a$ has the solution $u(t) = ae^{tA}$, which decays to 0 as $t \to \infty$ provided that $A < 0$, or $\mathrm{Re}\, A < 0$ if A is complex This ODE is linear, homogeneous, and autonomous

EXAMPLE 1.1.2. The example above becomes a vector initial-value problem if u and a are N-vectors and A is an $N \times N$ matrix The solution can still be written $u(t) = e^{tA}a$, if e^{tA} now denotes the $N \times N$ matrix defined by applying the usual Taylor series for the exponential to the matrix tA For generic initial vectors a, this solution $u(t)$ decays to the zero vector as $t \to \infty$, i e $\lim_{t\to\infty} \|u(t)\| = 0$, if and only if each eigenvalue λ of A satisfies $\mathrm{Re}\,\lambda < 0$

EXAMPLE 1.1.3. The scalar initial-value problem $u_t = u \cos t$, $u(0) = 1$ has the solution $u(t) = e^{\sin t}$ One can derive this by separation of variables by integrating the equation $du/u = \cos t\, dt$

EXAMPLE 1.1.4. The nonlinear initial-value problem $u_t = u + u^2$, $u(0) = 1$ has the solution $u(t) = 1/(2e^{-t} - 1)$, which is valid until the solution becomes infinite at $t = \log 2 \approx 0.693$ This ODE is an example of a Bernoulli differential equation One can derive the solution by the substitution $w(t) = 1/u(t)$, which leads to the linear initial-value problem $w_t = -1 - w$, $w(0) = 1$, with solution $w(t) = 2e^{-t} - 1$

EXAMPLE 1.1.5. The Lorenz equations, with the most familiar choice of parameters, can be written

$$u_t = -10u + 10v, \qquad v_t = 28u - v - uw, \qquad w_t = -\tfrac{8}{3}w + uv.$$

This is the classical example of a nonlinear system of ODEs whose solutions are chaotic The solution cannot be written in closed form

---

Equation (1.1.1) is an ODE of **fi**, for it contains only a first derivative with respect to . Many ODEs that occur in practice are of second order or higher, so the form we have chosen may seem unduly restrictive. However, any higher-order ODE can be reduced to an equivalent system of first-order ODEs in a mechanical fashion by introducing additional variables representing lower-order terms. The following example illustrates this reduction.

EXAMPLE 1.1.6. Suppose that a mass m at the end of a massless spring of length y experiences a force $F = -K(y - y^*)$, where K is the spring constant and y^* is the rest position (Hooke's Law) By Newton's First Law, the motion is governed by the autonomous equation

$$y_{tt} = -\frac{K}{m}(y - y^*), \qquad y(0) = a, \quad y_t(0) = b$$

for some a and b, a scalar second-order initial-value problem Now let u be the 2-vector with $u^{(1)} = y$, $u^{(2)} = y_t$ Then the equation can be rewritten as the first-order system

$$\begin{aligned} u_t^{(1)} &= u^{(2)}, & u^{(1)}(0) &= a, \\ u_t^{(2)} &= -\frac{K}{m}(u^{(1)} - y^*), & u^{(2)}(0) &= b. \end{aligned}$$

It should be clear from this example, and Exercise 1.1.1 below, how to reduce an arbitrary collection of higher-order equations to a first-order system. More formal treatments of this process can be found in the references.

Throughout the fields of ordinary and partial differential equations—and their numerical analysis—there are a variety of procedures in which one problem is reduced to another. e shall see inhomogeneous terms reduced to initial data, initial data reduced to boundary values, and so on. But this reduction of higher-order ODEs to first-order systems is unusual in that it is not just a convenient theoretical device, but highly practical. In fact, most of the general-purpose ODE software currently available assumes that the equation is written as a first-order system. One pays some price in efficiency for this, but it is usually not too great. For PDEs, on the other hand, such reductions are less often practical, and indeed there is less general-purpose software available of any kind.

It may seem obvious that (1.1.2) should have a unique solution for all ; after all, the equation tells us exactly how changes at each instant. But in fact, solutions can fail to exist or fail to be unique, and an example of nonexistence for log 2 appeared already in Example 1.1.4 (see also Exercises 1.1.4 and 1.1.5). To ensure existence and uniqueness, we must make some assumptions concerning. The standard assumption is that is continuous with respect to and satisfies a (uniform) **Lpcz c** with respect to . This means that there exists a constant L such tha t for all v and

$$\| (\quad) (v) \qquad \| \leq L \| \qquad v \| \tag{113}$$

where $\|\cdot\|$ denotes some norm on the set of - vectors. (For 1, i.e. a scalar system of equations, $\|\cdot\|$ is usually just the absolute value $|\cdot|$. For 2, the most important examples of norms are $\|\cdot\|$, $\|\cdot\|$, and $\|\cdot\|_\infty$, and the reader should make sure that he or she is familiar with these examples. See Appendix B for a review of norms.) A sufficient condition for Lipschit continuity is that the partial derivative of ( ) with respect to exists and is bounded in norm by L for all and

The following result goes back to Cauchy in 1824.

EXISTENCE AND UNIQUENESS FOR THE INITIAL VALUE PROBLEM
T **11** *L( ) wp* *nyw* *Lpz wp* . *T* *x* *q* () - *pm* *1.1.2.*

The standard proof of Theorem 1.1 nowadays, which can be found in many books on ODEs, is based on a construction called the method of successive approximations or the Picard iteration. This construction can be realied as a numerical method, but not one of the form we shall discuss in this book. Following Cauchys original idea, however, a more elementary proof can also be based on more standard numerical methods, such as Eulers formula, which is mentioned in the next section. See Henrici (1962) or Hairer, Nørsett & anner (1987).

Theorem 1.1 can be strengthened by allowing to be Lipschit continuous not for all , but for confined to an open subset D of . Unique solutions then still exist on the interval , where is either the first point at which the solution hits the boundary of D, or , whichever is smaller.

EXERCISES

▷ *1.1.1. Reduction to first-order system.* Consider the system of ODEs

$$u_{ttt} = u_{tt} + v_t, \qquad v_t t = u^2 + \sin(v) + e^t u_t v_t$$

with initial data

$$u(0) = u_t(0) = u_{tt}(0) = v_t(0) = 0, \qquad v(0) = 1.$$

Reduce this initial-value problem to a first-order system in the standard form (1 1 2)

▷ *1.1.2. The planets.* M planets (or suns, or spacecraft) orbit about each other in three space dimensions according to Newton's laws of gravitation and acceleration What are the dimension and order of the corresponding system of ODEs— *(a)* When written in their most natural form? *(b)* When reduced to a first-order system? (This problem is intended to be straightforward; do not attempt clever tricks such as reduction to center-of-mass coordinates)

▷ *1.1.3. Existence and uniqueness.* Apply Theorem 1 1 to show existence and uniqueness of the solutions we have given for the following initial-value problems, stating explicitly your choices of suitable Lipschitz constants L:

(a) Example 1 1 1

(b) Example 1 1 3

(c) Example 1 1 4 First, by considering $u_t = u + u^2$ itself, explain why Theorem 1 1 does not guarantee existence or uniqueness for all $t > 0$ Next, by considering the transformed equation $w_t = -1 - w$, show that Theorem 1 1 *does* guarantee existence and uniqueness

until such time as u may become infinite Exactly how does the proof of the theorem fail at that point?

(d) Example 1 1 2

▷ *1.1.4. Nonexistence and nonuniqueness.* Consider the scalar initial-value problem

$$u_t = u^\alpha, \qquad u(0) = u_0 \geq 0$$

for some constant $\alpha > 0$

(a) For which α does Theorem 1 1 guarantee existence and uniqueness of solutions for all $t > 0$?

(b) For $\alpha = 2$ and $u_0 = 1$, no solution for all $t > 0$ exists Verify this informally by finding an explicit solution that blows up to infinity in a finite time (Of course such an example by itself does not constitute a proof of nonexistence)

(c) For $\alpha = \frac{1}{2}$ and $u_0 = 0$, there is more than one solution Find one of them, an "obvious" one Then find another "obvious" one Now construct an infinite family of distinct solutions

▷ *1.1.5. Continuity with respect to t.* Theorem 1 1 requires continuity with respect to t as well as Lipschitz continuity with respect to u Show that this assumption cannot be dispensed with by finding an initial-value problem, independent of T, in which f is uniformly Lipschitz continuous with respect to u but discontinuous with respect to t, and for which no solution exists on any interval $[0,T]$ with $T > 0$ *(Hint:* consider the degenerate case in which $f(u,t)$ is independent of u)

1.2. Linear ultistep for ulas

Most numerical methods in every field are based on discretization For the solution of ordinary differential equations, one of the most powerful discretization strategies goes by the name of **linear multistep methods**

Suppose we are given an initial-value problem (1 1 2) that satisfies the hypotheses of Theorem 1 1 on some interval $[0,T]$ Then it has a unique solution $u(t)$ on that interval, but the solution can rarely be found analytically Let $k>0$ be a real number, the **time step**, and let $t_0,t_1,\ldots$ be defined by $t_n = nk$ Our goal is to construct a sequence of values $v^0,v^1,\ldots$ such that

$$v^n \approx u(t_n), \qquad n \geq 0. \tag{1.2.1}$$

(The superscripts are not exponents!—we are leaving room for subscripts to accommodate spatial discretization in later chapters) We also sometimes write $v(t_n)$ instead of v^n Let f^n be the abbreviation

$$f^n = f(v^n,t_n). \tag{1.2.2}$$

A linear multistep method is a formula for calculating each new value v^{n+1} from some of the previous values $v^0,\ldots,v^n$ and $f^0,\ldots,f^n$ * Equation (1 2 11) below will make this more precise

We shall take the attitude, standard in this field, that the aim in solving an initial-value problem numerically is to achieve a prescribed accuracy with the use of as few function evaluations $f(v^n,t_n)$ as possible n other words, obtaining function values is assumed to be so expensive that all subsequent manipulations—all "overhead" operations—are essentially free For easy problems this assumption may be unrealistic, but it is the harder problems that matter more For hard problems values of f may be very expensive to determine, particularly if they are obtained by solving an inner algebraic or differential equation

Linear multistep methods are designed to minimize function evaluations by using the approximate values v^n and f^n repeatedly

The simplest linear multistep method is a one-step method: the **Euler** formula, defined by

$$v^{n+1} = v^n + kf^n. \tag{1.2.3}$$

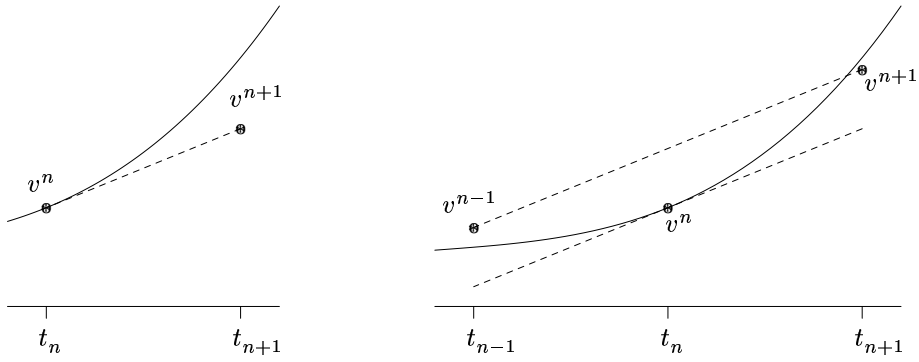
The motivation for this formula is linear extrapolation, as suggested in Figure 1 2 1a f v^0 is given (presumably set equal to the initial value u_0), it is a straightforward matter to apply (1 2 3) to compute successive values $v^1,v^2,\ldots$ Euler's method is an example of an **explicit one-step formula**

A related linear multistep method is the **backward Euler** (or **implicit Euler**) formula, also a one-step formula, defined by

$$v^{n+1} = v^n + kf^{n+1}. \tag{1.2.4}$$

The switch from f^n to f^{n+1} is a big one, for it makes the backward Euler formula **implicit** According to (1 2 2), f^{n+1} is an abbreviation for $f(v^{n+1},t_{n+1})$ Therefore, it would

*Ineneral v^n and f^n are vectors in but it is safe to think of them as scalars; the extension to systems of qua tions is easy.



(a) Euler *(b)* midpoint

Figure 1.2.1. One step of the Euler and midpoint formulas The solid curves represent not the exact solution to the initial-value problem, but the exact solution to the problem $u_t = f$, $u(t_n) = v^n$

appear that v^{n+1} cannot be determined from (1 2 4) unless it is already known! n fact, to implement an implicit formula, one must employ an iteration of some kind to solve for the unknown v^{n+1}, and this involves extra work, not to mention the questions of existence and uniqueness * But as we shall see, the advantage of (1 2 4) is that in some situations it may be stable when (1 2 3) is catastrophically unstable Throughout the numerical solution of differential equations, there is a tradeoff between explicit methods, which tend to be easier to implement, and implicit ones, which tend to be more stable Typically this tradeoff takes the form that an explicit method requires less work per time step, while an implicit method is able to take larger and hence fewer time steps without sacrificing accuracy to unstable oscillations *

An example of a more accurate linear multistep formula is the **trapezoid** rule,

$$v^{n+1} = v^n + \frac{k}{2}(f^n + f^{n+1}), \tag{1.2.5}$$

also an implicit one-step formula Another is the **midpoint** rule,

$$v^{n+1} = v^{n-1} + 2kf^n, \tag{1.2.6}$$

an explicit **two-step** formula (Figure 1 2 1b) The fact that (1 2 6) involves multiple time levels raises a new difficulty We can set $v^0 = u_0$, but to compute v^1 with this formula, where

*In so-called predictor-corrector methods not discussed in this book the iteration is terminated before converenceivin a class of methods intermediate between explicit and implicit.

*But don't listen to people who talk about "conservation of difficulty" as if there were never any clear winnersWe shall see that sometimes explicit methods are vas tly more efficient than implicit ones lare non-stiff systems of s and sometimes it is the reverse small stiff systems .

shall we get v^{-1}? Or if we want to begin by computing v^2, where shall we get v^1? This initialization problem is a general one for linear multistep formulas, and it can be addressed in several ways One is to calculate the missing values numerically by some simpler formula such as Euler's method—possibly applied several times with a smaller value of k to avoid loss of accuracy Another is to handle the initialization process by Runge-Kutta methods, to be discussed in Section 1 9

n Chapter 3 we shall see that the formulas (1 2 3)–(1 2 6) have important analogs for partial differential equations † For the easier problems of ordinary differential equations, however, they are too primitive for most purposes nstead one usually turns to more complicated and more accurate formulas, such as the **fourth-order Adams-Bashforth** formula,

$$v^{n+1} = v^n + \frac{k}{24}\left(55f^n - 59f^{n-1} + 37f^{n-2} - 9f^{n-3}\right), \tag{1.2.7}$$

an explicit four-step formula, or the **fourth-order Adams-Moulton** formula,

$$v^{n+1} = v^n + \frac{k}{24}\left(9f^{n+1} + 19f^n - 5f^{n-1} + f^{n-2}\right), \tag{1.2.8}$$

an implicit three-step formula Another implicit three-step formula is the **third-order backwards differentiation** formula,

$$v^{n+1} = \tfrac{18}{11}v^n - \tfrac{9}{11}v^{n-1} + \tfrac{2}{11}v^{n-2} + \tfrac{6}{11}kf^{n+1}, \tag{1.2.9}$$

whose advantageous stability properties will be discussed in Section 1 8

EXAMPLE 1.2.1. Let us perform an experiment to compare some of these methods The initial-value problem will be

$$u_t = u, \qquad t \in [0,2], \quad u(0) = 1, \tag{1.2.10}$$

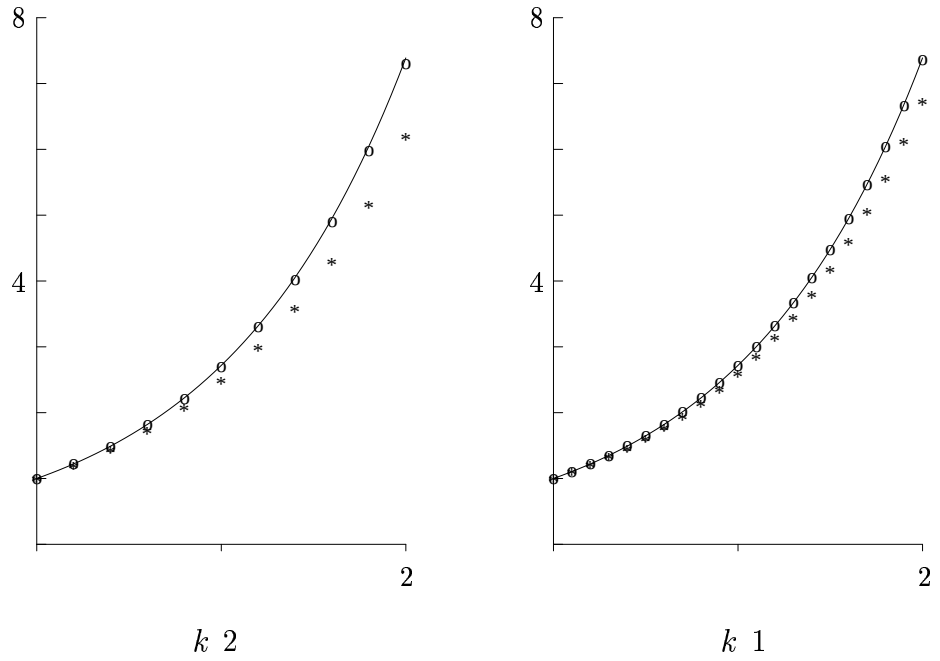
whose solution is simply $u(t) = e^t$ Figure 1 2 2 compares the exact solution with the numerical solutions obtained by the Euler and midpoint formulas with $k = 0.2$ and $k = 0.1$ (For simplicity, we took v^1 equal to the exact value $u(t_1)$ to start the midpoint formula) A solution with the fourth-order Adams-Bashforth formula was also calculated, but is indistinguishable from the exact solution in the plot Notice that the midpoint formula does much better than the Euler formula, and that cutting k in half improves both To make this precise, Table 1 2 1 compares the various errors $u(2) - v(2)$ The final column of ratios suggests that the errors for the three formulas have magnitudes $\Theta(k)$, $\Theta(k^2)$, and $\Theta(k^4)$ as $k \to 0$ The latter figure accounts for the name "fourth-order;" see the next section

---

Up to this point we have listed a few formulas, some with rather complicated coefficients, and shown that they work at least for one problem. Fortunately, the subject of linear multistep methods has a great deal more order in it than this. A number of questions suggest themselves:

- hat is the general form of a linear multistep method?

† uler Backward uler Crank-Nicolsonand Leap Fro.



k 2 k 1

F **122** Solution of (1.2.1) by the Euler (∗) and midpoint (o) formulas.

	k 2	k 1	k 5	ratio (k 1 to k 5)
Euler	1.19732	.66156	.3497	1.9
Midpoint	. 955	. 2382	. 67	3.92
AB4	. 422	. 38	. 3	12.7

T **121** Errors (2) $v(2)$ for the experiment of Figure 1.2.2.

- Can appropriate coefficients be derived in a systematic way?
- How accurate are these methods?
- Can anything go wrong?

In the following pages we shall see that these questions have very interesting answers.

e can take care of the first one right away by defining the general linear multistep formula:

A s- **p** **p***m*

$$\sum_{j=}^{s} \; {}_j v^{n+j} \qquad k \sum_{j=}^{s} \; {}_j \; {}^{n+j} \tag{12.11}$$

m{ } $_j$} { $_j$} *w* $_s$ 1 / /.
I $_s$ *m* **x** **p** $_s$ / **p**.

Readers familiar with electrical engineering may notice that (1.2.11) looks like the definition of a **cv** or **R fi** (Oppenheim & Schafer, 1989). Linear multistep formulas can indeed be thought of in this way, and many of the issues to be described here also come up in digital signal processing, such as the problem of stability and the idea of testing for it by looking for eros of a polynomial in the unit disk of the complex plane. A linear multistep formula is not quite a digital filter of the usual kind, however. In one respect it is more general, since the function depends on and ra ther than on alone. In another respect it is more narrow, since the coefficients are chosen so that the formula has the effect of integration. Digital filters are designed to accomplish a much wider variety of tasks, such as high-pass or low-pass filtering (smoothing), differentiation, or channel equali ation.

hat about the word "linear"? Do linear multistep formulas apply only to linear differential equations? Certainly not. Equation (1.2.11) is called linear because the quantities v^n and ^{n} are related linearly; may very well be a nonlinear function of and Some authors refer simply to "multistep formulas" to avoid this potential source of confusion.

EXERCISES

▷ *1.2.1.* What are s, $\{\alpha_j\}$, and $\{\beta_j\}$ for formulas (1 2 7)–(1 2 9)?

▷ *1.2.2. Linear multistep formula for a system of equations.* One of the footnotes above claimed that the extension of linear multistep methods to systems of equations is easy Verify this by writing down exactly what the 2×2 system of Example 1 1 5 becomes when it is approximated by the midpoint rule (1 2 6)

▷ *1.2.3. Exact answers for low-degree polynomials.* f L is a nonnegative integer, the initial-value problem

$$u_t(t) = \frac{L}{t+1} u(t), \qquad u(0) = 1$$

has the unique solution $u(t) = (t+1)^L$ Suppose we calculate an approximation $v(2)$ by a linear multistep formula, using exact values where necessary for the initialization
(a) For which L does (1 2 3) reproduce the exact solution? *(b)* (1 2 6)? *(c)* (1 2 7)?

▷ *1.2.4. Extrapolation methods.*

(a) The values $V_k = v(2) \approx u(2)$ computed by Euler's method in Figure 1 2 2 are $V_{0.1} \approx 6.72750$ and $V_{0.2} \approx 6.19174$ Use a calculator to compute the analogous value $V_{0.4}$

(b) t can be shown that these quantities V_k satisfy

$$V_k = u(2) + C_1 k + C_2 k^2 + O(k^3)$$

for some constants C_1, C_2 as $k \to 0$ n the process known as **Richardson extrapolation**, one constructs the higher-order estimates

$$V_k' = V_k + (V_k - V_{2k}) = u(2) + O(k^2)$$

and

$$V_k'' = V_k' + \tfrac{1}{3}(V_k' - V_{2k}') = u(2) + O(k^3).$$

Apply these formulas to compute $V_{0.1}''$ for this problem How accurate is it? This is an example of an **extrapolation method** for solving an ordinary differential equation (Gragg 1965; Bulirsch & Stoer 1966) The analogous method for calculating integrals is known as **Romberg integration** (Davis & Rabinowitz 1975)

1.3. Accuracy and consistency

n this section we define the consistency and order of accuracy of a linear multistep formula, as well as the associated characteristic polynomials $\rho(z)$ and $\sigma(z)$, and show how these definitions can be applied to derive accurate formulas by a method of undetermined coefficients We also show that linear multistep formulas are connected with the approximation of the function $\log z$ by the rational function $\rho(z)/\sigma(z)$ at $z=1$

With $\{\alpha_j\}$ and $\{\beta_j\}$ as in (1 2 11), the **characteristic polynomials** (or **generating polynomials**) for the linear multistep formula are defined by

$$\rho(z)=\sum_{j=0}^{s}\alpha_j z^j, \qquad \sigma(z)=\sum_{j=0}^{s}\beta_j z^j. \tag{1.3.1}$$

The polynomial ρ has degree exactly s, and σ has degree s if the formula is implicit or $<s$ if it is explicit Specifying ρ and σ is obviously equivalent to specifying the linear multistep formula by its coefficients We shall see that ρ and σ are convenient for analyzing accuracy and indispensable for analyzing stability

---

EXAMPLE 1.3.1. Here are the characteristic polynomials for the first four formulas of the last section:

Euler (1 2 3):	$s=1$,	$\rho(z)=z-1$,	$\sigma(z)=1$
Backward Euler (1 2 4):	$s=1$,	$\rho(z)=z-1$,	$\sigma(z)=z$
Trapezoid (1 2 5):	$s=1$,	$\rho(z)=z-1$,	$\sigma(z)=\frac{1}{2}(z+1)$
Midpoint (1 2 6):	$s=2$,	$\rho(z)=z^2-1$,	$\sigma(z)=2z$

Now let Z denote a **p** that acts both on discrete functions according to

$$Z v^n \quad v^{n+} \tag{132}$$

and on continuous functions according to

$$Z\ () \qquad (\quad k) \tag{133}$$

(In principle we should write $Z\ \ v^{\ n}$ and $Z\ \ \ ()$, but expressions like $Z\ v^n$ and even $Z(v^n)$ are irresistibly convenient.) The powers of Z have the obvious meanings, e.g., $Z\ v^n \quad v^{n+}$ and $Z^{-}\ \ () \qquad (\quad k)$.

Equation (1.2.11) can be rewritten compactly in terms of Z, ρ, and σ:

$$\rho(Z)v^n \quad k\sigma(Z)^{\ n} \tag{134}$$

hen a linear multistep formula is applied to solve an ODE, this equation is satisfied exactly since it is the definition of the numerical approximation $\{v^n\}$.

If the linear multistep formula is a good one, the analogous equation ought to be nearly satisfied when $\{v^n\}$ and $\{\ ^n\}$ are replaced by discreti a tions of any well-behaved function $(\ )$ and its deriva tive $(\ )$. ith this in mind, let us define the **pcp** $\mathcal{L}$, acting on the set of continuously differentiable functions $(\ )$, by

$$\mathcal{L} \quad \rho(Z) \quad k\mathcal{D}\sigma(Z) \tag{135}$$

where $\mathcal{D}$ is the time differentiation operator, that is,

$$\begin{aligned} \mathcal{L}\ (\ _n) \quad & \rho(Z)\ (\ _n) \quad k\sigma(Z)\ (\ _n) \\ & \sum_{j=}^{s}\ _j\ (\ _{n+j}) \quad k\sum_{j=}^{s}\ _j\ (\ _{n+j}) \end{aligned} \tag{136}$$

If the linear multistep formula is accurate, $\mathcal{L}\ (\ _n)$ ought to be small, and we make this precise as follows. Let a function $(\ )$ and its derivative $(\ )$ be expanded formally in Taylor series about $_n$,

$$(\ _{n+j}) \quad (\ _n) \quad jk\ (\ _n) \quad -(jk) \quad (\ _n) \quad \cdots \tag{137}$$

$$(\ _{n+j}) \quad (\ _n) \quad jk\ (\ _n) \quad -(jk) \quad (\ _n) \quad \cdots \tag{138}$$

Inserting these formulas in (1.3.6) gives the **ccz** (or **c c**) for the linear multistep formula,

$$\mathcal{L}\ (\ _n) \quad C\ (\ _n) \quad C\ k\ (\ _n) \quad C\ k \quad (\ _n) \quad \cdots \tag{139}$$

where

$$\begin{aligned} C \quad & \cdots \quad _s \\ C \quad & (\quad 2 \quad \cdots \quad s\ _s)\ (\quad \cdots \quad _s) \\ C \quad & -(\quad 4 \quad \cdots \quad s\ _s)\ (\quad 2 \quad \cdots \quad s\ _s) \\ & \vdots \\ C_m \quad & \sum_{j=}^{s}\frac{j^m}{m!}\ j \quad \sum_{j=}^{s}\frac{j^{m-}}{(m\quad 1)!}\ j \end{aligned} \tag{131}$$

e now define:

A *npm* **cc** **c** *p*

$$\mathcal{L}\ (\ _n) \quad \Theta(k^{p+}) \qquad k \to$$

. . *C C* $\cdots$ C_p *C* $_{p+}$ /. *T* **c** *C* $_{p+}$ ·
Tm **c** *C* *C* .. *y*
p 1.

EXAMPLE 1.3.1, CONTINUED. First let us analyze the local accuracy of the Euler formula in a lowbrow fashion that is equivalent to the definition above: we suppose that $v^0,\dots,v^n$ are exactly equal to $u(t_0),\dots,u(t_n)$, and ask how close v^{n+1} will then be to $u(t_{n+1})$ f $v^n = u(t_n)$ exactly, then by definition,

$$v^{n+1} = v^n + kf^n = u(t_n) + ku_t(t_n),$$

while the Taylor series (1 3 7) gives

$$u(t_{n+1}) = u(t_n) + ku_t(t_n) + \frac{k^2}{2}u_{tt}(t_n) + O(k^3).$$

Subtraction gives

$$u(t_{n+1}) - v^{n+1} = \frac{k^2}{2}u_{tt}(t_n) + O(k^3)$$

as the formal local discretization error of the Euler formula

Now let us restate the argument in terms of the operator $\mathcal{L}$ By combining (1 3 6) and (1 3 7), or by calculating $C_0 = C_1 = 0$, $C_2 = \frac{1}{2}$, $C_3 = \frac{1}{6}$ from the values $\alpha_0 = -1$, $\alpha_1 = 1$, $\beta_0 = 1$, $\beta_1 = 0$ in (1 3 10), we obtain

$$\begin{aligned}\mathcal{L}u(t_n) &= u(t_{n+1}) - u(t_n) - ku_t(t_n) \\ &= \frac{k^2}{2}u_{tt}(t_n) + \frac{k^3}{6}u_{ttt}(t_n) + \cdots. \end{aligned} \tag{1.3.11}$$

Thus again we see that the order of accuracy is 1, and evidently the error constant is $\frac{1}{2}$

Similarly, the trapezoid rule (1 2 5) has

$$\begin{aligned}\mathcal{L}u(t_n) &= u(t_{n+1}) - u(t_n) - \frac{k}{2}\big(u_t(t_{n+1}) + u_t(t_n)\big) \\ &= -\frac{k^3}{12}u_{ttt}(t_n) - \frac{k^4}{24}u_{tttt}(t_n) + \cdots, \end{aligned} \tag{1.3.12}$$

with order of accuracy 2 and error constant $-\frac{1}{12}$, and the midpoint rule (1 2 6) has

$$\mathcal{L}u(t_n) = \tfrac{1}{3}k^3u_{ttt}(t_n) + \tfrac{1}{3}k^4u_{tttt}(t_n) + \cdots, \tag{1.3.13}$$

with order of accuracy 2 and error constant $\frac{1}{3}$

The idea behind the definition of order of accuracy is as follows. Equation (1.3.9) suggests that if a linear multistep formula is applied to a problem with a sufficiently smooth solution (), an error of approxima tely $C_{p+}\ k^{p+}\ \frac{d^{p+1}u}{d\ ^{p+1}}(\ _n)$ $O(k^{p+}\ )$ will be introduced locally at step n.* This error is known as the **c cz** (or **c**) at step n for the given initial-value problem (as opposed to the formal local discreti a tion error, which is a

*As usual the limit associ ated with the Bi - symbol $k\to$ is omitted here because it is obvious.

formal expression that does not depend on the initial-value problem). Since there are k $\Theta(k^{-})$ steps in the interval , these local errors can be expected to add up to a global error $O(k^p)$. e shall make this argument more precise in Section 1.6.

ith hindsight, it is obvious by symmetry that the trapeoid and midpoint formulas had to have even orders of accuracy. Notice that in (1.2.6), the terms involving v^{n+j} are antisymmetric about $_n$, and those involving $^{n+j}$ are symmetric. In (1.3.13) the effect of these symmetries is disguised, because $_{n-}$ was shifted to $_n$ for simplicity of the formulas in passing from (1.2.6) to (1.2.11) and (1.3.4). However, if we had expressed $\mathcal{L}\ (\ _n)$ as a Taylor series about $_{n+}$ instead, we would have obtained

$$\mathcal{L}\ (\ _n) \quad \tfrac{}{3}k^3 \quad (\ _{n+}\ ) \quad \tfrac{}{6}k^5 \quad (\ _{n+}\ ) \quad O(k^7)$$

with all the even-order derivatives in the series vanishing due to symmetry. Now it can be shown that to leading order, it doesn t matter what point one expands $\mathcal{L}\ (\ _n)$ about: C C_{p+} are independent of this point, though the subsequent coefficients C_{p+} C_{p+3} are not. Thus the vanishing of the even-order terms above is a valid explanation of the second-order accuracy of the midpoint formula. For the trapeoid rule (1.2.5), we can make an analogous argument involving an expansion of $\mathcal{L}\ (\ _n)$ about $_{n+\ /\ }$.

As a rule, symmetry arguments do not go very far in the analysis of discrete methods for ODEs, since most of the formulas used in practice are of high order and fairly complicated. They go somewhat further with the simpler formulas used for PDEs.

The definition of order of accuracy suggests a
cffi c for deriving linear multistep formulas: having decided somehow which $_j$ and $_j$ are permitted to be nonero, simply ad just these parameters to make p as large as possible. If there are q parameters, not counting the fixed value $_s$ 1, then the equations (1.3.1) with $\leq m \leq q$ 1 constitute a q q linear system of equations. If this system is nonsingular, as it usually is, it must have a unique solution that defines a linear multistep formula of order at least p q 1. See E xercise 1.3.1.

The method of undetermined coefficients can also be described in another, equivalent way that was hinted at in Exercise 1.2.3. It can be shown that any consistent linear multistep formula computes the solution to an initial-value problem exactly in the special case when that solution is a polynomial $p()$ of degree L, provided that L is small enough. The order of accuracy p is the largest such L (see Exercise 1.3.6). To derive a linear multistep formula for a particular choice of available parameters $_j$ and $_j$, one can choose the parameters in such a way as to make the formula exact when applied to polynomials of as high a degree as possible.

At this point it may appear to the reader that the construction of linear multistep formulas is a rather uninteresting matter. Given s , why not simply pick $_{s-}$ and $_{s}$ so as to achieve order of accuracy $2s$? The answer is that for s 2, the resulting formulas are unstable and consequently useless (see Exercise 1.5.4). In fact, as we shall see in Section 1.5, a famous theorem of Dahlquist asserts that a stable s-step linear multistep formula can have order of accuracy at most s 2. Consequently, nothing can be gained by permitting all $2s$ 1 coefficients in an s-step formula to be nonero. Instead, the linear multistep formulas used in practice usually have most of the $_j$ or most of the $_j$ equal toero. In the next section we shall describe several important families of this kind.

And there is another reason why the construction of linear multistep formulas is interesting. It can be made exceedingly slick! e shall now describe how the formulas above can be analyed more compactly, and the question of order of accuracy reduced to a question of rational approximation, by manipulation of formal power series.

Taking j 1 in (1.3.7) gives the identity

$$(_{n+}) \quad (_n) \quad k \quad (_n) \quad -k \quad (_n) \quad \cdots$$

Since ($_{n+}$) Z ($_n$), here is another way to express the same fact:

$$Z \quad 1 \quad (k\mathcal{D}) \quad \frac{}{!}(k\mathcal{D}) \quad \frac{}{3!}(k\mathcal{D})^3 \quad \cdots \quad e^{k\mathcal{D}} \qquad (13.14)$$

Like (1.3.7), this formula is to be interpreted as a formal identity; the idea is to use it as a tool for manipulation of terms of series without making any claims about convergence. Inserting (1.3.14) in (1.3.5) and comparing with (1.3.9) gives

$$\mathcal{L} \quad \rho(e^{k\mathcal{D}}) \quad k\mathcal{D}\sigma(e^{k\mathcal{D}}) \quad C \quad C\ (k\mathcal{D}) \quad C\ (k\mathcal{D}) \quad \cdots \qquad (13.15)$$

In other words, the coefficients C_j of (1.3.9) are nothing else than the Taylor series coefficients of the function $\rho(e^{k\mathcal{D}})$ $k\mathcal{D}\sigma(e^{k\mathcal{D}})$ with respect to the argument $k\mathcal{D}$. If we let κ be an abbreviation for $k\mathcal{D}$, this equation becomes even simpler:

$$\mathcal{L} \quad \rho(e^{\kappa})\ \kappa\ \sigma(e^{\kappa}) \quad C \quad C\ \kappa \quad C\ \kappa \quad \cdots \qquad (13.16)$$

ith the aid of a symbolic algebra system such as Macsyma, Maple, or Mathematica, it is a trivial matter to make use of (1.3.16) to compute the coefficients C_j for a linear multistep formula if the parameters $_j$ and $_j$ are given. See Exercise 1.3.7.

By definition, a linear multistep formula has order of accuracy p if and only if the the term between the equal signs in (1.3.16) is $\Theta(\kappa^{p+})$ as $\kappa \to$.

Since $\sigma(e^{\kappa})$ is an analytic function of κ, if it is nonzero at κ we can divide through to conclude that the linear multistep formula has order of accuracy p if and only if

$$\frac{\rho(e^{\kappa})}{\sigma(e^{\kappa})} \quad \kappa \quad \Theta(\kappa^{p+}) \qquad \text{as } \kappa \to \qquad (1317)$$

The following theorem restates this conclusion in terms of the variable $z \quad e^{\kappa}$.

LINEAR MULTISTEP FORMULAS AND RATIONAL APPROXIMATION

T 12 *A* *npm* w $\sigma(1)$ /
y *y*

$$\frac{\rho(z)}{\sigma(z)} \quad \log z \quad \Theta((z \quad 1)^{p+}) \quad \left[(z \quad 1) \quad -(z \quad 1) \quad \tfrac{1}{3}(z \quad 1)^3 \quad \cdots\right] \quad \Theta((z \quad 1)^{p+}) \qquad (1318)$$

$z \to 1$. *I y*

$$\rho(1) \qquad \rho'(1) \quad \sigma(1) \qquad (1319)$$

P . T o get from (1.3.17) to the first equality of (1.3.18), we make the change of variables $z \quad e^{\kappa}$, κ log z, noting that $\Theta(\kappa^{p+})$ as $\kappa \to$ has the same meaning as $\Theta((z \quad 1)^{p+})$ as $z \to 1$ since $e^{\kappa} \quad 1$ and $(e^{\kappa}) \quad \kappa$ / at κ. The second equality of (1.3.18) is just the usual Taylor series for $\log z$.

To prove (1.3.19), let (1.3.18) be written in the form

$$\rho(z) \quad \sigma(z)(z \quad 1) \quad O((z \quad 1)^{}) \quad \Theta((z \quad 1)^{p+})$$

or by expanding $\rho(z)$ and $\sigma(z)$ about $z \quad 1$,

$$\rho(1) \quad (z \quad 1)\rho'(1) \quad (\quad z \quad 1)\sigma(1) \quad O((z \quad 1)^{}) \quad \Theta((z \quad 1)^{p+}) \qquad (1320)$$

Matching successive powers of $z \quad 1$, we obtain $\rho(1) \quad \Leftrightarrow p$ and p $1 \Rightarrow \rho'(1) \quad \sigma(1) \Rightarrow p$ /. Thus (1.3.19) is equivalent to p 1, which is the definition of consistency. ∎

In Theorem 1.2 the ODE context of a linear multistep formula has vanished entirely, leaving a problem in the mathematical field known as **pp x -** . Questions of approximation underlie most discrete methods for differential equations, both ordinary and partial.

EXAMPLE 1.3.2. The trapezoid rule (1 2 5) has $\rho(z) = z-1$ and $\sigma(z) = \frac{1}{2}(z+1)$ Since $\rho(1) = 0$ and $\rho'(1) = 1 = \sigma(1)$, the formula is consistent Comparing (1 3 18) with the expansion

$$\frac{\rho(z)}{\sigma(z)} = \frac{z-1}{\frac{1}{2}(z+1)} = \frac{z-1}{1+\frac{1}{2}(z-1)} = (z-1)\left[1 - \frac{z-1}{2} + \frac{(z-1)^2}{4} - \cdots\right]$$

confirms that the trapezoid rule has order 2 and error constant $-\frac{1}{12}$

This approach to linear multistep formulas by rational approximation is closely related to the methods of **c** described in several of the references. It is also the basis of the method of analysis by described later in this chapter. For ordinary differential equations with special structure, such as highly oscillatory behavior, it is sometimes advantageous to approximate $\log z$ at one or more points other than z 1; this is the idea behind the **c-fi** or- **p** formulas discussed by Gautschi (1961), Liniger and illoughby (1967), Kuo and Levy (199), and others. See Exercise 1.3.3.

EXERCISES

▷ *1.3.1.* Show by the method of undetermined coefficients that

$$v^{n+1} = -4v^n + 5v^{n-1} + k\left(4f^n + 2f^{n-1}\right) \tag{1.3.21}$$

is the most accurate 2-step explicit linear multistep formula, with order of accuracy $p = 3$ n Section 1 5 we shall see that (1 3 21) is unstable and hence useless in practice

▷ *1.3.2.* Consider the third-order backwards differentiation formula (1 2 9)

(a) What are $\rho(z)$ and $\sigma(z)$?

(b) Apply Theorem 1 2 to verify consistency

(c) Apply Theorem 1 2 to verify that the order of accuracy is 3

▷ *1.3.3. Optimal formulas with finite step size.* The concept of order of accuracy is based on the limit $k \to 0$, but one can also devise formulas on the assumption of a finite step size $k > 0$ For example, an Euler-like formula might be defined by

$$v^{n+1} = v^n + \gamma(k) k f^n \tag{1.3.22}$$

for some function $\gamma(k)$ with $\gamma(k) \to 1$ as $k \to 0$

(a) What choice of $\gamma(k)$ makes (1 3 22) exact when applied to the equation $u_t = u$?

(b) ODEs of practical interest contain various time scales, so it is not really appropriate to consider just $u_t = u$ Suppose the goal is to approximate all problems $u_t = au$ with $a \in [0, 1]$ as accurately as possible State a definition of "as accurately as possible" based on the L^∞ norm of the error over a *single* time step, and determine the resulting function $\gamma(k)$

▶ *1.3.4. Numerical experiments.* Consider the scalar initial-value problem

$$u_t(t) = e^{\cos(tu(t))} \quad \text{for} \quad t \in [0,3], \qquad u(0) = 0.$$

n this problem you will test four numerical methods: (i) Euler, (ii) Midpoint, (iii) Fourth-order Adams-Bashforth, and (iv) Fourth-order Runge-Kutta, defined by

$$\begin{aligned} a &:= kf(v^n, t_n), \\ b &:= kf(v^n + a/2, t_n + k/2), \\ c &:= kf(v^n + b/2, t_n + k/2), \\ d &:= kf(v^n + c, t_n + k), \\ v^{n+1} &:= v^n + \tfrac{1}{6}(a + 2b + 2c + d) \end{aligned} \qquad (1.3.23)$$

.

(a) Write a computer program to implement (1 3 23) in high precision arithmetic (16 digits or more) Run it with $k = 1/2, 1/4, \ldots$ until you are confident that you have a computed value $v(3)$ accurate to at least 6 digits This will serve as your "exact solution " Make a computer plot or a sketch of $v(t)$ Store appropriate values from your Runge-Kutta computations for use as starting values for the multistep formulas (ii) and (iii)

(b) Modify your program so that it computes $v(3)$ by each of the methods (i)–(iv) for the sequence of time steps $k = 1/2, 1/4, \ldots, 1/256$ Make a table listing $v(3)$ and $v(3) - u(3)$ for each method and each k

(c) Draw a plot on a log-log scale of four curves representing $|v(3) - u(3)|$ as a function of the *number of evaluations of* f, for each of the four methods (Make sure you calculate each f^n only once, and count the number of function evaluations for the Runge-Kutta formula rather than the number of time steps)

(d) What are the approximate slopes of the lines in (c), and why? (f you can't explain them, there may be a bug in your program—very likely in the specification of initial conditions) Which of the four methods is most efficient?

(e) f you are programming in Matlab, solve this same problem with the programs `ode23` and `ode45` with eight or ten different error tolerances Measure how many time steps are required for each run and how much accuracy is achieved in the value $u(3)$, and add these new results to your plot of (c) What are the observed orders of accuracy of these adaptive codes? How do they compare in efficiency with your non-adaptive methods (i)–(iv)?

▷ *1.3.5. Statistical effects?* t was stated above that if local errors of magnitude $O(k^{p+1})$ are made at each of $\Theta(k^{-1})$ steps, then the global error will have magnitude $O(k^p)$ However, one might argue that more likely, the local errors will behave like random numbers of order $O(k^{p+1})$, and will accumulate in a square-root fashion according to the principles of a random walk, giving a smaller global error $O(k^{p+1/2})$ Experiments show that for most problems, including those of Example 1 2 1 and Exercise 1 3 4, this optimistic prediction is invalid What is the fallacy in the random walk argument? Be specific, citing an equation in the text to support your answer

▷ *1.3.6.* Prove the lemma alluded to on p 23: *An s-step linear multistep formula has order of accuracy p if and only if, when applied to an ordinary differential equation* $u_t = q(t)$, *it gives exact results whenever q is a polynomial of degree* $\le p$, *but not whenever q is a polynomial*

of degree $p+1$. (Assume arbitrary continuous initial data u_0 and exact numerical initial data $v_0,\ldots,v^{s-1}$)

▷ *1.3.7.* f you have access to a symbolic computation system, carry out the suggestion on p 26: write a short program which, given the parameters α_j and β_j for a linear multistep formula, computes the coefficients $C_0,C_1,\ldots$ Use your program to verify the results of Exercises 1 3 1 and 1 3 2, and then explore other linear multistep formulas that interest you

1.4. Deri ation of linear ultistep for ulas

The oldest linear multistep formulas are the Adams formulas,

$$v^{n+s} - v^{n+s-1} = k\sum_{j=0}^{s} \beta_j f^{n+j}, \tag{1.4.1}$$

which date to the work of J C Adams* as early as 1855 n the notation of (1 2 11) we have $\alpha_s = 1$, $\alpha_{s-1} = -1$, and $\alpha_0 = \cdots = \alpha_{s-2} = 0$, and the first characteristic polynomial is $\rho(z) = z^s - z^{s-1}$ For each $s \geq 1$, the s-step **Adams-Bashforth** and **Adams-Moulton** formulas are the optimal explicit and implicit formulas of this form, respectively "Optimal" means that the available coefficients $\{\beta_j\}$ are chosen to maximize the order of accuracy, and in both Adams cases, this choice turns out to be unique

We have already seen the 1-step Adams-Bashforth and Adams-Moulton formulas: they are Euler's formula (1 2 3) and the trapezoid formula (1 2 5), respectively The fourth-order Adams-Bashforth and Adams-Moulton formulas, with $s = 4$ and $s = 3$, respectively, were listed above as (1 2 7) and (1 2 8) The coefficients of these and other formulas are listed in Tables 1 4 1–1 4 3 on the next page The "stencils" of various families of linear multistep formulas are summarized in Figure 1 4 1, which should be self-explanatory

To calculate the coefficients of Adams formulas, there is a simpler and more enlightening alternative to the method of undetermined coefficients mentioned in the last section Think of the values $f^n, \ldots, f^{n+s-1}$ (A-B) or $f^n, \ldots, f^{n+s}$ (A-M) as discrete samples of a continuous function $f(t) = f(u(t), t)$ that we want to integrate,

$$u(t_{n+s}) - u(t_{n+s-1}) = \int_{t_{n+s-1}}^{t_{n+s}} u_t(t)\,dt = \int_{t_{n+s-1}}^{t_{n+s}} f(t)\,dt,$$

as illustrated in Figure 1 4 2a (Of course $f^n, \ldots, f^{n+s}$ will themselves be inexact due to earlier errors in the computation, but we ignore this for the moment) Let $q(t)$ be the unique polynomial of degree at most $s-1$ (A-B) or s (A-M) that interpolates these data, and set

$$v^{n+s} - v^{n+s-1} = \int_{t_{n+s-1}}^{t_{n+s}} q(t)dt. \tag{1.4.2}$$

Since the integral is a linear function of the data $\{f^{n+j}\}$, with coefficients that can be computed once and for all, (1 4 2) implicitly defines a linear multistep formula of the Adams type (1 4 1)

EXAMPLE 1.4.1. Let us derive the coefficients of the 2nd-order Adams-Bashforth formula, which are listed in Table 1 4 1 n this case the data to be interpolated are f^n and

*the same Adams who first predicted the existence of the planet Neptune

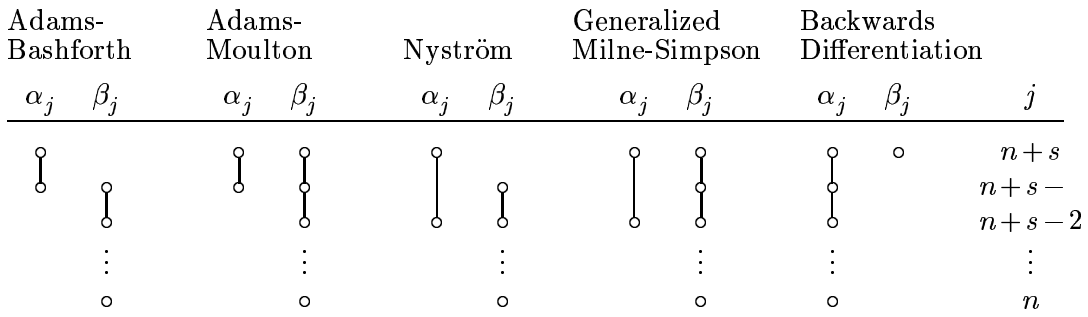


Figure 1.4.1. Stencils of various families of linear multistep formulas.

number of steps s	order p	β_s	β_{s-1}	β_{s-2}	β_{s-3}	β_{s-4}
					(EER)	
2	2		$\frac{3}{2}$	$-\frac{1}{2}$		
3	3		$\frac{23}{12}$	$-\frac{16}{12}$	$\frac{5}{12}$	
4	4		$\frac{55}{24}$	$-\frac{59}{24}$	$\frac{37}{24}$	$-\frac{9}{24}$

Table 1.4.1. Coefficients $\{\beta_j\}$ of Adams-Bashforth formulas.

number of steps s	order p	β_s	β_{s-1}	β_{s-2}	β_{s-3}	β_{s-4}
1	1	1		(BACKWARD EULER)		
1	2	$\frac{1}{2}$	$\frac{1}{2}$	(TRAPEZOID)		
2	3	$\frac{5}{12}$	$\frac{8}{12}$	$-\frac{1}{12}$		
3	4	$\frac{9}{24}$	$\frac{19}{24}$	$-\frac{5}{24}$	$\frac{1}{24}$	
4	5	$\frac{251}{720}$	$\frac{646}{720}$	$-\frac{264}{720}$	$\frac{106}{720}$	$-\frac{19}{720}$

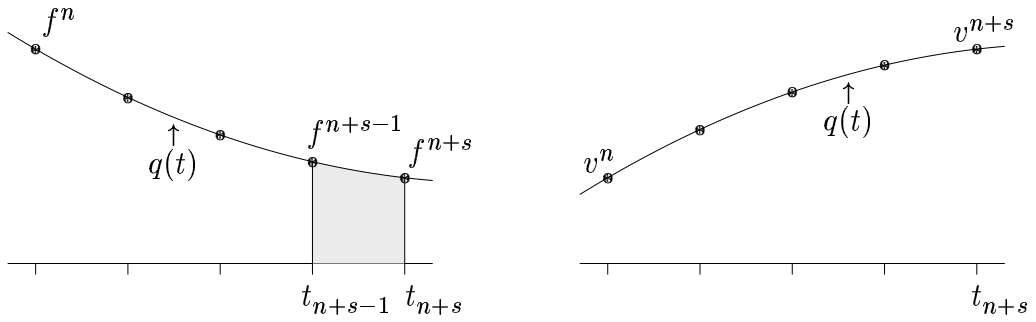
Table 1.4.2. Coefficients $\{\beta_j\}$ of Adams-Moulton formulas.

number of steps s	order p	α_s	α_{s-1}	α_{s-2}	α_{s-3}	α_{s-4}	β_s
1	1	1	-1	(BACKWARD EULER)			1
2	2	1	$-\frac{4}{3}$	$\frac{1}{3}$			$\frac{2}{3}$
3	3	1	$-\frac{18}{11}$	$\frac{9}{11}$	$-\frac{2}{11}$		$\frac{6}{11}$
4	4	1	$-\frac{48}{25}$	$\frac{36}{25}$	$-\frac{16}{25}$	$\frac{3}{25}$	$\frac{12}{25}$

Table 1.4.3. Coefficients $\{\alpha_j\}$ and β_s of backwards differentiation formulas.

f^{n+1}, and the interpolant is the linear polynomial $q(t) = f^{n+1} - k^{-1}(f^{n+1} - f^n)(t_{n+1} - t)$. Therefore (1.4.2) becomes

$$\begin{aligned} v^{n+2} - v^{n+1} &= \int_{t_{n+1}}^{t_{n+2}} \Big[f^{n+1} - k^{-1}(f^{n+1} - f^n)(t_{n+1} - t) \Big] dt \\ &= k f^{n+1} - k^{-1}(f^{n+1} - f^n) \int_{t_{n+1}}^{t_{n+2}} (t_{n+1} - t) dt \\ &= k f^{n+1} - k^{-1}(f^{n+1} - f^n)(-\tfrac{1}{2}k^2) \\ &= \frac{3}{2} k f^{n+1} - \frac{1}{2} k f^n. \end{aligned} \tag{1.4.3}$$



(a) Adams (b) backwards differentiation

Figure 1.4.2. Derivation of Adams and backwards differentiation formulas via polynomial interpolation.

More generally, the coefficients of the interpolating polynomial q can be determined with the aid of the **Newton interpolation formula.** To begin with, consider the problem of interpolating a discrete function $\{y^n\}$ in the points $0, \ldots, \nu$ by a polynomial $q(t)$ of degree at most ν. Let Δ and ∇ denote the **forward** and **backward difference operators,**

$$\Delta = Z - 1, \qquad \nabla = 1 - Z^{-1}, \tag{1.4.4}$$

where 1 represents the identity operator. For example,

$$\Delta y^n = y^{n+1} - y^n, \qquad \Delta^2 y^n = y^{n+2} - 2y^{n+1} + y^n.$$

Also, let $\binom{a}{j}$ denote the binomial coefficient "a choose j",

$$\binom{a}{j} = \frac{a(a-1)(a-2)\cdots(a-j+1)}{j!}, \tag{1.4.5}$$

defined for integers $j \geq 0$ and arbitrary $a \in \mathbb{C}$. The following is a standard result that can be found in many books of numerical analysis and approximation theory:

NEWTON INTERPOLATION FORMULA

Theorem 1.3. *The polynomial*

$$q(t) = \left[1 + \binom{t}{1}\Delta + \binom{t}{2}\Delta^2 + \cdots + \binom{t}{\nu}\Delta^\nu\right] y^0 \qquad (1.4.6)$$

is the unique polynomial of degree at most ν that interpolates the data $y^0, \ldots, y^\nu$ in the points $0, \ldots, \nu$.

Proof. First of all, from (1.4.5) it is clear that $\binom{t}{j}$ is a monomial of degree j, and since (1.4.6) describes a linear combination of such terms with $0 \leq j \leq \nu$, $q(t)$ is evidently a polynomial of degree at most ν.

We need to show that $q(t)$ interpolates $y^0, \ldots, y^\nu$. To this end, note that $Z = 1 + \Delta$, and therefore

$$Z^j = (1+\Delta)^j = 1 + \binom{j}{1}\Delta + \binom{j}{2}\Delta^2 + \cdots + + \binom{j}{j}\Delta^j$$

for any integer $j \geq 0$ (the binomial formula). If $0 \leq j \leq \nu$ we may equally well extend the series to term ν,

$$Z^j = 1 + \binom{j}{1}\Delta + \binom{j}{2}\Delta^2 + \cdots + \binom{j}{\nu}\Delta^\nu,$$

since $\binom{j}{m} = 0$ for $m > j$. By taking $t = j$ in (1.4.6), this identity implies that $q(j) = Z^j y^0$ for $0 \leq j \leq \nu$. In other words, $q(t)$ interpolates the data as required.

Finally, uniqueness of the interpolating polynomial is easy to prove. If $q_1(t)$ and $q_2(t)$ are two polynomials of degree $\leq \nu$ that interpolate the data, then $q_1 - q_2$ is polynomial of degree $\leq \nu$ that vanishes at the interpolation points, which implies $q_1 - q_2 = 0$ identically since a nonzero polynomial of degree $\leq \nu$ can have at most ν zeros. ∎

We want to apply Theorem 1.3 to the derivation of Adams-Bashforth formulas. To do this, we need a version of the theorem that is normalized differently and considerably uglier, though equivalent. Let the points $0, \ldots, \nu$ be replaced by the points $t_0, \ldots, t_{-\nu}$. Then from Theorem 1.3, or by a proof from scratch, one can readily show that the polynomial

$$q(t) = \left[1 - \binom{-t/k}{1}\nabla + \binom{-t/k}{2}\nabla^2 - \cdots + (-1)^\nu \binom{-t/k}{\nu}\nabla^\nu\right] y^0 \qquad (1.4.7)$$

is the unique polynomial of degree at most ν that interpolates the data $y^{-\nu},\dots,$ y^0 in the points $t_{-\nu},\dots,t_0$. Note that among other changes, Δ has been replaced by ∇.

Now let us replace y by f, ν by $s-1$, and t by $t-t_{n+s-1}$, hence $t_{-\nu},\dots,t_0$ by $t_n,\dots,t_{n+s-1}$. Equation (1.4.7) then becomes

$$q(t) = \left[1 - \binom{(t_{n+s-1}-t)/k}{1}\nabla + \cdots + (-1)^{s-1}\binom{(t_{n+s-1}-t)/k}{s-1}\nabla^{s-1}\right] f^{n+s-1}. \tag{1.4.8}$$

Inserting this expression in (1.4.2) gives

$$v^{n+s} - v^{n+s-1} = k\sum_{j=0}^{s-1}\gamma_j \nabla^j f^{n+s-1},$$

where

$$\gamma_j = \frac{(-1)^j}{k}\int_{t_{n+s-1}}^{t_{n+s}} \binom{(t_{n+s-1}-t)/k}{j} dt = (-1)^j \int_0^1 \binom{-\tau}{j} d\tau.$$

The first few values γ_j are

$$\begin{aligned} &\gamma_0 = 1, && \gamma_3 = \frac{3}{8}, && \gamma_6 = \frac{19087}{60480}, \\ &\gamma_1 = \frac{1}{2}, && \gamma_4 = \frac{251}{720}, && \gamma_7 = \frac{5257}{17280}, \\ &\gamma_2 = \frac{5}{12}, && \gamma_5 = \frac{95}{288}, && \gamma_8 = \frac{1070017}{3628800}. \end{aligned} \tag{1.4.9}$$

The following theorem summarizes this derivation and some related facts:

ADAMS-BASHFORTH FORMULAS

Theorem 1.4. *For any $s \geq 1$, the s-step Adams-Bashforth formula has order of accuracy s, and is given by*

$$v^{n+s} = v^{n+s-1} + k \sum_{j=0}^{s-1} \gamma_j \nabla^j f^{n+s-1}, \qquad \gamma_j = (-1)^j \int_0^1 \binom{-\tau}{j} d\tau. \quad (1.4.10)$$

For $j \geq 0$, the coefficients γ_j satisfy the recurrence relation

$$\gamma_j + \frac{1}{2}\gamma_{j-1} + \frac{1}{3}\gamma_{j-2} + \cdots + \frac{1}{j+1}\gamma_0 = 1. \quad (1.4.11)$$

Proof. We have already shown above that the coefficients (1.4.10) correspond to the linear multistep formula based on polynomial interpolation. To prove the theorem, we must show three things more: that the order of accuracy is as high as possible, so that these are indeed Adams-Bashforth formulas; that the order of accuracy is s; and that (1.4.11) holds.

The first claim follows from the lemma stated in Exercise 1.3.6. If $f(u,t)$ is a polynomial in t of degree $\leq s$, then $q(t) = f(u,t)$ in our derivation above, so the formula (1.4.2) gives exact results and its order of accuracy is accordingly $\geq s$. On the other hand any other linear multistep formula with different coefficients would fail to integrate q exactly, since polynomial interpolants are unique, and accordingly would have order of accuracy $< s$. Thus (1.4.10) is indeed the Adams-Bashforth formula.

The second claim can also be based on Exercise 1.3.6. If the s-step Adams-Bashforth formula had order of accuracy $> s$, it would be exact for any problem $u_t = q(t)$ with $q(t)$ equal to a polynomial of degree $s+1$. But there are nonzero polynomials of this degree that interpolate the values $f^0 = \cdots = f^s = 0$, from which we can readily derive counterexamples in the form of initial-value problems with $v(t_{s+1}) = 0$ but $u(t_{s+1}) \neq 0$.

Finally, for a derivation of (1.4.11), the reader is referred to Henrici (1962) or Hairer, Nørsett & Wanner (1987). ∎

EXAMPLE 1.4.1, CONTINUED. To rederive the 2nd-order Adams-Bashforth formula directly from (1.4.10), we calculate

$$v^{n+2} = v^{n+1} + k\gamma_0 f^{n+1} + k\gamma_1 (f^{n+1} - f^n) = v^{n+1} + k\left(\tfrac{3}{2} f^{n+1} - \tfrac{1}{2} f^n\right).$$

EXAMPLE 1.4.2. To obtain the third-order Adams-Bashforth formula, we increment n

to $n+1$ in the formula above and then add one more term $k\gamma_2\nabla^2 f^{n+2}$ to get

$$\begin{aligned} v^{n+3} &= v^{n+2} + k\left(\tfrac{3}{2}f^{n+2} - \tfrac{1}{2}f^{n+1}\right) + \tfrac{5}{12}k\left(f^{n+2} - 2f^{n+1} + f^n\right) \\ &= v^{n+2} + k\left(\tfrac{23}{12}f^{n+2} - \tfrac{16}{12}f^{n+1} + \tfrac{5}{12}f^n\right), \end{aligned}$$

which confirms the result listed in Table 1.4.1.

For Adams-Moulton formulas the derivation is entirely analogous. We have

$$v^{n+s} - v^{n+s-1} = k\sum_{j=0}^{s} \gamma_j^* \nabla^j f^{n+s},$$

where

$$\gamma_j^* = \frac{(-1)^j}{k}\int_{t_{n+s-1}}^{t_{n+s}} \binom{(t_{n+s}-t)/k}{j} dt = (-1)^j \int_{-1}^{0} \binom{-\tau}{j} d\tau,$$

and the first few values γ_j^* are

$$\begin{aligned} &\gamma_0^* = +1, && \gamma_3^* = -\frac{1}{24}, && \gamma_6^* = -\frac{863}{60480}, \\ &\gamma_1^* = -\frac{1}{2}, && \gamma_4^* = -\frac{19}{720}, && \gamma_7^* = -\frac{275}{24192}, \\ &\gamma_2^* = -\frac{1}{12}, && \gamma_5^* = -\frac{3}{160}, && \gamma_8^* = -\frac{33953}{3628800}. \end{aligned} \tag{1.4.12}$$

Notice that these numbers are smaller than before, an observation which is related to the fact that Adams-Moulton formulas generally have smaller error constants than the corresponding Adams-Bashforth formulas.

The analog of Theorem 1.4 is as follows:

ADAMS-MOULTON FORMULAS

Theorem 1.5. *For any $s \geq 0$, the s-step Adams-Moulton formula* has order of accuracy $s+1$, and is given by*

$$v^{n+s} = v^{n+s-1} + k\sum_{j=0}^{s} \gamma_j^* \nabla^j f^{n+s}, \qquad \gamma_j^* = (-1)^j \int_{-1}^{0} \binom{-\tau}{j} d\tau, \tag{1.4.13}$$

For $j \geq 1$, the coefficients γ_j^* satisfy the recurrence relation

$$\gamma_j^* + \frac{1}{2}\gamma_{j-1}^* + \frac{1}{3}\gamma_{j-2}^* + \cdots + \frac{1}{j+1}\gamma_0^* = 0. \tag{1.4.14}$$

* The "0-step Adams-Moulton formula" of this theorem actually has $s = 1$, as indicated in Table 1.4.2, because of the nonzero coefficient α_0.

Proof. Analogous to the proof of Theorem 1.4. ∎

Both sets of coefficients $\{\gamma_j\}$ and $\{\gamma_j^*\}$ can readily be converted into coefficients $\{\beta_j\}$ of our standard representation (1.2.11), and the results for $s \le 4$ are listed above in Tables 1.4.1 and 1.4.2.

Besides Adams formulas, the most important family of linear multistep formulas dates to Curtiss and Hirschfelder in 1952, and is also associated with the name of C. W. Gear. The s-step **backwards differentiation formula** is the optimal implicit linear multistep formula with $\beta_0 = \cdots = \beta_{s-1} = 0$. (An example was given in (1.2.9).) Unlike the Adams formulas, the backwards differentiation formulas allocate the free parameters to the $\{\alpha_j\}$ rather than the $\{\beta_j\}$. These formulas are "maximally implicit" in the sense that the function f enters the calculation only at the level $n+1$. For this reason they are the hardest to implement of all linear multistep formulas, but as we shall see in §§1.7–1.8, they are also the most stable.

To derive the coefficients of backwards differentiation formulas, one can again make use of polynomial interpolation. Now, however, the data are samples of v rather than of f, as suggested in Figure 1.4.2b. Let q be the unique polynomial of degree $\le s$ that interpolates $v^n, \ldots, v^{n+s}$. The number v^{n+s} is unknown, but it is natural to define it implicitly by imposing the condition

$$q_t(t_{n+s}) = f^{n+s}. \tag{1.4.15}$$

Like the integral in (1.4.2), the derivative in (1.4.15) represents a linear function of the data, with coefficients that can be computed once and for all, and so this equation constitutes an implicit definition of a linear multistep formula. The coefficients for $s \le 4$ were listed above in Table 1.4.3.

To convert this prescription into numerical coefficients, one can again apply the Newton interpolation formula (see Exercise 1.4.2). However, the proof below is a slicker one based on rational approximation and Theorem 1.2.

BACKWARDS DIFFERENTIATION FORMULAS

Theorem 1.6. *For any $s \ge 1$, the s-step backwards differentiation formula has order of accuracy s, and is given by*

$$\sum_{j=1}^{s} \frac{1}{j} \nabla^j v^{n+s} = k f^{n+s}. \tag{1.4.16}$$

(Note that (1.4.16) is not quite in the standard form (1.2.11); it must be normalized by dividing by the coefficient of v^{n+s}.)

Proof. Let $\rho(z)$ and $\sigma(z) = z^s$ be the characteristic polynomials corresponding to the s-step backwards differentiation formula. (Again we have normalized differently from usual.) By Theorem 1.2, since $\log z = -\log z^{-1}$, the order of accuracy is p if and only if

$$\begin{aligned}\frac{\rho(z)}{z^s} &= -\log z^{-1} + \Theta((z-1)^{p+1}) \\ &= -\left[(z^{-1}-1) - \tfrac{1}{2}(z^{-1}-1)^2 + \tfrac{1}{3}(z^{-1}-1)^3 - \cdots\right] + \Theta((z-1)^{p+1}),\end{aligned}$$

that is,

$$\rho(z) = z^s\left[(1-z^{-1}) + \tfrac{1}{2}(1-z^{-1})^2 + \tfrac{1}{3}(1-z^{-1})^3 + \cdots\right] + \Theta((z-1)^{p+1}).$$

By definition, $\rho(z)$ is a polynomial of degree at most s with $\rho(0) \neq 0$; equivalently, it is z^s times a polynomial in z^{-1} of degree exactly s. Since $\rho(z)$ maximizes the order of accuracy among all such polynomials, the last formula makes it clear that we must have

$$\rho(z) = z^s\left[(1-z^{-1}) + \cdots + \frac{1}{s}(1-z^{-1})^s\right],$$

with order of accuracy s. This is precisely (1.4.16). ∎

These three families of linear multistep formulas—Adams-Bashforth, Adams-Moulton, and backwards differentiation—are the most important for practical computations. Other families, however, have also been developed over the years. The s-step **Nyström** formula is the optimal explicit linear multistep formula with $\rho(z) = z^s - z^{s-2}$, that is, $\alpha_s = 1$, $\alpha_{s-2} = -1$, and $\alpha_j = 0$ otherwise. The s-step **generalized Milne-Simpson** formula is the optimal implicit linear multistep formula of the same type. Coefficients for these formulas can be obtained by the same process described in Figure 1.4.2a, except that now $q(t)$ is integrated from t_{n+s-2} to t_{n+s}. Like the Adams and backwards differentiation formulas, the s-step Nyström and generalized Milne-Simpson formulas have exactly the order of accuracy one would expect from the number of free parameters (s and $s+1$, respectively), with one exception: the generalized Milne-Simpson formula with $s = 2$ has order 4, not 3. This formula is known as the **Simpson** formula for ordinary differential equations:

$$v^{n+2} = v^n + \frac{1}{3}k(f^n + 4f^{n+1} + f^{n+2}). \tag{1.4.17}$$

EXERCISES

▷ *1.4.1. Second-order backwards differentiation formula.* Derive the coefficients of the 2-step backwards differentiation formula in Table 1.4.3:
(a) By the method of undetermined coefficients;
(b) By Theorem 1.2, making use of the expansion $z^2 = (z-1)^2 + 2(z-1) + 1$;
(c) By interpolation, making use of Theorem 1.3.

▷ *1.4.2. Backwards differentiation formulas.* Derive (1.4.16) from Theorem 1.3.

▷ *1.4.3. Third-order Nyström formula.* Determine the coefficients of the third-order Nyström formula by interpolation, making use of Theorem 1.3.

▷ *1.4.4. Quadrature formulas.* What happens to linear multistep formulas when the function $f(u,t)$ is independent of u? To be specific, what happens to Simpson's formula (1.4.17)? Comment on the effect of various strategies for initializing the point v^1 in an integration of such a problem by Simpson's formula.

1.5. Stability

It is time to introduce one of the central themes of this book: stability. Before 1950, the word stability rarely if ever appeared in papers on numerical methods, but by 1960, at which time computers were widely distributed, its importance had become universally recognized.* Problems of stability affect almost every numerical method for solving differential equations, and they must be confronted head-on.

For both ordinary and partial differential equations, there are two main stability questions that have proved important over the years:

Stability:† If $t > 0$ is held fixed, do the computed values $v(t)$ remain bounded as $k \to 0$?

Eigenvalue stability: If $k > 0$ is held fixed, do the computed values $v(t)$ remain bounded as $t \to \infty$?

These two questions are related, but distinct, and each has important applications. We shall consider the first in this and the following section, and the second in §§1.7,1.8.

The motivation for all discussions of stability is the most fundamental question one could ask about a numerical method: will it give the right answer? Of course one can never expect exact results, so a reasonable way to make the question precise for the initial-value problem (1.1.2) is to ask: if $t > 0$ is a fixed number, and the computation is performed with various step sizes $k > 0$ in exact arithmetic, will $v(t)$ converge to $u(t)$ as $k \to 0$?

A natural conjecture might be that for any consistent linear multistep formula, the answer must be yes. After all, as pointed out in §1.3, such a method commits local errors of size $O(k^{p+1})$ with $p \geq 1$, and there are a total of $\Theta(k^{-1})$ time steps. But a simple argument shows that this conjecture is false. Consider a linear multistep formula based purely on extrapolation of previous values $\{v^n\}$, such as

$$v^{n+2} = 2v^{n+1} - v^n. \tag{1.5.1}$$

This is a first-order formula with $\rho(z) = (z-1)^2$ and $\sigma(z) = 0$, and in fact we can construct extrapolation formulas of arbitrary order of accuracy by taking $\rho(z) = (z-1)^s$. Yet such a formula cannot possibly converge to the correct solution, for it uses no information from the differential equation!*

Thus local accuracy cannot be sufficient for convergence. The surprising fact is that accuracy plus an additional condition of stability *is* sufficient, and necessary too. In fact this is a rather general principle of numerical analysis, first formulated precisely by Dahlquist for ordinary differential equations and by Lax and Richtmyer for partial differential equations, both in the 1950's. Strang (1985) calls it the "fundamental theorem of numerical analysis."

---

*See G. Dahlquist, "33 years of numerical instability," *BIT 25 (1985)*, 188–204.

†Stability is also known as "zero-stability" or sometimes "D-stability" for ODEs, and "Lax stability" or "Lax-Richtmyer stability" for PDEs. Eigenvalue stability is also known as "weak stability" or "absolute stability" for ODEs, and "time-stability," "practical stability" or "P-stability" for PDEs. The reason for the word "eigenvalue" will become apparent in §§1.7,1.8.

*Where exactly did the argument suggesting global errors $O(k^p)$ break down? Try to pinpoint it yourself; the answer is in the next section.

Theorem 1.10 in the next section gives a precise statement for the case of linear multistep formulas, and for partial differential equations, see Theorem 4.1.

We begin with a numerical experiment.

EXAMPLE 1.5.1. In the spirit of Example 1.2.1, suppose we solve the initial-value problem

$$u_t = u, \qquad t \in [0,1], \qquad u(0) = 1 \tag{1.5.2}$$

by three different 2-step explicit methods: the extrapolation formula (1.5.1), the second-order Adams-Bashforth formula (1.4.3), and the "optimal" 2-step formula (1.3.21). Again we take exact quantities e^{nk} where needed for starting values. Figure 1.5.1 shows the computed functions $v(t)$ for $k = 0.2$ and 0.1. In the first plot, both of the higher-order formulas appear to be performing satisfactorily. In the second, however, large oscillations have begun to develop in the solution based on (1.3.21). Obviously (1.3.21) is useless for this problem.

Table 1.5.1 makes this behavior quantitative by listing the computed results $v(1)$ for $k = 0.2, 0.1, 0.05, 0.025$. As k decreases, the solution based on (1.5.1) converges to an incorrect solution, namely the function $t+1$, and the solution based on (1.3.21) diverges explosively. Notice that none of the numbers are preceded by a *. In this example the instability has been excited by discretization errors, not rounding errors.

Figure 1.5.2 gives a fuller picture of what is going on in this example by plotting the error $|v(1)-e|$ as a function of k (on a log scale, with smaller values of k to the right) for the same three linear multistep formulas as well as the fourth-order Adams-Bashforth formula (1.2.7). The two Adams-Bashforth formulas exhibit clean second-order and fourth-order convergence, as one would expect, showing in the plot as lines of slope approximately -2 and -4, respectively. The extrapolation formula (1.5.1) exhibits zeroth-order convergence—in other words divergence, with the error approaching the constant $e-2$. The formula (1.3.21) diverges explosively.

It is not difficult to see what has gone wrong with (1.3.21). For simplicity, assume k is negligible. Then (1.3.21) becomes

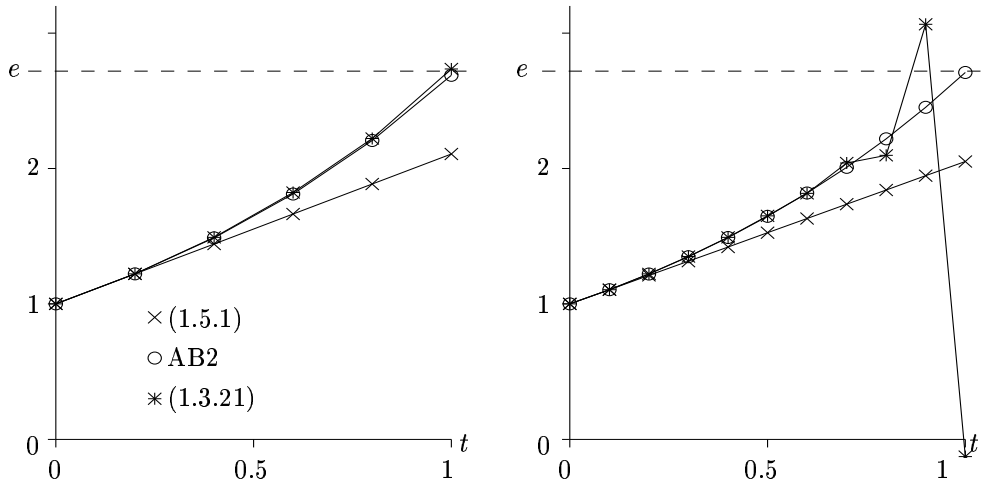
$$v^{n+2} + 4v^{n+1} - 5v^n = 0, \tag{1.5.3}$$

a second-order recurrence relation for $\{v^n\}$. It is easy to verify that both $v^n = 1$ and $v^n = (-5)^n$ are solutions of (1.5.3). (*Caution:* the n in v^n is a superscript, but the n in $(-5)^n$ is an exponent.) Since an arbitrary solution to (1.5.3) is determined by two initial values v^0 and v^1, it follows that any solution can be written in the form

$$v^n = a(1)^n + b(-5)^n \tag{1.5.4}$$

for some constants a and b. What has happened in our experiment is that b has ended up nonzero—there is some energy in the mode $(-5)^n$, and an explosion has taken place. In fact, if the computation with $k = 0.025$ is carried out to one more time step, v^n takes the value 4.60×10^{19}, which is $-4.93 \approx -5$ times the final quantity listed in the table. So the assumption that k was negligible was not too bad.

Although Example 1.5.1 is very simple, it exhibits the essential mechanism of instability in the numerical solution of ordinary differential equations by linear multistep formulas: a recurrence relation that admits an exponentially



(a) $k = 0.2$ (b) $k = 0.1$

Figure 1.5.1. Solution of (1.5.2) by three explicit two-step linear multistep formulas. The high-order formula (1.3.21) looks good at first, but becomes unstable when the time step is halved.

	$k = 0.2$	$k = 0.1$	$k = 0.05$	$k = 0.025$
Extrapolation (1.5.1)	2.10701	2.05171	2.02542	2.01260
2nd-order A-B (1.4.4)	2.68771	2.70881	2.71568	2.71760
"optimal" (1.3.21)	2.73433	−0.12720	-1.62×10^{6}	-9.34×10^{18}

Table 1.5.1. Computed values $v(1) \approx e$ for the initial-value problem (1.5.2).

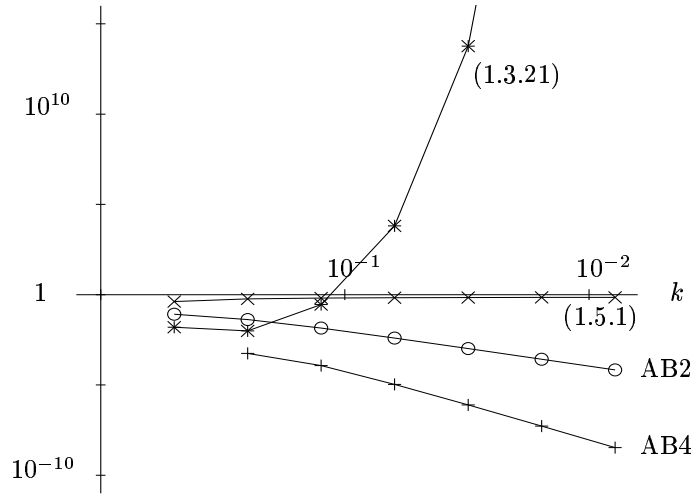


Figure 1.5.2. Error $|v(1) - e|$ as a function of time step k for the solution of (1.5.2) by four explicit linear multistep formulas.

growing solution z^n for some $|z| > 1$. Such a solution is sometimes known as a **parasitic solution** of the numerical method, since it is introduced by the discretization rather than the ordinary differential equation itself. It is a general principle that if such a mode exists, it will almost always be excited by either discretization or rounding errors, or by errors in the initial conditions. Of course in principle, the coefficient b in (1.5.4) might turn out to be identically zero, but in practice this possibility can usually be ignored. Even if b were zero at $t = 0$, it would soon become nonzero due to variable coefficients, nonlinearity, or rounding errors, and the z^n growth would then take over sooner or later.*

The analysis of Example 1.5.1 can be generalized as follows. Given any linear multistep formula, consider the associated recurrence relation

$$\rho(Z)v^n = \sum_{j=0}^{s} \alpha_j v^{n+j} = 0 \tag{1.5.5}$$

obtained from (1.3.4) with $k = 0$. We define:

A linear multistep formula is **stable** *if all solutions $\{v^n\}$ of the recurrence relation (1.5.5) are bounded as $n \to \infty$.*

This means that for any function $\{v^n\}$ that satisfies (1.5.5), there exists a constant $M > 0$ such that $|v^n| \le M$ for all $n \ge 0$. We shall refer to the recurrence relation itself as **stable**, as well as the linear multistep formula.

There is an elementary criterion for determining stability of a linear multistep formula, based on the characteristic polynomial ρ.

ROOT CONDITION FOR STABILITY

Theorem 1.7. *A linear multistep formula is stable if and only if all the roots of $\rho(z)$ satisfy $|z| \le 1$, and any root with $|z| = 1$ is simple.*

A "simple" root is a root of multiplicity 1.

First proof. To prove this theorem, we need to investigate all possible solutions $\{v^n\}$, $n \ge 0$, of the recurrence relation (1.5.5). Since any such solution is determined by its initial values $v^0, \ldots, v^{s-1}$, the set of all of them is a

*A general observation is that when any linear process admits an exponentially growing solution in theory, that growth will almost invariably appear in practice. Only in nonlinear situations can the existence of exponentially growing modes fail to make itself felt. A somewhat far-flung example comes from numerical linear algebra. In the solution of an $N \times N$ matrix problems by Gaussian elimination with partial pivoting—a highly nonlinear process—an explosion of rounding errors at the rate 2^{N-1} can occur in principle, but almost never does in practice (Trefethen & Schreiber, "Average-case stability of Gaussian elimination," *SIAM J. Matrix Anal. Applics.*, 1990).

vector space of dimension s. Therefore if we can find s linearly independent solutions v^n, these will form a basis of the space of all possible solutions, and the recurrence relation will be stable if and only if each of the basis solutions is bounded as $n \to \infty$.

It is easy to verify that if z is any root of $\rho(z)$, then

$$v^n = z^n \tag{1.5.6}$$

is one solution of (1.5.5). (Again, on the left n is a superscript and on the right it is an exponent. If $z = 0$ we define $z^0 = 1$.) If ρ has s distinct roots, then these functions constitute a basis. Since each function (1.5.6) is bounded if and only if $|z| \le 1$, this proves the theorem in the case of distinct roots.

On the other hand suppose that $\rho(z)$ has a root z of multiplicity $m \ge 2$. Then it can readily be verified that each of the functions

$$v^n = nz^n, \qquad v^n = n^2 z^n, \quad \ldots, \quad v^n = n^{m-1} z^n \tag{1.5.7}$$

is an additional solution of (1.5.5), and clearly they are all linearly independent since degree-$(m-1)$ polynomial interpolants in m points are unique, to say nothing of ∞ points! (If $z = 0$, we replace $n^j z^n$ by the function that takes the value 1 at $n = j$ and 0 elsewhere.) These functions are bounded if and only if $|z| < 1$, and this finishes the proof of the theorem in the general case. ∎

Alternative proof based on linear algebra. The proof above is simple and complete, but there is another way of looking at Theorem 1.7 that involves a technique of general importance. Let us rewrite the s-step recurrence relation as a 1-step matrix operation on vectors $\mathbf{v}$ of length s:

$$\begin{pmatrix} v^{n+1} \\ v^{n+2} \\ \vdots \\ v^{n+s} \end{pmatrix} = \begin{pmatrix} 0 & 1 & & & \\ & 0 & 1 & & \\ & & \ddots & \ddots & \\ & & & 0 & 1 \\ -\alpha_0 & & \cdots & -\alpha_{s-2} & -\alpha_{s-1} \end{pmatrix} \begin{pmatrix} v^n \\ v^{n+1} \\ \vdots \\ v^{n+s-1} \end{pmatrix}. \tag{1.5.8}$$

That is,

$$\mathbf{v}^{n+1} = A\mathbf{v}^n, \tag{1.5.9}$$

or after n steps,

$$\mathbf{v}^n = A^n \mathbf{v}^0, \tag{1.5.10}$$

where A^n denotes the nth power of the matrix A. This is a discrete analog of the reduction of higher-order ODEs to first-order systems described in §1.1.

The scalar sequence $\{v^n\}$ will be bounded as $n \to \infty$ if and only if the vector sequence $\{\mathbf{v}^n\}$ is bounded, and $\{\mathbf{v}^n\}$ will in turn be bounded if and only if the elements of A^n are bounded. Thus we have reduced stability to a problem of growth or boundedness of the powers of a matrix.

A matrix A of the form (1.5.8) is known as a **companion matrix**, and one can verify that $\det(zI - A) = \rho(z)$ for any z, where $\rho(z)$ is defined by (1.3.1) as usual.* In other words, the characteristic polynomial of the matrix A is the same as the characteristic polynomial of the linear multistep formula. Therefore the set of eigenvalues of A is the same as the set of roots of ρ, and these eigenvalues determine how the powers A^n behave asymptotically as $n \to \infty$. To make the connection precise one can look at the similarity transformation that brings A into **Jordan canonical form**,

$$A = SJS^{-1}.$$

Here S is an $s \times s$ nonsingular matrix, and J is an $s \times s$ matrix consisting of all zeros except for a set of **Jordan blocks** J_i along the diagonal with the form

$$J_i = \begin{pmatrix} z_i & 1 & & & \\ & z_i & 1 & & \\ & & \ddots & \ddots & \\ & & & z_i & 1 \\ & & & & z_i \end{pmatrix},$$

where z_i is one of the eigenvalues of A. Every matrix has a Jordan canonical form, and for matrices in general, each eigenvalue may appear in several Jordan blocks. For a companion matrix, however, there is exactly one Jordan block for each eigenvalue, with dimension equal to the multiplicity m_i of that eigenvalue. (Proof: $zI - A$ has rank $\geq s-1$ for any z, since its upper-right $(s-1) \times (s-1)$ block is obviously nonsingular. Such a matrix is called **nonderogatory**.) Now the powers of A are

$$A^n = (SJS^{-1}) \cdots (SJS^{-1}) = SJ^nS^{-1},$$

so their growth or boundedness is determined by the growth or boundedness

* One way to verify it is to look for eigenvectors of the form $(1, z, \ldots, z^{s-1})^T$, where z is the eigenvalue.

of the powers J^n, which can be written down explicitly:

$$J_i^n = \begin{pmatrix} z_i^n & \binom{n}{1} z_i^{n-1} & & \cdots & \binom{n}{m_i-1} z_i^{n+1-m_i} \\ & z_i^n & \binom{n}{1} z_i^{n-1} & & \\ & & \ddots & \ddots & \vdots \\ & & & z_i^n & \binom{n}{1} z_i^{n-1} \\ & \mathbf{0} & & & z_i^n \end{pmatrix}.$$

If $|z_i| < 1$, these elements approach 0 as $n \to \infty$. If $|z_i| > 1$, they approach ∞. If $|z_i| = 1$, they are bounded in the case of a 1×1 block, but unbounded if $m_i \geq 2$. ∎

The reader should study Theorem 1.7 and both of its proofs until he or she is quite comfortable with them. These tricks for analyzing recurrence relations come up so often that they are worth remembering.

EXAMPLE 1.5.2. Let us test the stability of various linear multistep formulas considered up to this point. Any Adams-Bashforth or Adams-Moulton formula has $\rho(z) = z^s - z^{s-1}$, with roots $\{1, 0, \ldots, 0\}$, so these methods are stable. The Nyström and generalized Milne-Simpson formulas have $\rho(z) = z^s - z^{s-2}$, with roots $\{1, -1, 0, \ldots, 0\}$, so they are stable too. The scheme (1.3.21) that caused trouble in Example 1.5.1 has $\rho(z) = z^2 + 4z - 5$, with roots $\{1, -5\}$, so it is certainly unstable. As for the less dramatically unsuccessful formula (1.5.1), it has $\rho(z) = z^2 - 2z + 1 = (z-1)^2$, with a multiple root $\{1, 1\}$, so it counts as unstable too since it admits the growing solution $v^n = n$. The higher-order extrapolation formula defined by $\rho(z) = (z-1)^s$, $\sigma(z) = 0$ admits the additional solutions $n^2, n^3, \ldots, n^{s-1}$, making for an instability more pronounced but still algebraic rather than exponential.

Note that by Theorem 1.2, any consistent linear multistep formula has a root of $\rho(z)$ at $z = 1$, and if the formula is stable, then Theorem 1.7 ensures that this root is simple. It is called the **principal root** of ρ, for it is the one that tracks the differential equation. The additional consistency condition $\rho'(1) = \sigma(1)$ of Theorem 1.2 amounts to the condition that if z is perturbed away from 1, the principal root behaves correctly to leading order.

In §1.3 an analogy was mentioned between linear multistep formulas and recursive digital filters. In digital signal processing, one demands $|z| < 1$ for stability; why then does Theorem 1.7 contain the weaker condition $|z| \leq 1$? The answer is that unlike the usual filters, an ODE integrator must remember the past: even for values of t where f is zero, u is in general nonzero. If all roots z satisfied $|z| < 1$, the influence of the past would die away exponentially.

Theorem 1.7 leaves us in a remarkable situation, summarized in Figure 1.5.3. By Theorem 1.2, consistency is the condition that $\rho(z)/\sigma(z)$ matches $\log z$ to at least second order at $z=1$. By Theorem 1.7, stability is the condition that all roots of $\rho(z)$ lie in $|z| \le 1$, with simple roots only permitted on $|z|=1$. Thus the two crucial properties of linear multistep formulas have been reduced completely to algebraic questions concerning a rational function. The proofs of many results in the theory of linear multistep formulas, such as Theorems 1.8 and 1.9 below, consist of arguments of pure complex analysis of rational functions, having nothing superficially to do with ordinary differential equations.

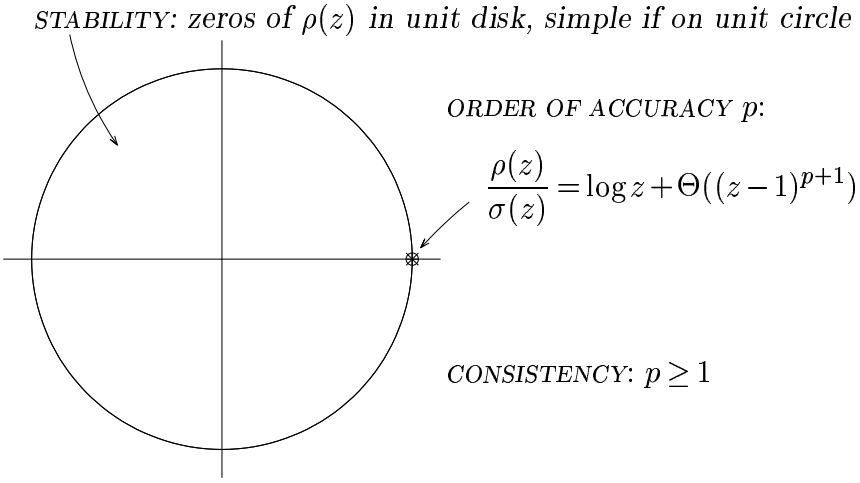


Figure 1.5.3. Stability, consistency, and order of accuracy as algebraic conditions on the rational function $\rho(z)/\sigma(z)$.

The following theorem summarizes the stability of the standard families of linear multistep formula that we have discussed.

STABILITY OF STANDARD LINEAR MULTISTEP FORMULAS
Theorem 1.8. *The s-step Adams-Bashforth, Adams-Moulton, Nyström, and generalized Milne-Simpson formulas are stable for all $s \ge 1$. The s-step backwards differentiation formulas are stable for $1 \le s \le 6$, but unstable for $s \ge 7$.*

Proof. The results for the first four families of linear multistep formulas follow easily from Theorem 1.7, as described in Example 1.5.2. The analysis of backwards differentiation formulas is a more complicated matter, since

the polynomials $\rho(z)$ are no longer trivial. Instability for $s \geq 7$ was recognized numerically in the 1950's (Mitchell and Craggs, 1953), but not proved mathematically until 1971 (Cryer, 1972). An elegant short proof of instability for $s \geq 12$ was devised by Hairer and Wanner in 1983 and can be found on pp. 328–331 of (Hairer, Nørsett & Wanner, 1987). ∎

To close this section we shall present an important result that is known as the **first Dahlquist stability barrier**. In §1.3 we mentioned that an s-step linear multistep formula can have order $2s$; why not simply use that high-order formula and forget special classes like Adams and Nyström methods? Dahlquist's famous theorem confirms that the answer is an impassable barrier of stability.

FIRST DAHLQUIST STABILITY BARRIER

Theorem 1.9. *The order of accuracy p of a stable s-step linear multistep formula satisfies*

$$p \leq \begin{cases} s+2 & \text{if } s \text{ is even,} \\ s+1 & \text{if } s \text{ is odd,} \\ s & \text{if the formula is explicit.} \end{cases} \tag{1.5.11}$$

Proof. Various proofs of Theorem 1.9 have been published, beginning with the original one by Dahlquist in 1956. More recent proofs have been based on the beautiful idea of "order stars" introduced by Wanner, Hairer, and Nørsett (*BIT*, 1978). The following argument is adapted from "A proof of the first Dahlquist barrier by order stars," by A. Iserles and S. P. Nørsett, *BIT*, 1984. Though fundamentally correct, it is somewhat casual about geometric details; for a more detailed treatment see that paper or the book *Order Stars* by the same authors.

Let $\rho(z)$ and $\sigma(z)$ be the characteristic polynomials corresponding to a stable s-step linear multistep formula of order of accuracy p. The key idea is to look at level lines of the function we have been dealing with since (1.3.18),

$$\varphi(z) = \frac{\rho(z)}{\sigma(z)} - \log z. \tag{1.5.12}$$

This function is analytic throughout the complex plane except near zeros of $\sigma(z)$, provided one introduces a branch cut going to the point $z = 0$. (Alternatively, one can work with $e^{\varphi(z)}$ and eliminate the need for a branch cut.) In particular, $\varphi(z)$ is analytic at $z = 1$. (If $\sigma(1) = 0$, $\rho(z)$ and $\sigma(z)$ have a

Figure 1.5.4. Order star defined by (1.5.14) for the 5th-order Adams-Bashforth formula. The zeros of $\rho(z)$ (at $z=0$) are marked by o and the zeros of $\sigma(z)$ are marked by $*$. The 6-fold daisy at $z=1$ reflects the order of accuracy 5 of this formula. If there were a bounded, shaded "finger" that did not intersect the unit disk (dashed), the formula would be unstable.

common factor $z-1$ and the linear multistep formula is not in simplest form.) By Theorem 1.2, its behavior near there is

$$\varphi(z) = C(z-1)^{p+1} + O((z-1)^{p+2}) \tag{1.5.13}$$

for $C = C_{p+1}/\sigma(1) \neq 0$. Now let A, the **order star** for the linear multistep formula, be the set defined by

$$A = \{z \in \mathbb{C}\colon \operatorname{Re}\varphi(z) > 0\}. \tag{1.5.14}$$

In other words, it is the inverse image of the right half-plane under φ. Figure 1.5.4 shows the order star for the 5th-order Adams-Bashforth formula.

The power of order stars comes from their ability to couple local and global properties of a function by way of geometric arguments. The global

property that concerns us is stability: the zeros of $\rho(z)$ must lie in the unit disk. The local property that concerns us is order of accuracy: from (1.5.13) it follows that near $z=1$, A must look like a daisy with $p+1$ evenly spaced petals—or "fingers", as they are sometimes called. In Figure 1.5.4 the number of fingers is 6.

Since the linear multistep formula is stable, all the zeros of $\rho(z)$ must lie in A (or possibly on its boundary in the case of a zero with $|z|=1$). This follows from (1.5.12) and (1.5.14) since the zeros have to satisfy $|z|\le 1$, hence $\mathrm{Re}\log z \le 0$. [At this point some reasoning by the argument principle of complex analysis is needed, which will be filled in later.] The conclusion is this: any bounded finger of A (i.e., not extending to ∞) must contain one of the zeros of $\rho(z)$. Consequence: *for stability, every bounded finger has to intersect the unit disk.*

To finish the argument, let us now assume that the linear multistep formula is explicit; similar arguments apply in the implicit case (Exercise 1.5.7). Our goal is then to prove $p \le s$. To do this we shall count zeros of $\mathrm{Re}\,\varphi(z)$ on the unit circle $|z|=1$. Let M be this number of zeros, counted with multiplicity; obviously M must be even. How big can M be? Note first that

$$2\,\mathrm{Re}\,\varphi(z) = \frac{\rho(z)}{\sigma(z)} + \frac{\overline{\rho(z)}}{\overline{\sigma(z)}} = \frac{\rho(z)\overline{\sigma(z)} + \overline{\rho(z)}\sigma(z)}{\sigma(z)\overline{\sigma(z)}}$$

on the unit circle $|z|=1$, since $\mathrm{Re}\log z = 0$ there. Since the linear multistep formula is explicit, $\sigma(z)$ has degree $\le s-1$, so the numerator is a polynomial of degree $\le 2s-1$, which implies that M, being even, satisfies

$$M \le 2s-2. \tag{1.5.15}$$

Now, how many zeros are implied by order of accuracy p? First, there is a zero of multiplicity $p+1$ at $z=1$. In addition, there is a zero wherever the boundary of A crosses the unit circle—in Figure 1.5.4, four such zeros altogether. To count these, we note that $(p+1)/2$ fingers of A begin at $z=1$ outside the unit circle. One of these may be unbounded and go to infinity, but the other $(p-1)/2$ must intersect the unit disk and cross the unit circle on the way. Two of those may cross the unit circle just once (one each in the upper and lower half-planes), but all the remaining $(p-5)/2$ fingers are trapped inside those fingers and must cross the unit circle twice. All told, we count

$$M \ge p+1+2+2(p-5)/2 = 2p-2. \tag{1.5.16}$$

Combining (1.5.15) and (1.5.16) gives $p \le s$, as required.

Similar arguments apply if the linear multistep formula is implicit. ∎

The restrictions of Theorem 1.9 are tight! In effect, they show, half of the available parameters in a linear multistep formula are wasted.

Theorems 1.3, 1.8, and 1.9 imply that the Adams-Bashforth and Nyström formulas are all optimal in the sense that they attain the bound $p = s$ for stable explicit formulas, and the Adams-Moulton and generalized Milne-Simpson formulas of odd step number s are also optimal, with $p = s+1$. Simpson's rule, with $p = s+2$, is an example of an optimal implicit formula with even step number. It can be shown that for any optimal implicit formula with even step number s, the roots of $\rho(z)$ all lie on the unit circle $|z| = 1$ (Henrici, 1962, p. 232). Thus the stability of these formulas is always of a borderline kind.

EXERCISES

▷ *1.5.1. Backwards differentiation formulas.* Show that the following backwards differentiation formulas from Table 1.4.3 are stable: *(a)* $s = 2$, *(b)* $s = 3$.

▷ *1.5.2.* Prove:

(a) Any consistent 1-step linear multistep formula is stable.

(b) Any consistent linear multistep formula with $\rho(z) = \sigma(z)$ is unstable.

▷ *1.5.3.* Consider the s-step explicit linear multistep formula of optimal order of accuracy of the form $v^{n+s} = v^n + k\sum_{j=0}^{s-1} \beta_j f^{n+j}$.

(a) For which s is it stable?

(b) Derive the coefficients for the case $s = 2$.

(c) Likewise for $s = 3$.

▷ *1.5.4. Padé approximation.* The *type* (μ, ν) *Padé approximant* to $\log z$ at $z = 1$ is defined as the unique rational function $\rho(z)/\sigma(z)$ of type (μ, ν) (i.e., with numerator of degree $\le \mu$ and denominator of degree $\le \nu$) that satisfies

$$\frac{\rho(z)}{\sigma(z)} = \log z + O((z-1)^{\mu+\nu+1}) \qquad \text{as } z \to 1. \tag{1.5.17}$$

By Theorem 1.2, taking $\mu = \nu = s$ gives the maximally accurate implicit s-step formula, with order of accuracy at least $p = 2s$, and taking $\mu = s$, $\nu = s-1$ gives the maximally accurate explicit s-step formula, with order of accuracy at least $p = 2s-1$.

Without performing any calculations, but appealing only to the uniqueness of Padé approximants and to theorems stated in this text, determine whether each of the following Padé schemes is stable or unstable, and what it is called if we have given a name for it. In each case, state exactly what theorems you have used.

(a) $s = 1$, explicit. *(b)* $s = 1$, implicit.

(c) $s = 2$, explicit. *(d)* $s = 2$, implicit.

(e) $s = 3$, explicit. *(f)* $s = 3$, implicit.

(g) If you have access to a symbolic calculator such as Macsyma, Maple, or Mathematica, use it to calculate the coefficients of the linear multistep formulas *(a)–(f)*, and confirm that the names you have identified above are correct.

▷ *1.5.5. Linear combinations of linear multistep formulas.* In Example 1.3.1 we showed that the trapezoid formula has error constant $-\frac{1}{12}$ and the midpoint formula has error constant $\frac{1}{3}$.

(a) Devise a linear combination of these two formulas that has order of accuracy higher than 2. What is the order of accuracy?

(b) Show that the formula of *(a)* is stable.

(c) In general, is a convex linear combination of two stable linear multistep formulas always stable? (A convex linear combination has the form $a \times formula_1 + (1-a) \times formula_2$ with $0 \le a \le 1$.) Prove it or give a counterexample.

▶ *1.5.6. Order stars.* Write a program to generate order star plots like that of Figure 1.5.4. This may be reasonably easy in a higher-level language like Matlab. Plot order stars for the following linear multistep formulas, among others:

(a) 4th-order Adams-Bashforth;

(b) 4th-order Adams-Moulton;

(c) 4th-order backwards differentiation;

(d) the unstable formula (1.3.21).

▷ *1.5.7.* The proof of Theorem 1.9 in the text covered only the case of an explicit linear multistep formula. Show what modifications are necessary for the implicit case.

1.6. Convergence and the Dahlquist equivalence theorem

Up to this point we have talked about accuracy, consistency, and stability, but we have yet to establish that a linear multistep formula with these admirable properties will actually work. After one more definition we shall be able to remedy that. To set the stage, let us return to the footnote on p. 41. If a linear multistep formula has local accuracy $O(k^{p+1})$, how can it fail to have global accuracy $O(k^p)$? The answer has nothing to do with the fact that f may be nonlinear, for we have assumed that f is Lipschitz continuous, and that is enough to keep the behavior of small perturbations essentially linear, so long as they remain small.

The flaw in the $O(k^p)$ argument is as follows. Even though a discretization error may be small when it is first introduced, *from that point on it may grow*—often exponentially. The global error at step n consists of the superposition not simply of the local errors at all previous steps, but of what these local errors *have become* at step n. Consistency is the condition that the local errors are small at the time that they are introduced, provided that the function being dealt with is smooth. The additional condition of stability is needed to make sure that they do not become bigger as the calculation proceeds.

The purpose of this section is to describe the ideas related to the Dahlquist Equivalence Theorem that have been built to describe these phenomena. The rigorous statement of the argument above appears in the proof of Theorem 1.10, below. If f is linear, the argument becomes simpler; see Exercise 1.6.3.

To begin we must define the notion of **convergence**. The standard definition requires the linear multistep formula to be applicable not just to a particular initial-value problem, but to an arbitrary initial-value problem with Lipschitz continuous data f. It must also work for any starting values $v^0,\dots,v^{s-1}$ that satisfy the consistency condition*

$$\|v^n - u_0\| = o(1) \quad \text{as } k \to 0, \qquad 0 \le n \le s-1. \tag{1.6.1}$$

Equivalent statements of the same condition would be

$$\lim \|v^n - u_0\| = 0, \qquad \lim v^n = u_0, \qquad \text{or} \quad v^n = u_0 + o(1)$$

as $k \to 0$ for $0 \le n \le s-1$ (see Exercise α.3(b)).

A linear multistep formula is **convergent** *if, for all initial-value problems (1.1.2) satisfying the conditions of Theorem 1.1 on an interval $[0,T]$, and all starting values $v^0,\dots,v^{s-1}$ satisfying (1.6.1), the solution v^n satisfies*

$$\|v(t) - u(t)\| = o(1) \qquad \text{as } k \to 0 \tag{1.6.2}$$

uniformly for all $t \in [0,T]$.

*The choice of the norm $\|\cdot\|$ doesn't matter, since all norms on a finite-dimensional space are equivalent. For a scalar ODE the norm can be replaced by the absolute value.

("Uniformly" means that $\|v(t)-u(t)\|$ is bounded by a fixed function $\phi(k)=o(1)$ as $k\to 0$, independent of t.) To put it in words, a convergent linear multistep formula is one that is guaranteed to get the right answer in the limit $k\to 0$, for each t in a bounded interval $[0,T]$—assuming there are no rounding errors.

Some remarks on this definition are in order. First, (1.6.2) is a statement about the limit $k\to 0$ for each *fixed* value of t. Since $v(t)$ is defined only on a discrete grid, this means that k is implicitly restricted to values t/n in this limit process. However, it is possible to loosen this restriction by requiring only $t_{n_k}\to t$ as $k\to 0$.

Second, to be convergent a linear multistep formula must work for *all* well-posed initial-value problems, not just one. This may seem an unnaturally strict requirement, but it is not. A formula that worked for only a restricted class of initial-value problems—for example, those with sufficiently smooth coefficients—would be a fragile object, sensitive to perturbations.

Third, v^n refers to the exact solution of the linear multistep formula; rounding errors are not accounted for in the definition of convergence. This is a reasonable simplification because discretization errors and errors in the starting values *are* included, via (1.6.1), and the stability phenomena that govern these various sources of error are nearly the same. Alternatively, the theory can be broadened to include rounding errors explicitly.

Finally, note that condition (1.6.1) is quite weak, requiring only $o(1)$ accuracy of starting values, or $O(k)$ if the errors happen to follow an integral power of k. This is in contrast to the $O(k^2)$ accuracy required at subsequent time steps by the definition of consistency. The reason for this discrepancy is simple enough: starting errors are introduced only at s time steps, while subsequent errors are introduced $\Theta(k^{-1})$ times. Therefore one can get away with one order lower accuracy in starting values than in the time integration. For partial differential equations we shall find that analogously, one can usually get away with one order lower accuracy in discretizing boundary conditions.

We come now to a remarkable result that might be called the fundamental theorem of linear multistep formulas. Like much of the material in this and the last section, it first appeared in a classic paper by Germund Dahlquist in 1956.*

DAHLQUIST EQUIVALENCE THEOREM

Theorem 1.10. *A linear multistep formula is convergent if and only if it is consistent and stable.*

Before proving this theorem, let us pause for a moment to consider what it says. It seems obvious that a linear multistep formula that admits unstable solutions is unlikely to be useful, but it is not so obvious that instability is the *only* thing that can go wrong. For example, clearly the extrapolation formula (1.5.1) of Example 1.5.1 must be useless, since it ignores $\{f^n\}$, but why should that have anything to do with instability? Yet Theorem 1.10 asserts that any useless consistent formula must be unstable! And indeed, we have verified this prediction for (1.5.1) in Example 1.5.2.

* G. Dahlquist, "Convergence and stability in the numerical integration of ordinary differential equations," *Math. Scand. 4* (1956), 33–53. There are important earlier references in this area, too, notably an influential article by Richard von Mises in 1930, who proved convergence of Adams methods. The classic reference in book form on this material is P. Henrici, *Discrete Variable Methods in Ordinary Differential Equations*, Wiley, 1962. A modern classic is the two-volume series by Hairer, Nørsett and Wanner.

An indication of the power of Theorem 1.10 is that here, as in all of the developments of this chapter, the initial-value problems under consideration may be linear or nonlinear. For partial differential equations we shall see that the best known analog of Theorem 1.10—the Lax Equivalence Theorem—requires linearity.

Proof. Theorem 1.10 is not mathematically deep; the art is in the definitions. To prove it we shall establish three implications:

(a) convergent $\Rightarrow$ stable;

(b) convergent $\Rightarrow$ consistent;

(c) consistent + stable $\Rightarrow$ convergent.

In outline, *(a)* and *(b)* are proved by applying the linear multistep formula to particular initial-value problems (1.1.2), and *(c)* is proved by verifying that so long as one has stability, local errors cannot add up too much.

(a) Convergent $\Rightarrow$ stable. If the linear multistep formula is convergent, then (1.6.2) must hold in particular for the initial-value problem $u_t = 0$, $u(0) = 0$, whose solution is $u(t) \equiv 0$. Now suppose the formula is unstable. Then by the proof of Theorem 1.7, it admits a particular solution $V^n = z^n$ with $|z| > 1$, or $V^n = nz^n$ with $|z| = 1$. In either case, suppose the starting values for the time integration are taken as

$$v^n = \sqrt{k}\, V^n, \qquad 0 \le n \le s-1, \tag{1.6.3}$$

where k is as always the time step. Then the computed solution for all n will be precisely $\sqrt{k}V^n$. But whereas the factor $\sqrt{k}$ ensures that the starting values approach $u_0 = 0$ as $k \to 0$, as required by (1.6.1), $|\sqrt{k}\, V^n|$ approaches ∞ for any $t = nk > 0$. Thus (1.6.2) does not hold, and the formula is not convergent.

(b) Convergent $\Rightarrow$ consistent. To prove consistency, by Theorem 1.2, we must show $\rho(1) = 0$ and $\rho'(1) = \sigma(1)$. For the first, consider the particular initial-value problem $u_t = 0$, $u(0) = 1$, whose solution is $u(t) \equiv 1$, and the particular initial values $v^0 = \cdots = v^{s-1} = 1$. Since $f(u,t) = 0$, the linear multistep formula reduces to (1.5.5), and convergence implies that the solutions to this recurrence relation for the given initial data satisfy $v(1) \to 1$ as $k \to 0$. Since v^n does not depend on k, this is the same as saying $v^n \to 1$ as $n \to \infty$ for fixed $k > 0$, and by (1.5.5), this implies $\rho(1) = 0$.

To show $\rho'(1) = \sigma(1)$, consider the particular initial-value problem $u_t = 1$, $u(0) = 0$, with exact solution $u(t) = t$. Since $f(u,t) = 1$, the linear multistep formula (1.3.4) reduces to $\rho(Z)v^n = k\sigma(1)$. Now since $\rho(Z)$ is a polynomial of degree s with $\rho(1) = 0$, it has a finite Taylor expansion

$$\rho(Z) = \rho'(1)(Z-1) + \frac{1}{2}\rho''(1)(Z-1)^2 + \cdots + \frac{1}{s!}\rho^{(s)}(1)(Z-1)^s.$$

Let us apply $\rho(Z)$ to the particular function $V^n = kn\sigma(1)/\rho'(1) = t_n\sigma(1)/\rho'(1)$. Since V^n is linear in n, all but the first term in the series are 0, and we get $\rho(Z)V^n = \rho'(1)(Z-1)V^n$, which reduces to $k\sigma(1)$. In other words, V^n is a solution to the linear multistep formula if the prescribed initial values are $v^n = V^n$ for $0 \le n \le s-1$. Obviously this solution satisfies condition (1.6.1) for the given initial data $u(0) = 0$. For the linear multistep formula to be convergent, it follows that V^n must satisfy (1.6.2), and thus we must have $\sigma(1) = \rho'(1)$, as claimed.

(c) Consistent + stable $\Rightarrow$ convergent. [This part of the proof is not yet written; see the references, such as pp. 342–345 of Hairer, Nørsett, & Wanner (1987). Following work of

Butcher (1966) and Skeel (1976), it is now standard to carry out the proof by first reducing the linear multistep formula to a one-step recursion, as in (1.5.8). The underlying idea is *backward error analysis*—i.e., one shows that the numerical method for a fixed time step computes the exact solution to a problem with a slightly perturbed function $f(u,t)$.] ∎

Besides convergence, it is desirable to know something about the accuracy of solutions computed with linear multistep formulas. To ensure pth-order accuracy, condition (1.6.1) may be strengthened as follows:

$$\|v^n - u(t_n)\| = O(k^p) \quad \text{as } k \to 0, \qquad 0 \le n \le s-1. \tag{1.6.4}$$

The following theorem confirms that for a stable linear multistep formula applied to an initial-value problem with sufficiently smooth coefficients, the local errors add up as expected:

GLOBAL p^{TH}-ORDER ACCURACY

Theorem 1.11. *Consider an initial-value problem (1.1.2) that satisfies the conditions of Theorem 1.1 on an interval $[0,T]$, and in addition, assume that $f(u,t)$ is p times continuously differentiable with respect to u and t. Let an approximate solution be computed by a convergent linear multistep formula of order of accuracy $\ge p$ with starting values satisfying (1.6.4). Then this solution satisfies*

$$\|v(t) - u(t)\| = O(k^p) \qquad \text{as } k \to 0, \tag{1.6.5}$$

uniformly for all $t \in [0,T]$.

EXERCISES

▷ *1.6.1.* Which of the following linear multistep formulas are convergent? Are the nonconvergent ones inconsistent, or unstable, or both?

(a) $v^{n+2} = \frac{1}{2}v^{n+1} + \frac{1}{2}v^n + 2kf^{n+1}$,

(b) $v^{n+1} = v^n$,

(c) $v^{n+4} = v^n + \frac{4}{3}k(f^{n+3} + f^{n+2} + f^{n+1})$,

(d) $v^{n+3} = v^{n+1} + \frac{1}{3}k(7f^{n+2} - 2f^{n+1} + f^n)$,

(e) $v^{n+4} = \frac{8}{19}(v^{n+3} - v^{n+1}) + v^n + \frac{6}{19}k(f^{n+4} + 4f^{n+3} + 4f^{n+1} + f^n)$,

(f) $v^{n+3} = -v^{n+2} + v^{n+1} + v^n + 2k(f^{n+2} + f^{n+1})$.

▷ *1.6.2. Borderline cases.*

(a) Consider the unstable extrapolation formula (1.5.1). What is the order of magnitude (power of k) of the errors introduced locally at each step? What is the order of magnitude factor by which these are amplified after $\Theta(k^{-1})$ time steps? Verify that this factor is large enough to cause nonconvergence.

(b) What are the corresponding numbers for the s-step generalization with $\rho(z) = (z-1)^s$, $\sigma(z) = 0$?

(c) On the other hand, devise another unstable linear multistep formula in which the local discretization errors are of a high enough order of accuracy relative to the unstable amplification factors that they will *not* cause nonconvergence.

(d) Why does the example of *(c)* not contradict Theorem 1.10? Why is it appropriate to consider this example unstable?

▷ *1.6.3. Convergence for scalar linear initial-value problems.* Prove as briefly and elegantly as you can that stability and consistency imply convergence, for the special case in which u is scalar and the function $f(u,t)$ is scalar and linear.

1.7. Stability regions

The results of the last two sections concerned the behavior of linear multistep formulas in the limit $k \to 0$. But for the important class of stiff ODEs, which involve widely varying time scales, it is impractical to take k small enough for those results to apply. Analysis of behavior for finite k becomes indispensable. In Chapter 4 we shall see that consideration of finite k is also essential for the analysis of discretizations of partial differential equations.

Stiffness is a subtle idea, and our discussion of it will be deferred to the next section. Here, we shall simply consider the problem of finite k analysis of linear multistep formulas. The key idea that emerges is the idea of a *stability region.*

EXAMPLE 1.7.1. Let us begin by looking again at the unstable third-order formula (1.3.21). In Example 1.5.1 we applied this formula to the initial-value problem $u_t = u$, $u(0) = 1$ with time step $k = 0.025$ and observed the oscillatory instability shown in Figure 1.5.1(b). From one time step to the next, the solution grew by a factor about -4.93, and to explain this we looked at the recurrence relation

$$\rho(Z)v^n = v^{n+2} + 4v^{n+1} - 5v^n = 0. \tag{1.7.1}$$

The zeros of $\rho(z)$ are 1 and -5, corresponding to solutions $v^n = 1$ and $v^n = (-5)^n$ to (1.7.1), and the second of these solutions explains the factor -4.93 at least approximately.

For this scalar, linear example, we can do much better by retaining the terms f^{n+j} in the analysis. The function f is simply $f(u) = u$, and thus an exact rather than approximate model of the calculation is the recurrence relation

$$\rho(Z)v^n - k\sigma(Z)f^n = (\rho(Z) - k\sigma(Z))v^n = 0, \tag{1.7.2}$$

that is,

$$(Z^2 + (4-4k)Z - (5+2k))v^n = v^{n+2} + (4-4k)v^{n+1} - (5+2k)v^n = 0.$$

Setting $k = 0.025$ gives the zeros

$$z_1 \approx 1.025315, \qquad z_2 \approx -4.925315.$$

And now we have a virtually exact explanation of the ratio ≈ -4.93 of Exercise 1.5.1—accurate, as it happens, to about 20 digits of accuracy (see Exercise 1.7.5).

For an arbitrary ODE, f is not just a multiple of u, and finite k analysis is not so simple as in the example just considered. We want to take the terms $\beta_j f^{n+j}$ into consideration, but we certainly don't want to carry out a separate analysis for each function f. Instead, let us assume that $f(u,t) = au$ for some constant $a \in \mathbb{C}$. In other words, we consider the linear *model equation*

$$u_t = au. \tag{1.7.3}$$

In the next section we shall see that a nonlinear system of ODEs can be reduced to a set of problems of the form (1.7.3) by linearization, freezing of coefficients, and diagonalization.

If a linear multistep formula (1.2.11) is applied to the model equation (1.7.3), it reduces to the recurrence relation

$$\sum_{j=0}^{s} \alpha_j v^{n+j} - \bar{k} \sum_{j=0}^{s} \beta_j v^{n+j} = 0,$$

or equivalently

$$\left[\rho(Z) - \bar{k}\sigma(Z)\right] v^n = 0, \tag{1.7.4}$$

where we have defined

$$\bar{k} = ak. \tag{1.7.5}$$

Now let $\pi_{\bar{k}}(z)$ be the **stability polynomial**

$$\pi_{\bar{k}}(z) = \rho(z) - \bar{k}\sigma(z) = \sum_{j=0}^{s} (\alpha_j - \bar{k}\beta_j) z^j, \tag{1.7.6}$$

whose coefficients depend on the parameter $\bar{k}$. Then the solutions to (1.7.4) are related to the zeros of $\pi_{\bar{k}}$ exactly as the solutions to (1.5.5) were related to the zeros of $\rho(z)$. In analogy to the developments of §1.5, we define:

A linear multistep formula is **absolutely stable*** *for a particular value $\bar{k} = ak$ if all solutions $\{v^n\}$ of the recurrence relation (1.7.4) are bounded as $n \to \infty$.*

Just as in Theorem 1.7, it is easy to characterize those linear multistep formulas that are absolutely stable:

ROOT CONDITION FOR ABSOLUTE STABILITY

Theorem 1.12. *A linear multistep formula is absolutely stable for a particular value $\bar{k} = ak$ if and only if all the zeros of $\pi_{\bar{k}}(z)$ satisfy $|z| \le 1$, and any zero with $|z| = 1$ is simple.*

What is different here from what we did in §1.5 is that everything depends on $\bar{k}$. For some $\bar{k}$ a linear multistep formula will be absolutely stable, and for others it will be absolutely unstable. Here now is the key definition:

*Other terms are "weakly stable" and "time-stable."

The **stability region** *S of a linear multistep formula is the set of all $\bar{k} \in \mathbb{C}$ for which the formula is absolutely stable.*

Note that according to this definition, a linear multistep formula is stable if and only if the point 0 belongs to its stability region.

Let us now derive the four most familiar examples of stability regions, illustrated in Figure 1.7.1.

EXAMPLE 1.7.2. For the Euler formula (1.2.3), (1.7.6) becomes

$$\pi_{\bar{k}}(z) = (z-1) - \bar{k} = z - (1+\bar{k}), \tag{1.7.7}$$

with zero $1+\bar{k}$. Therefore S is the set of $\bar{k} \in \mathbb{C}$ with $|1+\bar{k}| \le 1$, that is, the disk $|\bar{k}-(-1)| \le 1$. Figure 1.7.1a plots this region.

EXAMPLE 1.7.3. For the backward Euler formula (1.2.4), the stability polynomial is

$$\pi_{\bar{k}}(z) = (z-1) - \bar{k}z = (1-\bar{k})z - 1, \tag{1.7.8}$$

with zero $(1-\bar{k})^{-1}$. S is the set of $\bar{k} \in \mathbb{C}$ with $|1-\bar{k}|^{-1} \le 1$, that is, the exterior of the disk $|\bar{k}-1| \ge 1$. See Figure 1.7.1b.

EXAMPLE 1.7.4. For the trapezoid formula (1.2.5), we have

$$\pi_{\bar{k}}(z) = (z-1) - \tfrac{1}{2}\bar{k}(z+1) = (1-\tfrac{1}{2}\bar{k})z - (1+\tfrac{1}{2}\bar{k}) \tag{1.7.9}$$

with zero $(1+\frac{1}{2}\bar{k})/(1-\frac{1}{2}\bar{k}) = (2+\bar{k})/(2-\bar{k})$. S is the set of points in $\mathbb{C}$ that are no farther from -2 than from 2—i.e., $\mathrm{Re}\, z \le 0$, the left half-plane. See Figure 1.7.1c.

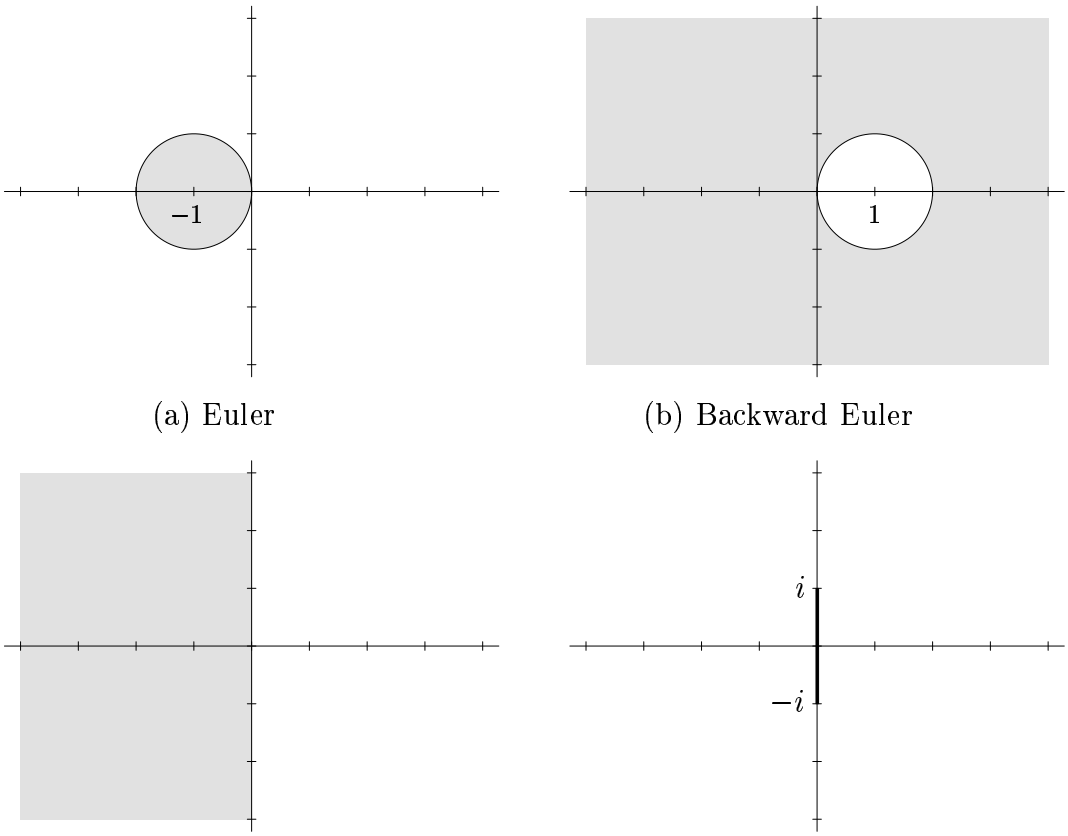
EXAMPLE 1.7.5. The midpoint formula (1.2.6) has

$$\pi_{\bar{k}}(z) = z^2 - 2\bar{k}z - 1, \tag{1.7.10}$$

which has two zeros z satisfying

$$z - \frac{1}{z} = 2\bar{k}. \tag{1.7.11}$$

Obviously there is always one zero with $|z_1| \le 1$ and another with $|z_2| \ge 1$, so for absolute stability, we must have $|z_1| = |z_2| = 1$ —both zeros on the unit circle, which will occur if and only if $\bar{k}$ lies in the closed complex interval $[-i, i]$. The two extreme values $\bar{k} = \pm i$ give double zeros $z_1 = z_2 = \pm i$, so we conclude that S is the open complex interval $(-i, i)$, as shown in Figure 1.7.1d.



(a) Euler (b) Backward Euler

(c) Trapezoid (d) Midpoint

Figure 1.7.1. Stability regions (shaded) for four linear multistep formulas. In case (d) the stability region is the open complex interval $(-i, i)$.

As mentioned on p. 17, the four linear multistep formulas just considered are the bases of four important classes of finite difference formulas for partial differential equations: Euler, backward Euler, trapezoid, and leap frog. We shall refer often to Figure 1.7.1 in examining the stability properties of these finite difference formulas.

The boundaries of the stability regions for various Adams-Bashforth, Adams-Moulton, and backwards differentiation formulas are plotted in Figures 1.7.2–1.7.4. Note that the scales in these three figures are very different—the Adams-Moulton stability regions are much larger than those for Adams-Bashforth. Note also that for all of the higher-order Adams formulas, the stability regions are bounded, whereas for the backwards differentiation for-

mulas they are unbounded. As will be explained in the next section, this is why backwards differentiation formulas are important.

How does one compute stability regions like these? One approach would be to calculate the zeros of $\pi_{\bar{k}}$ for a large number of values of $\bar{k} \in \mathbb{C}$ and draw a contour plot, but there is a simpler and more accurate method. By (1.7.6), if z is a zero of $\pi_{\bar{k}}(z)$ for some $\bar{k}$ and $\sigma(z) \neq 0$, then

$$\bar{k} = \frac{\rho(z)}{\sigma(z)} \tag{1.7.12}$$

(cf. (1.3.19)). To determine the boundary of the stability region, first calculate the curve of values $\bar{k}$ corresponding to $z = e^{i\theta}$ with $\theta \in [0, 2\pi]$. This **root locus curve** has the property that at each point $\bar{k}$ on the curve, one zero of $\pi_{\bar{k}}$ just touches the unit circle. It follows that the boundary of S is a subset of the root locus curve—only a subset, in general, because the other zeros of $\pi_{\bar{k}}$ might lie either in the disk or outside. By the principle of the argument of complex analysis, one can determine which components are which by checking just one value $\bar{k}$ in each loop enclosed by the root locus curve. See Exercise 1.7.3.

EXERCISES

▷ *1.7.1.* Prove that for the unstable linear multistep formula (1.3.21), the stability region S is the empty set.

▷ *1.7.2.* Find exact formulas for the boundaries of S for *(a)* the second-order Adams-Bashforth formula, *(b)* the third-order backwards differentiation formula.

▶ *1.7.3.* Write a computer program to plot root locus curves. In a high-level language like Matlab, it is most convenient to define a variable $\nabla = 1 - z^{-1}$, where z is a vector of 200 or so points on the unit circle, and then work directly with formulas such as (1.4.10), (1.4.13), (1.4.16) rather than with the coefficients α_j and β_j.

(a) Reproduce Figure 1.7.2, and then generate the root locus curve for $p = 4$. What is S? (Be careful.)

(b) Reproduce Figure 1.7.4, and then generate the root locus curve for $p = 7$. What is S?

(c) Plot the root locus curve for (1.3.21). Label each component by the number of zeros of $\pi_{\bar{k}}$ outside the unit disk, and explain how this pictures relates to Exercise 1.7.1.

▷ *1.7.4.* What is the maximum time step k for which the third-order Adams-Bashforth formula is absolutely stable when applied to *(i)* $u_t = -u$, *(ii)* $u_t = iu$?

(a) First, estimate the time step limits with the aid of Figure 1.7.2 and a ruler.

(b) Now derive the exact answers. For *(ii)* this is hard, but it can be done.

▷ *1.7.5.* Explain the remark about 20-digit accuracy at the end of Example 1.7.1. Approximately where does the figure of 20 digits come from?

▷ *1.7.6. True or False:* the stability region for any linear multistep formula is a closed subset of the complex plane.

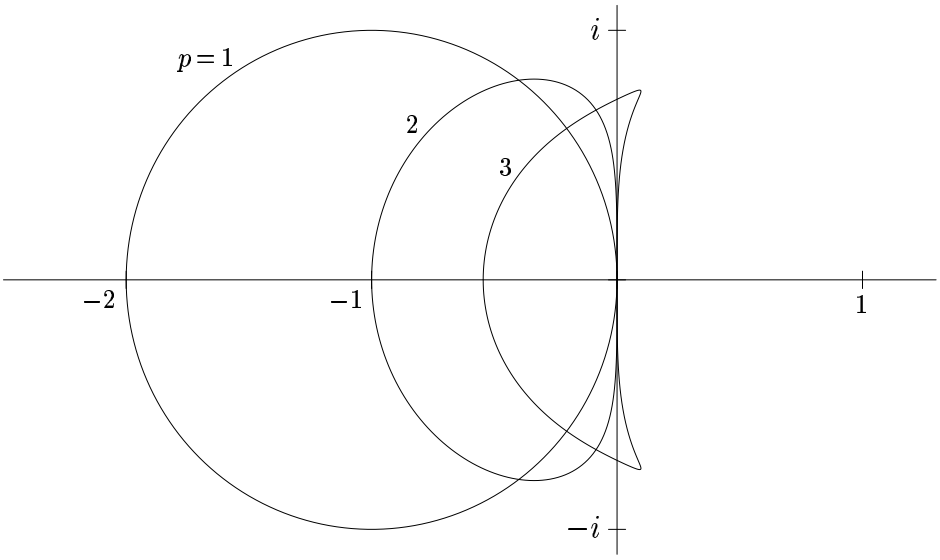


Figure 1.7.2. Boundaries of stability regions for Adams-Bashforth formulas of orders 1–3.

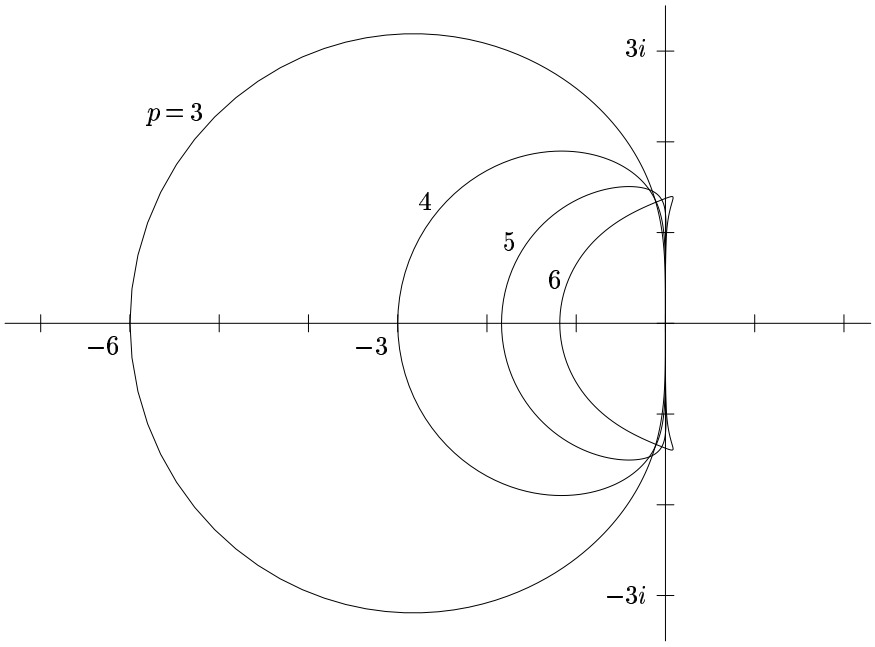


Figure 1.7.3. Boundaries of stability regions for Adams-Moulton formulas of orders 3–6. (Orders 1 and 2 were displayed already in Figure 1.7.1(b,c).) Note that the scale is very different from that of the previous figure.

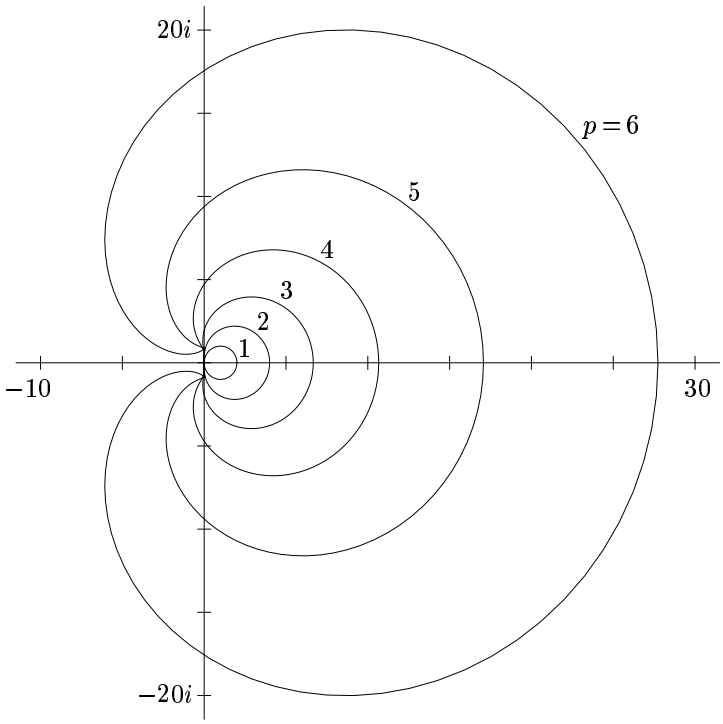


Figure 1.7.4. Boundaries of stability regions for backwards differentiation formulas of orders 1–6 (exteriors of curves shown).

▷ *1.7.7. Simpson's formula.* Determine the stability region for Simpson's formula (1.4.17), and draw a sketch.

▷ *1.7.8.* Prove that any convergent linear multistep formula is absolutely unstable for all sufficiently small positive $\bar{k} > 0$.

1.8. Stiffness

In 1962 Henrici's *Discrete Variable Methods in Ordinary Differential Equations* appeared. This landmark book, presenting the material of the previous sections in great detail, made the "Dahlquist theory" of linear multistep formulas widely known. With the availability of formulas of arbitrarily high order of accuracy and a general convergence theory to prove that they worked, it may have seemed that little was left to do except turn theory into software.

As ODE computations became commonplace in the 1960s, however, it became clear that certain problems were handled badly by the usual methods. Unreasonably small time steps would be required to achieve the desired accuracy. What was missing in the standard theory was the notion of *stiffness*. As it happens, the missing piece had been supplied a decade earlier in an eight-page paper by a pair of chemists at the University of Wisconsin, C.F. Curtiss and J.O. Hirschfelder (*Proc. Nat. Acad. Sci. 38*, 1952). Curtiss and Hirschfelder had identified the phenomenon of stiffness, coined the term, and invented backwards differentiation formulas to cope with it. However, their paper received little attention for a number of years, and for example, it does not appear among Henrici's three hundred references.

What is a stiff ODE? The following are the symptoms most often mentioned:

1. *The problem contains widely varying time scales.*
2. *Stability is more of a constraint on k than accuracy.*
3. *Explicit methods don't work.**

Each of these statements has been used as a characterization of stiffness by one author or another. In fact, they are all correct, and they are not independent statements but part of a coherent whole. After presenting an example, we shall make them more precise and explain the logical connections between them.

EXAMPLE 1.8.1. The linear initial-value problem

$$u_t = -100(u - \cos(t)) - \sin(t), \qquad u(0) = 1 \tag{1.8.1}$$

has the unique solution $u(t) = \cos(t)$, for if $u(t) = \cos(t)$ the first term on the right becomes 0, so that the large coefficient -100 drops out of the equation. That coefficient has a dominant effect on *nearby* solutions of the ODE corresponding to different initial data, however, as illustrated in Figure 1.8.1. A typical trajectory $u(t)$ of a solution to this ODE begins by shooting rapidly towards the curve $\cos(t)$ on a time scale ≈ 0.01. This is the hallmark of stiffness: rapidly changing components that are present in an ODE even when they are absent from the solution being tracked.

---

*This last item is quoted from p. 2 of the book by Hairer and Wanner (1991). For all kinds of information on numerical methods for stiff ODEs, including historical perspectives and lighthearted humor, that is the book to turn to. Another earlier reference worth noting is L. F. Shampine and C. W. Gear, "A user's view of solving stiff ordinary differential equations," *SIAM Review 21* (1979), 1–17.

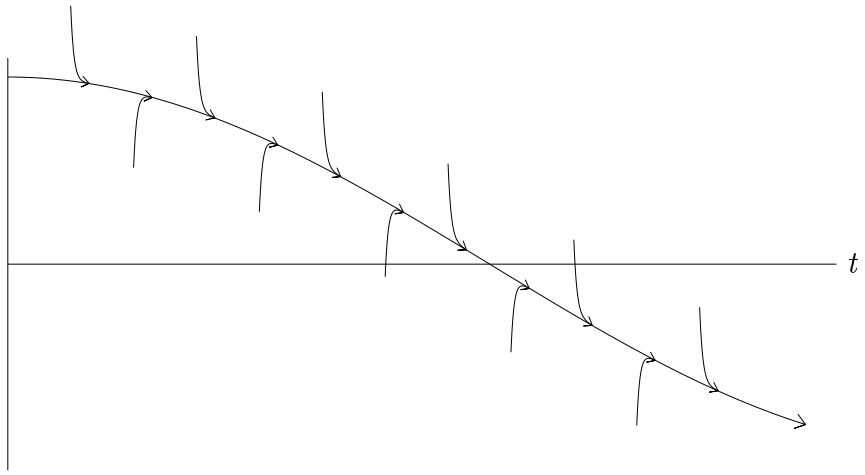


Figure 1.8.1. $u(t) = \cos(t)$ and nearby solutions of the initial-value problem (1.8.1).

k	AB2	BD2	
0.2	14.40	0.5404	
0.1	-5.70×10^4	0.54033	
0.05	-1.91×10^9	0.540309	
0.02	-5.77×10^{10}	0.5403034	
0.01	0.54030196	0.54030258	
0.005	0.54030222	0.54030238	
⋮	⋮	⋮	
0	0.540302306	0.540302306	$= \cos(1)$

Table 1.8.1. Computed values $v(1)$ for the same problem.

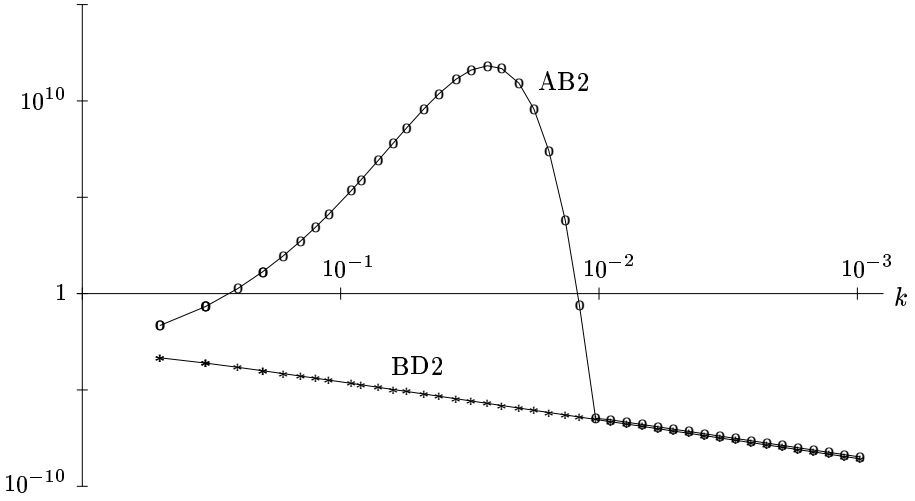


Figure 1.8.2. Computed errors $|v(1) - \cos(1)|$ as a function of step size k for the same problem (log-log scale).

Table 1.8.1 indicates the curious effect that this property of the ODE has on numerical computations. For six values of k, the table compares the results at $t=1$ computed by the second-order Adams-Bashforth and backwards differentiation formulas. (Since the ODE is linear, implementing the backwards differentiation formula is easy.) The difference between the two columns of numbers is striking. The backwards differentiation formula behaves beautifully, converging smoothly and quadratically to the correct answer, but the Adams-Bashforth formula generates enormous and completely erroneous numbers for moderate k. Yet when k becomes small enough it settles down to be just as accurate as backwards differentiation.

This behavior is shown graphically in Figure 1.8.2, which is a log-log plot of the error $|v(1)-\cos(1)|$ as a function of k. The Adams-Bashforth formula is obviously useless for $k>0.01$. What if we only want two or three digits of accuracy? With the Adams-Bashforth formula, that request cannot be satisfied; seven digits is the minimum.

Since the example just considered is linear, one can analyze what went wrong with the Adams-Bashforth formula exactly. In the notation of the last section, the trouble is that there is a root of the recurrence relation $\pi_{\bar{k}}(Z)v^n=0$ that lies outside the unit disk. We make the argument precise as follows.

EXAMPLE 1.8.1, CONTINUED. If $u(t)$ is any solution to $u_t=-100(u-\cos(t))-\sin(t)$, then $(\delta u)(t)=u(t)-\cos(t)$ satisfies the equation

$$(\delta u)_t = -100(\delta u). \tag{1.8.2}$$

This is a linear, scalar, constant-coefficient ODE of the form (1.7.3) of the last section, with $a=-100$. If we apply the 2nd-order Adams-Bashforth formula to it, we get the recurrence relation

$$v^{n+2}-v^{n+1} = -100k(\tfrac{3}{2}v^{n+1}-\tfrac{1}{2}v^n),$$

that is,

$$v^{n+2}+(150k-1)v^{n+1}-50kv^n = 0,$$

with characteristic polynomial

$$\pi_{\bar{k}}(z) = z^2+(150k-1)z-50k = z^2+(\tfrac{3}{2}\bar{k}-1)z-\tfrac{1}{2}\bar{k}.$$

For $k>0.01$, one of the two roots of this polynomial lies in the negative real interval $(-\infty,-1)$, whereas for $k\le 0.01$ that root crosses into the interval $[-1,0)$. This is why $k=0.01$ is the critical value for this problem, as is so strikingly evident in Figure 1.8.2.

Figure 1.8.3 shows how this analysis relates to the stability region of the second-order Adams-Bashforth formula. The shaded region in the figure is the stability region (see Figure 1.7.2), and the crosses mark the quantities $\bar{k}=-100k$ corresponding to the values k (except $k=0.2$) in Table 1.8.1. When k becomes as small as 0.01, i.e., $\bar{k}\ge -1$, the crosses move into the stability region and the computation is successful.

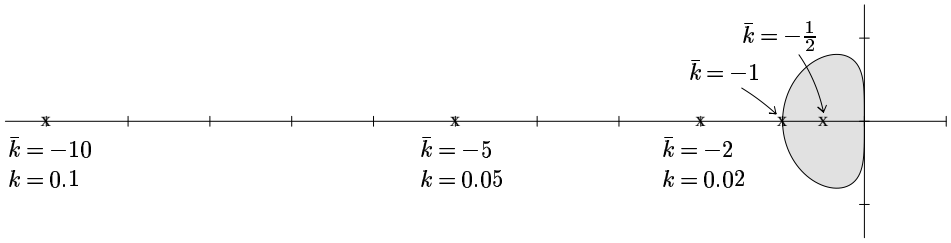


Figure 1.8.3. The stability region for the 2nd-order Adams-Bashforth formula, with crosses marking the values $\bar{k} = -100k$ from Table 1.8.1.

With this example in mind, let us turn to a more careful discussion of statements 1–3 on p. 65. The following summary describes what it means to say that the ODE $u_t = f(u,t)$ is stiff with respect to the solution $u^*(t)$ for times $t \approx t^*$.

1. Widely varying time scales. Stiffness arises when the time scale that characterizes the evolution of $u^*(t)$ for $t \approx t^*$ is much slower than the time scale that characterizes the evolution of small perturbations $(\delta u)(t)$ on that solution for $t \approx t^*$.

2. Stability is more of a constraint than accuracy. If widely varying time scales in this sense are present, and if the discrete ODE formula has a bounded stability region, then there is a mismatch between the relatively large time steps that are adequate to resolve $u^*(t)$ and the relatively small time steps that are necessary to prevent unstable growth of small perturbations $(\delta u)(t)$. A successful computation will necessitate the use of these small time steps.

3. Explicit methods don't work. In particular, since explicit methods always have bounded stability regions (Exercise 1.8.1), the solution of stiff ODEs by explicit methods necessitates the use of small time steps. For better results one must usually turn to implicit formulas with unbounded stability regions.

If the real world supplied ODEs to be solved "at random," then widely varying time scales might not be a common concern. In actuality, the applications that the real world presents us with are often exceedingly stiff. One example is the field of chemical kinetics. Here the ordinary differential equations describe reactions of various chemical species to form other species, and the stiffness is a consequence of the fact that different reactions take place on vastly different time scales. Time scale ratios of 10^6 or more are common. Two other fields in which stiff ODEs arise frequently are control theory, where controlling forces tend to be turned on and off again suddenly, and circuit simulation, since different circuit components tend to have widely differing natural

frequencies.* A fourth important class of stiff ordinary differential equations are the systems of ODEs that arise after discretizing time-dependent partial differential equations, particularly parabolic problems, with respect to their space variables—the "method of lines." This example will be discussed at length in Chapter 4.

To be able to determine whether a particular problem is stiff, we need tools for quantifying the notion of the "time scales" present in an ODE. The way this is generally done is by reducing an arbitrary system of ODEs to a collection of scalar, linear model problems of the form (1.7.3), to which the idea of stability regions will be applicable. This is achieved by the following sequence of three steps, a sequence that has been familiar to mathematical scientists for a century:

$$u_t = f(u,t) \tag{1.8.3}$$

$$\downarrow \quad \textit{LINEARIZE}$$

$$u_t = J(t)u \tag{1.8.4}$$

$$\downarrow \quad \textit{FREEZE COEFFICIENTS}$$

$$u_t = Au \tag{1.8.5}$$

$$\downarrow \quad \textit{DIAGONALIZE}$$

$$\{u_t = \lambda u\}, \quad \lambda \in \Lambda(A). \tag{1.8.6}$$

In (1.8.6), $\Lambda(A)$ denotes the spectrum of A, i.e., the set of eigenvalues.

We begin with (1.8.3), a system of N first-order ODEs, with $u^*(t)$ denoting the particular solution that we are interested in. If we make the substitution

$$u(t) = u^*(t) + (\delta u)(t),$$

as in (1.8.2), then stability and stiffness depend on the evolution of $(\delta u)(t)$.

The first step is to *linearize* the equation by assuming δu is small. If f is differentiable with respect to each component of u, then we have

$$f(u,t) \;=\; f(u^*,t) + J(t)(\delta u)(t) + o(\|\delta u\|),$$

where $J(t)$ is the $N \times N$ **Jacobian matrix** of partial derivatives of f with respect to u:

$$[J(t)]_{ij} = \frac{\partial f^{(i)}}{\partial u^{(j)}}(u^*(t),t), \qquad 1 \le i,j \le N.$$

*See A. Sangiovanni-Vincentelli and J. K. White, *Relaxation Techniques for the Solution of VLSI Circuits,* Kluwer, 1987.

This means that if δu is small, the ODE can be accurately approximated by a linear problem:

$$u_t = f(u^*,t) + J(t)(\delta u).$$

If we subtract the identity $u_t^* = f(u^*,t)$ from this equation, we get

$$(\delta u)_t = J(t)(\delta u).$$

One can think of this result as approximate, if δu is small, or exact, if δu is infinitesimal. Rewriting δu as a new variable u gives (1.8.4).

The second step is to *freeze coefficients* by setting

$$A = J(t^*)$$

for the particular value t^* of interest. The idea here is that stability and stiffness are local phenomena, which may appear at some times t^* and not others. The result is the constant-coefficient linear problem (1.8.5).

Finally, assuming A is diagonalizable, we *diagonalize* it. In general, any system of N linear ordinary differential equations with a constant, diagonalizable coefficient matrix A is exactly equivalent to N scalar equations (1.7.3) with values a equal to the eigenvalues of A. To exhibit this equivalence, let V be an $N \times N$ matrix such that $V^{-1}AV$ is diagonal. (The columns of V are eigenvectors of A.) Then $u_t = Au$ can be rewritten as $u_t = V(V^{-1}AV)V^{-1}u$, i.e., $(V^{-1}u)_t = (V^{-1}AV)(V^{-1}u)$, a diagonal system of equations in the variables $V^{-1}u$. This diagonal system of ODEs can be interpreted as a representation of the original system (1.8.5) in the basis of eigenvectors of A defined by the columns of V. And thus, since a diagonal system of ODEs is the same as a collection of independent scalar ODEs, we end up with the N problems (1.8.6).

Having reduced (1.8.3) to a collection of N scalar, linear, constant-coefficient model problems, we now estimate stability and stiffness as follows:

> *RULE OF THUMB. For a successful computation of $u^*(t)$ for $t \approx t^*$, k must be small enough that for each eigenvalue λ of the Jacobian matrix $J(t^*)$, $\bar{k} = k\lambda$ lies inside the stability region S (or at least within a distance $O(k)$ of S).*

This Rule of Thumb is not a theorem; exceptions can be found.* But if it is violated, some numerical mode will very likely blow up and obliterate the correct solution. Regarding the figure $O(k)$, see Exercise 1.8.4.

*It is interesting to consider what keeps the Rule of Thumb from being a rigorous statement. The various gaps in the argument can be associated with the three reductions (1.8.3)→(1.8.6). The step (1.8.3)→(1.8.4) may fail if the actual discretization errors or rounding errors present in the problem are large enough that they cannot be considered infinitesimal, i.e., the behavior of δu is nonlinear.

A stiff problem is one for which satisfying the conditions of the Rule of Thumb is more of a headache than resolving the underlying solution $u^*(t)$.

Here are some examples to illustrate the reduction (1.8.3)→(1.8.6).

EXAMPLE 1.8.2. If the equation in Example 1.8.1 had been

$$u_t = -100\sin(u-\cos(t))-\sin(t),$$

then linearization would have been called for. Setting $(\delta u)(t) = u(t)-\cos(t)$ gives

$$(\delta u)_t = -100\sin(\delta u),$$

and if $u(t)\approx\cos(t)$, then $(\delta u)(t)$ is small, and the linear model becomes (1.8.2) again.

EXAMPLE 1.8.3. For the initial-value problem

$$u_t = -100u^2(t)\sin(u-\cos(t))-\sin(t),$$

we end up with a model equation with a time-dependent coefficient:

$$(\delta u)_t = -100\cos^2(t)(\delta u).$$

These are scalar examples. However, the usual reason that an ODE has widely varying time scales is that it is a *system* of differential equations. If the solution has N components, it is natural that some of them may evolve much more rapidly than others. Here are three examples.

EXAMPLE 1.8.5. Consider the 2×2 linear initial-value problem

$$\begin{pmatrix} u^{(1)} \\ u^{(2)} \end{pmatrix}_t = \begin{pmatrix} -1000 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} u^{(1)} \\ u^{(2)} \end{pmatrix}, \quad t\in[0,1], \qquad \begin{pmatrix} u^{(1)} \\ u^{(2)} \end{pmatrix}(0) = \begin{pmatrix} 1 \\ 1 \end{pmatrix}. \tag{1.8.7}$$

Suppose we decide that an acceptable solution at $t=1$ is any vector with $|v^{(1)}(1)-e^{-1000}|<\epsilon$ and $|v^{(2)}(1)-e^{-1}|<\epsilon$ for some $\epsilon>0$. If (1.8.7) is solved by a pth order linear multistep formula with time step k, the $u^{(1)}$ and $u^{(2)}$ components decouple completely, so the results

---

The step (1.8.4)→(1.8.5) may fail if the density of time steps is not large compared with the time scale on which $J(t)$ varies (D.J. Higham 1992). The finite size of the time steps also implies that if an eigenvalue drifts outside the stability region for a short period, the error growth it causes may be small enough not to matter. Finally, the step (1.8.5)→(1.8.6) may fail in the case where the matrix A is far from *normal*, i.e., its eigenvectors are far from orthogonal. In this case the matrix V is ill-conditioned and instabilities may arise even though the spectrum of A lies in the stability region; it becomes necessary to consider the *pseudospectra* of A as well (D.J. Higham and L.N.T., 1992). These effects of non-normality will be discussed in §§4.5–4.6, and turn out to be particularly important in the numerical solution of time-dependent PDEs by spectral methods, discussed in Chapter 8.

in each component will be those for the corresponding model equation (1.7.3). To obtain $v^{(2)}$ sufficiently accurately, we need $k = O(\epsilon^{1/p})$. But to obtain $v^{(1)}$ sufficiently accurately, if the formula has a stability region of finite size like the Euler formula, we need k to be on the order of 10^{-3}. Most likely this is a much tighter restriction.

EXAMPLE 1.8.6. The linear problem

$$\begin{pmatrix} u^{(1)} \\ u^{(2)} \end{pmatrix}_t = \begin{pmatrix} -5 & 6 \\ 4 & -5 \end{pmatrix} \begin{pmatrix} u^{(1)} \\ u^{(2)} \end{pmatrix}, \qquad \begin{pmatrix} u^{(1)} \\ u^{(2)} \end{pmatrix}(0) = \begin{pmatrix} 1 \\ 1 \end{pmatrix} \tag{1.8.8}$$

is only superficially less trivial. The eigenvalues of the matrix are approximately -9.90 and -0.10; if 6 and 4 are changed to 5.01 and 4.99 they become approximately -9.999990 and -0.000010. As a result this system will experience a constraint on k just like that of Example 1.8.5, or worse.

EXAMPLE 1.8.7. The nonlinear ODE

$$\begin{pmatrix} u^{(1)} \\ u^{(2)} \end{pmatrix}_t = \begin{pmatrix} -u^{(1)}u^{(2)} \\ \cos(u^{(1)}) - \exp(u^{(2)}) \end{pmatrix}$$

has Jacobian matrix

$$J = -\begin{pmatrix} u^{(2)} & u^{(1)} \\ \sin(u^{(1)}) & \exp(u^{(2)}) \end{pmatrix}.$$

Near a point t with $u^{(1)}(t) = 0$ and $u^{(2)}(t) \gg 1$, the matrix is diagonal with widely differing eigenvalues, and the behavior will probably be stiff.

In general, a system of ODEs $u_t = f(u,t)$ is likely to be stiff at a point $u = u^*$, $t = t^*$ if the eigenvalues of $J(u^*, t^*)$ differ greatly in magnitude, especially if the large eigenvalues have negative real parts so that the corresponding solution components tend to die away rapidly.

To solve stiff problems effectively, one needs discrete ODE formulas with large stability regions. The particular shape of the stability region required will depend on the problem, but as a rule, the most important property one would like is for the stability region to include a large part of the left half-plane. (Why the left half-plane? See Exercise 1.8.3.) The following definitions are standard:

A linear multistep formula is A**-stable** *if the stability region contains the entire left half-plane* $\mathrm{Re}\,\bar{k} < 0$. *It is* $A(\alpha)$**-stable**, $\alpha \in (0, \pi/2)$, *if the stability region contains the infinite sector* $|\arg \bar{k} - \pi| < \alpha$. *It is* $A(0)$**-stable** *if it is* $A(\alpha)$*-stable for some* $\alpha > 0$.

Roughly speaking, an A-stable formula will perform well on almost any stiff problem (and of course also on non-stiff problems, at some cost in extra work per time step). An $A(0)$-stable formula will perform well on stiff problems whose component modes exhibit exponential decay (negative real eigenvalues) but not oscillation (imaginary eigenvalues). Discretizations of parabolic and hyperbolic partial differential equations, respectively, will provide good examples of these two situations later in the book.

Figure 1.7.1 showed that both the backward Euler and trapezoid formulas are A-stable. This seems encouraging; perhaps to play it safe, we can simply use A-stable formulas all the time? But the following famous theorem shows that that policy is too limiting:

SECOND DAHLQUIST STABILITY BARRIER

Theorem 1.13. *The order of accuracy of an implicit A-stable linear multistep formula satisfies $p \leq 2$. An explicit linear multistep formula cannot be A-stable.*

Proof. (Dahlquist, *BIT* 1963). *[Not yet written]* ∎

For serious computations, second-order accuracy is often not good enough. This is why backwards differentiation formulas are so important. For all of the higher-order Adams formulas, the stability regions are bounded, but the backwards differentiation formulas are $A(0)$-stable for $p \leq 6$.* Most of the software in use today for stiff ordinary differential equations is based on these formulas, though Runge-Kutta methods are also contenders (see the next section). Because of Theorem 1.8, the usable backwards differentiation formulas are only those with $p \leq 6$. In practice, some codes restrict attention to $p \leq 5$.

Of course there are a dozen practical issues of computation for stiff ODEs that we have not touched upon here. For example, how does one solve the nonlinear equations at each time step? (The answer is invariably some form of Newton's method; see the references.) The state of software for stiff problems is highly advanced, as is also true for non-stiff problems. A non-stiff solver applied to a stiff problem will typically get the right answer, but it will take excessively small time steps. Many such codes incorporate devices to detect that stiffness is present and alert the user to that fact, or in some cases, to trigger an automatic change to a stiff solution method.

EXERCISES

▷ *1.8.1.* Prove that the stability region of any explicit linear multistep formula is bounded.

*For $p = 3, 4, 5, 6$ the values of α involved are approximately $86°$, $73°$, $52°$, $18°$. See Figure 1.7.4.

▷ *1.8.2.* Suppose Figure 1.8.2 had been based on the error $|v(2)-\cos(2)|$ instead of $|v(1)-\cos(1)|$. Qualitatively speaking, how would the plot have been different?

▷ *1.8.3.* What is special about the left half-plane that makes it appear in the definition of A-stability? Avoid an answer full of waffle; write two or three sentences that hit the nail on the head.

▷ *1.8.4.* The Rule of Thumb of p. 70 mentions the distance $O(k)$ from the stability region. Explain where this figure comes from, perhaps with the aid of an example. Why is it $O(k)$ rather than, say, $O(k^2)$ or $O(k^{1/2})$?

▷ *1.8.5.* Here is a second-order initial-value problem:

$$u_{tt}(t) = \cos(t\,u_t(t)) + (u(t))^2 + t, \qquad u(0) = 1, \qquad u_t(0) = 0.$$

(a) Rewrite it as a first-order initial-value problem of dimension 2.
(b) Write down the precise formula used to obtain v^{n+1} at each step if this initial-value problem is solved by the second-order Adams-Bashforth formula.
(c) Write down the 2×2 Jacobian matrix J for the system *(a)*.

▷ *1.8.6.* An astronomer decides to use a non-stiff ODE solver to predict the motion of Jupiter around the sun over a period of 10,000 years. Saturn must certainly be included in the calculation if high accuracy is desired, and Neptune and Uranus might as well be thrown in for good measure. Now what about Mars, Earth, Venus, and Mercury? Including these inner planets will improve the accuracy slightly, but it will increase the cost of the computation. Estimate as accurately as you can the ratio by which it will increase the computation time on a serial computer. (*Hint:* your favorite almanac or encyclopedia may come in handy.)

▷ *1.8.7.* A linear multistep formula is A-stable. What can you conclude about the roots of $\sigma(z)$?

▷ *1.8.8. Van der Pol oscillator.* The equation $\epsilon u_{tt} = -u+(1-u^2)u_t$, known as the Van der Pol equation, represents a simple harmonic oscillator to which has been added a nonlinear term that introduces positive damping for $|u|>1$ and negative damping for $|u|<1$. Solutions to this equation approach limit cycles of finite amplitude, and if ϵ is small, the oscillation is characterized by periods of slow change punctuated by short intervals during which $u(t)$ swings rapidly from positive to negative or back again.

(a) Reduce the Van der Pol equation to a system of first-order ODEs, and compute the Jacobian matrix of this system.

(b) What can you say about the stiffness of this problem?

▷ *1.8.9.* Given $\epsilon \ll 1$, suppose you solve an initial-value problem involving the linear equation $u_t = -Au + f(t)$, where A is a constant 3×3 matrix with eigenvalues 1, $\epsilon^{-1/2}$, and ϵ^{-1}, and $f(t)$ is a slowly-varying vector function of dimension 3. You solve this equation by a linear multistep program that is smart enough to vary the step size adaptively as it integrates, and your goal is to compute the solution $u(1)$ accurate to within an absolute error on the order of $\delta \ll 1$. How many time steps (order of magnitude functions of δ and ϵ) will be needed if the program uses *(a)* the second-order Adams-Moulton method, and *(b)* the third-order Adams-Moulton method? (*Hint:* be careful!)

▶ *1.8.10. A nonlinear stiff ODE. [To appear]*

1.9. Runge-Kutta methods

Linear multistep formulas represent one extreme—one function evaluation per time step—and Runge-Kutta methods represent the other. They are *one-step, multistage* methods, in which $f(u,t)$ may be evaluated at any number of *stages* in the process of getting from v^n to v^{n+1}, but those stages are intermediate evaluations that are never used again. As a result, Runge-Kutta methods are often more stable than linear multistep formulas, but at the price of more work per time step. They tend to be easier to implement than linear multistep formulas, since starting values are not required, but harder to analyze. These methods were first developed a century ago by Runge (1895), Heun (1900), and Kutta (1901).

In this book we will not discuss Runge-Kutta methods in any detail; we merely list a few of them, below, and state two theorems without proof. This is not because they are less important than linear multistep methods, but merely to keep the scale of the book manageable. Fortunately, the fundamental concepts of stability, accuracy and convergence are much the same for Runge-Kutta as for linear multistep formulas. The details are quite different, however, and quite fascinating, involving a remarkable blend of ideas of combinatorics and graph theory. For information, see the books by Butcher and by Hairer, Nørsett, and Wanner.

Runge-Kutta formulas tend to be "not very unique". If we define

$$s = \text{numbers of stages}, \qquad p = \text{order of accuracy},$$

then for most values of s there are infinitely many Runge-Kutta formulas with the maximal order of accuracy p. Here are some of the best-known examples [which should properly be presented in tableau form, but I haven't gotten around to that]:

"Modified Euler" or "improved polygon" formula $(s=p=2)$

$$\begin{aligned} a &:= kf(v^n, t_n), \\ b &:= kf(v^n + a/2, t_n + k/2), \\ v^{n+1} &:= v^n + b. \end{aligned} \tag{1.9.1}$$

"Improved Euler" or "Heun" formula $(s=p=2)$

$$\begin{aligned} a &:= kf(v^n, t_n), \\ b &:= kf(v^n + a, t_n + k), \\ v^{n+1} &:= v^n + \tfrac{1}{2}(a+b). \end{aligned} \tag{1.9.2}$$

"Heun's third-order formula" $(s=p=3)$

$$\begin{aligned} a &:= kf(v^n, t_n), \\ b &:= kf(v^n + a/3, t_n + k/3), \\ c &:= kf(v^n + 2b/3, t_n + 2k/3), \\ v^{n+1} &:= v^n + \tfrac{1}{4}(a+3c). \end{aligned} \tag{1.9.3}$$

"Fourth-order Runge-Kutta formula" $(s=p=4)$*

$$\begin{aligned} a &:= kf(v^n, t_n), \\ b &:= kf(v^n + a/2, t_n + k/2), \\ c &:= kf(v^n + b/2, t_n + k/2), \\ d &:= kf(v^n + c, t_n + k), \\ v^{n+1} &:= v^n + \tfrac{1}{6}(a + 2b + 2c + d). \end{aligned} \tag{1.9.4}$$

If f is independent of u, the first two of these formulas reduce to the midpoint and trapezoid formulas, respectively, and the fourth-order formula reduces to Simpson's rule. This last formula is sometimes called "the" Runge-Kutta formula, and is very widely known.

How accurate can a Runge-Kutta formula be? Here is what was known as of 1987:

ORDER OF ACCURACY OF EXPLICIT RUNGE-KUTTA FORMULAS

Theorem 1.14. *An explicit Runge-Kutta formula of order of accuracy p has the following minimum number of stages s:*

Order p	*Minimum number of stages s*
$1,2,3,4$	$1,2,3,4$ *(respectively)*
5	6
6	7
7	9
8	11
9	12–17 *(precise minimum unknown)*
10	13–17 *(precise minimum unknown)*

Proof. See Butcher (1987). ∎

Figure 1.9.1 shows the stability regions for the explicit Runge-Kutta formulas with $s = 1,2,3,4$. Since these formulas are not unique, the reader may wonder which choice the figure illustrates. As it turns out, it illustrates *all* the choices, for they all have the same stability regions, which are the level curves $|p(z)| = 1$ for the truncated Taylor series approximations to e^z. (This situation changes for $s \geq 5$.)

The classical Runge-Kutta formulas were explicit, but in recent decades implicit Runge-Kutta formulas have also been studied extensively. Unlike linear multistep formulas, Runge-Kutta formulas are not subject to an A-stability barrier. The following result should be compared with Theorems 1.9 and 1.13.

IMPLICIT RUNGE-KUTTA FORMULAS

Theorem 1.15. *An s-stage implicit Runge-Kutta formula has order of accuracy $p \leq 2s$. For each s, there exists an A-stable implicit Runge-Kutta formula with $p = 2s$.*

*already given in Exercise 1.3.4.

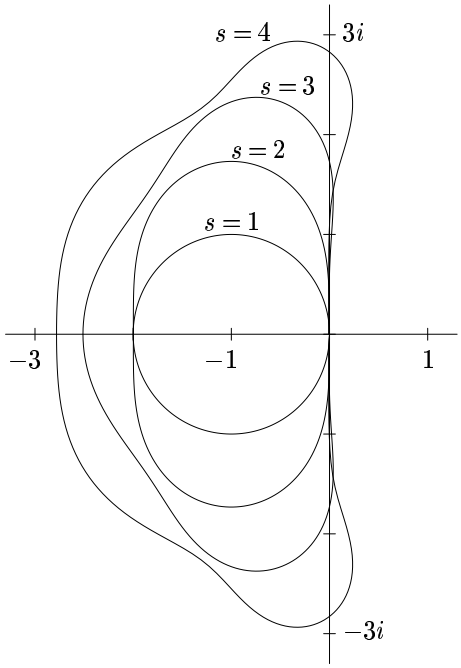


Figure 1.9.1. Boundaries of stability regions for Runge-Kutta formulas with $s = 1, 2, 3, 4$.

Theorem 1.15 looks like a panacea; why not use implicit Runge-Kutta methods all the time for stiff problems? The answer is that they lead to large systems of equations to be solved. For an initial-value problem involving N variables, an implicit linear multistep formula requires the solution of a system of N equations at each step, whereas in the Runge-Kutta case the dimension becomes sN. Since the work involved in solving a system of equations usually depends superlinearly on its size, it follows that implicit Runge-Kutta formulas tend to be more advantageous for small systems than for the larger ones that arise, for example, in discretizing partial differential equations. On the other hand, many special tricks have been devised for those problems, especially to take advantage of sparsity, so no judgment can be considered final. The book by Hairer and Wanner contains experimental comparisons of explicit and implicit multistep and Runge-Kutta codes, reaching the conclusion that they all work well.

1.10. Notes and References

This chapter has said little about the practical side of numerical solution of ordinary differential equations. One topic barely mentioned is the actual implementation of implicit methods—the solution of the associated equations at each time step by Newton's method and its variants. Another gap is that we have not discussed the idea of **adaptive step size and order control**, which, together with the development of special methods for stiff problems, are the most important developments in the numerical solution of ODEs during the computer era. The software presently available for solving ODEs is so powerful and reliable that there is little reason other than educational to write one's own program, except for very easy problems or specialized applications. Essentially all of this software makes use of adaptive step size control, and some of it controls the order of the formula adaptively too. Trustworthy codes can be found in the IMSL, NAG, and Harwell libraries, in Matlab, and in numerous other sources. The best-known programs are perhaps those developed at Sandia Laboratories and the Lawrence Livermore Laboratory, which can be obtained from the Netlib facility described in the Preface.

Although this book does not discuss boundary-value problems for ODEs, the reader should be aware that in that area too there is excellent adaptive software, which is far more efficient and reliable than the program a user is likely to construct by combining an initial-value problem solver with the idea of "shooting". Two well-known programs are PASVA and its various derivatives, which can be found in the NAG Library, and COLSYS, which has been published in the *Transactions on Mathematical Software* and is available from Netlib. A standard textbook on the subject is U. M. Ascher, R. M. M. Mattheij and R. D. Russell, *Numerical Solution of Boundary Value Problems for Ordinary Differential Equations*, Prentice Hall, 1988.

Some applications lead to systems of differential equations that must be solved in conjunction with additional non-differential conditions. Such systems are called **differential-algebraic equations (DAEs)** and have been investigated by Gear, Petzold, and many others. Software is available for these problems too, including a well-known program by Petzold known as DASSL.

The scale of ODEs that occur in practice can be enormous. One example of engineering interest arises in the simulation of VLSI circuits, where one encounters sparse stiff systems containing thousands or tens of thousands of variables. In such cases it is of paramount importance to exploit the special properties of the systems, and one special method that has been devised goes by the name of **waveform relaxation;** see the footnote on p. 69.

Chapter 2.
Fourier Analysis

... Untwisting all the chains that tie
the hidden soul of harmony.
— *JOHN MILTON, L'Allegro (1631)*

The last chapter dealt with time dependence, and this one is motivated by space dependence. Later chapters will combine the two.

Fourier analysis touches almost every aspect of partial differential equations and their numerical solution. Sometimes Fourier ideas enter into the analysis of a numerical algorithm derived from other principles—especially in the stability analysis of finite-difference formulas. Sometimes they underlie the design of the algorithm itself—spectral methods. And sometimes the situation is a mixture of both, as with iterative and multigrid methods for elliptic equations. For one reason or another, Fourier analysis will appear in all of the remaining chapters of this book.

The impact of Fourier analysis is also felt in many fields besides differential equations and their numerical solution, such as quantum mechanics, crystallography, signal processing, statistics, and information theory.

There are four varieties of Fourier transform, depending on whether the spatial domain is unbounded or bounded, continuous or discrete:

Name	Space variable	Transform variable
Fourier transform	unbounded, continuous	continuous, unbounded
Fourier series	bounded, continuous	discrete, unbounded
semidiscrete Fourier transform or *z-transform*	unbounded, discrete	continuous, bounded
discrete Fourier transform (DFT)	bounded, discrete	discrete, bounded

(The second and third varieties are mathematically equivalent.) This chapter will describe the essentials of these operations, emphasizing the parallels between them. In discrete methods for partial differential equations, one looks for a representation that will converge to a solution of the continuous problem as the mesh is refined. Our definitions are chosen so that the same kind of convergence holds also for the transforms.

Rigorous Fourier analysis is a highly technical and highly developed area of mathematics, which depends heavily on the theory of Lebesgue measure and integration. We shall make use of L^2 and ℓ^2 spaces, but for the most part this chapter avoids the technicalities. In particular, a number of statements made in this chapter hold not at every point of a domain, but "almost everywhere" — everywhere but on a set of measure zero.

2.1. The Fourier transform

If $u(x)$ is a (Lebesgue-measurable) function of $x \in \mathbb{R}$, the **L^2-norm** of u is the nonnegative or infinite real number

$$\|u\| = \Big[\int_{-\infty}^{\infty} |u(x)|^2 dx\Big]^{1/2}. \tag{2.1.1}$$

The symbol L^2 ("L-two") denotes the set of all functions for which this integral is finite:

$$L^2 = \{u : \|u\| < \infty\}. \tag{2.1.2}$$

Similarly, L^1 and L^∞ are the sets of functions having finite L^1- and L^∞-norms, defined by

$$\|u\|_1 = \int_{-\infty}^{\infty} |u(x)| dx, \qquad \|u\|_\infty = \sup_{-\infty<x<\infty} |u(x)|. \tag{2.1.3}$$

Note that since the L^2 norm is the norm used in most applications, because of its many desirable properties, we have reserved the symbol $\|\cdot\|$ without a subscript for it.

The **convolution** of two functions u, v is the function $u*v$ defined by

$$(u*v)(x) = (u*v)(x) = \int_{-\infty}^{\infty} u(x-y)v(y)dy = \int_{-\infty}^{\infty} u(y)v(x-y)dy, \tag{2.1.4}$$

assuming these integrals exist. One way to think of $u*v$ is as a weighted moving average of values $u(x)$ with weights defined by $v(x)$, or vice versa.

For any $u \in L^2$, the **Fourier transform** of u is the function $\hat{u}(\xi)$ defined by

$$\hat{u}(\xi) = (\mathcal{F}u)(\xi) = \int_{-\infty}^{\infty} e^{-i\xi x} u(x) dx, \qquad \xi \in \mathbb{R}.$$

The quantity ξ is known as the **wave number,** the spatial analog of frequency. For many functions $u \in L^2$, this integral converges in the usual sense for all $\xi \in \mathbb{R}$, but there are situations where this is not true, and in these cases one must interpret the integral as a limit in a certain L^2-norm sense of integrals $\int_{-M}^{M}$ as $M \to \infty$. The reader interested in such details should consult the various books listed in the references.*

*If $u \in L^1$, then $\hat{u}(\xi)$ exists for every ξ and is continuous with respect to ξ. According to the **Riemann-Lebesgue Lemma,** it also satisfies $|\hat{u}(\xi)| \to 0$ as $\xi \to \infty$.

Figure 2.1.1. Space and wave number domains for the Fourier transform (compare Figures 2.2.1 and 2.4.1).

The following theorem summarizes some of the fundamental properties of Fourier transforms.

THE FOURIER TRANSFORM

Theorem 2.1.* *If $u \in L^2$, then the Fourier transform*

$$\hat{u}(\xi) = (\mathcal{F}u)(\xi) = \int_{-\infty}^{\infty} e^{-i\xi x} u(x)dx, \qquad \xi \in \mathbb{R} \tag{2.1.5}$$

belongs to L^2 also, and u can be recovered from $\hat{u}$ by the **inverse Fourier transform**

$$u(x) = (\mathcal{F}^{-1}\hat{u})(x) = \frac{1}{2\pi}\int_{-\infty}^{\infty} e^{i\xi x}\hat{u}(\xi)d\xi, \qquad x \in \mathbb{R}. \tag{2.1.6}$$

The L^2-norms of u and $\hat{u}$ are related by **Parseval's equality**,

$$\|\hat{u}\| = \sqrt{2\pi}\|u\|. \tag{2.1.7}$$

If $u \in L^2$ and $v \in L^1$ (or vice versa), then $u * v \in L^2$, and $\widehat{u*v}$ satisfies

$$\widehat{u*v}(\xi) = \hat{u}(\xi)\hat{v}(\xi). \tag{2.1.8}$$

These four equations are of such fundamental importance that they are worth commenting on individually, although it is assumed the reader has already been exposed to Fourier analysis.

*As mentioned in the introduction to this chapter, some of these properties—namely equations (2.1.6) and (2.1.8)—hold merely for "almost every" value of x or ξ. In fact even if $f(z)$ is a continuous function in L^2, its Fourier transform may fail to converge at certain points x. To ensure pointwise convergence one needs additional assumptions such as that f is of **bounded variation** (defined below before Theorem 2.4) and belongs to L^1. These assumptions also ensure that at any point x where f has a jump discontinuity, its Fourier transform converges to the average value $(f(x^-)+f(x^+))/2$.

First of all, (2.1.5) indicates that $\hat{u}(\xi)$ is a measure of the correlation of $u(x)$ with the function $e^{i\xi x}$. The idea behind Fourier analysis is to interpret $u(x)$ as a superposition of monochromatic waves $e^{i\xi x}$ with various wave numbers ξ, and $\hat{u}(\xi)$ represents the complex amplitude (more precisely: amplitude density with respect to ξ) of the component of u at wave number ξ.

Conversely, (2.1.6) expresses the synthesis of $u(x)$ as a superposition of its components $e^{i\xi x}$, each multiplied by the appropriate factor $\hat{u}(\xi)$. The factor 2π is a nuisance that could have been put in various places in our formulas, but is hard to eliminate entirely.

Equation (2.1.7), Parseval's equality, is a statement of energy conservation: the L^2 energy of any signal $u(x)$ is equal to the sum of the energies of its component vibrations (except for the factor $\sqrt{2\pi}$). By "energy" we mean the square of the L^2 norm.

Finally, the convolution equation (2.1.8) is perhaps the most subtle of the four. The left side, $\widehat{u*v}(\xi)$, represents the strength of the wave number ξ component that results when u is convolved with v—in other words, the degree to which u and v beat in and out of phase with each other at wave number ξ when multiplied together in reverse order with a varying offset. Such beating is caused by a quadratic interaction of the wave number component ξ in u with the same component of v—hence the right-hand side $\hat{u}(\xi)\hat{v}(\xi)$.

All of the assertions of Theorem 2.1 can be verified in the following example, which the reader should study carefully.

EXAMPLE 2.1.1. *B-splines.* Suppose u is the function

$$u(x) = \begin{cases} \frac{1}{2} & \text{for } -1 \le x \le 1, \\ 0 & \text{otherwise} \end{cases} \tag{2.1.9}$$

(Figure 2.1.2). Then by (2.1.1) we have $\|u\| = 1/\sqrt{2}$, and (2.1.5) gives

$$\hat{u}(\xi) = \tfrac{1}{2}\int_{-1}^{1} e^{-i\xi x}\,dx = \left.\frac{e^{-i\xi x}}{-2i\xi}\right|_{-1}^{1} = \frac{\sin\xi}{\xi}. \tag{2.1.10}$$

(This function $\hat{u}(\xi)$ is called a **sinc function;** more on these in §2.3.) From (2.1.1) and the indispensable identity*

$$\int_{-\infty}^{\infty} \frac{\sin^2 s}{s^2}\,ds = \pi, \tag{2.1.11}$$

which can be derived by complex contour integration, we calculate $\|\hat{u}\| = \sqrt{\pi}$, which confirms (2.1.7).

From the definition (2.1.4) it is readily verified that in this example

$$(u*u)(x) = \begin{cases} \frac{1}{2}(1-|x|/2) & \text{for } -2 \le x \le 2, \\ 0 & \text{otherwise} \end{cases} \tag{2.1.12}$$

---

*worth memorizing!

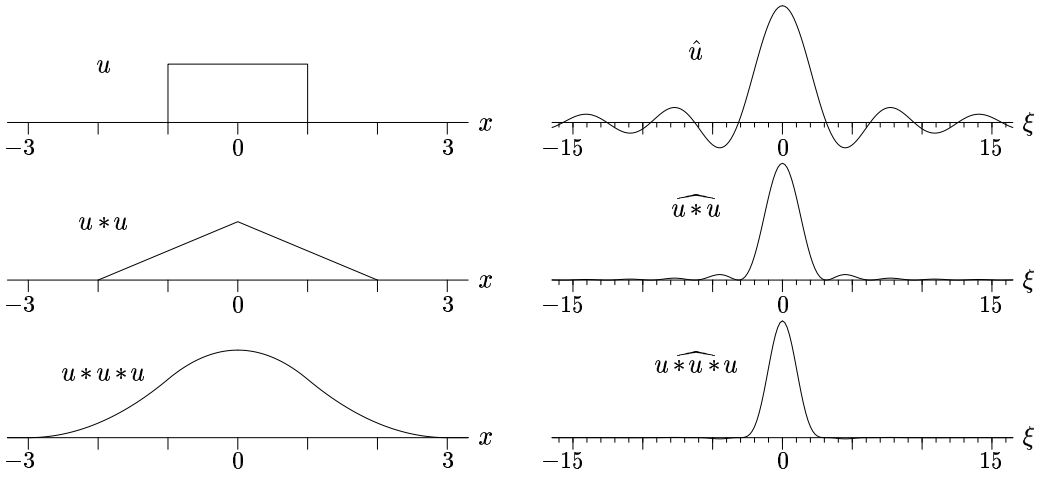


Figure 2.1.2. The first three B-splines of Example 2.1.1 and their Fourier transforms.

and

$$(u*u*u)(x) = \begin{cases} \frac{3}{4} - \frac{1}{4}x^2 & \text{for } -1 \le x \le 1, \\ \frac{1}{8}(9 - 6|x| + x^2) & \text{for } 1 \le |x| \le 3, \\ 0 & \text{otherwise,} \end{cases} \tag{2.1.13}$$

and by (2.1.8) and (2.1.10), the corresponding Fourier transforms must be

$$\widehat{u*u}(\xi) = \frac{\sin^2\xi}{\xi^2}, \qquad \widehat{u*u*u}(\xi) = \frac{\sin^3\xi}{\xi^3}. \tag{2.1.14}$$

See Figure 2.1.2. In general, a convolution $u_{(p)}$ of p copies of u has the Fourier transform

$$\widehat{u_{(p)}}(\xi) = \mathcal{F}\{u*u*\cdots*u\}(\xi) = \left(\frac{\sin\xi}{\xi}\right)^p. \tag{2.1.15}$$

Note that whenever $u_{(p)}$ or any other function is convolved with the function u of (2.1.9), it becomes smoother, since the convolution amounts to a local moving average. In particular, u itself is piecewise continuous, $u*u$ is continuous and has a piecewise continuous first derivative, $u*u*u$ has a continuous derivative and a piecewise continuous second derivative, and so on. In general $u_{(p)}$ is a piecewise polynomial of degree $p-1$ with a continuous $(p-2)$nd derivative and a piecewise continuous $(p-1)$st derivative, and is known as a **B-spline**. (See, for example, C. de Boor, *A Practical Guide to Splines*, Springer, 1978.)

Thus convolution with u makes a function smoother, while the effect on the Fourier transform is to multiply it by $\sin\xi/\xi$ and thereby make it decay more rapidly $\xi \to \infty$. This relationship is evident in Figure 2.1.2.

For applications to numerical methods for partial differential equations, there are two properties of the Fourier transform that are most important.

One is equation (2.1.8): the Fourier transform converts convolution into multiplication. The second can be derived by integration by parts:

$$\widehat{u_x}(\xi) = \int_{-\infty}^{\infty} e^{-i\xi x} u_x(x)dx = -\int_{-\infty}^{\infty} (-i\xi)e^{-i\xi x} u(x)dx = i\xi\,\hat{u}(\xi), \qquad (2.1.16)$$

assuming $u(x)$ is smooth and decays at ∞. That is, the Fourier transform converts differentiation into multiplication by $i\xi$. This result is rigorously valid for any absolutely continuous function $u \in L^2$ whose derivative belongs to L^2. Note that differentiation makes a function less smooth, so the fact that it makes the Fourier transform decay less rapidly fits the pattern mentioned above for convolution.

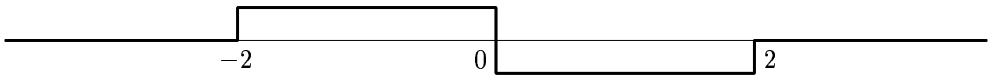


Figure 2.1.3.

EXAMPLE 2.1.2. The function

$$u(x) = \begin{cases} \frac{1}{4} & \text{for } -2 \le x < 0, \\ -\frac{1}{4} & \text{for } 0 < x \le 2, \\ 0 & \text{otherwise,} \end{cases} \qquad (2.1.17)$$

illustrated in Figure 2.1.3, has Fourier transform

$$\begin{aligned} \hat{u}(\xi) &= \tfrac{1}{4}\int_{-2}^{0} e^{-i\xi x}dx - \tfrac{1}{4}\int_{0}^{2} e^{-i\xi x}dx \\ &= \frac{1}{-4i\xi}(1 - e^{2i\xi} - e^{-2i\xi} + 1) = \frac{1}{4i\xi}(e^{i\xi} - e^{-i\xi})^2 = \frac{i\sin^2\xi}{\xi}, \end{aligned} \qquad (2.1.18)$$

which is $i\xi$ times the Fourier transform (2.1.14) of the triangular hat function (2.1.12). In keeping with (2.1.16), (2.1.17) is the derivative of (2.1.12).

The following theorem collects (2.1.16) together with a number of additional properties of the Fourier transform:

PROPERTIES OF THE FOURIER TRANSFORM

Theorem 2.2. *Let $u, v \in L^2$ have Fourier transforms $\hat{u} = \mathcal{F}u$, $\hat{v} = \mathcal{F}v$. Then:*

(a) Linearity. $\mathcal{F}\{u+v\}(\xi) = \hat{u}(\xi) + \hat{v}(\xi)$; $\mathcal{F}\{cu\}(\xi) = c\hat{u}(\xi)$.

(b) Translation. If $x_0 \in \mathbb{R}$, *then* $\mathcal{F}\{u(x+x_0)\}(\xi) = e^{i\xi x_0}\hat{u}(\xi)$.

(c) Modulation. If $\xi_0 \in \mathbb{R}$, *then* $\mathcal{F}\{e^{i\xi_0 x}u(x)\}(\xi) = \hat{u}(\xi - \xi_0)$.

(d) Dilation. If $c \in \mathbb{R}$ *with* $c \neq 0$, *then* $\mathcal{F}\{u(cx)\}(\xi) = \hat{u}(\xi/c)/|c|$.

(e) Conjugation. $\mathcal{F}\{\overline{u}\}(\xi) = \overline{\hat{u}(-\xi)}$.

(f) Differentiation. If $u_x \in L^2$, *then* $\mathcal{F}\{u_x\}(\xi) = i\xi\hat{u}(\xi)$.

(g) Inversion. $\mathcal{F}^{-1}\{u\}(\xi) = \dfrac{1}{2\pi}\hat{u}(-\xi)$.

Proof. See Exercise 2.1.2. ∎

In particular, taking $c = -1$ in part *(d)* above gives $\mathcal{F}\{u(-x)\} = \hat{u}(-\xi)$. Combining this result with part *(e)* leads to the following elementary but useful results. Definitions: $u(x)$ is even, odd, real, or imaginary if $u(x) = u(-x)$, $u(x) = -u(-x)$, $u(x) = \overline{u(x)}$, or $u(x) = -\overline{u(x)}$, respectively; $u(x)$ is hermitian or skew-hermitian if $u(x) = \overline{u(-x)}$ or $u(x) = -\overline{u(-x)}$, respectively.

SYMMETRIES OF THE FOURIER TRANSFORM

Theorem 2.3. *Let $u \in L^2$ have Fourier transform $\hat{u} = \mathcal{F}u$. Then*

(a) $u(x)$ is even (odd) $\iff$ $\hat{u}(\xi)$ is even (odd);

(b) $u(x)$ is real (imaginary) $\iff$ $\hat{u}(\xi)$ is hermitian (skew-hermitian);

and therefore

(c) $u(x)$ is real and even $\iff$ $\hat{u}(\xi)$ is real and even;

(d) $u(x)$ is real and odd $\iff$ $\hat{u}(\xi)$ is imaginary and odd;

(e) $u(x)$ is imaginary and even $\iff$ $\hat{u}(\xi)$ is imaginary and even;

(f) $u(x)$ is imaginary and odd $\iff$ $\hat{u}(\xi)$ is real and odd.

Proof. See Exercise 2.1.3. ∎

In the discussion above we have twice observed the following relationships between the smoothness of a function and the decay of its Fourier transform:

$u(x)$		$\hat{u}(\xi)$
smooth	$\leftrightarrow$	decays rapidly as $\lvert\xi\rvert \to \infty$
decays rapidly as $\lvert x\rvert \to \infty$	$\leftrightarrow$	smooth

(Of course since the Fourier transform is essentially the same as the inverse Fourier transform, by Theorem 2.2g, the two rows of this summary are equivalent.) The intuitive explanation is that if a function is smooth, then it can be accurately represented as a superposition of slowly-varying waves, so one does not need much energy in the high wave number components. Conversely, a non-smooth function requires a considerable amplitude of high wave number components to be represented accurately. These relationships are the bedrock of analog and digital signal processing, where all kinds of smoothing operations are effected by multiplying the Fourier transform by a "windowing function" that decays suitably rapidly.

The following theorem makes these connections between u and $\hat{u}$ precise. This theorem may seem forbidding at first, but it is worth studying carefully. Each of the four parts of the theorem makes a stronger smoothness assumption on u than the last, and reaches a correspondingly stronger conclusion about the rate of decay of $\hat{u}(\xi)$ as $|\xi| \to \infty$. Parts *(c)* and *(d)* are known as the **Paley-Wiener theorems**.

First, a standard definition. A function u defined on $\mathbb{R}$ is said to have **bounded variation** if there is a constant M such that for any finite m and any points $x_0 < x_1 < \cdots < x_m$, $\sum_{j=1}^{m} |u(x_j) - u(x_{j-1})| \le M$.

SMOOTHNESS OF u AND DECAY OF $\hat{u}$

Theorem 2.4. *Let u be a function in L^2.*

(a) If u has $p-1$ continuous derivatives in L^2 for some $p \geq 0$, and a pth derivative in L^2 that has bounded variation, then

$$\hat{u}(\xi) = O(|\xi|^{-p-1}) \qquad \text{as} \quad |\xi| \to \infty. \tag{2.1.19}$$

(b) If u has infinitely many continuous derivatives in L^2, then

$$\hat{u}(\xi) = O(|\xi|^{-M}) \qquad \text{as} \quad |\xi| \to \infty \quad \text{for all} \quad M, \tag{2.1.20}$$

and conversely.

(c) If u can be extended to an analytic function of $z = x+iy$ in the complex strip $|\operatorname{Im} z| < a$ for some $a > 0$, with $\|u(x+iy)\| \leq$ const uniformly for each constant $-a < y < a$, then

$$e^{a|\xi|}\hat{u}(\xi) \in L^2, \tag{2.1.21}$$

and conversely.

(d) If u can be extended to an entire function of $z = x+iy$ with $|u(z)| = O(e^{a|z|})$ as $|z| \to \infty$ $(z \in \mathbb{C})$ for some $a > 0$, then $\hat{u}$ has compact support contained in $[-a, a]$, i.e.*

$$\hat{u}(\xi) = 0 \qquad \text{for all} \quad |\xi| > a, \tag{2.1.22}$$

and conversely.

Proof. See, for example, §VI.7 of Y. Katznelson, *An Introduction to Harmonic Analysis*, Dover, 1976. [Also see Rudin (p. 407), Paley & Wiener, Reed & Simon v. 2, Hoŕmander v. 1 (p. 181), entire functions books...] ∎

A function of the kind described in *(d)* is said to be **band-limited**, since only a finite band of wave numbers are represented in it.

Since the Fourier transform and its inverse are essentially the same, by Theorem 2.2g, Theorem 2.4 also applies if the roles of $u(x)$ and $\hat{u}(\xi)$ are interchanged.

EXAMPLE 2.1.1, CONTINUED. The square wave u of Example 2.1.1 (Figure 2.1.2) satisfies condition *(a)* of Theorem 2.4 with $p = 0$, so its Fourier transform should satisfy

*An entire function is a function that is analytic throughout the complex plane $\mathbb{C}$.

$|\hat{u}(\xi)| = O(|\xi|^{-1})$, as is verified by (2.1.10). On the other hand, suppose we interchange the roles of u and $\hat{u}$ and apply the theorem again. The function $u(\xi) = \sin\xi/\xi$ is entire, and since $\sin(\xi) = (e^{i\xi} - e^{-i\xi})/2i$, it satisfies $u(\xi) = O(e^{|\xi|})$ as $|\xi| \to \infty$ (with ξ now taking complex values). By part *(d)* of Theorem 2.4, it follows that $u(x)$ must have compact support contained in $[-1,1]$, as indeed it does.

Repeating the example for $u*u$, condition *(a)* now applies with $p=1$, and the Fourier transform (2.1.14) is indeed of magnitude $O(|\xi|^{-2})$, as required. Interchanging u and $\hat{u}$, we note that $\sin^2\xi/\xi^2$ is an entire function of magnitude $O(e^{2|\xi|})$ as $|\xi| \to \infty$, and $u*u$ has support contained in $[-2,2]$.

EXERCISES

▷ *2.1.1.* Show that the two integrals in the definition (2.1.4) of $u*v$ are equivalent.

▷ *2.1.2.* Derive conditions *(a)–(g)* of Theorem 2.2. (Do not worry about justifying the usual operations on integrals.)

▷ *2.1.3.* Prove Theorem 2.3.

▷ *2.1.4.*

(a) Which functions $u \in L^2 \cap L^1$ satisfy $u*u = 0$?

(b) How about $u*u = u$?

▷ *2.1.5. Integration.*

(a) What does part *(f)* of Theorem 2.2 suggest should be the Fourier transform of the function $U(x) = \int_{-\infty}^{x} u(s)ds$?

(b) Obviously $U(x)$ cannot belong to L^2 unless $\int_{-\infty}^{\infty} u(x)dx = 0$, so by Theorem 2.1, this is a necessary condition for $\hat{U}$ to be in L^2 also. Explain how the condition $\int_{-\infty}^{\infty} u(x)dx = 0$ relates to your formula of *(a)* for $\hat{U}$ in terms of $\hat{u}$.

▷ *2.1.6.*

(a) Calculate the Fourier transform of $u(x) = (1+x^2)^{-1}$. (*Hint:* use a complex contour integral if you know how. Otherwise, look the result up in a table of integrals.)

(b) How does this example fit into the framework of Theorem 2.4? Which parts of the theorem apply to u?

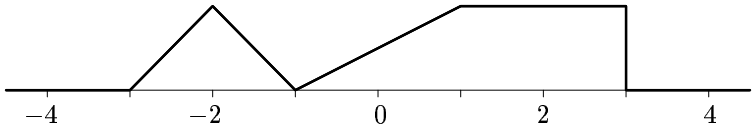
(c) If the roles of u and $\hat{u}$ are interchanged, how does the example now fit Theorem 2.4? Which parts of the theorem apply to $\hat{u}$?

▷ *2.1.7.* The **autocorrelation function** of a function $u \in L^2 \cap L^1$ may be defined by

$$\phi(c) = \frac{1}{\|u\|^2} \int_{-\infty}^{\infty} u(x)u(x+c)dx.$$

Find an expression for $\phi(c)$ as an inverse Fourier transform of a product of Fourier transforms involving u. This expression is the basis of some algorithms for computing $\phi(c)$.

▷ *2.1.8.* Without evaluating any integrals, use Theorem 2.2 and (2.1.10) to determine the Fourier transform of the following function:



▷ *2.1.9. The uncertainty principle.* Show by using Theorem 2.4 that if $u(x)$ and $\hat{u}(\xi)$ both have compact support, with $u \in L^2$, then $u(x) \equiv 0$.

2.2. The semidiscrete Fourier transform

The semidiscrete Fourier transform is the reverse of the more familiar Fourier series: instead of a bounded, continuous spatial domain and an unbounded, discrete transform domain, it involves the opposite. This is just what is needed for the analysis of numerical methods for partial differential equations, where we are perpetually concerned with functions defined on discrete grids. For many analytical purposes it is simplest to think of these grids as infinite in extent.

Let $h > 0$ be a real number, the **space step**, and let $\dots, x_{-1}, x_0, x_1, \dots$ be defined by $x_j = jh$. Thus $\{x_j\} = h\mathbb{Z}$, where $\mathbb{Z}$ is the set of integers. We are concerned now with spatial **grid functions** $v = \{v_j\}$, which may or may not be approximations to a continuous function u,

$$v_j \approx u(x_j).$$

As in the last chapter, it will be convenient to write $v(x_j)$ sometimes for v_j.

Figure 2.2.1. Space and wave number domains for the semidiscrete Fourier transform.

For functions defined on discrete domains it is standard to replace the upper-case letter L by a lower-case script letter ℓ. (Both symbols honor Henri Lebesgue, the mathematician who laid the foundations of modern functional analysis early in the twentieth century.) The ℓ_h^2**-norm** of a grid function v is the nonnegative or infinite real number

$$\|v\| = \left[h \sum_{j=-\infty}^{\infty} |v_j|^2 \right]^{1/2}. \tag{2.2.1}$$

Notice the h in front of the summation. One can think of (2.2.1) as a discrete approximation to the integral (2.1.1) by the trapezoid rule or the rectangle rule for quadrature. (On an unbounded domain these two are equivalent.) The symbol ℓ_h^2 ("little L-two sub h") denotes the set of grid functions of finite norm,

$$\ell_h^2 = \{v : \|v\| < \infty\},$$

and similarly with ℓ_h^1 and ℓ_h^∞. In contrast to the situation with L^1, L^2, and L^∞, these spaces are nested:

$$\ell_h^1 \subseteq \ell_h^2 \subseteq \ell_h^\infty. \tag{2.2.2}$$

(See Exercise 2.1.1.)

The **convolution** $v*w$ of two functions v, w is the function $v*w$ defined by

$$(v*w)_m = h\sum_{j=-\infty}^{\infty} v_{m-j}w_j = h\sum_{j=-\infty}^{\infty} v_j w_{m-j}, \tag{2.2.3}$$

provided that these sums exist. This formula is a trapezoid or rectangle rule approximation to (2.1.4).

For any $v \in \ell_h^2$, the **semidiscrete Fourier transform** of v is the function $\hat{v}(\xi)$ defined by

$$\hat{v}(\xi) = (\mathcal{F}_h v)(\xi) = h\sum_{j=-\infty}^{\infty} e^{-i\xi x_j} v_j, \qquad \xi \in [-\pi/h, \pi/h],$$

a discrete approximation to (2.1.5). A priori, this sum defines a function $\hat{v}(\xi)$ for all $\xi \in \mathbb{R}$. However, notice that for any integer m, the exponential $e^{2\pi i m x_j/h} = e^{2\pi i m j}$ is exactly 1 at all of the grid points x_j. More generally, any wave number ξ is indistinguishable on the grid from all other wave numbers $\xi + 2\pi m/h$, where m is any integer—a phenomenon called **aliasing**. This means that the function $\hat{v}(\xi)$ is $2\pi/h$-periodic on $\mathbb{R}$. To make sense of the idea of analyzing v into component oscillations, we shall normally restrict attention to one period of $\hat{v}$ by looking only at wave numbers in the range $[-\pi/h, \pi/h]$, and it is in this sense that the Fourier transform of a grid function is defined on a bounded domain. But the reader should bear in mind that the restriction of ξ to any particular interval is a matter of labeling, not mathematics; in principle e^0 and $e^{100\pi i j}$ are equally valid representations of the grid function $v_j \equiv 1$.

Thus for discretized functions v, the transform $\hat{v}(\xi)$ inhabits a bounded domain. On the other hand the domain is still continuous. This reflects the fact that arbitrarily fine gradations of wave number are distinguishable on an unbounded grid.

Since x and ξ belong to different sets, it is necessary to define an additional vector space for functions $\hat{v}$. The L_h^2**-norm** of a function $\hat{v}$ is the number

$$\|\hat{v}\| = \Big[\int_{-\pi/h}^{\pi/h} |\hat{v}(\xi)|^2 d\xi\Big]^{1/2}. \tag{2.2.4}$$

One can think of this as an approximation to (2.1.1) in which the wave number components with $|\xi| > \pi/h$ have been replaced by zero. The symbol L_h^2 denotes the set of (Lebesgue-measurable) functions on $[-\pi/h, \pi/h]$ of finite norm,

$$L_h^2 = \{\hat{v} : \|\hat{v}\| < \infty\}. \tag{2.2.5}$$

Now we can state a theorem analogous to Theorem 2.1:

THE SEMIDISCRETE FOURIER TRANSFORM

Theorem 2.5. *If $v \in \ell_h^2$, then the semidiscrete Fourier transform*

$$\hat{v}(\xi) = (\mathcal{F}_h v)(\xi) = h \sum_{j=-\infty}^{\infty} e^{-i\xi x_j} v_j, \qquad \xi \in [-\pi/h, \pi/h] \tag{2.2.6}$$

belongs to L_h^2, and v can be recovered from $\hat{v}$ by the **inverse semidiscrete Fourier transform**

$$v_j = (\mathcal{F}_h^{-1}\hat{v})(x) = \frac{1}{2\pi} \int_{-\pi/h}^{\pi/h} e^{i\xi x_j} \hat{v}(\xi) d\xi, \qquad j \in \mathbb{Z}. \tag{2.2.7}$$

The ℓ_h^2-norm of v and the L_h^2-norm of $\hat{v}$ are related by **Parseval's equality**,

$$\|\hat{v}\| = \sqrt{2\pi}\|v\|. \tag{2.2.8}$$

If $u \in \ell_h^2$ and $v \in \ell_h^1$ (or vice versa), then $v * w \in \ell_h^2$, and $\widehat{v*w}$ satisfies

$$\widehat{v*w}(\xi) = \hat{v}(\xi)\hat{w}(\xi). \tag{2.2.9}$$

As in the continuous case, the following properties of the semidiscrete Fourier transform will be useful. In *(c)*, and throughout this book wherever convenient, we take advantage of the fact that $\hat{v}(\xi)$ can be treated as a periodic function defined for all $\xi \in \mathbb{R}$.

PROPERTIES OF THE SEMIDISCRETE FOURIER TRANSFORM

Theorem 2.6. *Let $v, w \in \ell_h^2$ have Fourier transforms $\hat{v}, \hat{w}$. Then:*

(a) Linearity. $\mathcal{F}_h\{v+w\}(\xi) = \hat{v}(\xi) + \hat{w}(\xi)$; $\mathcal{F}_h\{cv\}(\xi) = c\hat{v}(\xi)$.

(b) Translation. If $j_0 \in \mathbb{Z}$, *then* $\mathcal{F}_h\{v_{j+j_0}\}(\xi) = e^{i\xi x_{j_0}}\hat{v}(\xi)$.

(c) Modulation. If $\xi_0 \in \mathbb{R}$, *then* $\mathcal{F}_h\{e^{i\xi_0 x_j} v_j\}(\xi) = \hat{v}(\xi - \xi_0)$.

(d) Dilation. If $m \in \mathbb{Z}$ *with* $m \neq 0$, *then* $\mathcal{F}_h\{v_{mj}\}(\xi) = \hat{v}(\xi/m)/|m|$.

(e) Conjugation. $\mathcal{F}_h\{\overline{v}\}(\xi) = \overline{\hat{v}(-\xi)}$.

The symmetry properties of the Fourier transform summarized in Theorem 2.3 apply to the semidiscrete Fourier transform too; we shall not repeat the list here.

We come now to a fundamental result that describes the relationship of the Fourier transform of a continuous function u to that of a discretization v of u—or if x and ξ are interchanged, the relationship of Fourier series to Fourier transforms. Recall that because of the phenomenon of **aliasing**, all wave numbers $\xi + 2\pi j/h$, $j \in \mathbb{Z}$, are indistinguishable on the grid $h\mathbb{Z}$. Suppose that $u \in L^2$ is a sufficiently smooth function defined on $\mathbb{R}$, and let $v \in \ell_h^2$ be the discretization obtained by sampling $u(x)$ at the points x_j. The aliasing principle implies that $\hat{v}(\xi)$ must consist of the sum of all of the values $\hat{u}(\xi + 2\pi j/h)$. This result is known as the **Poisson summation formula** or the **aliasing formula**:

ALIASING FORMULA

Theorem 2.7. *Let $u \in L^2$ be sufficiently smooth [?], with Fourier transform $\hat{u}$, and let $v \in \ell_h^2$ be the restriction of u to the grid $h\mathbb{Z}$. Then*

$$\hat{v}(\xi) = \sum_{j=-\infty}^{\infty} \hat{u}(\xi + 2\pi j/h), \qquad \xi \in [-\pi/h, \pi/h]. \tag{2.2.10}$$

Proof. Not yet written. See P. Henrici, *Applied and Computational Complex Analysis, v. 3*, Wiley, 1986. ∎

In applications, we are very often concerned with functions v obtained by discretization, and it will be useful to know how much the Fourier transform is affected in the process. Theorems 2.4 and 2.7 combine to give the following answers to this question:

EFFECT OF DISCRETIZATION ON THE FOURIER TRANSFORM

Theorem 2.8. *Let v be the restriction to the grid $h\mathbb{Z}$ of a function $u \in L^2$. The following estimates hold uniformly for all $\xi \in [-\pi/h, \pi/h]$, or a forteriori, for ξ in any fixed interval $[-A, A]$.*

(a) If u has $p-1$ continuous derivatives in L^2 for some $p \geq 1$ [?], and a pth derivative in L^2 that has bounded variation, then

$$|\hat{v}(\xi) - \hat{u}(\xi)| = O(h^{p+1}) \qquad \text{as } h \to 0. \tag{2.2.11}$$

(b) If u has infinitely many continuous derivatives in L^2, then

$$|\hat{v}(\xi) - \hat{u}(\xi)| = O(h^M) \qquad \text{as } h \to 0 \quad \text{for all } M. \tag{2.2.12}$$

(c) If u can be extended to an analytic function of $z = x+iy$ in the complex strip $|\operatorname{Im} z| < a$ for some $a > 0$, with $\|u(\cdot + iy)\| \leq \text{const}$ uniformly for each $-a < y < a$, then for any $\epsilon > 0$,

$$|\hat{v}(\xi) - \hat{u}(\xi)| = O(e^{-\pi(a-\epsilon)/h}) \qquad \text{as } h \to 0. \tag{2.2.13}$$

(d) If u can be extended to an entire function of $z = x+iy$ with $u(z) = O(e^{a|z|})$ as $|z| \to \infty$ ($z \in \mathbb{C}$) for some $a > 0$, then

$$\hat{v}(\xi) = \hat{u}(\xi) \qquad \text{provided } h < \pi/a. \tag{2.2.14}$$

In part *(c)*, $u(\cdot + iy)$ denotes a function of x, namely $u(x+iy)$ with x interpreted as a variable and y as a fixed parameter.

Proof. In each part of the theorem, $u(x)$ is smooth enough for Theorem 2.7 to apply, which gives the identity

$$\hat{v}(\xi) - \hat{u}(\xi) = \sum_{j=1}^{\infty} \hat{u}(\xi + 2\pi j/h) + \hat{u}(\xi - 2\pi j/h). \tag{2.2.15}$$

Note that since $\xi \in [-\pi/h, \pi/h]$, the arguments of $\hat{u}$ in this series have magnitudes at least π/h, $2\pi/h$, $3\pi/h, \ldots$.

For part *(a)*, Theorem 2.4*(a)* asserts that $|\hat{u}(\xi)| \leq C_1 |\xi|^{-p-1}$ for some constant C_1. With (2.2.15) this implies

$$|\hat{v}(\xi) - \hat{u}(\xi)| \leq C_1 \sum_{j=1}^{\infty} (j\pi/h)^{-p-1} = C_2 h^{p+1} \sum_{j=1}^{\infty} j^{-p-1}.$$

Since $p \geq 1$ this sum converges to a constant, which implies (2.2.11) as required.

Part *(b)* follows from part *(a)*.

For part *(c)*, ... [?]

For part *(d)*, note that if $h < \pi/a$, then $\pi/h > a$. Thus (2.2.15) reduces to 0 for all $\xi \in [-\pi/h, \pi/h]$, as claimed. ∎

Note that part *(d)* of Theorem 2.8 asserts that on a grid of size h, the semidiscrete Fourier transform is exact for band-limited functions containing energy only at wave numbers

$|\xi|$ smaller than π/h—the **Nyquist wave number**, corresponding to two grid points per wavelength. This two-points-per-wavelength restriction is famous among engineers, and has practical consequences in everything from fighter planes to compact disc players. When we come to discretize solutions of partial differential equations, two points per wavelength will be the coarsest resolution we can hope for under normal circumstances.

EXERCISES

▷ *2.2.1.*
(a) Prove (2.2.2): $\ell_h^1 \subseteq \ell_h^2 \subseteq \ell_h^\infty$.
(b) Give examples to show that these inclusions are proper: $\ell_h^2 \not\subseteq \ell_h^1$ and $\ell_h^\infty \not\subseteq \ell_h^2$.
(c) Give examples to show that neither inclusion in *(a)* holds for functions on continuous domains: $L^1 \not\subseteq L^2$ and $L^2 \not\subseteq L^\infty$.

▷ *2.2.2.* Let $\delta, \mu : \ell_h^2 \to \ell_h^2$ be the discrete differentiation and smoothing operators defined by

$$(\delta v)_j = \frac{1}{2h}(v_{j+1} - v_{j-1}), \qquad (\mu v)_j = \tfrac{1}{2}(v_{j-1} + v_{j+1}). \tag{2.2.16}$$

(a) Show that δ and μ are equivalent to convolutions with appropriate sequences $d, m \in \ell_h^2$. (Be careful with factors of h.)
(b) Compute the Fourier transforms $\hat{d}$ and $\hat{m}$. How does $\hat{d}$ compare to the transform of the exact differentiation operator for functions defined on $\mathbb{R}$ (Theorem 2.2f)? Illustrate this comparison with a sketch of $\hat{d}(\xi)$ against ξ.
(c) Compute $\|d\|$, $\|\hat{d}\|$, $\|m\|$, and $\|\hat{m}\|$, and verify Parseval's equality.
(d) Compute the Fourier transforms of the convolution sequences corresponding to the iterated operators δ^p and μ^p $(p \geq 2)$. Discuss how these results relate to the rule of thumb discussed in the last section: the smoother the function, the more rapidly its Fourier transform decays as $|\xi| \to \infty$. What imperfection in μ does this analysis bring to light?

▷ *2.2.3. Continuation of Exercise 2.1.6.* Let v be the discretization on the grid $h\mathbb{Z}$ of the function $u(x) = (1+x^2)^{-1}$.
(a) Determine $\hat{v}(\xi)$. (*Hint:* calculating it from the definition (2.2.6) is very difficult.)
(b) How fast does $\hat{v}(\xi)$ approach $\hat{u}(\xi)$ as $h \to 0$? Give a precise answer based on *(a)*, then compare your answer with the prediction of Theorem 2.8.
(c) What would the answer to *(b)* have been if the roles of u and $\hat{u}$ had been interchanged—that is, if v had been the discretization not of $u(x)$ but of its Fourier transform?

▷ *2.2.4. Integration by the trapezoid rule.* A function $u \in L^2 \cap L^1$ can be integrated approximately by the trapezoid rule:

$$I = \int_{-\infty}^{\infty} u(x)\,dx \quad \approx \quad I_h = h \sum_{j=-\infty}^{\infty} u(x_j). \tag{2.2.17}$$

This is an infinite sum, but in practice one might delete the tails if u decays sufficiently rapidly as $|x| \to \infty$. (This idea leads to excellent quadrature algorithms even for finite intervals, which are first transformed to the real axis by a change of variables; for a survey

see M. Mori, "Quadrature formulas obtained by variable transformation and the DE-rule," *J. Comp. Appl. Math. 12 & 13* (1985), 119–130.)

As $h \to 0$, how good an approximation is I_h to the exact integral I? Of course the answer will depend on the smoothness of $u(x)$.

(a) State how I_h is related to the semidiscrete Fourier transform.

(b) Give a bound for $|I_h - I|$ based on the theorems of this section.

(c) In particular, what can you say about $|I_h - I|$ for the function $u(x) = e^{-x^2}$?

(d) Show that your bound can be improved in a certain sense by a factor of 2.

▶ *2.2.5.* Draw a plot of $\sin n$ as a function of n, where n ranges over the integers (*not* the real numbers) from 1 to 5000. (That is, your plot should contain 5000 dots; in Matlab this can be done in one line.) Explain why the plot looks the way it does to the human eye, and what this has to do with aliasing. Make your explanation precise and quantitative. (See G. Strang, *Calculus*, Wellesley-Cambridge Press, 1991.)

2.3. Interpolation and sinc functions

[This section is not written yet, but here's a hint as to what will be in it.]

If δ_j is the Kronecker delta function

$$\delta_j = \begin{cases} 1 & \text{if } j = 0, \\ 0 & \text{if } j \neq 0, \end{cases} \tag{2.3.1}$$

then (2.2.6) gives the semidiscrete Fourier transform

$$\hat{\delta}_j(\xi) = h \qquad \text{(for all } \xi\text{)}.$$

If we now apply the inverse transform formula (2.2.7), we find after a little algebra

$$\delta_j = \frac{\sin(\pi x_j/h)}{\pi x_j/h}, \tag{2.3.2}$$

at least for $j \neq 0$. Since x_j/h is a nonzero integer for each $j \neq 0$, the sines are zero and this formula matches (2.3.1).

Suppose, however, that we evaluate (2.3.2) not just for $x = x_j$ but for all values $x \in \mathbb{R}$. Then we've got a sinc function again, one that can be called a **grid sinc function**:

$$S_h(x) = \frac{\sin(\pi x/h)}{\pi x/h}. \tag{2.3.3}$$

The plot of $S_h(x)$ is the same as the upper-right plot of Figure 2.1.2, except scaled so that the zeros are on the grid (i.e. at integer multiples of h). Obviously $S_h(x)$ is a continuous interpolant to the discrete delta function δ_j. Which one? It is the unique **band-limited interpolant**, band-limited in the sense that its Fourier transform $\widehat{S_h}(\xi)$ is zero for $\xi \notin [-\pi/h, \pi/h]$. (Proof: by construction it's band-limited in that way, and uniqueness can be proved via an argument by contradiction, making use of Parseval's equality (2.2.8).)

More generally, suppose we have an arbitrary grid function v_j (well, not quite arbitrary; we'll need certain integrability assumptions, but let's forget that for now). Then the **band-limited interpolant** to v_j is the unique function $v(x)$ defined for $x \in \mathbb{R}$ with $v(x_j) = v_j$ and $\hat{v}(\xi) = 0$ for $\xi \notin [-\pi/h, \pi/h]$. It can be derived in two equivalent ways:

Method 1: Fourier transform. Given v_j, compute the semidiscrete Fourier transform $\hat{v}(\xi)$. Then invert that transform, and evaluate the resulting formula for all x rather than just on the grid.

Method 2: linear combination of sinc functions. Write

$$v_j = \sum_{m=-\infty}^{\infty} v_m \delta_{m-j},$$

and then set

$$v(x) = \sum_{m=-\infty}^{\infty} v_m S_h(x - x_m).$$

The equivalence of Methods 1 and 2 is trivial; it follows from the linearity and translation-invariance of all the processes in question.

The consideration of band-limited interpolation is a good way to get insight into the Aliasing Formula presented as Theorem 2.7. (In fact, maybe that should go in this section.) The following schema summarizes everything. Study it!

$$\begin{array}{ccc} u(x) & \xrightarrow{F.T.} & \hat{u}(\xi) \\ \downarrow \textit{DISCRETIZE} & & \updownarrow \begin{array}{l}\textit{ALIASING}\\ \textit{FORMULA}\end{array} \\ v_j & \xrightarrow{F.T.} & \hat{v}(\xi) \\ \downarrow \begin{array}{l}\textit{BAND-LIMITED}\\ \textit{INTERPOLATION}\end{array} & & \downarrow \begin{array}{l}\textit{ZERO HIGH}\\ \textit{WAVE NOS.}\end{array} \\ v(x) & \xrightarrow{F.T.} & \underline{\hat{v}}(\xi) \end{array}$$

The **Gibbs phenomenon** is a particular phenomenon of band-limited interpolation that has received much attention. After an initial discovery by Wilbraham in 1848, it was made famous by Michelson in 1898 in a letter to *Nature*, and then by an analysis by Gibbs in *Nature* the next year. Gibbs showed that if the step function

$$u(x) = \begin{cases} +1 & x < 0, \\ -1 & x > 0 \end{cases}$$

is sampled on a grid and then interpolated in the band-limited manner, then the resulting function $v(x)$ exhibits a ringing effect: it overshoots the limits ± 1 by about 9%, achieving a maximum amplitude

$$\int_{-1}^{1} \frac{\sin(\pi y)}{\pi y} dy \approx 1.089490. \tag{2.3.4}$$

The ringing is scale-invariant; it does not go away as $h \to 0$. In the final text I will illustrate the Gibbs phenomenon and include a quick derivation of (2.3.4).

2.4. The discrete Fourier transform

Note: although the results of the last two sections will be used throughout the remainder of the book, the material of the present section will not be needed until Chapters 8 and 9.

For the discrete Fourier transform, both x and ξ inhabit discrete, bounded domains—or if your prefer, they are periodic functions defined on discrete, unbounded domains. Thus there is a pleasing symmetry here, as with the Fourier transform, that was missing in the semidiscrete case.

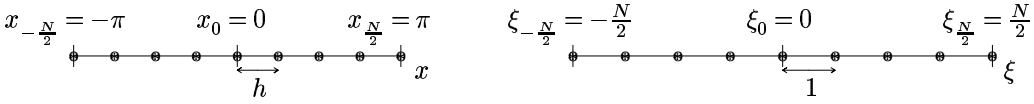


Figure 2.4.1. Space and wave number domains for the discrete Fourier transform.

For the fundamental spatial domain we shall take $[-\pi,\pi)$, as illustrated in Figure 2.4.1. Let N be a positive even integer, set

$$h = \frac{2\pi}{N} \qquad (N \text{ even}), \tag{2.4.1}$$

and define $x_j = jh$ for any j. The grid points in the fundamental domain are

$$x_{-N/2} = -\pi, \ldots, \quad x_0 = 0, \ldots, \quad x_{N/2-1} = \pi - h.$$

An invaluable identity to keep in mind is this:

$$\frac{N}{2} = \frac{\pi}{h}. \tag{2.4.2}$$

Let ℓ_N^2 denote the set of functions on $\{x_j\}$ that are N-periodic with respect to j, i.e, 2π-periodic with respect to x, with the norm

$$\|v\| = \Big[h \sum_{j=-N/2}^{N/2-1} |v_j|^2\Big]^{1/2}. \tag{2.4.3}$$

(Since the sum is finite, the norm is finite, so every function of the required type is guaranteed to belong to ℓ_N^2—and to ℓ_N^1 and ℓ_N^∞.) The **discrete Fourier transform (DFT)** of a function $v \in \ell_N^2$ is defined by

$$\hat{v}(\xi) = (\mathcal{F}_N v)(\xi) = h \sum_{j=-N/2}^{N/2-1} e^{-i\xi x_j} v_j, \qquad \xi \in \mathbb{Z}.$$

Since the spatial domain is periodic, the set of wave numbers ξ is discrete, and in fact ξ ranges precisely over the set of integers $\mathbb{Z}$. Thus it is natural to use ξ as a subscript,

$$\hat{v}_\xi = (\mathcal{F}_N v)_\xi = h \sum_{j=-N/2}^{N/2-1} e^{-i\xi jh} v_j, \qquad \xi \in \mathbb{Z},$$

and since $h = 2\pi/N$, $\hat{v}_\xi$ is N-periodic as a function of ξ. We shall take $[-N/2, N/2]$ as the fundamental domain of wave numbers, and let L_N^2 denote the set of N-periodic functions on the grid $\mathbb{Z}$, with norm

$$\|\hat{v}\| = \Big[\sum_{\xi=-N/2}^{N/2-1} |\hat{v}_\xi|^2 \Big]^{1/2}. \tag{2.4.4}$$

This is nonstandard notation, for an upper case L is normally reserved for a family of functions defined on a continuum. We use it here to highlight the relationship of the discrete Fourier transform with the semidiscrete Fourier transform.

The convolution of two functions in ℓ_N^2 is defined by

$$(v * w)_m = h \sum_{j=-N/2}^{N/2-1} v_{m-j} w_j = h \sum_{j=-N/2}^{N/2-1} v_j w_{m-j}. \tag{2.4.5}$$

Again, since the sum is finite, there is no question of convergence.

Here is a summary of the discrete Fourier transform:

THE DISCRETE FOURIER TRANSFORM

Theorem 2.9. *If $v \in \ell_N^2$, then the discrete Fourier transform*

$$\hat{v}_\xi = (\mathcal{F}_N v)_\xi = h \sum_{j=-N/2}^{N/2-1} e^{-i\xi jh} v_j, \qquad -\frac{N}{2} \le \xi \le \frac{N}{2} - 1 \tag{2.4.6}$$

belongs to L_N^2, and v can be recovered from $\hat{v}$ by the **inverse discrete Fourier transform**

$$v_j = (\mathcal{F}_N^{-1} \hat{v})_j = \frac{1}{2\pi} \sum_{\xi=-N/2}^{N/2-1} e^{i\xi jh} \hat{v}_\xi. \tag{2.4.7}$$

The ℓ_N^2-norm of v and the L_N^2-norm of $\hat{v}$ are related by **Parseval's equality**,

$$\|\hat{v}\| = \sqrt{2\pi}\|v\|. \tag{2.4.8}$$

If $v, w \in \ell_N^2$, then $\widehat{v * w}$ satisfies

$$(\widehat{v * w})_\xi = \hat{v}_\xi \hat{w}_\xi. \tag{2.4.9}$$

As with the other Fourier transforms we have considered, the following properties of the discrete Fourier transform will be useful. Once again we take advantage of the fact that $\hat{v}(\xi)$ can be treated as a periodic function defined for all $\xi \in \mathbb{Z}$.

PROPERTIES OF THE DISCRETE FOURIER TRANSFORM

Theorem 2.10. *Let $v, w \in \ell_N^2$ have discrete Fourier transforms $\hat{v}, \hat{w}$. Then:*

(a) Linearity. $\mathcal{F}_N\{v+w\}(\xi) = \hat{v}(\xi) + \hat{w}(\xi)$; $\mathcal{F}_N\{cv\}(\xi) = c\hat{v}(\xi)$.

(b) Translation. If $j_0 \in \mathbb{Z}$, then $\mathcal{F}_N\{v_{j+j_0}\}(\xi) = e^{i\xi x_{j_0}} \hat{v}(\xi)$.

(c) Modulation. If $\xi_0 \in \mathbb{Z}$, then $\mathcal{F}_N\{e^{i\xi_0 x_j} v_j\}(\xi) = \hat{v}(\xi - \xi_0)$.

(e) Conjugation. $\mathcal{F}_N\{\overline{v}\}(\xi) = \overline{\hat{v}(-\xi)}$.

(g) Inversion. $\mathcal{F}_N^{-1}\{v\}(\xi) = \dfrac{1}{2\pi h} \hat{v}(-\xi)$.

An enormously important fact about discrete Fourier transforms is that they can be computed rapidly by the recursive algorithm known as the **fast Fourier transform** (FFT).* A direct implementation of (2.4.6) or (2.4.7) requires $\Theta(N^2)$ arithmetic operations, but the

*The fast Fourier transform was discovered by Gauss in 1805 at the age of 28, but although he wrote a paper on the subject, he did not publish it, and the idea was more or less lost until its celebrated

FFT is based upon a recursion that reduces this figure to $\Theta(N \log N)$. We shall not describe the details of the FFT here, but refer the reader to various books in numerical analysis, signal processing, or other fields. However, to illustrate how simple an implementation of this idea may be, Figure 2.4.2 reproduces the original Fortran program that appeared in a 1965 paper by Cooley, Lewis, and Welch.† Assuming that N is a power of 2, it computes $2\pi\mathcal{F}_N^{-1}$, in our notation: the vector $A(1:N)$ represents $\hat{v}_0,\ldots,\hat{v}_{N-1}$ on input and $2\pi v_0,\ldots,2\pi v_{N-1}$ on output.

```
      subroutine fft(a,m)
      complex a(1),u,w,t
      n = 2**m                          do 20 l = 1,m
      nv2 = n/2                           le = 2**l
      nm1 = n-1                           le1 = le/2
      j=1                                 u = 1.
      do 7 i = 1,nm1                      ang = 3.14159265358979/le1
        if (i.ge.j) goto 5                w = cmplx(cos(ang),sin(ang))
        t = a(j)                          do 20 j = 1,le1
        a(j) = a(i)                         do 10 i = j,n,le
        a(i) = t                              ip = i+le1
5     k = nv2                                 t = a(ip)*u
6     if (k.ge.j) goto 7                      a(ip) = a(i)-t
        j = j-k                     10        a(i) = a(i)+t
        k = k/2                     20      u = u*w
        goto 6                            return
7     j = j+k                             end
```

Figure 2.4.2. Complex inverse FFT program of Cooley, Lewis, and Welch (1965).

As mentioned above, this program computes the *inverse* Fourier transform according to our definitions, times 2π. The same program can be used for the forward transform by making use of the following identity:

rediscovery by Cooley and Tukey in 1965. (See M. T. Heideman, et al., "Gauss and the history of the fast Fourier transform," *IEEE ASSP Magazine*, October 1984.) Since then, fast Fourier transforms have changed prevailing computational practices in many areas.

†Before publication, permission to print this program will be secured.

$$\hat{v}_\xi = \overline{\mathcal{F}_N\{\bar{v}\}(-\xi)} = 2\pi h \overline{\mathcal{F}_N^{-1}\{\bar{v}\}(\xi)}. \tag{2.4.10}$$

These equalities follow from parts *(e)* and *(g)* of Theorem 2.10, respectively.

2.5. Vectors and multiple space dimensions

Fourier analysis generalizes with surprising ease to situations where the independent variable x and/or the dependent variable u are vectors. We shall only sketch the essentials, which are based on the following two ideas:

- If x is a d-vector, then the dual variable ξ is a d-vector too, and the Fourier integral is a multiple integral involving the inner product $x \cdot \xi$;
- If u is an N-vector, then its Fourier transform $\hat{u}$ is an N-vector too, and is defined componentwise.

As these statements suggest, our notation will be as follows:

$$\begin{aligned} d &= \text{ number of space dimensions: } x = (x_1, \ldots, x_d)^T, \\ N &= \text{ number of dependent variables: } u = (u_1, \ldots, u_N)^T. \end{aligned}$$

Both ξ and $\hat{u}$ become vectors of the same dimensions,

$$\xi = (\xi_1, \ldots, \xi_d)^T, \qquad \hat{u} = (\hat{u}_1, \ldots, \hat{u}_N)^T,$$

and ξx becomes the dot product $\xi \cdot x = \xi_1 x_1 + \cdots + \xi_d x_d$. The formulas for the Fourier transform and its inverse read

$$\begin{aligned} \hat{u}(\xi) = (\mathcal{F}u)(\xi) &= \int e^{-i\xi \cdot x} u(x)\, dx \\ &= \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} e^{-i\xi \cdot x} u(x)\, dx_1 \cdots dx_d \end{aligned} \tag{2.5.1}$$

for $\xi \in \mathbb{R}^d$, and

$$\begin{aligned} u(x) = (\mathcal{F}^{-1}\hat{u})(x) &= (2\pi)^{-d} \int e^{i\xi \cdot x} \hat{u}(\xi) d\xi \\ &= (2\pi)^{-d} \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} e^{i\xi \cdot x} \hat{u}(\xi)\, d\xi_1 \cdots d\xi_d \end{aligned} \tag{2.5.2}$$

for $x \in \mathbb{R}^d$. In other words, u and $\hat{u}$ are related componentwise:

$$\hat{u}(\xi) = (\hat{u}^{(1)}(\xi), \ldots, \hat{u}^{(N)}(\xi))^T. \tag{2.5.3}$$

If the vector L^2-norm is defined by

$$\|u\|^2 = \int \|u(x)\|^2 dx = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} \|u(x)\|^2 dx_1 \cdots dx_d, \tag{2.5.4}$$

where the symbol $\|\cdot\|$ in the integrand denotes the 2-norm on vectors of length N, then **Parseval's equality** for vector Fourier transforms takes the form

$$\|\hat{u}\| = (2\pi)^{d/2}\|u\|. \tag{2.5.5}$$

The set of vector functions with bounded vector 2-norms can be written simply as $(L^2)^N$.

Before speaking of convolutions, we have to go a step further and allow $u(x)$ and $\hat{u}(\xi)$ to be $M \times N$ matrices rather than just N-vectors. The definitions above extend to this further case unchanged, if the symbol $\|\cdot\|$ in the integrand of (2.5.4) now represents the 2-norm (largest singular value) of a matrix. If $u(x)$ is an $M \times P$ matrix and $v(x)$ is a $P \times N$ matrix, then the convolution $u*v$ is defined by

$$\begin{aligned}(u*v)(x) &= \int u(x-y)v(y)\,dy \\ &= \int u(y)v(x-y)\,dy \\ &= \int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty} u(x-y)v(y)\,dy_1\cdots dy_d,\end{aligned} \tag{2.5.6}$$

and it satisfies

$$\widehat{u*v}(\xi) = \hat{u}(\xi)\hat{v}(\xi). \tag{2.5.7}$$

Since matrices do not commute in general, it is no longer possible to exchange u and v as in (2.1.4).

This generalization of Fourier transforms and convolutions to matrix functions is far from idle, for we shall need it for the Fourier analysis of multistep finite difference approximations such as the leap frog formula.

Similar generalizations of our scalar results hold for semidiscrete and discrete Fourier transforms.

EXERCISES

▷ *2.5.1.* What is the Fourier transform of the vector function

$$u(x) = \Big(\frac{\sin x}{x}, \frac{\sin 2x}{2x}\Big)^T,$$

defined for $x \in \mathbb{R}$?

▷ *2.5.2.* What is the Fourier transform of the scalar function

$$u(x) = e^{-\frac{1}{2}(x_1^2+x_2^2)},$$

defined for $x = (x_1, x_2)^T \in \mathbb{R}^2$?

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

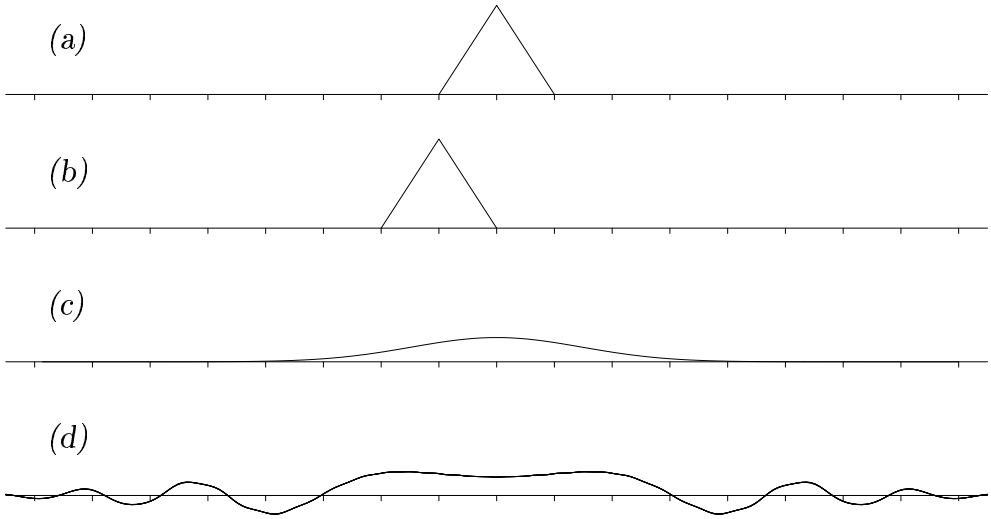


Figure 3.1.1. Evolution to $t = 1$ of *(a)* hat-shaped initial data under *(b)* the wave equation (3.1.1), *(c)* the heat equation (3.1.3), and *(d)* the Schrödinger equation (3.1.5) (the real part is shown).

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,

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

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

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

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,

$$\omega = \omega(\xi), \tag{3.1.8}$$

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 $\mathrm{Im}\,\omega > 0$, 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 $-1/a(x,t)$:

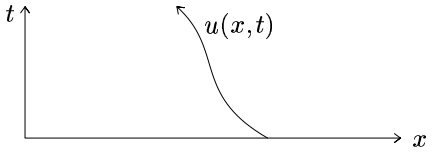


Figure 3.1.2

(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.

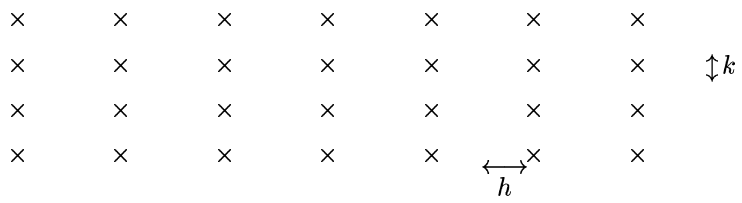


Figure 3.2.1. Regular finite difference grid in x and t.

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

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

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:

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

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 $k = 2h/5$, $h = 1/2, 1/4, \ldots, 1/64$. 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 $k = 2h^2/5$, $h = 1/2, 1/4, \ldots, 1/16$. 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 $k = \frac{1}{2}h$, $h = 1/4, 1/8, \ldots, 1/64$. 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:

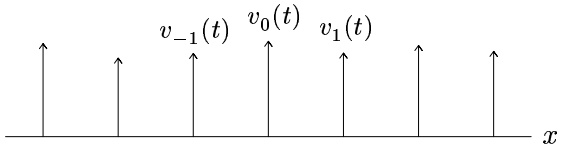


Figure 3.3.1. The "method of lines"—semidiscretization of a time-dependent PDE.

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 $200 \times 200 \times 200$ 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:

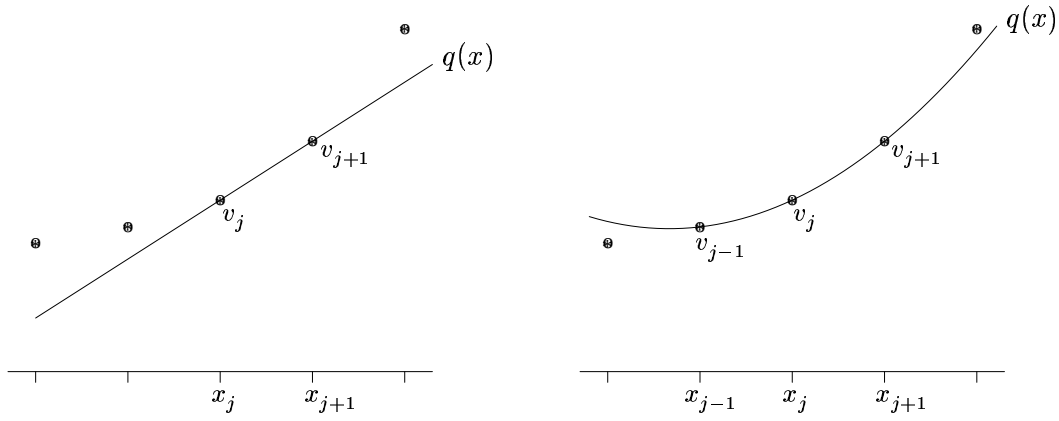
(1) Interpolate the data;
(2) Differentiate the interpolant at the mesh points. (3.3.3)

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 $x_0, x_1, \ldots, x_{n_{\max}}$ 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 $x_0, x_1, \ldots, x_n$ 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 $m_{\max} \geq 0$ and $n_{\max} \geq m_{\max}$, the following algorithm computes coefficients of finite difference approximations in arbitrary distinct points $x_0, \ldots, x_{n_{\max}}$ to $\partial^m/\partial x^m$ $(0 \leq m \leq m_{\max})$ at $x = 0$, as described above:*

$c_{00}^0 := 1, \quad \alpha := 1$
for $n := 1$ **to** $n_{\max}$
  $\beta := 1$
  for $j := 0$ **to** $n-1$
    $\beta := \beta(x_n - x_j)$
    if $n \leq m_{\max}$ **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 $v_{j-p}, \ldots, v_{j+p}$ 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 $p \geq 1$, 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 $v_{j-p}, \ldots, v_{j+p}$, 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

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

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

$$\vdots$$

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

(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 $(\ldots, v_{-1}, v_0, v_1, , \ldots,)^T$, then $\delta_0 v$ becomes the left-multiplication of v by the *infinite* matrix of the form

$$\delta_0 := \frac{1}{h}\begin{pmatrix} 0 & \frac{1}{2} & & & \\ -\frac{1}{2} & 0 & \frac{1}{2} & & \\ & -\frac{1}{2} & 0 & \frac{1}{2} & \\ & & -\frac{1}{2} & 0 & \frac{1}{2} \\ & & & -\frac{1}{2} & 0 \end{pmatrix}. \tag{3.3.17}$$

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 $x_j = hs^j$, $s > 1$. 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

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

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*

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

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

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

In principle, the operator S might be implemented by computing a semi-discrete 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.

Chapter 4.
Accuracy, Stability, and Convergence

Mighty oaks from little acorns grow!
— *ANONYMOUS*

The problem of stability is pervasive in the numerical solution of partial differential equations. In the absence of computational experience, one would hardly be likely to guess that instability was an issue at all,* yet it is a dominant consideration in almost every computation. Its impact is visible in the nature of algorithms all across this subject—most basically, in the central importance of linear algebra, since stability so often necessitates the use of implicit or semi-implicit formulas whose implementation involves large systems of discrete equations.

The relationship between stability and convergence was hinted at by Courant, Friedrichs, and Lewy in the 1920's, identified more clearly by von Neumann in the 1940's, and brought into organized form by Lax and Richtmyer in the 1950's—the Lax Equivalence Theorem. After presenting an example, we shall begin with the latter, and then relate it to the CFL and von Neumann conditions. After that we discuss the important problem of determining stability of the method of lines. For problems that lead to normal matrices, it is enough to make sure that the spectra of the spatial discretization operators lie within a distance $O(k)$ of the stability region of the time-stepping formula, but if the matrices are not normal, one has to consider pseudospectra instead.

The essential issues of this chapter are the same as those that came up for ordinary differential equations in Sections 1.5–1.7. For partial differential equations, however, the details are more complicated, and more interesting.

In addition to Richtmyer and Morton, a good reference on the material of this chapter is V. Thomée, "Stability theory for partial difference operators," *SIAM Review 11* (1969), 152–195.

*In particular, L. F. Richardson, the originator of finite-difference methods for partial differential equations, did not discover instability; see his book *Weather Prediction by Numerical Processes*, Cambridge University Press, 1922 (!), reprinted by Dover in 1965.

4.1. An example

Consider the model partial differential equation

$$u_t = u_x, \qquad x \in \mathbb{R}, \quad t \geq 0 \tag{4.1.1}$$

together with initial data

$$u(x,0) = \begin{cases} \cos^2 x & |x| \leq \frac{\pi}{2}, \\ 0 & |x| \geq \frac{\pi}{2}. \end{cases} \tag{4.1.2}$$

Let us solve this initial-value problem numerically by the leap frog formula (3.2.2), with space and time steps

$$h = 0.04\pi, \qquad k = \lambda h,$$

where λ is a constant known as the **mesh ratio.** Thus the leap frog formula takes the form

$$v_j^{n+1} = v_j^{n-1} + \lambda(v_{j+1}^n - v_{j-1}^n), \tag{4.1.3}$$

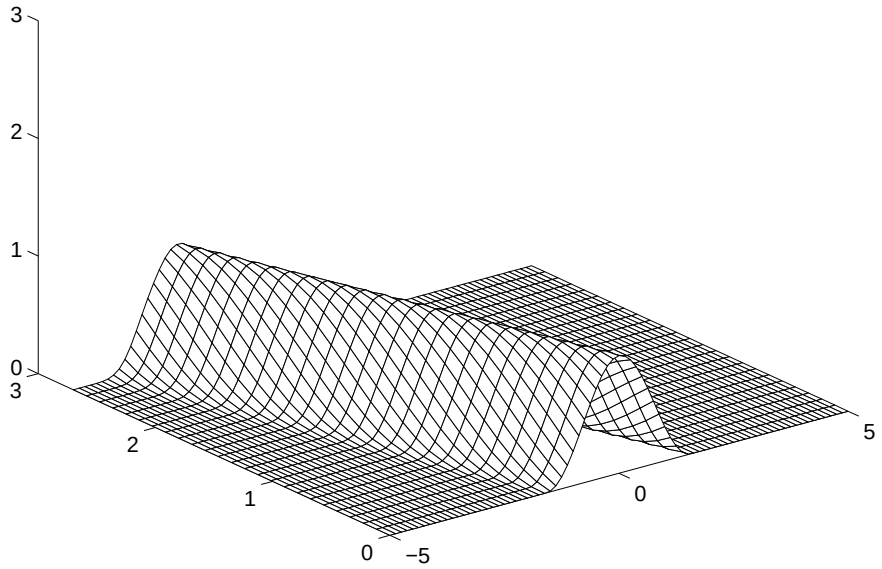
with the bump in the initial function represented by 25 grid points. The starting values at $t = 0$ and k will both be taken from the exact solution $u(x,t) = u(x+t,0)$.

Figure 4.1.1 shows computed results with $\lambda = 0.9$ and $\lambda = 1.1$, and they are dramatically different. For $\lambda < 1$ the leap frog formula is stable, generating a left-propagating wave as expected. For $\lambda > 1$ it is unstable. The errors introduced at each step are not much bigger than before, but they grow exponentially in subsequent time steps until the wave solution is obliterated by a sawtooth oscillation with 4 points per wavelength. This rapid blow-up of a sawtooth mode is typical of unstable finite difference formulas.

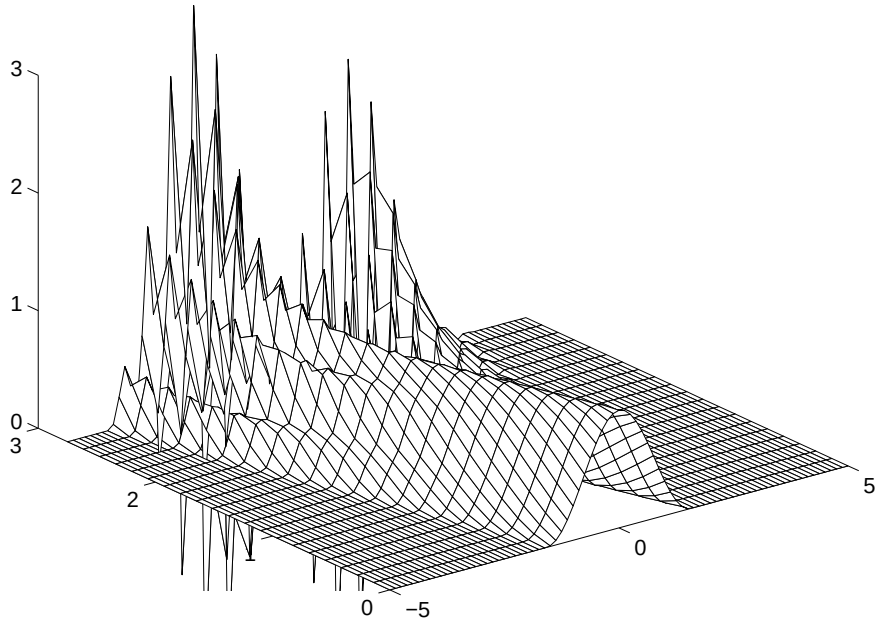
Although rounding errors can excite an instability, more often it is discretization errors that do so, and this particular experiment is quite typical in this respect. Figure 4.1.1 would have looked the same even if the computation had been carried out in exact arithmetic.

This chapter is devoted to understanding instability phenomena in a general way. Let us briefly mention how each of the sections to follow relates to the particular example of Figure 4.1.1.

First, §4.2 presents the celebrated Lax Equivalence Theorem: a consistent finite difference formula is convergent if and only if it is stable. Our example



(a) $\lambda = 0.9$



(b) $\lambda = 1.1$

Figure 4.1.1. Stable and unstable leap frog approximations to $u_t = u_x$.

(a) largest value $2i\lambda \sin \xi h$ *(b)* corresponding amplification factors z

Figure 4.1.2. Right-hand sides and corresponding amplification factors z of (4.1.6). The circles correspond to Figure 4.1.1a and the crosses to Figure 4.1.1b.

is consistent for any λ, but stable only for $\lambda < 1$. Thus as Figure 4.1.1 suggests, the numerical results would converge to the correct solution as $h, k \to 0$ in case *(a)*, but not in case *(b)*.

Next, §4.3 presents the CFL condition: a finite difference formula can be stable only if its numerical domain of dependence is at least as large as the mathematical domain of dependence. In the space-time grid of Figure 4.1.1*(b)*, information travels under the leap frog model at speed at most $(1.1)^{-1}$, which is less than the propagation speed 1 for the PDE itself. Thus *something* had to go wrong in that computation.

Section 4.4 presents the von Neumann approach to stability: Fourier analysis and amplification factors. This is the workhorse of stability analysis, and the foundations were given already in §§3.5,3.6. For our leap frog model (4.1.3), inserting the trial solution

$$v_j^n = z^n e^{i\xi x_j}, \qquad \xi \in \mathbb{R} \tag{4.1.4}$$

leads to the equation

$$z = z^{-1} + \lambda(e^{i\xi h} - e^{-i\xi h}), \tag{4.1.5}$$

that is,

$$z - z^{-1} = 2i\lambda \sin \xi h. \tag{4.1.6}$$

This is a quadratic equation in z with two complex roots (in general). As ξ varies, the right-hand side ranges over the complex interval $[-2i\lambda, 2i\lambda]$. For $|\lambda| \le 1$ this interval is a subset of $[-2i, 2i]$, and so the roots z lie in symmetric

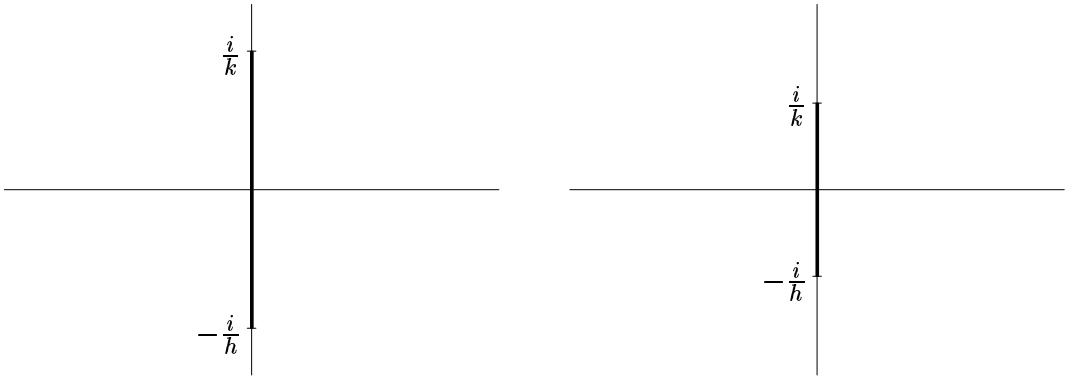
positions on the unit circle $|z| = 1$ (Figure 4.1.2). For $|\lambda| > 1$, on the other hand, some values of ξ lead to right-hand side values not contained in $[-2i, 2i]$, and the roots z then move off the unit circle—one inside and one outside. The root z outside the circle amounts to an "amplification factor" greater than 1, and causes instability. The largest z occurs for $\sin \xi h = \pm 1$, which explains why the instability of Figure 4.1.1*(b)* had 4 points per wavelength.

Finally, §§4.5,4.6 discuss stability analysis via stability regions for problems in the form of the method of lines. Our leap frog example can be interpreted as the midpoint rule in time coupled with the centered difference operator δ_0 in space. Figure 4.1.3*(a)* shows the stability region in the a-plane (not the $\bar{k} = ka$ plane) for the midpoint rule, repeated from Figure 1.7.1: it is the complex open interval $(-i/k, i/k)$ on the imaginary axis. Figure 4.1.3*(b)* shows the eigenvalues* of δ_0—its eigenfunctions are the functions $v_j = e^{i\xi x_j}$, $\xi \in \mathbb{R}$, with corresponding eigenvalues

$$\frac{1}{2h}(e^{i\xi h} - e^{-i\xi h}) = \frac{i}{h} \sin \xi h. \tag{4.1.7}$$

Thus the eigenvalues cover the complex interval $[-i/h, i/h]$. For absolute stability of the system of ODEs that arise in the method of lines, these eigenvalues must lie in the stability region, leading once again to the condition $h^{-1} < k^{-1}$, that is, $k < h$. Section 4.6 shows that this condition relates to true stability as well as to absolute stability.

*We are being careless with the term "eigenvalue." Since the functions $v_j = e^{i\xi x_j}$ do not belong to ℓ_h^2, they are not true eigenfunctions, and the proper term for the quantities (4.1.7) is "spectral values." However, this technicality is of little importance for our purposes, and goes away when one considers problems on a bounded interval or switches to the ℓ_h^∞ norm, so we shall ignore it and speak of "eigenvalues" anyway.



(a) Stability region for midpoint rule *(b)* Eigenvalues of δ_0

Figure 4.1.3. Absolute stability analysis of Example 4.1.1.

EXERCISES

▷ *4.1.1.* Determine the unstable mode that dominates the behavior of Figure 4.1.1(b). In particular, what is the factor by which the unstable solution is amplified from one step to the next?

4.2. The Lax Equivalence Theorem

[This section is rather sketchy at the moment, omitting some essential points of rigor as well as explanation, especially in two areas: the application of the operator A on a dense subspace rather than the whole space $\mathcal{B}$, and the relationship between continuous and discrete norms. For a more precise discussion see Richtmyer and Morton.]

The essential idea of the Lax Equivalence Theorem is this: for consistent *linear* finite difference models, stability is a necessary and sufficient condition for convergence. This is an analog of the Dahlquist Equivalence Theorem for ordinary differential equations (Theorem 1.10), except that the latter is valid for nonlinear problems too.

Aside from the assumption of linearity, the formulation of the Lax Equivalence Theorem is very general. Let $\mathcal{B}$ be a Banach space (a complete normed vector space) with norm denoted by $\|\cdot\|$. In applications of interest here, each element of $\mathcal{B}$ will be a function of one or more space variables x. Let $A:\mathcal{B}\to\mathcal{B}$ be a linear operator on this space. Here, A will be a differential operator. We are given the **initial value problem**

$$u_t(t) = Au(t), \quad 0 \le t \le T, \qquad u(0) = u_0, \tag{4.2.1}$$

where A is fixed but u_0 may range over all elements of $\mathcal{B}$. [Actually, A only has to be defined on a dense subset of $\mathcal{B}$.] This initial value problem is assumed to be **well-posed**, which means that a unique solution $u(t)$ exists for any initial data u_0 and $u(t)$ depends continuously upon the initial data.

EXAMPLE 4.2.1. Suppose we wish to solve $u_t = u_x$ for $x \in (-\infty,\infty)$, $t \ge 0$, with initial data $f(x)$, and we wish to look for solutions in the space L^2. In this case each $u(t)$ in (4.2.1) is a function of x, namely $u(x,t)$. (Technically we should not use the same symbol u in both places.) The Banach space $\mathcal{B}$ is L^2, A is the first-order differentiation operator ∂_x, and $u_0 = f$.

EXAMPLE 4.2.2. Suppose we wish to solve $u_t = u_{xx}$ for $x \in [-1,1]$, $t \ge 0$, with initial data $f(x)$ and boundary conditions $u(-1,t) = u(1,t) = 0$. Again, each $u(t)$ is a function of x. Now an appropriate Banach space might be $C_0[-1,1]$, the set of continuous functions of $x \in [-1,1]$ with value 0 at $x = \pm 1$, together with the supremum norm.

Abstractly, $u(t)$ is nothing more than an element in a Banach space $\mathcal{B}$, and this leaves room for applications to a wide variety of problems in differential equations. As in the examples above, $\mathcal{B}$ might be L^2 and A might be ∂_x or ∂_x^2, and homogeneous boundary conditions could be included by restricting $\mathcal{B}$ appropriately. More generally, $u(t)$ could be an vector-valued function of multiple space variables.

The next step is to define a general finite difference formula. In the abstract setting, this is a family of bounded linear operators

$$S_k : \mathcal{B} \to \mathcal{B}, \tag{4.2.2}$$

where the subscript k indicates that the coefficients of the finite difference formula depend on the time step. We advance from one step to the next by a single application of S_k:

$$v^{n+1} = S_k v^n, \quad \text{hence} \quad v^n = S_k^n v^0, \tag{4.2.3}$$

where S_k^n abbreviates $(S_k)^n$. (Watch out for the usual confusion of notation: the n in v^n is a superscript, while in S_k^n it is an exponent.) For simplicity, but no more essential reason, we are assuming that the problem (4.2.1) and hence S_k have no explicit dependence on t. But S_k does potentially depend on k, and this is an important point. On the other hand it does not explicitly depend on the space step h, for we adopt the following rule:

$$h \text{ is a fixed function } h(k) \text{ of } k.$$

For example, we might have $h = k/\lambda$ (λ constant) or $h = \sqrt{k/\sigma}$ (σ constant). If there are several space dimensions, each may have its own function $h_j(k)$. More generally, what we really need is grid(k), not $h(k)$; there is no need at all for the grid to be regular in the space dimensions.

EXAMPLE 4.2.3. *Lower-order terms.* For the UW model of $u_t = u_x$, the discrete solution operator is defined by $S_k v_j^n = v_j^n + \lambda(v_{j+1}^n - v_j^n)$, and if λ is held constant as $k \to 0$, this formula happens to be independent of k. The natural extension of UW to $u_t = u_x + u$, on the other hand, is $S_k v_j^n = v_j^n + \lambda(v_{j+1}^n - v_j^n) + k v_j^n$, and here there is an explicit dependence on k. This kind of k-dependence appears whenever the operator A involves derivatives of different orders.

Implicit or multistep finite difference formulas are not excluded by this formulation. As explained in §3.5, an implicit formula may still define a bounded operator S_k on an appropriate space such as ℓ_h^2, and a multistep formula can be reduced to an equivalent one-step formula by the introduction of a vector $w^n = (v^n, \ldots, v^{n+1-s})$.

Let us now be a bit more systematic in summarizing how the setup for the Lax Equivalence Theorem does or does not handle the various complications that make real problems differ from $u_t = u_x$ and $u_t = u_{xx}$.

- *Nonlinearity*
  The restriction here is essential: the Lax-Richtmyer theory does not handle nonlinear problems. (However, see various more recent papers by Sanz-Serna and others.)
- *Multiple space dimensions*
- *Implicit finite difference formulas*
  Both of these are included as part of the standard formulation.
- *Time-varying coefficients*
  Initial-value problems with time-varying coefficients are not covered in the description given here or in Richtmyer and Morton, but this restriction is not essential. The theory can be straightforwardly extended to such problems.
- *Boundary conditions*
- *Space-varying coefficients*
- *Lower-order terms*
  All of these are included as part of the standard formulation, and they have in common the property that they all lead to finite-difference approximations S_k that depend on k, as illustrated in Example 4.2.3 above.
- *Systems of equations*
- *Higher-order initial-value problems*
- *Multistep finite difference formulas*
  These are covered by the theory, if we make use of the usual device of reducing a one-step vector finite difference approximation to a first-order initial-value problem.

As in Chapter 1, we begin a statement of the Lax-Richtmyer theory by defining the order of accuracy and consistency of a finite difference formula.

$\{S_k\}$ *has* **order of accuracy** p *if*

$$\|u(t+k) - S_k u(t)\| = O(k^{p+1}) \qquad \text{as } k \to 0 \tag{4.2.4}$$

for any $t \in [0,T]$, *where* $u(t)$ *is any sufficiently smooth solution to the initial-value problem (4.2.1). It is* **consistent** *if it has order of accuracy* $p > 0$.

There are differences between this definition and the definition of order of accuracy for linear multistep formulas in §1.3. Here, the finite difference formula is applied not to an arbitrary function u, but to a solution of the initial value

problem. In practice, however, one still calculates order of accuracy by substituting formal Taylor expansions and determining up to what order the terms cancel (Exercise 4.2.1).

Another difference is that in the case of linear multistep formulas for ordinary differential equations, the order of accuracy was always an integer, and so consistency amounted to $p \geq 1$. Here, non-integral orders of accuracy are possible, although they are uncommon in practice.

EXAMPLE 4.2.4. *Non-integral orders of accuracy.* The finite difference approximation to $u_t = u_x$,

$$S_k v_j^n = v_j^n + \lambda(v_{j+1}^n - v_j^n) + k^{p+1}, \qquad \lambda = \text{constant} \tag{4.2.5}$$

is a (contrived) example with order of accuracy p, if p is any constant in the range $[0,1]$. A slightly less contrived example with order of accuracy p is

$$S_k v_j^n = v_j^n + \frac{k}{2h}(v_{j+1}^n - v_{j-1}^n), \quad \text{with} \quad h = k^{p/2} \tag{4.2.6}$$

for any $p \in [0,2]$.

As with ordinary differential equations, a finite difference formula for a partial differential equation is defined to be convergent if and only if it converges to the correct solution as $k \to 0$ for arbitrary initial data:

$\{S_k\}$ *is* **convergent** *if*

$$\lim_{\substack{k \to 0 \\ nk = t}} \| S_k^n u(0) - u(t) \| = 0 \tag{4.2.7}$$

for any $t \in [0,T]$, where $u(t)$ is the solution to the initial-value problem (4.2.1) for any initial data u_0.

Note that there is a big change in this definition from the definition of convergence for linear multistep formulas in §1.5. There, a fixed formula had to apply successfully to any differential equation and initial data. Here, the differential equation is fixed and only the initial data vary.

The definition of stability is slightly changed from the ordinary differential equation case, because of the dependence on k:

$\{S_k\}$ *is* **stable** *if for some $C > 0$,*

$$\| S_k^n \| \leq C \tag{4.2.8}$$

for all n and k such that $0 \leq nk \leq T$.

This bound on the operator norms $\|S_k^n\|$ is equivalent to

$$\|v^n\| = \|S_k^n v^0\| \le C\|v^0\|$$

for all $v^0 \in \mathcal{B}$ and $0 \le nk \le T$.

Here is the Lax Equivalence Theorem (compare Theorem 1.10):

LAX EQUIVALENCE THEOREM

Theorem 4.1. *Let $\{S_k\}$ be a consistent approximation to a well-posed linear initial-value problem (4.2.1). Then $\{S_k\}$ is convergent if and only if it is stable.*

Proof. *[Not yet written]* ∎

The following analog to Theorem 1.11 establishes that stable discrete formulas have the expected rate of convergence.

GLOBAL ACCURACY

Theorem 4.2. *Let a convergent approximation method of order of accuracy p be applied to a well-posed initial-value problem (4.2.1) [with some additional smoothness assumptions...]. Then the computed solution satisfies*

$$\|v(t) - u(t)\| = O(k^p) \qquad \text{as } k \to 0 \tag{4.2.9}$$

uniformly for all $t \in [0,T]$.

Proof. *[Not yet written]* ∎

A number of remarks should be made about the developments of this section.

• The definitions of convergence and of consistency make reference to the initial-value problem (4.2.1), but the definition of stability does not. One can ask whether a finite difference formula is stable or unstable without having any knowledge of what partial differential equation, if any, it approximates.

• No assumption has been made that the initial-value problem is hyperbolic or parabolic, so long as it is well-posed. Indeed, as far as the theory is concerned, the initial-value problem may not involve a differential operator or an x variable at all. (There are some useful applications of the Lax Equivalence Theorem of this kind, one of which involves the so-called Trotter product formula of mathematical physics.)

• As in §1.4, we are dealing with the limit in which t is fixed and $k \to 0$. The situation for $t \to \infty$ with k fixed will be discussed in §4.5.

• The definition of the finite difference formula (4.2.2) depends upon the mesh function $h = h(k)$. Consequently, whether the formula is stable or not may depend on $h(k)$ too. Examples are given in §4.4.

• As in §1.4, it is quite possible for an unstable finite difference formula to give convergent results for some initial data. For example, this might happen if the initial data were particularly smooth. But the definition of convergence requires good results for all possible initial data.

• Relatedly, the theory assumes exact arithmetic: discretization errors are included, but not rounding errors. The justification for this omission is that the same phenomena of stability and instability govern the propagation of both kinds of errors, so that in most situations a prediction based on discretization errors alone will be realistic. On the other hand, the fact that rounding errors occur in practice is one motivation for requiring convergence for all initial data. The initial data prescribed mathematically for a particular computation might be smooth, but the rounding errors superimposed on them will not be.

• Consistency, convergence, and stability are all defined in terms of a norm $\|\cdot\|$, and it must be the same norm in each case.

• It is quite possible for a finite-difference model to be stable in one norm and unstable in others; see §5.5. This may sound like a defect in the theory, but in such cases the instability is usually so weak that the behavior is more or less stable in practice.

We close this section by mentioning a general theorem that follows from the definition of stability:

PERTURBATIONS OF A STABLE FAMILY

Theorem 4.3. *Let $\{S_k\}$ be a stable family of operators, and let T_k be a family of operators satisfying $\|T_k\| = O(k)$ as $k \to 0$. Then $\{S_k + T_k\}$ is also a stable family.*

Proof. [not yet written; see Richtmyer & Morton, §3.9.]

Theorem 4.3 has an important corollary that generalizes Example 4.2.3:

LOWER ORDER TERMS

Theorem 4.4. *Let $\{S_k\}$ be a consistent finite difference approximation to a well-posed linear initial-value problem (4.2.1) in which A is a differential operator acting on one or more space variables. The stability of $\{S_k\}$ is determined only by the terms that relate to spatial derivatives.*

Proof. (sketch) If the finite difference approximation is consistent, then lower-order terms modify the finite difference formula only by terms of order $O(k)$, so by Theorem 4.3 they do not affect stability. ∎

EXAMPLE 4.2.4. If $\{S_k\}$ is a stable finite difference approximation of $u_t = u_x$ on $(-\infty, \infty)$, then the approximation remains stable if additional terms are added so that it becomes consistent with $u_t = u_x + f(x,u)$, for any function $f(x,u)$. The same is true for the equation $u_t = u_{xx}$. A consistent change from $u_t = u_{xx}$ to $u_t = u_{xx} + f(x,u,u_x)$, on the other hand, might destroy stability.

References:

- Chapters 4 and 5 of R. D. Richtmyer and K. W. Morton, *Difference Methods for Initial-Value Problems*, Wiley, 1967.

- P. D. Lax and R. D. Richtmyer, "Survey of the stability of linear finite difference equations," *Comm. Pure Appl. Math. 9* (1956), 267–293.

EXERCISES

▷ *4.2.1. Order of accuracy of the Lax-Wendroff formula.* Consider the Lax-Wendroff model of $u_t = u_x$ with $k/h = \lambda =$ constant. In analogy to the developments of §1.3, insert a formal power series for $u(x_{j+\Delta j}, t_{n+\Delta n})$ to obtain a formula for the leading-order nonzero term of the discretization error. Verify that the order of accuracy is 2.

4.3. The CFL condition

In 1928 Richard Courant, Kurt Friedrichs, and Hans Lewy, of the University of Göttingen in Germany, published a famous paper entitled "On the partial difference equations of mathematical physics."* This paper was written long before the invention of digital computers, and its purpose in investigating finite difference approximations was to apply them to prove existence of solutions to partial differential equations. But the "CFL" paper laid the theoretical foundations for practical finite difference computations, too, and in particular, it identified a fundamental necessary condition for convergence of any numerical scheme that has subsequently come to be known as the **CFL condition.**

What Courant, Friedrichs, and Lewy pointed out was that a great deal can be learned by considering the **domains of dependence** of a partial differential equation and of its discrete approximation. As suggested in Figure 4.3.1a, consider an initial-value problem for a partial differential equation, and let (x,t) be some point with $t>0$. (Despite the picture, the spatial grid need not be regular, or one-dimensional.) The **mathematical domain of dependence** of $u(x,t)$, denoted by $X(x,t)$, is the set of all points in space where the initial data at $t=0$ may have some effect on the solution $u(x,t)$.†

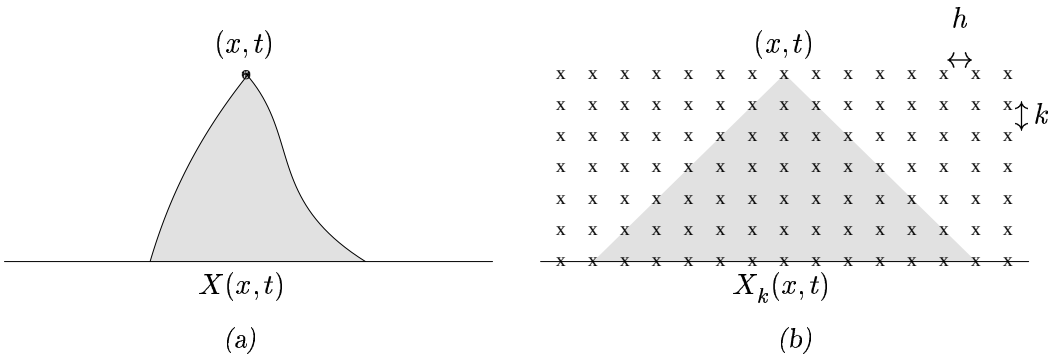


Figure 4.3.1. Mathematical and numerical domains of dependence.

For example, for $u_t = u_{xx}$ or any other parabolic partial differential equation in one space dimension, $X(x,t)$ will be the entire real axis, because under a parabolic equation, information travels infinitely fast. The magnitude of the influence of far-away data may decay exponentially with distance, but in the definition of the domain of dependence it matters only whether this influence is zero or nonzero. The same conclusion holds for the Schrödinger equation $u_t = iu_{xx}$.

On the other hand for $u_t = u_x$, $u_t = u_x + u$, $u_{tt} = u_{xx}$, or any other hyperbolic partial differential equation or system of equations, including nonlinear equations such as $u_t = (\frac{1}{2}u^2)_x$,

*In German: "Über die partiellen Differenzengleichungen der mathematischen Physik," *Math. Ann. 100* (1928), 32–74. An English translation appeared much later in *IBM Journal 11* (1967), 215–234.

†More precisely, for a problem in d space dimensions, $X(x,t)$ is the intersection of all closed sets $E \subseteq \mathbb{R}^d$ with the property that the data on $\mathbb{R}^d \backslash E$ has no effect on $u(x,t)$.

$X(x,t)$ is finite for each x and t. The reason is that in hyperbolic problems, information travels at a finite speed. Figure 4.3.1a suggests a problem of this sort, since the domain of dependence shown there is finite. For the model problem $u_t = u_x$, $X(x,t)$ is the single point $\{x+t\}$, but the more typical situation for hyperbolic problems is that the domain of dependence covers a bounded range of values of x. In one-dimensional problems the curves that bound this range, as in Figure 4.3.1a, are the **characteristic curves** for the partial differential equation, and these are straight lines in simple examples but usually more general curves in problems containing variable coefficients or nonlinearity.

A numerical approximation also has a domain of dependence, and this is suggested in Figure 4.3.1b. With an implicit finite difference formula, each value v_j^n depends on all the values at one or more earlier steps, and the domain of dependence is unbounded. On the other hand with an explicit formula, v_j^n depends on only a finite range of values at previous steps. For any fixed k and h, the domain of dependence will then fan out in a triangle that goes backwards in time. The triangle will be symmetrical for a three-point formula like Lax-Wendroff or leap frog, symmetrical and twice as wide for a five-point formula like fourth-order leap frog, asymmetrical for a one-sided formula like upwind, and so on.

The numerical domain of dependence for a fixed value k, denoted by $X_k(x,t)$, is defined to be the set of points x_j whose initial data v_j^0 enter into the computation of $v(x,t)$. For each time step k, this set is discrete, but what really matters is the limit $k \to 0$. We define the limiting **numerical domain of dependence**, $X_0(x,t)$, to be the set of all limit points of the sets $X_k(x,t)$ as $k \to 0$.* This will be a closed subset of the spatial domain—typically an interval if there is one space variable, or a parallelepiped if there are several.

From the point of view of domain of dependence, there are three general classes of discrete approximations. In the case of an implicit finite difference model, $X_k(x,t)$ is unbounded for each k, and therefore, provided only that the spatial grid becomes finer everywhere as $k \to 0$, $X_0(x,t)$ will be the entire spatial domain. (Spectral methods are also essentially of this type: although the time stepping may be explicit, their stencils cover the entire spatial domain, and so $X_k(x,t)$ is unbounded for each k.)

At the other extreme is the case of an explicit finite difference formula with a spatial grid that scales in proportion to k, so that $X_k(x,t)$ and $X_0(x,t)$ are bounded sets for each x and t. In the particular case of a regular grid in space with mesh ratio $k/h = \lambda =$ constant, their size is proportional to the width of the stencil and inversely proportional to λ. The latter statement should be obvious from Figure 4.3.1b—if λ is cut in half, it will take twice as many time steps to get to (x,t) for a fixed h, and so the width of the triangle will double.

Between these two situations lies the case of an explicit finite difference formula whose spatial grid is refined more slowly than the time step as $k \to 0$—for example, a regular grid with $k = o(h)$. Here, $X_k(x,t)$ is bounded for each k, but $X_0(x,t)$ is unbounded. This in-between situation shows that even an explicit formula can have an unbounded domain of dependence in the limiting sense.

The observation made by Courant, Friedrichs, and Lewy is beautifully simple: a numerical approximation cannot converge for arbitrary initial data unless it takes all of the necessary data into account. "Taking the data into account" means the following:

---

*More precisely, for a problem in d space dimensions, $X_0(x,t)$ is the set of all points $s \in \mathbb{R}^d$ every open neighborhood of which contains a point of $X_k(x,t)$ for all sufficiently small k.

THE CFL CONDITION. For each (x,t), the mathematical domain of dependence is contained in the numerical domain of dependence:

$$X(x,t) \subseteq X_0(x,t).$$

Here is their conclusion:

CFL THEOREM

Theorem 4.5. *The CFL condition is a necessary condition for the convergence of a numerical approximation of a partial differential equation, linear or nonlinear.*

The justification of Theorem 4.5 is so obvious that we shall not attempt to state or prove it more formally. But a few words are in order to clarify what the terms mean. First, the theorem is certainly valid for the particular case of the linear initial-value problem (4.2.1) with the definition of convergence (4.2.7) provided in the last section. In particular, unlike the von Neumann condition of the next section, it holds for any norm $\|\cdot\|$ and any partial differential equation, including problems with boundary conditions, variable coefficients, or nonlinearity. But one thing that cannot be changed is that Theorem 4.5 must always be interpreted in terms of a definition of convergence that involves arbitrary initial data. The reason is that for special initial data, a numerical method might converge even though it had a seemingly inadequate domain of dependence. The classic example of this is the situation in which the initial data are so smooth that they form an analytic function. Since an analytic function is determined globally by its behavior near any point, a finite difference model might sample "too little" initial data in an analytic case and still converge to the correct solution—at least in the absence of rounding errors.

A priori, the CFL condition is a necessary condition for *convergence.* But for linear problems, the Lax Equivalence Theorem asserts that convergence is equivalent to stability. From Theorems 4.1 and 4.5 we therefore obtain:

CFL CONDITION AND STABILITY

Theorem 4.6. *Let $\{S_k\}$ be a consistent approximation to a well-posed linear initial-value problem (4.2.1). Then the CFL condition is a necessary condition for the stability of $\{S_k\}$.*

Unlike Theorem 4.5, Theorem 4.6 is valid only if its meaning is restricted to the linear formulations of the last section. The problem of stability of finite difference models had not yet been identified when the CFL paper appeared in 1928, but Theorem 4.6 is the form in which the CFL result is now generally remembered. The reason is that the connection between convergence and stability is so universally recognized now that one habitually thinks of stability as the essential matter to be worried about.

The reader may have noticed a rather strange feature of Theorem 4.6. In the last section it was emphasized that the stability of $\{S_k\}$ has nothing to do with what initial-value problem, if any, it approximates. Yet Theorem 4.6 states a stability criterion based on

an initial-value problem. This is not a logical inconsistency, for nowhere has it been claimed that the theorem is applicable to every finite difference model $\{S_k\}$; an initial-value problem is brought into the act only when a consistency condition happens to be satisfied. For more on this point, see Exercise 4.4.4.

Before giving examples, we shall state a fundamental consequence of Theorem 4.6:

EXPLICIT MODELS OF PARABOLIC PROBLEMS

Theorem 4.7. *If an explicit finite difference approximation of a parabolic initial-value problem is convergent, then the time and space steps must satisfy $k = o(h)$ as $k \to 0$.*

Proof. This assertion follows from Theorem 4.6 and the fact that an explicit finite difference model with $k \neq o(h)$ has a bounded numerical domain of dependence $X_0(x,t)$ for each (x,t), whereas the mathematical domain of dependence $X(x,t)$ is unbounded. ∎

The impact of Theorem 4.7 is far-reaching: parabolic problems must always be solved by implicit formulas, or by explicit formulas with small step sizes. This would make them generally more difficult to treat than hyperbolic problems, were it not that hyperbolic problems tend to feature shock waves and other strongly nonlinear phenomena—a different source of difficulty that evens the score somewhat.

In computations involving more complicated equations with both convective and diffusive terms, such as the Navier-Stokes equations of fluid dynamics, the considerations of Theorem 4.7 often lead to numerical methods in which the time iteration is based on a splitting into an explicit substep for the convective terms and an implicit substep for the diffusive terms. See any book on computational fluid dynamics.

EXAMPLE 4.3.1. *Approximations of $u_t = u_x$.* For the equation $u_t = u_x$, all information propagates leftward at speed exactly 1, and so the mathematical domain of dependence for each (x,t) is the single point $X(x,t) = \{x+t\}$. Figure 4.3.2 suggests how this relates to the domain of dependence for various finite difference formulas. As always, the CFL condition is necessary but not sufficient for stability: it can prove a method unstable, but not stable. For the finite difference formulas of Table 3.2.1 with $k/h = \lambda =$ constant, we reach the following conclusions:

LF4: unstable for $\lambda > 2$;

UW, LF, LW, EU_x, LXF: unstable for $\lambda > 1$;

"Downwind" formula: unstable for all λ;

BE_x, CN_x, BOX_x: no restriction on λ.

We shall see in the next section that these conclusions are sharp except in the cases of LF4 and EU_x (and LF, marginally, whose precise condition for instability is $\lambda \geq 1$ rather than $\lambda > 1$.)

EXAMPLE 4.3.2. *Approximations of $u_t = u_{xx}$.* Since $u_t = u_{xx}$ is parabolic, Theorem 4.7 asserts that no consistent explicit finite difference approximation can be stable unless $k = o(h)$ as $k \to 0$. Thus for the finite difference formulas of Table 3.3.2, it implies

EU_{xx}, LF_{xx}: unstable unless $k = o(h)$,

BE_{xx}, CN, BOX_{xx}, CN4: no restriction on $h(k)$.

(a) LW or EU, $\lambda < 1$ *(b)* LW or EU, $\lambda > 1$ *(c)* UW, $\lambda < 1$

Figure 4.3.2. The CFL condition and $u_t = u_x$. The dashed line represents the characteristic of the PDE and the solid line represents the stencil of the finite difference formula.

The next section will show that these conclusions for LF_{xx} and EU_{xx} are not sharp. In fact, for example, EU_{xx} is stable only if $k \le \frac{1}{2}h^2$.

The virtue of the CFL condition is that it is extremely easy to apply. Its weakness is that it is necessary but not sufficient for convergence. As a practical matter, the CFL condition often suggests the correct limits on stability, but not always, and therefore it must be supplemented by more careful analysis.

EXERCISES

▷ *4.3.1. Multidimensional wave equation.* Consider the second-order wave equation in d space dimensions:

$$u_{tt} = u_{x_1 x_1} + \cdots + u_{x_d x_d}.$$

(a) Write down the d-dimensional analog of the simple second-order finite difference approximation

$$v_j^{n+1} - 2v_j^n + v_j^{n-1} = \lambda^2 (v_{j+1}^n - 2v_j^n + v_{j-1}^n)$$

for a regular grid with space step h in all directions and $\lambda = k/h =$ constant.

(b) What does the CFL condition tell you about values of λ for which this model must be unstable? (*Hint:* if you are in doubt about the speed of propagation of energy under the multidimensional wave equation, consider the fact that the equation is isotropic—i.e., energy propagates in the same manner regardless of direction.) (*Another hint:* be careful!)

4.4. The von Neumann condition for scalar one-step formulas

Von Neumann analysis is the analysis of stability by Fourier methods. In principle this restricts its applicability to a narrow range of linear, constant-coefficient finite difference formulas on regular grids, with errors measured in the ℓ_h^2 norm. In practice, the von Neumann approach has something to say about almost every finite difference model. It is the mainstay of practical stability analysis.

The essential idea is to combine the ideas of §4.2, specialized to the ℓ_h^2 norm, with those of §§3.5,3.6. For one-step scalar models the ensuing results give a complete characterization of stability in terms of the "amplification factors" introduction in §3.5, which for stability must lie within a distance $O(k)$ of the unit disk as $k \to 0$. For multistep or vector problems one works with the associated "amplification matrices" introduced in §3.6, and a complete analysis of stability requires looking not only at the spectra of these matrices, which amount to amplification factors, but also at their resolvents or pseudospectra. We shall treat the one-step scalar case in this section and the general case in the next two sections.

We begin with two easy lemmas. The first is a generalization of the well-known inequality $(1+\epsilon)^{1/\epsilon} < e$:

Lemma 4.4.1. *For any real numbers $a \ge -1$ and $b \ge 0$,*

$$(1+a)^b \le e^{ab}. \tag{4.4.1}$$

Proof. Both sides are nonnegative, so taking the bth root shows that it is enough to prove $1+a \le e^a$. This inequality is trivial (just draw a graph!). ∎

The second lemma deals with arbitrary sets of numbers $s_k \ge 0$, which in our applications will be norms such as $\|S_k\|$:

Lemma 4.4.2. *Let $\{s_k\}$ be a set of nonnegative numbers indexed by $k > 0$, and let $T > 0$ be fixed. Then $(s_k)^n \le C_1$ for some $C_1 \ge 0$, uniformly for all k and n with $kn \le T$, if and only if $s_k \le 1 + O(k)$ as $k \to 0$.*

Proof. If $s_k \le 1 + C_2 k$ for each k, then by Lemma 4.4.1, $(s_k)^n \le (1+C_2 k)^n \le e^{C_2 kn} \le e^{C_2 T}$. Conversely, if $(s_k)^n \le C_1$ for all $kn \le T$, then in particular, for each k, $(s_k)^n \le C_1$ for some value of n with $nk > T/2$. This implies $s_k \le C_1^{1/n} = (C_1^{2/T})^{T/2n} \le (C_1^{2/T})^k \le 1 + O(k)$. ∎

Now we are ready for von Neumann analysis. Suppose that (4.2.1) is a well-posed linear initial-value problem in which $u(t)$ is a scalar function of x and $\mathcal{B}$ is the space ℓ_h^2, with $\|\cdot\|$ denoting the ℓ_h^2-norm. Suppose also that for each $k > 0$, the operator S_k of (4.2.2) denotes an explicit or implicit one-step finite difference formula (3.5.20) with coefficients $\{\alpha_\mu\}$ and $\{\beta_\mu\}$ that are constant except for the possible dependence on k. If S_k is implicit, it is assumed to satisfy the solvability condition (3.5.26), which ensures that it has a bounded amplification factor function $g_k(\xi) = \hat{a}_k(\xi)/\hat{b}_k(\xi)$.

By (4.2.8), the formula is stable if and only if

$$\|S_k^n\| \le C \qquad \text{for } 0 \le nk \le T$$

for some constant C. By Theorem 3.4, this is equivalent to the condition

$$(\|g_k\|_\infty)^n \le C \qquad \text{for } 0 \le nk \le T,$$

or equivalently,

$$|g_k(\xi)|^n \le C \qquad \text{for } 0 \le nk \le T, \tag{4.4.2}$$

where C is a constant independent of ξ. By Lemma 4.4.2, this is equivalent to

$$|g_k(\xi)| \le 1 + O(k) \tag{4.4.3}$$

as $k \to 0$, uniformly in ξ. What this means is that there exists a constant C' such that for all n and k with $0 \le nk \le T$, and all $\xi \in [-\pi/h, \pi/h]$,

$$|g_k(\xi)| \le 1 + C'k. \tag{4.4.4}$$

We have obtained the following complete characterization of stable finite difference formulas in our scalar ℓ_h^2 problem.

VON NEUMANN CONDITION
FOR SCALAR ONE-STEP FINITE DIFFERENCE FORMULAS

Theorem 4.8. *A linear, scalar, constant-coefficient one-step finite difference formula as described above is stable in ℓ_h^2 if and only if the amplification factors $g_k(\xi)$ satisfy*

$$|g_k(\xi)| \le 1 + O(k) \tag{4.4.5}$$

as $k \to 0$, uniformly for all $\xi \in [-\pi/h, \pi/h]$.

With Theorem 4.8 in hand, we are equipped to analyze the stability of many of the finite difference formulas of §3.2.

EXAMPLE 4.4.1. *Upwind formula for $u_t = u_x$.* In (3.5.5) we computed the amplification factor for the upwind formula as

$$g(\xi) = (1-\lambda) + \lambda e^{i\xi h}, \tag{4.4.6}$$

assuming that $\lambda = k/h$ is a constant. For any λ, this formula describes a circle in the complex plane, shown in Figure 4.4.1, as ξ ranges over $[-\pi/h, \pi/h]$. The circle will lie in the closed unit disk, as required by (4.4.5), if and only if $\lambda \le 1$, which is accordingly the stability condition for the upwind formula. This matches the restriction suggested by the CFL condition (Example 4.3.1).

(a) $\lambda < 1$ (stable) (b) $\lambda > 1$ (unstable)

Figure 4.4.1. Amplification factors for the upwind model of $u_t = u_x$ (solid curve). The shaded region is the unit disk.

EXAMPLE 4.4.2. *Crank-Nicolson formulas for $u_t = u_x$, $u_t = u_{xx}$, $u_t = iu_{xx}$.* In (3.5.35) we found the amplification factor for the Crank-Nicolson formula to be

$$g(\xi) = \frac{1 - 2\frac{k}{h^2}\sin^2\frac{\xi h}{2}}{1 + 2\frac{k}{h^2}\sin^2\frac{\xi h}{2}}. \tag{4.4.7}$$

Here $|g(\xi)| \le 1$ for all ξ, regardless of k and h. Therefore the Crank-Nicolson formula is stable as $k \to 0$, no matter how k and h are related. (It will be consistent, hence convergent, so long as $h(k) = o(1)$ as $k \to 0$.)

For the Crank-Nicolson model of $u_t = u_x$, the corresponding formula is

$$g(\xi) = \frac{1 + \frac{ik}{2h}\sin\xi h}{1 - \frac{ik}{2h}\sin\xi h}. \tag{4.4.8}$$

Now $|g(\xi)| = 1$ for all ξ, so the formula is again unconditionally stable. The same is true of the Crank-Nicolson model of $u_t = iu_{xx}$, whose amplification factor function is

$$g(\xi) = \frac{1 - 2i\frac{k}{h^2}\sin^2\frac{\xi h}{2}}{1 + 2i\frac{k}{h^2}\sin^2\frac{\xi h}{2}}. \tag{4.4.9}$$

EXAMPLE 4.4.3. *Euler formulas for $u_t = u_x$ and $u_t = u_{xx}$.* The amplification factor for the Euler model of $u_t = u_{xx}$ was given in (3.5.19) as

$$g(\xi) = 1 - 4\frac{k}{h^2}\sin^2\frac{\xi h}{2}. \tag{4.4.10}$$

As illustrated in Figure 4.4.2a, this expression describes the interval $[1 - 4k/h^2, 1]$ in the complex plane as ξ ranges over $[-\pi/h, \pi/h]$. If $\sigma = k/h^2$ is held constant as $k \to 0$, we conclude that the Euler formula is stable if and only if $\sigma \le \frac{1}{2}$. This is a tighter restriction than the one provided by the CFL condition, which requires only $k = o(h)$.

(a) $u_t = u_{xx}$ (b) $u_t = u_x$

Figure 4.4.2. Amplification factors for the Euler models of $u_t = u_{xx}$ and $u_t = u_x$.

The amplification factor for the Euler model of $u_t = u_x$ is

$$g(\xi) = 1 + \frac{ik}{h}\sin\xi h, \tag{4.4.11}$$

which describes a line segment tangent to the unit circle in the complex plane (Figure 4.4.2b). Therefore the largest amplification factor is

$$\sqrt{1+\frac{k^2}{h^2}}.$$

If $\lambda = k/h$ is held fixed as $k \to 0$, then this finite difference formula is therefore unstable regardless of λ. On the other hand the square root will have the desired magnitude $1+O(k)$ if $k^2/h^2 = O(k)$, i.e. $k = O(h^2)$, and this is accordingly the stability condition for arbitrary mesh relationships $h = h(k)$. Thus in principle the Euler formula is usable for hyperbolic equations, but it is not used in practice since there are alternatives that permit larger time steps $k = O(h)$, as well as having higher accuracy.

EXAMPLE 4.4.4. *Lax-Wendroff formula for* $u_t = u_x$. The amplification factor for the Lax-Wendroff formula was given in (3.5.17) as

$$g(\xi) = 1 + i\lambda\sin\xi h - 2\lambda^2\sin^2\frac{\xi h}{2}, \tag{4.4.12}$$

if $\lambda = k/h$ is a constant. Therefore $|g(\xi)|^2$ is

$$|g(\xi)|^2 = (1-4\lambda^2\sin^2\frac{\xi h}{2} + 4\lambda^4\sin^4\frac{\xi h}{2}) + \lambda^2\sin^2\xi h.$$

Applying the identity $\sin^2\theta = 4\sin^2\frac{\theta}{2}\cos^2\frac{\theta}{2} = 4(\sin^2\frac{\theta}{2} - \sin^4\frac{\theta}{2})$ to the last term converts this expression to

$$\begin{aligned} |g(\xi)|^2 &= 1 - 4\lambda^2\sin^2\frac{\xi h}{2} + 4\lambda^4\sin^4\frac{\xi h}{2} + 4\lambda^2\sin^2\frac{\xi h}{2} - 4\lambda^2\sin^4\frac{\xi h}{2} \\ &= 1 + 4(\lambda^4 - \lambda^2)\sin^4\frac{\xi h}{2}. \end{aligned} \tag{4.4.13}$$

If λ is fixed, it follows that the Lax-Wendroff formula is stable provided that $\lambda^4 - \lambda^2 \in [-\frac{1}{2}, 0]$. This is true if and only if $\lambda \le 1$, which is accordingly the stability condition.

Tables 4.4.1 and 4.4.2 summarize the orders of accuracy and the stability limits for the finite difference formulas of Tables 3.2.1 and 3.2.2. The results listed for multistep formulas will be justified in §4.6.

Formula	order of accuracy	CFL stability restriction	Exact stability restriction
(EU_x = Euler)	1	$\lambda \le 1$	unstable
(BE_x = Backward Euler)	1	none	none
(CN_x = Crank-Nicolson)	2	none	none
LF = Leap frog	2	$\lambda \le 1$	$\lambda < 1$
BOX_x = Box	2	none	none
LF4 = Fourth-order Leap frog	2	$\lambda \le 2$	$\lambda < 0.728\ldots$*
LXF = Lax-Friedrichs	1	$\lambda \le 1$	$\lambda \le 1$
UW = Upwind	1	$\lambda \le 1$	$\lambda \le 1$
LW = Lax-Wendroff	2	$\lambda \le 1$	$\lambda \le 1$

Table 4.4.1. Orders of accuracy and stability limits for various finite difference approximations to the wave equation $u_t = u_x$, with $\lambda = k/h =$ constant (see Table 3.2.1).

Formula	order of accuracy	CFL stability restriction	Exact stability restriction
EU_{xx} = Euler	1†	none	$\sigma \le \frac{1}{2}$
BE_{xx} = Backward Euler	1	none	none
CN = Crank-Nicolson	2	none	none
(LF_{xx} = Leap frog)	2	none	unstable
BOX_{xx} = Box	2	none	none
CN4 = Fourth-order CN	2	none	none
DF = DuFort-Frankel	2**	none	none**

Table 4.4.2. Orders of accuracy and stability limits for various finite difference approximations to the heat equation $u_t = u_{xx}$, with $\sigma = k/h^2 =$ constant (see Table 3.2.2). (The orders of accuracy are with respect to h, not k as in (4.2.4).)

*See Exercise 4.5.3.
†See Exercise 4.4.5.
**See Exercise 4.5.4.

EXERCISES

▷ *4.4.1. Generalized Crank-Nicolson or "theta method."* Let the heat equation $u_t = u_{xx}$ be modeled by the formula

$$v_j^{n+1} = v_j^n + \frac{k(1-\theta)}{h^2}(v_{j+1}^n - 2v_j^n + v_{j-1}^n) + \frac{k\theta}{h^2}(v_{j+1}^{n+1} - 2v_j^{n+1} + v_{j-1}^{n+1}) \tag{4.4.19}$$

with $0 \le \theta \le 1$. For $\theta = 0$, $\frac{1}{2}$, 1 this is Euler, Crank-Nicolson, or backward Euler formula, respectively.

(a) Determine the amplification factor function $g(\xi)$.

(b) Suppose $\sigma = k/h^2$ is held constant as $k \to 0$. For which σ and θ is (4.4.19) stable?

(c) Suppose $\lambda = k/h$ is held constant as $k \to 0$. For which λ and θ is (4.4.19) stable?

(d) Your boss asks you to solve a heat conduction problem involving a space interval of length 1 and a time interval of length 1. She wants an answer with errors on the order of some number $\delta \ll 1$. Roughly speaking (order of magnitude in δ), how many floating-point operations will you have to perform if you use $\theta = 0$, $\theta = \frac{1}{2}$, and $\theta = 1$?

▷ *4.4.2. The downwind formula.* The downwind approximation to $u_t = u_x$ is

$$v_j^{n+1} = S_k v_j^n = v_j^n + \frac{k}{h}(v_j^n - v_{j-1}^n).$$

In this problem, do not assume that k/h is held constant as $k \to 0$; let $h(k)$ be a completely arbitrary function of k.

(a) For what functions $h(k)$, if any, is this formula stable? (Use Theorem 4.8 and be careful!)

(b) For what functions $h(k)$, if any, is it consistent?

(c) For what functions $h(k)$, if any, is it convergent? (Use the Lax Equivalence Theorem.)

(d) How does the result of *(c)* match the prediction by the CFL condition?

▷ *4.4.3. The CFL-stability link.* (This problem was originally worked out by G. Strang in the 1960's.) **Bernstein's inequality** asserts that if

$$g(\xi) = \sum_{\nu=-m}^{m} \alpha_\nu e^{i\nu\xi}$$

for some constants $\{\alpha_\nu\}$, then

$$\|g'\|_\infty \le m\|g\|_\infty,$$

where $\|\cdot\|_\infty$ denotes the maximum over $[-\pi,\pi]$, and g' is the derivative $dg/d\xi$.

Derive from this the CFL condition: if an explicit finite difference formula

$$v_j^{n+1} = \sum_{\nu=-m}^{m} \alpha_\nu v_{j+\nu}^n$$

is stable as $h \to 0$ with $k/h = \lambda =$ constant, and consistent with $u_t = u_x$, then $\lambda \le m$. (*Hint:* what does consistency imply about the behavior of $g(\xi)$ near $\xi = 0$?)

▷ *4.4.4. A thought experiment.* Suppose the Lax-Wendroff model of $u_t = u_x$ is applied on an infinite grid with $h = 0.01$ to compute an approximate solution at $t = 1$. The initial data are

$$u_1(x) = \cos^2 \pi x \qquad \text{or} \qquad u_2(x) = \max\{0, 1 - |x|\},$$

and the time step is $k = \lambda h$ with

$$\lambda_a = \frac{4}{5} \qquad \text{or} \qquad \lambda_b = \frac{4}{3}.$$

Thus we have four problems: 1a, 1b, 2a, 2b. Let E_{1a}, E_{1b}, E_{2a}, E_{2b} denote the corresponding maximum (ℓ_h^∞) errors over the grid at $t = 1$.

One of the four numbers E_{1a}, E_{1b}, E_{2a}, E_{2b} depends significantly on the machine precision, ϵ, and thus deserves a star * in front of it (see §β); the other three do not. Which one? Explain carefully why each of them does or does not depend on ϵ. In the process, give order-of-magnitude predictions for the four numbers, such as $E \approx 10^{-2}$, $E \approx 10^{10}$, $E \approx {}^\star\epsilon 10^{10}$. Explain your reasoning!

If you wish, you are welcome to turn this thought experiment into a computer experiment.

▷ *4.4.5. The Euler formula for the heat equation.* Show that if $\sigma = 1/6$, the order of accuracy of the Euler formula for $u_t = u_{xx}$ increases from 1 to 2. (*Note:* Such bits of good fortune are not always easy to take advantage of in practice, since coefficients and hence effective mesh ratios may vary from point to point.)

▷ *4.4.6. Weak inequalities.* In Tables 4.4.1 and 4.4.2, the stability restrictions for one-step formulas all involve weak ("$\le$") rather than strong ("$<$") inequalities. Prove that this is a general phenomenon: the stability condition for a formula of the type considered in Theorem 4.8 may be of the form $k \le f(h)$ but never $k < f(h)$.

4.5. Resolvents, pseudospectra, and the Kreiss matrix theorem

Powers of matrices appear throughout numerical analysis. In this book they arise mainly from equation (4.2.3), which expresses the computed solution at step n for a numerical model of a linear problem in terms of the solution at step 0:

$$v^n = S_k^n v^0. \tag{4.5.1}$$

In this formula S_k represents a bounded operator on a Banach space $\mathcal{B}$, possibly different for each time step $k > 0$. Thus we are dealing with a family of operators $\{S_k\}$ indexed by k. According to the theory presented in §4.2, a key question to ask about such a family of operators is whether it is stable, i.e., whether a bound

$$\|S_k^n\| \le C \tag{4.5.2}$$

holds for all n and k with $0 \le nk \le T$.

This question about operators may be reduced to a question about matrices in two distinct ways. The first is by Fourier or von Neumann analysis, applicable in cases where one is dealing with a regular grid, constant coefficients, and the 2-norm. For one-step scalar formulas Fourier analysis reduces S_k to a scalar, the amplification factor $g(\xi)$ treated in the last section. For vector formulas, the subject of this and the next section, we get the more interesting Fourier analogue

$$\widehat{v^n}(\xi) = G_k(\xi)^n \, \widehat{v^0}(\xi). \tag{4.5.3}$$

Here $G_k(\xi)^n$ is the nth power of the amplification matrix, defined in §3.6, and the equivalence of (4.5.1) and (4.5.3) implies that the norm of $G_k(\xi)$ determines the norm of S_k^n:

$$\|S_k^n\|_2 = \sup_{\xi \in [-\pi/h, \pi/h]} \|G_k(\xi)^n\|.$$

The question of stability becomes, are the norms $\|G_k(\xi)^n\|$ bounded by a constant for all $nk \le T$, uniformly with respect to ξ? Note that here the dimension N of $G_k(\xi)$ is fixed, independent of k and h.

For problems involving variable coefficients, variable grids, or translation-dependent discretizations such as Chebyshev spectral methods (Chapter 8), Fourier analysis cannot be applied. Here a second and more straightforward reduction to matrices becomes relevant. One can simply work with S_k itself as a matrix—that is, work in space itself, not Fourier space. If the grid is unbounded, then the dimension of S_k is infinite, but since the grids one computes with are usually bounded, S_k is usually finite in practice. The dimension increases to ∞, however, as $k \to 0$.

In summary, the stability analysis of numerical methods for partial differential equations leads naturally to questions of whether the powers of a family of matrices have uniformly bounded norms, or, as the problem is often put, whether a family of matrices is **power-bounded**. Depending on the circumstances, the matrices in the family may be of fixed dimension or varying dimensions.

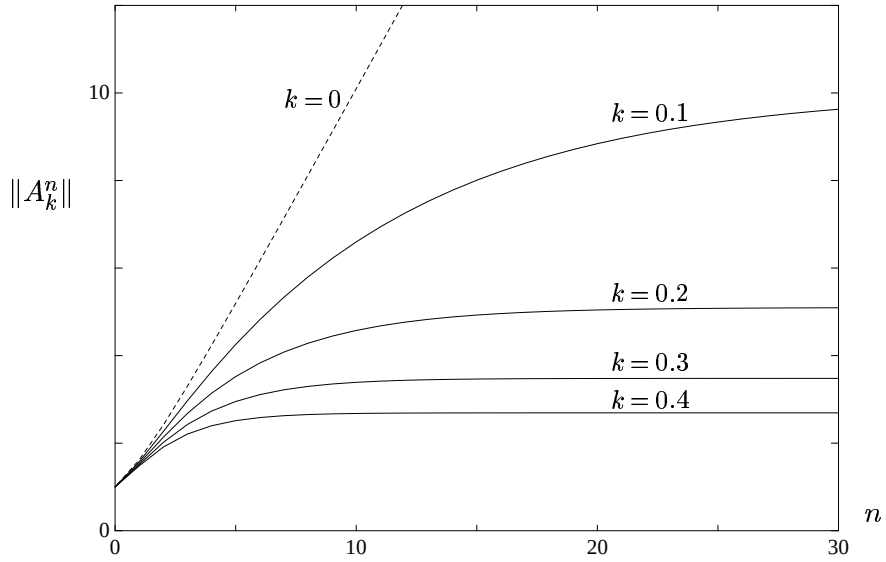


Figure 4.5.1. Norms of powers of the matrix A_k of (4.5.4) for various k. Each A_k for $k>0$ is individually power-bounded, but the family $\{A_k\}_{(k>0)}$ is not power-bounded, since the powers come arbitrarily close to those of the limiting defective matrix A_0 (dashed).

In §1.5 we have already presented an algebraic criterion for power-boundedness of a matrix A. According to the "alternative proof" of Theorem 1.6, the powers $\|A^n\|$ are bounded if and only if the eigenvalues of A lie in the closed unit disk and any eigenvalues on the unit circle are nondefective. For families of matrices, however, matters are not so simple. The eigenvalue condition is still necessary for power-boundedness, but it is not sufficient. To illustrate this, consider the two families of matrices

$$A_k = \begin{pmatrix} 1 & 1 \\ 0 & 1-k \end{pmatrix}, \qquad B_k = \begin{pmatrix} 0 & k^{-1} \\ 0 & 0 \end{pmatrix} \tag{4.5.4}$$

for $k>0$. For each $k>0$, A_k and B_k are individually power-bounded, but neither $\{A_k\}$ nor $\{B_k\}$ is power-bounded as a family. For $\{B_k\}$ the explanation is quite obvious: the upper-right entry diverges to ∞ as $k\to 0$, and thus even the first power $\|B_k\|$ is unbounded as $k\to 0$. For $\{A_k\}$ the explanation is more interesting. As $k\to 0$, these matrices come closer and closer to a defective matrix whose powers increase in norm without bound. Figure 4.5.1 illustrates this effect.

In the next section we shall see that an unstable family much like $\{A_k\}$ arises in the von Neumann analysis of the leap frog discretization of $u_t = u_x$ with mesh ratio $\lambda = 1$. In most applications, however, the structure of the matrices that arise is far from obvious by inspection and one needs a general tool for determining power-boundedness.

What is needed is to consider not just the spectra of the matrices but also their *pseudospectra.* The idea of pseudospectra is as follows. An eigenvalue of a matrix A is a number $z\in\mathbb{C}$ with the property that the **resolvent** matrix $(zI-A)^{-1}$ is undefined. By convention we may write $\|(zI-A)^{-1}\|=\infty$ in this case. A pseudo-eigenvalue of A is a

number z where $\|(zI-A)^{-1}\|$, while not necessarily infinite, is very large. Here is the definition:

Given $\epsilon > 0$, the number $\lambda \in \mathbb{C}$ is an **ϵ-pseudo-eigenvalue** *of A if either of the following equivalent conditions is satisfied:*

(i) $\|(\lambda I - A)^{-1}\| \geq \epsilon^{-1}$;

(ii) λ *is an eigenvalue of $A+E$ for some $E \in \mathbb{C}^{N\times N}$ with $\|E\| \leq \epsilon$.*

The **ϵ-pseudospectrum** *of A, denoted by $\Lambda_\epsilon(A)$, is the set of all of its ϵ-pseudo-eigenvalues.*

Condition *(i)* expresses the idea of an ϵ-pseudo-eigenvalue just described. Condition *(ii)* is a mathematical equivalent with quite a different flavor: though an ϵ-pseudo-eigenvalue of A need not be near to any exact eigenvalue of A, it is an exact eigenvalue of some nearby matrix. The equivalence of conditions *(i)* and *(ii)* is easy to prove (Exercise 4.5.1).

If a matrix is **normal**, which means that its eigenvectors are orthogonal,* then the 2-norm of the resolvent is just $\|(zI-A)^{-1}\|_2 = 1/\mathrm{dist}\,(z,\Lambda(A))$, where $\mathrm{dist}\,(z,\Lambda(A))$ denotes the distance between the point z and the set $\Lambda(A)$ (Exercise 4.5.2). Therefore for each $\epsilon \geq 0$, $\Lambda_\epsilon(A)$ is equal to the union of the closed balls of radius ϵ about the eigenvalues of A; by condition *(ii)*, the eigenvalues of A are insensitive to perturbations. The interesting cases arise when A is non-normal, where the ϵ-pseudospectra may be much larger and the eigenvalues may be highly sensitive to perturbations. Figure 4.5.2 illustrates this comparison rather mildly by comparing the ϵ-pseudospectra of the two matrices

$$A = \begin{pmatrix} 0.9 & 0 \\ 0 & 0.8 \end{pmatrix}, \qquad \tilde{A} = \begin{pmatrix} 0.9 & 1 \\ 0 & 0.8 \end{pmatrix}, \tag{4.5.5}$$

motivated by (4.5.4), for $\epsilon = 0.05, 0.10, 0.15, \ldots, 0.50$. These two matrices both have the spectrum $\{0.8, 0.9\}$, but the pseudospectra of $\tilde{A}$ are considerably larger than those of A.

Roughly speaking, matrices with larger pseudospectra tend to have larger powers, even if the eigenvalues are the same. Figure 4.5.3 illustrates this phenomenon for the case of the two matrices A and $\tilde{A}$ of (4.5.5). Asymptotically as $n \to \infty$, the powers decrease in both cases at the rate $(0.9)^n$ determined by the spectral radius, $\rho(A) = \rho(\tilde{A}) = 0.9$. The transient behavior is noticeably different, however, with the curve for $\|\tilde{A}^n\|$ showing a hump by a factor of about 2.8 centered around $n = 6$ before the eventual decay.

Factors of 2.8 are of little consequence for applications, but then, $\tilde{A}$ is not a very highly non-normal matrix. In more extreme examples the hump in a figure like Figure 4.5.3 may be arbitrarily high. To control it we must have a bound on the pseudospectra, and this is the subject of the Kreiss matrix theorem, first proved in 1962.

*More precisely, a matrix is normal if *there exists* a complete set of orthogonal eigenvectors.

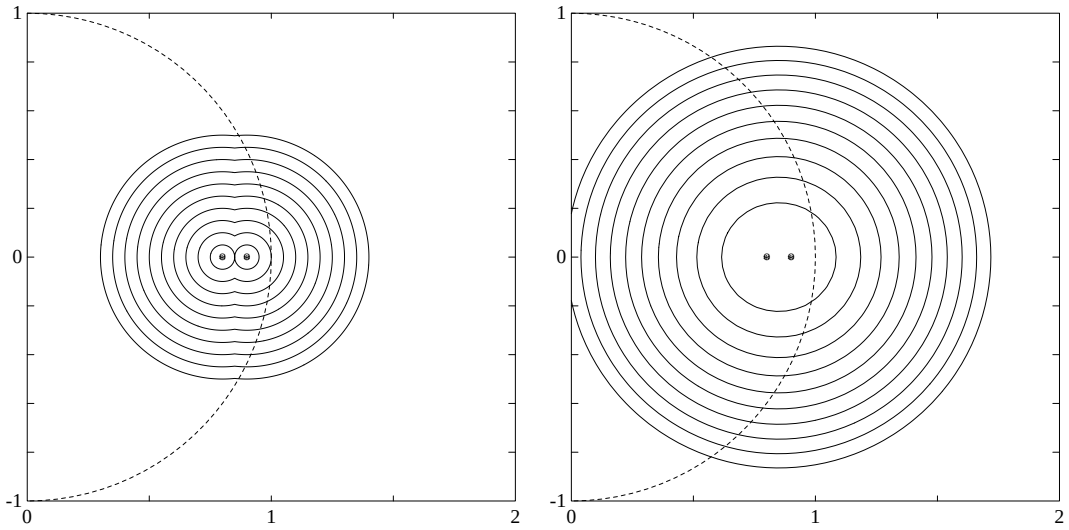


Figure 4.5.2. Boundaries of ϵ-pseudospectra of the matrices A and $\tilde{A}$ of (4.5.6) for $\epsilon = 0.05, 0.10, \ldots, 0.50$. The dashed curve is the right half of the unit circle, whose location is important for the Kreiss matrix theorem. The solid dots are the eigenvalues.

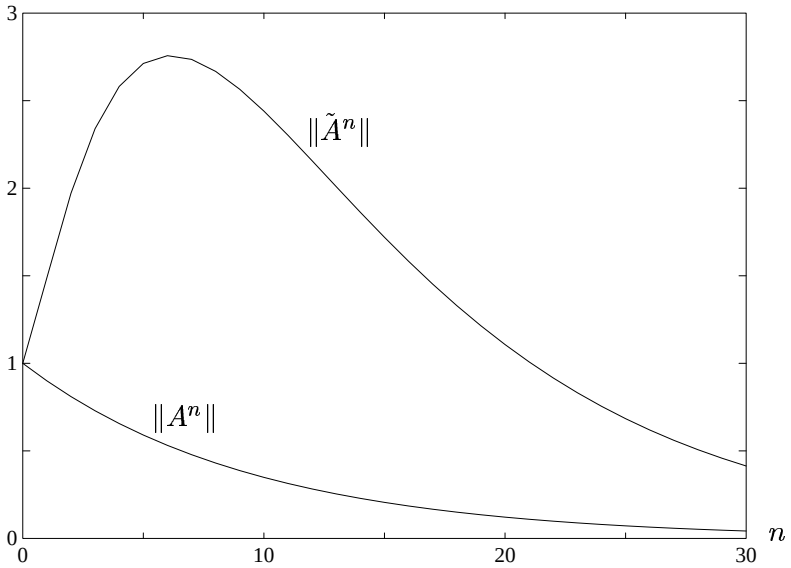


Figure 4.5.3. $\|A^n\|$ and $\|(\tilde{A})^n\|$ for the matrices A and $\tilde{A}$ of (4.5.5).

KREISS MATRIX THEOREM

Theorem 4.10. *Let A be a matrix of dimension N. If*

$$\|A^n\| \le C \qquad \forall n \ge 0 \tag{4.5.6}$$

for some constant C, then

$$|\lambda_\epsilon| \le 1 + C\epsilon \qquad \forall \lambda_\epsilon \in \Lambda_\epsilon(A), \quad \forall \epsilon \ge 0, \tag{4.5.7}$$

or equivalently,

$$\|(zI - A)^{-1}\| \le \frac{C}{|z| - 1} \qquad \forall z \in \mathbb{C}, \quad |z| > 1. \tag{4.5.8}$$

Conversely, (4.5.7)-(4.5.8) imply

$$\|A^n\| \le eC \min\{N, n+1\} \qquad \forall n \ge 0. \tag{4.5.9}$$

Theorem 4.10 is stated for an individual matrix, but since the bounds asserted are explicit and quantitative, the same result applies to families of matrices A_ν satisfying a uniform bound $\|A_\nu^n\| \le C$, provided the dimensions N are all the same. If the dimensions N vary unboundedly, then in (4.5.9), only the factor $n+1$ remains meaningful. The inequality (4.5.8) is sometimes called the **Kreiss condition**, and the constant C there is the **Kreiss constant**.

Proof. The equivalence of (4.5.7) and (4.5.8) follows trivially from definition *(i)* of $\Lambda_\epsilon(A)$. The proof that (4.5.6) implies (4.5.8) is also straightforward if one considers the power series expansion

$$(zI - A)^{-1} = z^{-1}I + z^{-2}A + z^{-3}A^2 + \cdots.$$

Thus the real substance of the Kreiss Matrix theorem lies in the assertion that (4.5.8) implies (4.5.9). This is really two assertions: one involving a factor N, the other involving a factor $n+1$.

The starting point for both proofs is to write the matrix A^n as the Cauchy integral

$$A^n = \frac{1}{2\pi i} \int_\Gamma z^n (zI - A)^{-1}\, dz, \tag{4.5.10}$$

where Γ is any curve in the complex plane that encloses the eigenvalues of A. To prove that $\|A^n\|$ satisfies the bound (4.5.9), it is enough to show that $|v^* A^n u|$ satisfies the same bound for any vectors u and v with $\|u\| = \|v\| = 1$. Thus let u and v be arbitrary N-vectors of this kind. Then (4.5.10) implies

$$v^* A^n u = \frac{1}{2\pi i} \int_\Gamma z^n q(z)\, dz, \tag{4.5.11}$$

where $q(z) = v^*(zI - A)^{-1}u$. It can be shown that $q(z)$ is a rational function of order N, i.e., a quotient of two polynomials of degrees at most N, and by (4.5.8), it satisfies

$$|q(z)| \le \frac{C}{|z| - 1}. \tag{4.5.12}$$

Let us take Γ to be the circle $\Gamma = \{z \in \mathbb{C} : \ |z| = 1 + (n+1)^{-1}\}$, which certainly encloses the eigenvalues of A if (4.5.8) holds. On this circle (4.5.12) implies $|q(z)| \le C(n+1)$. Therefore (4.5.11) yields the bound

$$\begin{aligned} |v^* A^n u| &\le \frac{1}{2\pi} \int_\Gamma |z|^n C(n+1)\, |dz| \\ &\le \frac{1}{2\pi} (1+(n+1)^{-1})^n C(n+1) 2\pi (1+(n+1)^{-1}) \\ &\le (1+(n+1)^{-1})^{n+1} C(n+1) \le e\, C(n+1). \end{aligned}$$

In the last of these inequalities we have used Lemma 4.4.1. This proves the part of (4.5.9) involving the factor $n+1$.

The other part of (4.5.9), involving the factor N, is more subtle. Integration by parts of (4.5.11) gives

$$v^* A^n u = \frac{-1}{2\pi i (n+1)} \int_\Gamma z^{n+1} q'(z)\, dz.$$

Using the same contour of integration Γ as before and again the estimate $|z^{n+1}| \le e$, we obtain

$$|v^* A^n u| \le \frac{e}{2\pi (n+1)} \int_\Gamma |q'(z)|\, |dz|.$$

The integral in this formula can be interpreted as the arc length of the image of the circle Γ under the rational function $q(z)$. Now according to a result known as Spijker's Lemma,* if $q(z)$ is a rational function of order N, the arc length of the image of a circle under $q(z)$ can be at most $2\pi N$ times the maximum modulus that $q(z)$ attains on that circle, which in this case is at most $C(n+1)$. Thus we get

$$|v^* A^n u| \le \frac{e}{2\pi (n+1)} (2\pi N) C(n+1) = e\, C N.$$

This completes the proof of the Kreiss matrix theorem. ∎

We close this section with a figure to further illustrate the idea of pseudospectra. Consider the 32×32 matrix of the form

$$A = \begin{pmatrix} 0 & 1 & 1 & & & \\ & 0 & 1 & 1 & & \\ & & 0 & 1 & 1 & \\ & & & 0 & 1 & 1 \\ & & & & 0 & 1 \\ & & & & & 0 \end{pmatrix}, \tag{4.5.13}$$

*Spijker's Lemma was conjectured in 1983 by LeVeque and Trefethen (*BIT* '84) and proved up to a factor of 2. The sharp result was proved by Spijker in 1990 (*BIT* '92). Shortly thereafter it was pointed out by E. Wegert that the heart of the proof is a simple estimate in integral geometry that generalizes the famous "Buffon needle problem" of 1777. See Wegert and Trefethen, "From the Buffon needle problem to the Kreiss matrix theorem," *Amer. Math. Monthly 101* (1994), pp. 132–139.

whose only eigenvalue is $\lambda = 0$. The upper plot of Figure 4.5.4 depicts the boundaries of the ϵ-pseudospectra of this matrix for $\epsilon = 10^{-1}, 10^{-2}, \ldots, 10^{-8}$. Even for $\epsilon = 10^{-8}$, the ϵ-pseudospectrum of this matrix is evidently quite a large set, covering a heart-shaped region of the complex plane that extends far from the spectrum $\{0\}$. Thus by condition *(ii)* of the definition of the ϵ-pseudospectrum, it is evident that the eigenvalues of this matrix are exceedingly sensitive to perturbations. The lower plot of Figure 4.5.4 illustrates this fact more directly. It shows a superposition of the eigenvalues of 100 matrices $A+E$, where each E is a random matrix of norm $\|E\|_2 = 10^{-3}$. (The elements of each E are taken as independent, normally distributed complex numbers of mean 0, and then the whole matrix is scaled to achieve this norm.) Thus there are 3200 dots in Figure 4.5.4b, which by definition must lie within in second-largest of the curves in Figure 4.5.4a.

For further illustrations of matrices with interesting pseudospectra, see L. N. Trefethen, "Pseudospectra of matrices," in D. F. Griffiths and G. A. Watson, eds., *Numerical Analysis 1991*, Longman, 1992, pp. 234–266.

EXERCISES

▷ *4.5.1. Equivalent definitions of the pseudospectrum.*

(a) Prove that conditions *(i)* and *(ii)* on p. 175 are equivalent.

(b) Prove that another equivalent condition is
(iii) $\exists u \in \mathbb{C}^n$, $\|u\| = 1$, such that $\|(A-\lambda)u\| \le \epsilon$.
Such a vector u is called an *ϵ-pseudo-eigenvector* of A.

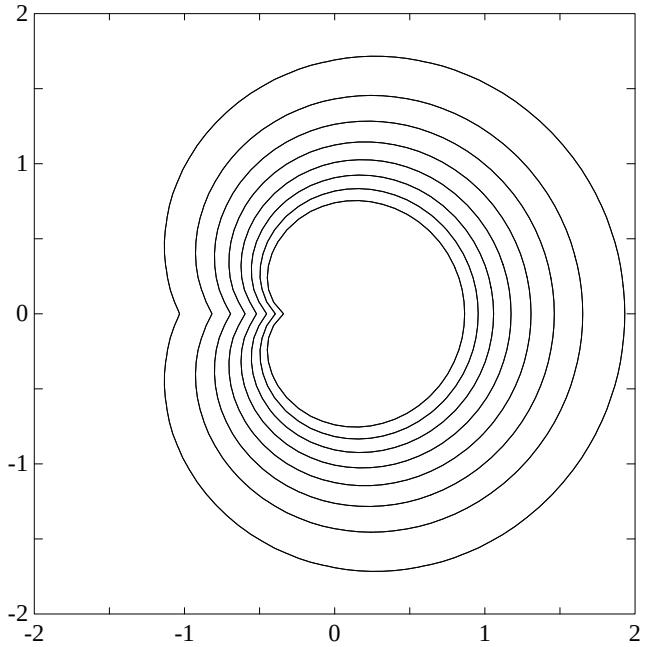
(c) Prove that if $\|\cdot\| = \|\cdot\|_2$, then a further equivalent condition is
(iv) $\sigma_N(\lambda I - A) \le \epsilon$,
where $\sigma_N(\lambda I - A)$ denotes the smallest singular value of $\lambda I - A$.

▷ *4.5.2.* Prove that if A is a normal matrix, then $\|(zI-A)^{-1}\|_2 = 1/\mathrm{dist}\,(z, \Lambda(A))$ for all $z \in \mathbb{C}$. (*Hint:* if A is normal, then it can be unitarily diagonalized.)

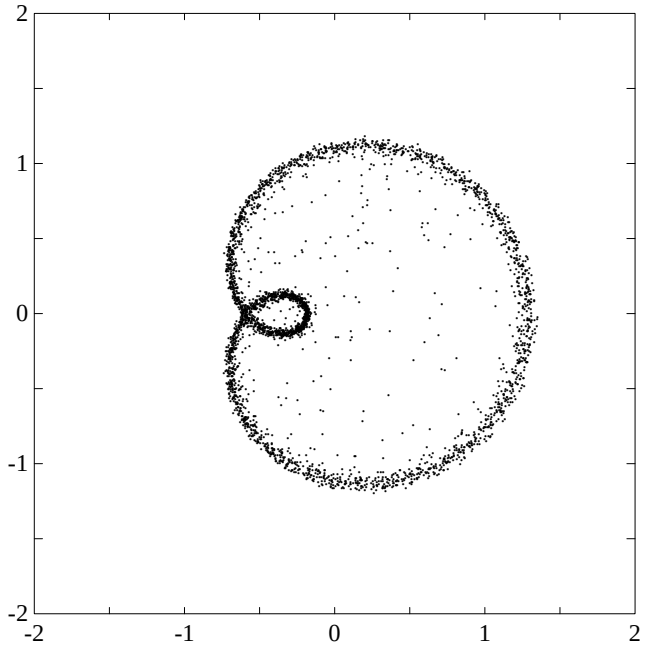
▶ *4.5.3.*

(a) Making use of Figure 4.5.4a, a ruler, and the Kreiss matrix theorem, derive lower and upper bounds as sharp as you can manage for the quantity $\sup_{n\ge 0} \|A^n\|_2$ for the matrix A of (4.5.13).

(b) Find the actual actual number with Matlab. How does it compare with your bounds?



(a) Boundaries of $\Lambda_\epsilon(A)$ for $\epsilon = 10^{-1}, 10^{-2}, \ldots, 10^{-8}$.



(b) Eigenvalues of 100 randomly perturbed matrices $A+E$, $\|E\|_2 = 10^{-3}$.

Figure 4.5.4. Pseudospectra of the 32×32 matrix A of (4.5.13).

4.6. The von Neumann condition for vector or multistep formulas

Just as §3.6 followed §3.5, Fourier analysis applies to vector or multistep finite difference formulas as well as to the scalar one-step case. The determination of stability becomes more complicated, however, because one must estimate norms of powers of matrices.

Consider a linear, constant-coefficient finite-difference formula on a regular grid, where the dependent variable is an N-vector. By introducing new variables as necessary, we may assume that the formula is written in one-step form $v^{n+1} = S_k v^n$. It may be explicit or implicit, provided that in the implicit case, the solvability condition (3.6.5) is satisfied.

If $\|\cdot\|$ is the 2-norm, then the condition (4.2.8) for stability is equivalent to the condition

$$\|G_k(\xi)^n\|_2 \le C \tag{4.6.1}$$

for all $\xi \in [-\pi/h, \pi/h]$ and n, k with $0 \le nk \le T$. Here $G_k(\xi)$ denotes the amplification matrix, an $N \times N$ function of ξ, as described in §3.6. Stability is thus a question of power-boundedness of a family of $N \times N$ matrices, a family indexed by the two parameters ξ and k. This is just the question that was addressed in the last section.

The simplest estimates of the powers $\|G_k(\xi)^n\|$ are based on the norm $\|G_k(\xi)\|$ or the spectral radius $\rho(G_k(\xi))$, that is, the largest of the moduli of the eigenvalues of $G_k(\xi)$. These two quantities provide a lower and an upper bound on (4.6.1) according to the easily proved inequalities

$$\rho(G_k(\xi))^n \le \|G_k(\xi)^n\| \le \|G_k(\xi)\|^n. \tag{4.6.2}$$

Combining (4.6.1) and (4.6.2) yields:

VON NEUMANN CONDITION
FOR VECTOR FINITE DIFFERENCE FORMULAS

Theorem 4.10. *Let $\{S_k\}$ be a linear, constant-coefficient finite difference formula as described above. Then*

(a) $\rho(G_k(\xi)) \le 1 + O(k)$ *is necessary for stability, and* (4.6.3)

(b) $\|G_k(\xi)\| \le 1 + O(k)$ *is sufficient for stability.* (4.6.4)

Both (a) and (b) are assumed to hold as $k \to 0$, uniformly for all $\xi \in [-\pi/h, \pi/h]$. Condition *(a)* is called the **von Neumann condition.** For the record, let us give it a formal statement:

VON NEUMANN CONDITION. The spectral radius of the amplification matrix satisfies

$$\rho(G_k(\xi)) \le 1 + O(k) \tag{4.6.5}$$

as $k \to 0$, uniformly for all $\xi \in [-\pi/h, \pi/h]$.

In summary, for vector or multistep problems, the von Neumann condition is a statement about eigenvalues of amplification matrices, and it is necessary but not sufficient for stability.

Obviously there is a gap between conditions (4.6.3) and (4.6.4). To eliminate this gap one can apply the Kreiss matrix theorem. For any matrix A and constant $\epsilon \geq 0$, let us define the **ϵ-pseudospectral radius** $\rho_\epsilon(A)$ of A to be the largest of the moduli of its ϵ-pseudo-eigenvalues, that is,

$$\rho_\epsilon(A) = \sup_{\lambda_\epsilon \in \Lambda_\epsilon(A)} |\lambda_\epsilon|. \tag{4.6.6}$$

From condition *(ii)* of the definition of $\Lambda_\epsilon(A)$, it is easily seen that an equivalent definition is

$$\rho_\epsilon(A) = \sup_{\|E\| \leq \epsilon} \rho(A+E). \tag{4.6.7}$$

Theorem 4.10 can be restated in terms of the ϵ-pseudospectral radius as follows: a matrix A is power-bounded if and only if

$$\rho_\epsilon(A) \leq 1 + O(\epsilon) \tag{4.6.8}$$

as $\epsilon \to 0$. For the purpose of determining stability we need to modify this condition slightly in recognition of the fact that only powers n with $nk \leq T$ are of interest. Here is the result:

STABILITY VIA THE KREISS MATRIX THEOREM

Theorem 4.11. *A linear, constant-coefficient finite difference formula $\{S_k\}$ is stable in the 2-norm if and only if*

$$\rho_\epsilon(G_k(\xi)) \leq 1 + O(\epsilon) + O(k) \tag{4.6.9}$$

as $\epsilon \to 0$ and $k \to 0$.

The "O" symbols in this theorem are understood to apply uniformly with respect to $\xi \in [-\pi/h, \pi/h]$.

Proof. A more explicit expression of (4.6.9) is

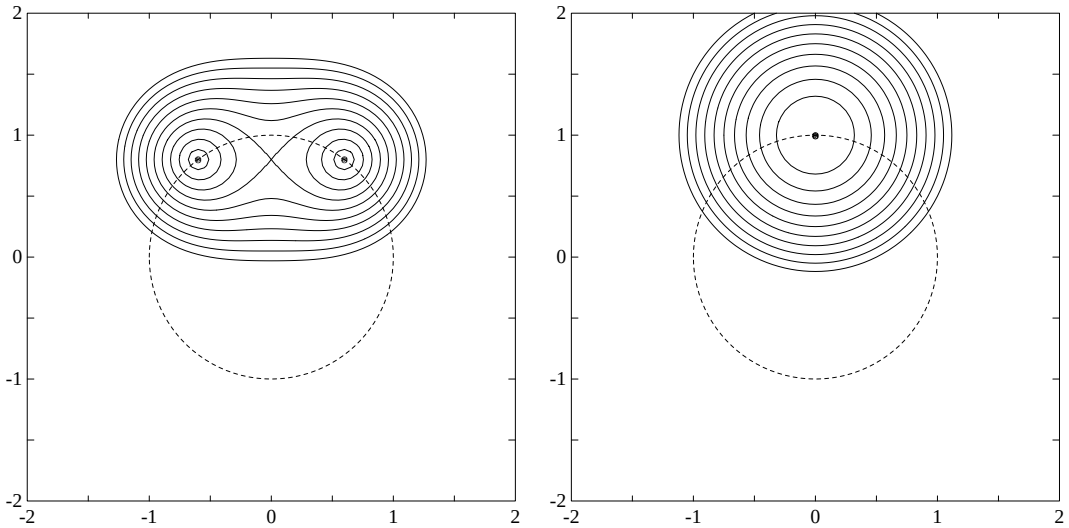
$$|\lambda_\epsilon| \leq 1 + C_1\epsilon + C_2 k \tag{4.6.10}$$

for all λ_ϵ in the ϵ-pseudospectrum $\Lambda_\epsilon(G_k(\xi))$, all $\epsilon \geq 0$, all $k > 0$, and all $\xi \in [-\pi/h, \pi/h]$. Equivalently,

$$\|(zI - G(\xi))^{-1}\|_2 \leq \frac{C}{|z| - (1 + C_2 k)} \tag{4.6.11}$$

for all $|z| > 1 + C_2 k$. [From here it's easy; to be completed later.] ∎

Theorem 4.11 is a powerful result, giving a necessary and sufficient condition for stability of a wide variety of problems, but it has two limitations. The first is that determining resolvent norms and pseudospectra is not always an easy matter, and that is why it is convenient to have Theorem 4.10 available as well. The second is that not all numerical methods satisfy the rather narrow requirements of constant coefficients and regular unbounded grids that make Fourier analysis applicable. This difficulty will be addressed in the next section.



(a) $\lambda = 0.8$ (b) $\lambda = 1$

Figure 4.6.1. Boundaries of ϵ-pseudospectra of the leap frog amplification matrix $G(\xi)$ with $\xi = \pi/2$ ($\epsilon = 0.05, 0.10, \ldots, 0.50$). For $\lambda = 1$ there is a defective eigenvalue at $z = i$, the condition $\rho_\epsilon(G(\xi)) \le 1 + O(\epsilon)$ fails, and the formula is unstable.

EXAMPLE 4.6.1. For the leap frog model of $u_t = u_t$, the amplification matrix

$$G(\xi) = \begin{pmatrix} 2i\lambda \sin \xi h & 1 \\ 1 & 0 \end{pmatrix} \tag{4.6.12}$$

was derived in Example 3.6.1. For $\lambda < 1$ this family of matrices is power-bounded, but for $\lambda = 1$, the matrix $G(\pi/2)$ is defective and not power-bounded. The pseudospectra for $G(\pi/2)$ for $\lambda = 0.8$ and $\lambda = 1$ are illustrated and Figure 4.6.1. See Exercise 4.6.2.

In practice, how do people test for stability of finite difference methods? Usually, by computing eigenvalues and checking the von Neumann condition. If it is satisfied, the method is often stable, but sometimes it is not. It is probably accurate to say that when instability occurs, the problem is more often in the boundary conditions or nonlinearities than in the gap between Theorems 4.10 and 4.11 associated with non-normality of the amplification matrices. In Chapter 6 we shall discuss additional tests that can be used to check for instability introduced by boundary conditions.

EXERCISES

▷ *4.6.1. Multidimensional wave equation.* Consider again the second-order wave equation in d space dimensions

$$u_{tt} = u_{x_1x_1} + \cdots + u_{x_dx_d},$$

and the finite difference approximation discussed in Exercise 4.3.1. Use d-dimensional Fourier analysis to determine the stability bound on λ. (You do not have to use matrices and do it rigorously, which would involve an amplification matrix with a defective eigenvalue under certain conditions; just plug in the appropriate Fourier mode solution ("Ansatz") and compute amplification factors. You need not worry about keeping track of strong vs. weak inequalities.) Is it the same as the result of Exercise 4.3.1?

▷ *4.6.2. Stability of leap frog.* Consider the leap frog model of $u_t = u_x$ with $\lambda = k/h =$ constant ≤ 1 (Example 4.6.1 and Figure 4.6.1).

(a) Compute the eigenvalues and spectral radius of $G_k(\xi)$, and verify that the von Neumann condition does not reveal leap frog to be unstable.

(b) Compute the 2-norm $\|G_k(\xi)\|$, and verify that the condition (4.6.4), which would guarantee that leap frog is stable, does not hold.

(c) In fact, leap frog is stable for $\lambda < 1$. Prove this by any means you wish, but be sure that you have shown boundedness of the powers $G_k(\xi)$ uniformly in ξ, not just for each ξ individually. One way to carry out the proof is to argue first that for each fixed ξ, $\|G^n(\xi)\| \le M(\xi)$ for all n for an appropriate function $M(\xi)$, and then argue that $M(\xi)$ must be bounded as a function of ξ. Another method is to compute the eigenvalue decomposition of $G(\xi)$.

▷ *4.6.3. The fourth-order leap frog formula.* In Table 4.4.1 the stability restriction for the fourth-order leap frog formula is listed as $\lambda < 0.728\ldots$. What is this number?

▷ *4.6.4. The DuFort-Frankel formula.* The DuFort-Frankel model of $u_t = u_{xx}$, listed in Table 3.2.2 on p. 120, has some remarkable properties.

(a) Derive an amplification matrix for this formula. (This was done already in Exercise 3.6.1.)

(b) Show that it is unconditionally stable in ℓ_h^2.

(c) What does the result of *(b)*, together with the theorems of this chapter, imply about the consistency of the DuFort-Frankel formula with $u_t = u_{xx}$? Be precise, and state exactly what theorems you have appealed to.

(d) By manipulating Taylor series, derive the precise consistency restriction (involving k and h) for the DuFort-Frankel formula. Is it is same as the result of *(c)*?

4.7. Stability of the method of lines

[This section is not yet written. Most of its results can be found in S. C. Reddy and L. N. Trefethen, "Stability of the method of lines," *Numerische Mathematik 62* (1992), 235–267.]

The Kreiss matrix theorem (Theorem 4.9) asserts that a family of matrices $\{A_\nu\}$ is power-bounded if and only if its ϵ-pseudospectra $\Lambda_\epsilon(A_\nu)$ lie within a distance $O(\epsilon)$ of the unit disk, or equivalently, if and only if its resolvent norms $\|(zI-A_\nu)^{-1}\|$ increase at most inverse-linearly as z approaches the unit disk D from the outside. If the matrices all have a fixed dimension N, then this statement is valid exactly as it stands, and if the dimensions N_ν are variable, one loses a factor $\min\{N, n+1\}$, that is, $O(N)$ or $O(n)$.

The Kreiss matrix theorem has many consequences for stability of numerical methods for time-dependent PDEs. To apply it to this purpose, we first make a small modification so that we can treat stability on a finite interval $[0,T]$, as in §4.2, rather than the infinite interval $[0,\infty)$. All that is involved is to replace $O(\epsilon)$ by $O(\epsilon)+O(k)$, as in Theorem 4.11.

It turns out that there are four particularly important consequences of the Kreiss matrix theorem, which can be arranged in a two-by-two table according to the following two binary choices. Theorem 4.11 was one of these four consequences.

First, one can work either with the operators $\{S_k\}$ as matrices, or with the amplification matrices $\{G_k(\xi)\}$. The latter choice is only possible when Fourier analysis is applicable, i.e., under the usual restrictions of constant coefficients, regular grids, no boundaries, etc. It has the great advantage that the dimensions of the matrices involved are fixed, so there is no factor $O(n)$ or $O(N)$ to worry about, and indeed, $G_k(\xi)$ is often independent of k. When Fourier analysis is inapplicable, however, one always has the option of working directly with the matrices $\{S_k\}$ themselves—in "space space" instead of Fourier space. This is customary for example in the stability analysis of spectral methods on bounded domains.

Thus our first pair of theorems are as follows:

STABILITY
Theorem 4.11 (again). *A linear, constant-coefficient finite difference formula $\{S_k\}$ is stable in the 2-norm if and only if the pseudo-eigenvalues $\lambda_\epsilon \in \Lambda_\epsilon(G_k(\xi))$ of the amplification matrices satisfy* $$dist(\lambda_\epsilon, D) = O(\epsilon)+O(k) \qquad (4.7.1)$$

STABILITY IN FOURIER SPACE
Theorem 4.12. *A linear finite difference formula $\{S_k\}$ is stable, up to a factor* $\min\{N, n+1\}$, *if and only if the pseudo-eigenvalues $\lambda_\epsilon \in \Lambda_\epsilon(S_k)$ satisfy* $$dist(\lambda_\epsilon, D) = O(\epsilon)+O(k) \qquad (4.7.2)$$

In these theorems the order symbols $O(\epsilon)+O(k)$ should be understood to apply uniformly for all k and (where appropriate) ξ as $k\to 0$ and $\xi\to 0$.

As a practical matter, the factor $\min\{N,n+1\}$ is usually not important, because most often the instabilities that cause trouble are exponential. "One derivative of smoothness" in the initial and forcing data for a time-dependent PDE is generally enough to ensure that such a factor will not prevent convergence as the mesh is refined. In a later draft of the book this point will be emphasized in Chapter 4.

The other pair of theorems comes when we deal with the method of lines, discussed previously in §3.3. Following the standard formulation of the Lax-Richtmyer stability theory in Chapter 4, suppose we are given an autonomous linear partial differential equation

$$u_t = \mathcal{L}u, \tag{4.7.3}$$

where $u(t)$ is a function of one or more space variables on a bounded or unbounded domain and $\mathcal{L}$ is a differential operator, independent of t. For each sufficiently small time step $k>0$, let a corresponding finite or infinite spatial grid be defined and let (4.7.3) first be discretized with respect to the space variables only, so that it becomes a system of ordinary differential equations,

$$v_t = L_k v, \tag{4.7.4}$$

where $v(t)$ is a vector of dimension $N_k \le \infty$ and L_k is a matrix or bounded linear operator. With the space discretization determined in this way, let (4.7.4) then be discretized with respect to t by a linear multistep or Runge-Kutta formula with time step k. We assume that the stability region S is bounded by a cusp-free curve. Then one can show:

STABILITY OF THE METHOD OF LINES

Theorem 4.13. *The method of lines discretization described above is stable, up to a factor* $\min\{N,n+1\}$, *if and only if the pseudo-eigenvalues* $\lambda_\epsilon \in \Lambda_\epsilon(kL_k)$ *satisfy*

$$dist(\lambda_\epsilon, S) = O(\epsilon)+O(k) \tag{4.7.5}$$

STABILITY OF THE METHOD OF LINES IN FOURIER SPACE

Theorem 4.14. *A linear, constant-coefficient method of lines discretization as described above is stable in the 2-norm if and only if the pseudo-eigenvalues* $\lambda_\epsilon \in \Lambda_\epsilon(k\hat{L}_k(\xi))$ *satisfy*

$$dist(\lambda_\epsilon, S) = O(\epsilon)+O(k) \tag{4.7.6}$$

When the matrices S_k or $G_k(\xi)$ or L_k or $\hat{L}_k(\xi)$ appearing in these theorems are normal, one can simplify the statements by replacing ϵ-pseudo-eigenvalues by eigenvalues and $O(\epsilon)$ by 0. In particular, for a method of lines calculation in which the space discretization matrices L_k are normal, a necessary and sufficient condition for stability is that the eigenvalues of $\{kL_k\}$ lie within a distance $O(k)$ of the stability region S as $k\to 0$.

Chapter 5.
Dissipation, Dispersion, and Group Velocity

Things fall apart; the center cannot hold;
Mere anarchy is loosed upon the world.
— *W. B. YEATS, The Second Coming (190–)*

It would be a fine thing if discrete models calculated solutions to partial differential equations exactly, but of course they do not. In fact in general they could not, even in principle, since the solution depends on an infinite amount of initial data. Instead, the best we can hope for is that the errors introduced by discretization will be small when those initial data are reasonably well-behaved.

This chapter is devoted to understanding the behavior of numerical errors. From truncation analysis we may have a bound on the magnitude of discretization errors, depending on the step sizes h and k, but much more can be said, for the behavior of discretization errors exhibits great regularity, which can be quantified by the notions of numerical dissipation and dispersion. Rounding errors too, though introduced essentially at random, propagate in the same predictable ways.

So long as we can estimate the magnitude of the discretization and rounding errors, what is the point in trying to investigate their behavior in more detail? There are several answers to this question. One is that it is a good idea to train the eye: a practitioner familiar with artificial numerical effects is less likely to mistake spurious features of a numerical solution for mathematical or physical reality. Another is that in certain situations it may be advantageous to design schemes with special properties—low dissipation, for example, or low dispersion. A third is that in more complicated circumstances, the magnitude of global errors may depend on the behavior of local errors in ways that ordinary analysis of discretization and rounding errors cannot predict. In particular, we shall see in the next chapter that the stability of boundary conditions for hyperbolic partial differential equations depends upon phenomena of numerical dispersion.

One might say that this chapter is built around an irony: finite difference approximations have a more complicated "physics" than the equations they are designed to simulate. The irony is no paradox, however, for finite differences are used not because the numbers they generate have simple properties, but because those numbers are simple to compute.

5.1. Dispersion relations

Any time-dependent scalar, linear partial differential equation with constant coefficients on an unbounded space domain admits plane wave solutions

$$u(x,t) = e^{i(\xi x + \omega t)}, \qquad \xi \in \mathbb{R}, \tag{5.1.1}$$

where ξ is the **wave number** and ω is the **frequency**. (Vector differential equations admit similar modes multiplied by a constant vector; the extension to multiple space dimensions is described at the end of this section.) For each ξ, not all values of ω can be taken in (5.1.1). Instead, the PDE imposes a relationship between ξ and ω,

$$\omega = \omega(\xi), \tag{5.1.2}$$

which is known as the **dispersion relation**, mentioned already in §3.1. In general each wave number ξ corresponds to m frequencies ω, where m is the order of the differential equation with respect to t, and that is why (5.1.2) is called a relation rather than a function. For most purposes it is appropriate to restrict attention to values of ξ that are real, in which case ω may be real or complex, depending on the PDE. The wave (5.1.1) decays as $t \to \infty$ if $\operatorname{Im} \omega > 0$, and grows if $\operatorname{Im} \omega < 0$.

For example, here again are the dispersion relations for the model equations of §3.1, and also for the second-order wave equation:

$$u_t = u_x: \qquad \omega = \xi, \tag{5.1.3}$$

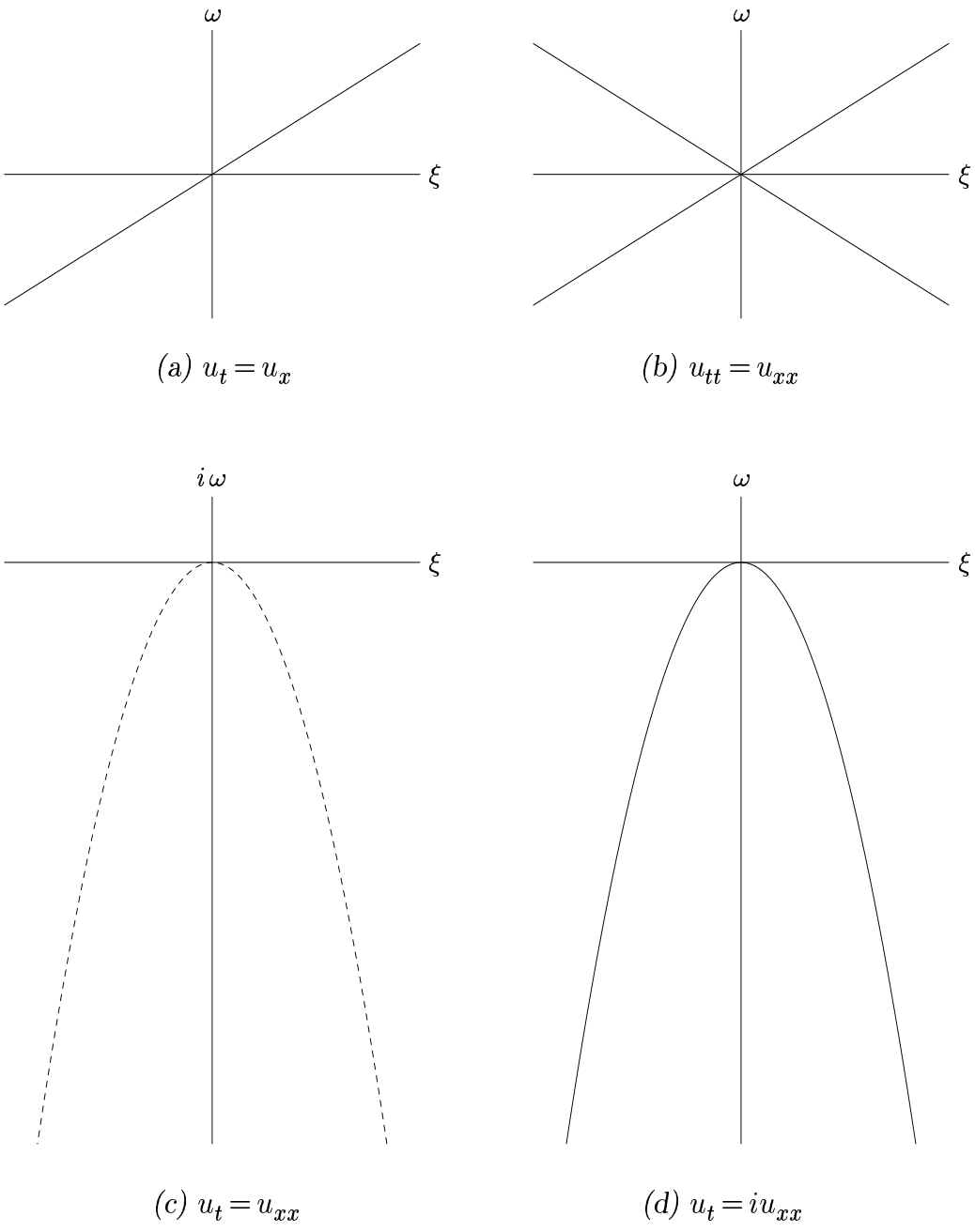
$$u_{tt} = u_{xx}: \qquad \omega^2 = \xi^2, \text{ i.e., } \omega = \pm\xi, \tag{5.1.4}$$

$$u_t = u_{xx}: \qquad i\omega = -\xi^2, \tag{5.1.5}$$

$$u_t = iu_{xx}: \qquad \omega = -\xi^2. \tag{5.1.6}$$

These relations are plotted in Figure 5.1.1. Notice the double-valuedness of the dispersion relation for $u_{tt} = u_{xx}$, and the dashed curve indicating complex values for $u_t = u_{xx}$.

More general solutions to these partial differential equations can be obtained by superimposing plane waves (5.1.1), so long as each component satisfies the dispersion relation; the mathematics behind such Fourier synthesis



(a) $u_t = u_x$ (b) $u_{tt} = u_{xx}$

(c) $u_t = u_{xx}$ (d) $u_t = iu_{xx}$

Figure 5.1.1. Dispersion relations for four model partial differential equations. The dashed curve in *(c)* is a reminder that ω is complex.

was described in Chapter 2, and examples were given in §3.1. For a PDE of first order in t, the result is

$$u(x,t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} e^{i(\xi x+\omega(\xi)t)}\hat{u}_0(\xi)\,d\xi. \tag{5.1.7}$$

Since most partial differential equations of practical importance have variable coefficients, nonlinearity, or boundary conditions, it is rare that this integral representation is exactly applicable, but it may still provide insight into local behavior.

Discrete approximations to differential equations also admit plane wave solutions (5.1.1), at least if the grid is uniform, and so they too have dispersion relations. To begin with, let us discretize in x only so as to obtain a semidiscrete formula. Here are the dispersion relations for the standard centered semidiscretizations of (5.1.3)–(5.1.6):

$$u_t = \delta_0 u: \qquad \omega = \frac{1}{h}\sin\xi h, \tag{5.1.8}$$

$$u_{tt} = \delta_\times u: \qquad \omega^2 = \frac{4}{h^2}\sin^2\frac{\xi h}{2}, \tag{5.1.9}$$

$$u_t = \delta_\times u: \qquad i\omega = -\frac{4}{h^2}\sin^2\frac{\xi h}{2}, \tag{5.1.10}$$

$$u_t = i\delta_\times u: \qquad \omega = -\frac{4}{h^2}\sin^2\frac{\xi h}{2}. \tag{5.1.11}$$

These formulas are obtained by substituting (5.1.1) into the finite difference formulas with $x = x_j$. In keeping with the results of §2.2, each dispersion relation is $2\pi/h$-periodic in ξ, and it is natural to take $\xi \in [-\pi/h, \pi/h]$ as a fundamental domain. The dispersion relations are plotted in Figure 5.1.2, superimposed upon dotted curves from Figure 5.1.1 for comparison.

Stop for a moment to compare the continuous and semidiscrete curves in Figure 5.1.2. In each case the semidiscrete dispersion relation is an accurate approximation *when ξ is small*, which corresponds to *many grid points per wavelength*. (The number of points per spatial wavelength for the wave (5.1.1) is $2\pi/\xi h$.) In general, the dispersion relation for a partial differential equation is a polynomial relation between ξ and ω, while a discrete model amounts to a trigonometric approximation. Although other design principles are possible, the standard discrete approximations are chosen so that the trigonometric function matches the polynomial to as high a degree as possible at the origin $\xi = \omega = 0$. To illustrate this idea, Figure 5.1.3 plots dispersion relations for the standard semidiscrete finite difference approximations to $u_t = u_x$ and $u_t = iu_{xx}$ of orders 2, 4, and 6. The formulas were given in §3.3.

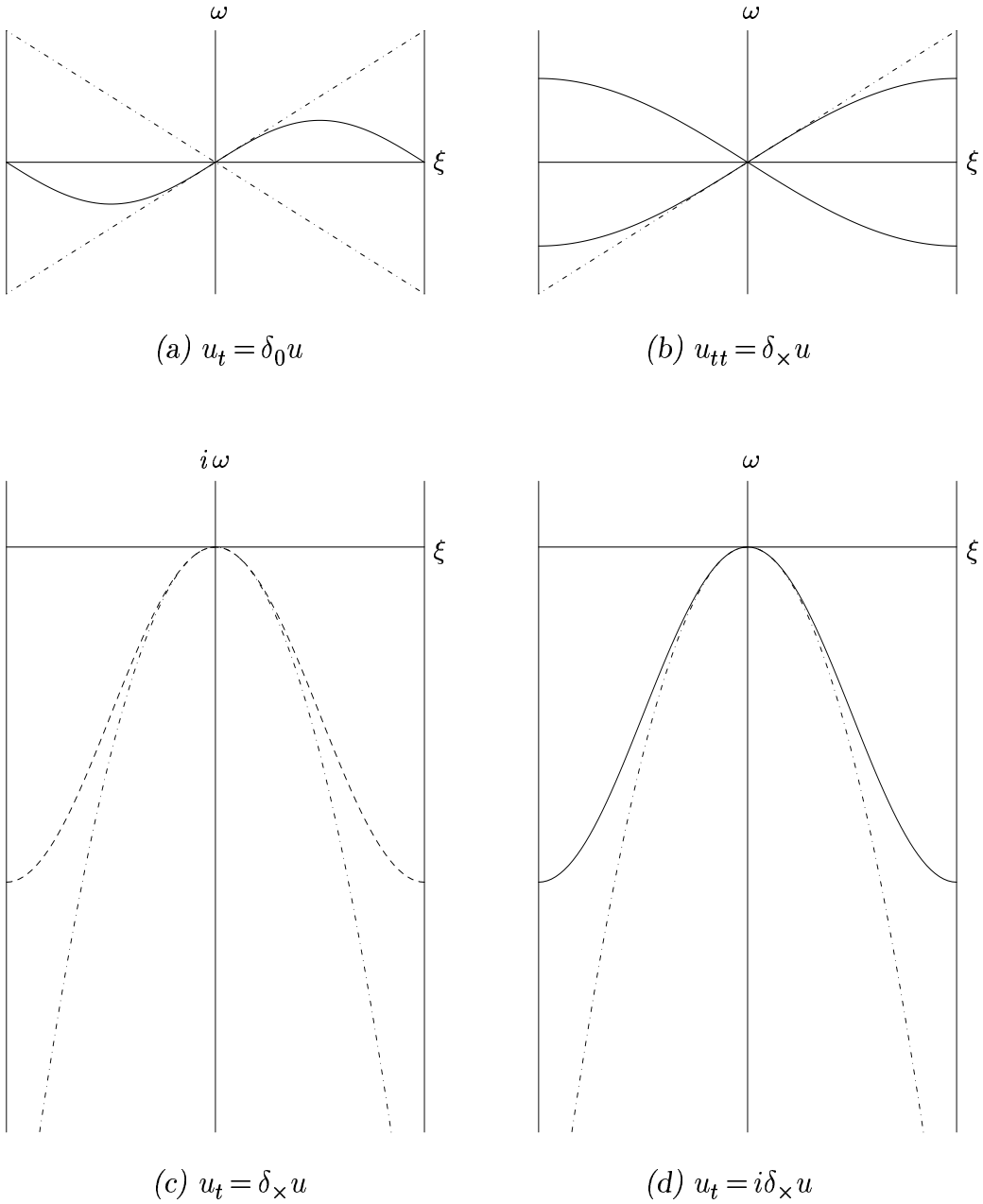
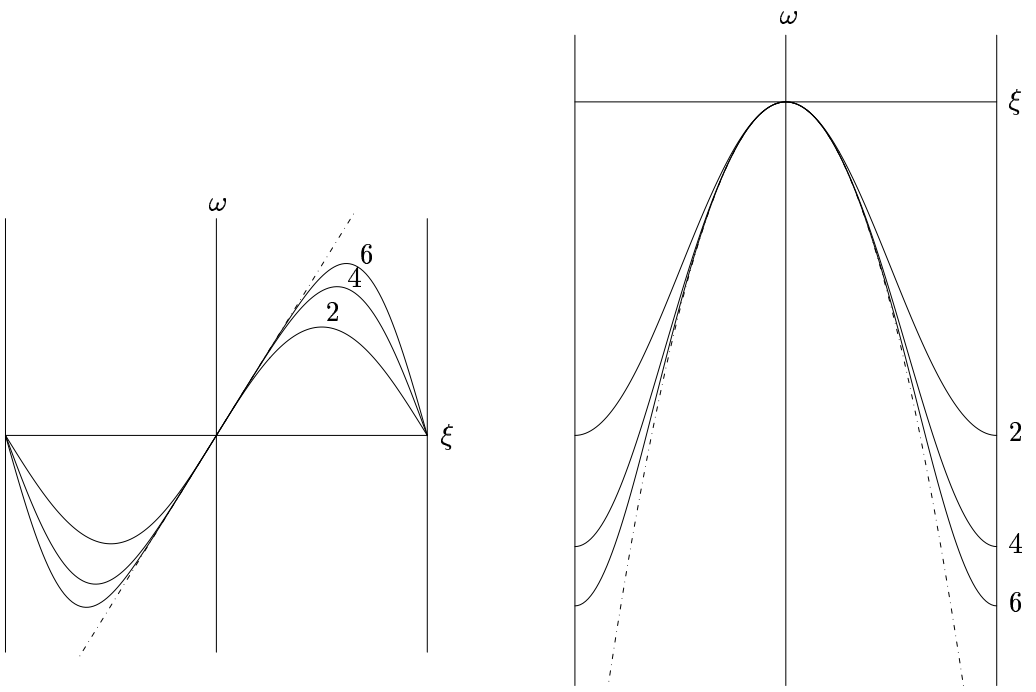


Figure 5.1.2. Dispersion relations for centered semidiscrete approximations to the four model partial differential equations. Each function is $2\pi/h$-periodic in ξ; the plots show the fundamental domain $\xi \in [-\pi/h, \pi/h]$.



(a) $u_t = u_x$ (b) $u_t = iu_{xx}$

Figure 5.1.3. Dispersion relations for semidiscrete centered difference approximations to $u_t = u_x$ and $u_t = iu_{xx}$ of orders 2, 4, 6.

Now let us turn to fully discrete finite difference formulas: discrete in time as well as space. The possibilities become numerous. For example, substituting the plane wave

$$v_j^n = e^{i(\xi x_j + \omega t_n)} = e^{i(\xi jh + \omega nk)}$$

into the leap frog approximation of $u_t = u_x$ yields the dispersion relation

$$e^{i\omega k} - e^{-i\omega k} = \lambda(e^{i\xi x} - e^{-i\xi x}),$$

where $\lambda = k/h$, that is,

$$\sin \omega k = \lambda \sin \xi h. \tag{5.1.12}$$

Similarly, the Crank-Nicolson approximation of $u_t = u_{xx}$ has the dispersion relation

$$e^{i\omega k} - 1 = \frac{\sigma(e^{i\omega k} + 1)}{2}\left[e^{i\xi h} - 2 + e^{i\xi h}\right],$$

which reduces to

$$i \tan \frac{\omega k}{2} = -2\sigma \sin^2 \frac{\xi h}{2}. \tag{5.1.13}$$

These dispersion relations are $2\pi/h$-periodic in ξ and $2\pi/k$-periodic in ω. With the use of inverse trigonometric functions it is possible to solve such equations for ω, so as to exhibit the functional dependence explicitly, but the resulting formulas are less appealing and often harder to work with. Equations like (5.1.12) and (5.1.13) have a certain elegance—one sees at a glance that the time and space derivatives have been replaced by trigonometric analogs.

Tables 5.1.1 and 5.1.2* consider once again the various finite difference approximations to $u_t = u_x$ and $u_t = u_{xx}$ that appeared in Tables 3.2.1/3.2.2 and 4.4.1/4.4.2. In each case the dispersion relation is both listed and plotted. Since h and k are independent parameters, there is now a range of possible plots; we have arbitrarily taken $\lambda = k/h = 0.5$ in the first table and $\sigma = k/h^2 = 0.5$ in the second. That is why each plot occupies a rectangle of aspect ratio 2 in wave number space.† Notice that the multistep (leap frog) formulas contain two branches of ω values.

For partial differential equations in several space dimensions, the notion of dispersion relation generalizes just as §2.6 would suggest: a plane wave takes the form

$$u(x,t) = e^{i(\xi\cdot x+\omega t)}, \qquad \xi \in \mathbb{R}, \tag{5.1.14}$$

where ξ and x are vectors, and (5.1.2) becomes a scalar function (or relation) of a vector argument. For example, the wave equation

$$u_{tt} = u_{xx} + u_{yy}$$

has the dispersion relation

$$\omega^2 = \xi^2 + \eta^2, \tag{5.1.15}$$

if the vector ξ is written (ξ,η), so that the lines of constant ω in the (ξ,η) plane are concentric circles. On the other hand the leap frog approximation

$$v_{ij}^{n+1} - 2v_{ij}^n + v_{ij}^{n-1} = \lambda^2(v_{i+1,j}^n + v_{i-1,j}^n + v_{i,j+1}^n + v_{i,j-1}^n - 4v_{ij}^n),$$

appropriate to a uniform grid with $h = \Delta x = \Delta y$, has the dispersion relation

$$\sin^2\frac{\omega k}{2} = \lambda^2\Big[\sin^2\frac{\xi h}{2} + \sin^2\frac{\eta h}{2}\Big], \tag{5.1.16}$$

which is plotted in Figure 5.1.4 for $\lambda \approx 0$. Once again the dispersion relation is accurate for small wave numbers but diverges dramatically elsewhere.

EXERCISES

▷ *5.1.1.* What are the coefficients as trigonometric functions of the dispersion relations plotted in Figure 5.1.3?

▷ *5.1.2.* Sketch the dispersion relation for the leap frog model of $u_t = u_x$ with $\lambda > 1$—say, $\lambda = 2$. How is the instability of this finite difference formula reflected in your sketch?

*Not yet written.

†analogous to a **Brillouin zone** in crystallography (C. Kittel, *Introduction to Solid State Physics*, Wiley, 1976).

BE_x = Backward Euler	$-i(1-e^{-i\omega k}) = \lambda \sin \xi h$
CN_x = Crank-Nicolson	$2 \tan \frac{\omega k}{2} = \lambda \sin \xi h$
LF = Leap Frog	$\sin \omega k = \lambda \sin \xi h$
BOX_x = Box	$\tan \frac{\omega k}{2} = \lambda \tan \frac{\xi h}{2}$
LF4 = 4th-order Leap Frog	$\sin \omega k = \frac{4}{3}\lambda \sin \xi h - \frac{1}{6}\lambda \sin 2\xi h$
LXF = Lax-Friedrichs	$e^{i\omega k} = \cos \xi h + i\lambda \sin \xi h$
UW = Upwind	$e^{i\omega k} - 1 = \lambda(e^{i\xi h} - 1)$
LW = Lax-Wendroff	$-i(e^{i\omega k} - 1) = \lambda \sin \xi h + 2i\lambda^2 \sin^2 \frac{\xi h}{2}$

Table 5.1.1. Dispersion relations for various finite difference approximations to $u_t = u_x$ with $\lambda = k/h = 0.5$. See Tables 3.2.1 and 4.4.1. The dashed lines indicate the slope $d\omega/d\xi$ at isolated points where ω is real.

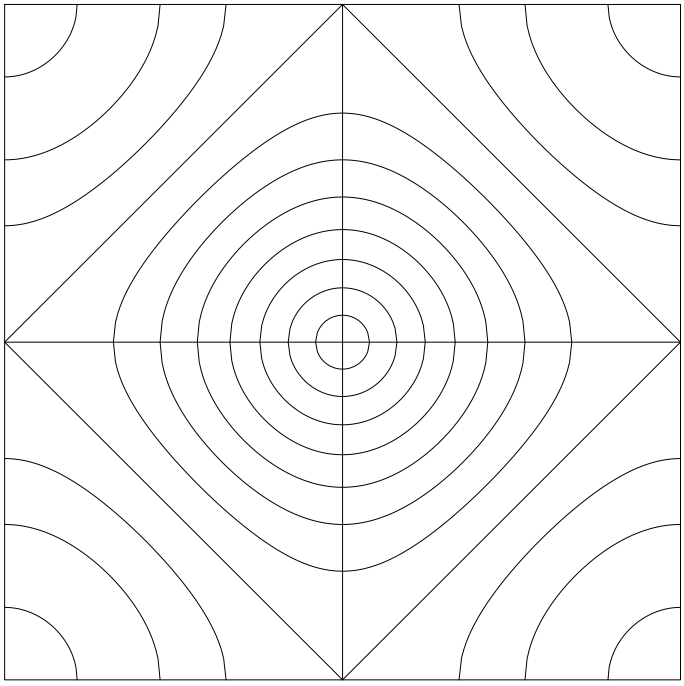


Figure 5.1.4. Dispersion relation for the leap frog model of $u_{tt} = u_{xx} + u_{yy}$ in the limit $\lambda \to 0$. The region shown is the domain $[-\pi/h, \pi/h]^2$ of the (ξ, η) plane. The concentric curves are lines of constant ω for $\omega h = \frac{1}{4}, \frac{1}{2}, \ldots, \frac{11}{4}$.

5.2. Dissipation

Even though a partial differential equation may conserve energy in the L^2 norm, its finite difference models will often lose energy as t increases, especially in the wave numbers comparable to the grid size. This property is **numerical dissipation**, and it is often advantageous, since it tends to combat instability and unwanted oscillations. In fact, **artificial dissipation*** is often added to otherwise nondissipative formulas to achieve those ends. An example of this kind appeared in Exercise 3.2.1.

To make the matter quantitative, suppose we have a linear partial differential equation or finite difference approximation that admits waves (5.1.1) with ω given by a dispersion relation (5.1.2). Since ξ is assumed to be real, it follows that the wave has absolute value

$$|e^{i(\xi x + \omega t)}| = e^{-t \operatorname{Im} \omega} \tag{5.2.1}$$

*In computational fluid dynamics one encounters the more dramatic term **artificial viscosity**.

as a function of t, and thus decays exponentially if $\mathrm{Im}\,\omega > 0$. By Parseval's equality, the L^2 norm of a superposition of waves (5.1.1) is determined by a superposition of such factors:

$$\|u(\cdot,t)\| = \frac{1}{2\pi}\left\|e^{-t\,\mathrm{Im}\,\omega(\xi)}\hat{u_0}(\xi)\right\|. \tag{5.2.2}$$

As an extreme case the heat equation $u_t = u_{xx}$, with dispersion relation $\omega = i\xi^2$, dissipates nonzero wave numbers strongly—and indeed it does nothing else; at time t, only wave numbers $\xi = O(\sqrt{t})$ remain with close to their initial amplitudes. But it is principally the dissipation introduced by finite difference formulas that we are concerned with here.

The following definitions are standard:

A finite difference formula is **nondissipative** *if* $\mathrm{Im}\,\omega = 0$ *for all* ξ. *It is* **dissipative** *if* $\mathrm{Im}\,\omega > 0$ *for all* $\xi \neq 0$. *It is* **dissipative of order** $2r$ *if* ω *satisfies*

$$\mathrm{Im}\,\omega k \geq \gamma_1(\xi h)^{2r}, \qquad \textit{i.e.,} \qquad |e^{i\omega k}| \leq 1 - \gamma_2(\xi h)^{2r} \tag{5.2.3}$$

for some constants $\gamma_j > 0$. *In each of these statements,* ξ *varies over the interval* $[-\pi/h, \pi/h]$, *and* ω *represents all possible values* ω *corresponding to a given* ξ. *For problems in multiple space dimensions,* (ξh) *is replaced by* $\|\xi h\|$ *in any norm.*

For example, the leap frog and Crank-Nicolson models of $u_t = u_x$ are nondissipative, while the upwind and Lax-Wendroff formulas are dissipative.

Dissipative and nondissipative are mutually exclusive, but not exhaustive: a finite difference formula can be neither dissipative nor nondissipative. See Exercise 5.2.1.

According to the definition, no consistent finite difference approximation of $u_t = u_x + u$ could be dissipative, but customary usage would probably use the term dissipative sometimes for such a problem anyway. One could modify the definition to account for this by including a term $O(k)$ in (5.2.3).

A more serious problem with these standard definitions arises in the case of multistep formulas, for which each ξ corresponds to several values of ω. In such cases the definition of dissipative, for example, should be interpreted as requiring $\mathrm{Im}\,\omega > 0$ for every value of ω that corresponds to some $\xi \neq 0$. The difficulty arises because for multistep formulas, that condition ensures only that the formula dissipates oscillations in space, not in time. For example, the leap frog model becomes dissipative if a small term such as $k\,\delta_x v^n$ is added to it, according to our definitions, yet the resulting formula still admits the wave (5.1.1) with $\xi = 0$, $\omega = \pi/h$, which is sawtoothed in time. To exclude possibilities of that kind it is sometimes desirable to use a stronger definition:

A finite difference formula is **totally dissipative** *if it is dissipative and in addition,* $\mathrm{Im}\,\omega = 0$ *implies* $\omega = 0$.

EXERCISES

▷ *5.2.1.* Determine whether each of the following models of $u_t = u_x$ is nondissipative, dissipative, or neither. If it is dissipative, determine the order of dissipativity.
(a) Lax-Wendroff, *(b)* Backward Euler, *(c)* Fourth-order leap frog, *(d)* Box, *(e)* Upwind.

5.3. Dispersion and group velocity

[This section is not yet properly written. The next few pages contain a few remarks, followed by an extract from my paper "Dispersion, dissipation, and stability."]

The general idea. Whereas dissipation leads to decay of a wave form, dispersion leads to its gradual separation into a train of oscillations. This phenomenon is a good deal less obvious intuitively, for it depends upon constructive and destructive interference of Fourier components. It is of central importance in finite difference modeling, because although many partial differential equations are nondispersive, their discrete approximations are almost invariably dispersive. (Spectral methods are an exception.)

Caution. Phase and group velocity analysis depend upon the problem being linear and nondissipative—i.e., ω must be real when ξ is real. (However, similar predictions hold if there is a sufficiently slight amount of dissipation.)

Phase velocity. Suppose that a PDE or finite difference formula admits a solution $e^{i(\omega t+\xi x)}$. It's then a triviality to see that any individual "wave crest" of this wave, i.e., a point moving in such a way that the quantity inside the parentheses has a constant value (the phase), moves at the velocity

$$\textbf{Phase velocity:} \quad c(\xi,\omega) = -\frac{\omega}{\xi}. \tag{5.3.1}$$

Group velocity. However, early in the twentieth century it was realized that wave energy propagates at a different velocity,

$$\textbf{Group velocity:} \quad c_g(\xi,\omega) = -\frac{d\omega}{d\xi}. \tag{5.3.2}$$

The algebraic meaning of this expression is that we differentiate the dispersion relation with respect to ξ. (In a plot in ξ–ω space, c is minus the slope of the line through (ξ,ω) and the origin, and c_g is minus the slope of the line tangent to the dispersion relation at (ξ,ω).) The physical meaning is that, for example, a **wave packet**—a smooth envelope times an oscillatory carrier wave with parameters (ξ,ω)—will move approximately at the velocity c_g. The same goes for a wave front and for any other signal that can carry information.

Dispersive vs. nondispersive systems. If the dispersion relation is linear, i.e., $\omega = \text{const}\cdot\xi$, then (5.3.1) and (5.3.2) are equal and the system is **nondispersive**. If the dispersion relation is nonlinear, the system is **dispersive**. Finite difference formulas are almost always dispersive, since their dispersion relations are periodic and therefore nonlinear. (However, see Exercise 5.3.2.)

Precise meaning of group velocity. The meaning of group velocity can be made precise in various asymptotic fashions, for example by considering the limit $t \to \infty$. The mathematics behind this is usually a **stationary phase** or **steepest descent** argument.

Simple explanations of group velocity. There are a number of intuitive ways to understand where the derivative (5.3.2) comes from. One is to superimpose two waves with nearby

parameters (ξ_1,ω_1) and (ξ_2,ω_2). It is then readily seen that the superposition consists of a smooth envelope times a carrier wave at frequency $\frac{1}{2}(\omega_1+\omega_2)$ and wave number $\frac{1}{2}(\xi_1+\xi_2)$, and the envelope moves at velocity $-(\omega_2-\omega_1)/(\xi_2-\xi_1)$, which approaches (5.3.2) in the limit $\xi_2\to\xi_1$, $\omega_2\to\omega_1$. Another approach is to take a pure exponential $e^{i(\omega t+\xi x)}$, with ξ and ω real, and change ξ slightly to a complex value $\xi+i\Delta\xi$ "in order to visualize which way the envelope is moving." If the effect on ω is to make it change to $\omega+i\Delta\omega$, it is readily calculated that the resulting evanescent wave has an envelope that moves laterally at the velocity $-\Delta\omega/\Delta\xi$, and this again approaches (5.3.2) in the limit $\Delta\xi\to 0$, $\Delta\omega\to 0$. A third, more "PDE-style" explanation is based upon advection of local wave number according to a simple hyperbolic equation with coefficient c_g; see Lighthill.

Group velocity in multiple dimensions. If there are several space dimensions, the group velocity becomes the gradient of ω with respect to the vector ξ, i.e., $c_g=-\nabla_\xi\omega$.

Phase and group velocities on a grid. There is another sense in which group velocity has more physical meaning than phase velocity on a finite difference grid: the former is well-defined, but the latter is not. On a periodic grid, any Fourier mode can be represented in terms of infinitely many possible choices of ξ and ω that are indistinguishable physically, and according to (5.3.1), each choice gives a different phase velocity. What's going on here is that naturally one can't tell how fast a pure complex exponential wave is moving if one sees it at only intermittent points in space or time, for one wave crest is indistinguishable from another. By contrast, the group velocity is well-defined, since it depends only on the slope, which is a local property; formula (5.3.2) has the same periodicity as the dispersion relation itself.

Computation of a group velocity on a grid. To compute the group velocity for a finite difference formula, differentiate the dispersion relation implicitly and then solve for $c_g=-d\omega/d\xi$. For example, the wave equation $u_t=u_x$ has $c=c_g=-1$ for all ξ. For the leap frog approximation the dispersion relation is $\sin\omega k=\lambda\sin\omega\xi$, which implies $k\cos\omega k\,d\omega=h\lambda\cos\omega\xi\,d\xi$, and since $k=h\lambda$, $c_g(\xi,\omega)=-\cos\xi h/\cos\omega k$.

Parasitic waves. Many finite difference formulas admit parasitic waves as solutions, i.e., waves that are sawtoothed with respect to space or time. These correspond to $\xi=\pm\pi/h$, $\omega=\pm\pi/k$, or both. It is common for such waves to have group velocities opposite in sign to what is correct physically. In the example of the leap frog formula, all four parasitic modes 1, $(-1)^j$, $(-1)^n$, and $(-1)^{j+n}$ are possible, with group velocities -1, 1, 1, and -1, respectively.

Spurious wiggles near interfaces and boundaries. It is common to observe spurious wiggles in a finite difference calculation, and they appear most often near boundaries, interfaces, or discontinuities in the solution itself. The explanation of where they appear is usually a matter of group velocity. Typically a smooth wave has passed through the discontinuity and generated a small reflected wave of parasitic form, which propagates backwards into the domain because its group velocity has the wrong sign. More on this in the next chapter.

Waves in crystals. Dispersion relations for vibrations in crystals are also periodic with respect to ξ. As a result, sound waves in crystals exhibit dispersive effects much like those associated with finite difference formulas, including the existence of positive and negative group velocities.

Figure 5.3.1. Dispersion under the leap frog model of $u_t = u_x$ with $\lambda = 0.5$. The lower mesh is twice as fine as the upper.

EXERCISES

▷ *5.3.1. The Box formula.*

(a) Write out the BOX_x formula of Table 3.2.1 in terms of v_j^n, v_{j+1}^n, etc.

(b) Determine the dispersion relation (expressed in as simple a form as possible).

(c) Sketch the dispersion relation.

(d) Determine the group velocity as a function of ξ and ω.

▷ *5.3.2. Schrödinger equation.*

(a) Calculate and plot the dispersion relation for the Crank-Nicolson model of $u_t = iu_{xx}$ of Exercise 3.2.1*(f)*.

(b) Calculate the group velocity. How does it compare to the group velocity for the equation $u_t = iu_{xx}$ itself?

▷ *5.3.3. A paradox.* Find the resolution of the following apparent paradox, and be precise in stating where the mistake is. Draw a sketch of an appropriate dispersion relation to explain your answer.

One the one hand, if we solve $u_t = u_x$ by the leap frog formula with $\lambda = 1$, the results will be exact, and in particular, no dispersion will take place.

On the other hand, as discussed above, the dispersion relation on any discrete grid must be periodic, hence nonlinear—and so dispersion must take place after all.

Chapter 6.
Boundary Conditions

O God! I could be bounded in a nutshell,
and count myself a king of infinite space,
were it not that I have bad dreams.

— *W. SHAKESPEARE, Hamlet II, ii (1601)*

The difficulties caused by boundary conditions in scientific computing would be hard to overemphasize. Boundary conditions can easily make the difference between a successful and an unsuccessful computation, or between a fast and a slow one. Yet in many important cases, there is little agreement about what the proper treatment of the boundary should be.

One of the sources of difficulty is that although some numerical boundary conditions come from discretizing the boundary conditions for the continuous problem, other "artificial" or "numerical boundary conditions" do not. The reason is that the number of boundary conditions required by a finite difference formula depends on its stencil, not on the equation being modeled. Thus even a complete mathematical understanding of the initial boundary value problem to be solved—which is often lacking—is in general not enough to ensure a successful choice of numerical boundary conditions. This situation reaches an extreme in the design of what are variously known as "open", "radiation", "absorbing", "non-reflecting", or "far-field" boundary conditions, which are numerical artifacts designed to limit a computational domain in a place where the mathematical problem has no boundary at all.

Despite these remarks, perhaps the most basic and useful advice one can offer concerning numerical boundary conditions is this: before worrying about discretization, make sure to understand the mathematical problem. If the IBVP is ill-posed, no amount of numerical cleverness will lead to a successful computation. This principle may seem too obvious to deserve mention, but it is surprising how often it is forgotten.

[Only portions of this chapter have been written so far. The preceding typewritten pages from Chapter 5, however, include some of the material that belongs here.]

6.2. Scalar hyperbolic equations

[This section is not yet properly written, but some of the essential ideas are summarized below.]

z and κ. The following are standard abbreviations:

$$v_j^n = z^n \kappa^j = e^{i(\xi x + \omega t)}, \qquad z = e^{i\omega k}, \qquad \kappa = e^{i\xi h}. \tag{6.2.1}$$

Thus z is the "temporal amplification factor" and κ is the "spatial amplification factor" for a wave mode $e^{i(\xi x+\omega t)}$. We shall often write x and t, as here, even though the application may involve only their discretizations x_j and t_n.

History. The four papers listed first in the references for this chapter fit the following neat pattern. The 1970 paper by Kreiss is the classic reference on well-posedness of initial boundary value problems, and the 1986 paper by Higdon presents an interpretation of this theory in terms of dispersive wave propagation. The 1972 "GKS" paper by Gustafsson, Kreiss, and Sundström is the classic reference on stability of finite difference models of initial boundary value problems (although there are additional important references by Strang, Osher, and others), and the 1984 paper by Trefethen presents the dispersive waves interpretation for that.

Leftgoing and rightgoing waves. For any real frequency ω (i.e., $|z| = 1$), a three-point finite difference formula typically admits two wave numbers ξ (i.e., κ), and often, one will have $c_g \geq 0$ and the other $c_g \leq 0$. This is true, for example, for the leap frog and Crank-Nicolson models of $u_t = u_x$. We shall label these wave numbers ξ_L and ξ_R, for "leftgoing" and "rightgoing".

Interactions at boundaries and interfaces. If a plane wave hits a boundary or interface, then typically a reflected wave is generated that has the same ω (i.e., z) but different ξ (i.e., κ). The reflected value ξ must also satisfy the finite difference formula for the same ω, so for the simple formulas mentioned above, it will simply be the "other" value, e.g., ξ_R at a left-hand boundary.

Reflection coefficients. Thus at a boundary it makes sense to look for single-frequency solutions of the form

$$v_j^n = z^n(\alpha_L \kappa_L^j + \alpha_R \kappa_R^j) = e^{i\omega t}(\alpha_L e^{i\xi_L x} + \alpha_R e^{i\xi_R x}). \tag{6.2.1}$$

If there is such a solution, then it makes sense to define the **reflection coefficient** $R(\omega)$ by

$$R(\omega) = \frac{\alpha_R}{\alpha_L}. \tag{6.2.3}$$

Stable and unstable boundary conditions. Very roughly, the GKS theory asserts that a left-hand boundary condition is stable if and only if it admits no solutions (6.2.2) with $\alpha_L = 0$. The idea is that such a solution corresponds to spurious energy radiating into the domain from nowhere. Algebraically, one checks for stability by checking whether there are any modes (6.2.2) that satisfy three conditions:

(1) v_j^n satisfies the interior finite difference formula;

(2) v_j^n satisfies the discrete boundary conditions;

(3) v_j^n is a "rightgoing" mode; that is, either
(a) $|z| = |\kappa| = 1$ and $c_g \geq 0$, or
(b) $|z| \geq 1 > |\kappa|$.

Finite vs. infinite reflection coefficients. If $R(\omega) = \infty$ for some ω, the boundary condition must be unstable, because that implies $\alpha_L(\omega) = 0$ above. The converse, however, does not hold. That is, $R(\omega)$ may be finite if $\alpha_R(\omega)$ and $\alpha_L(\omega)$ happen to be zero simultaneously, and this is a situation that comes up fairly often. A boundary instability tends to be more pronounced if the associated reflection coefficient is infinite. However, a pronounced instability is sometimes less dangerous than a weak one, for it is less likely to go undetected.

The effect of dissipation. Adding dissipation, or shifting from centered to one-sided interior or boundary formulas, often makes an unstable model stable. Theorems to this effect can be found in various papers by Goldberg & Tadmor; see the references. However, dissipation is not a panacea, for if a slight amount of dissipation is added to an unstable formula, the resulting formula may be technically stable but still subject to oscillations large enough to be troublesome.

Hyperbolic problems with two boundaries. A general theorem by Kreiss shows that in the case of a finite difference model of a linear hyperbolic system of equations on an interval such as $x \in [0,1]$, the two-boundary problem is stable if and only if each of the two boundary conditions is individually stable.

Hyperbolic problems in corners. Analogously, one might expect that if a hyperbolic system of equations in two space variables x and y is modeled with stable boundary conditions for $x \geq 0$ and for $y \geq 0$, then the same problem would be stable in the quarter-domain $x \geq 0$, $y \geq 0$. However, examples by Osher and Sarason & Smoller show that this is not true in general.

(a) $v_0^{n+1} = v_1^n$ (b) $v_0^{n+1} = v_1^{n+1}$

Figure 6.2.1. Stable and unstable left-hand boundary conditions for the leap frog model of $u_t = u_x$ with $\lambda = 0.9$, $v = 0$ at the right-hand boundary, initial data exact.

EXERCISES

▷ *6.2.1. Interpretation of Figure 6.2.1.* In Figure 6.2.1*(b)*, various waves are evidently propagating back and forth between the two boundaries. In each of the time segments marked *a*, *b*, *c*, and *d*, one particular mode (6.2.1) dominates. What are these four modes? Explain your answer with reference to a sketch of the dispersion relation.

▶ *6.2.2. Experiments with Crank-Nicolson.*

Go back to your program of Exercise 3.2.1, and modify it now to implement CN_x—the Crank-Nicolson model of $u_t = u_x$. This is not a method one would use in practice, since there's no need for an implicit formula in this hyperbolic case, but there's no harm in looking at it as an example.

The computations below should be performed on the interval $[0,1]$ with $h = 0.01$ and $\lambda = 1$. Let $f(x)$ be defined by

$$f(x) = e^{-400(x-1/2)^2},$$

and let the boundary conditions at the endpoints be $v_0^{n+1} = v_{1/h}^{n+1} = 0$, except where otherwise specified.

(a) Write the down the dispersion relation for CN_x model of $u_t = u_x$, and sketch it. In the problems below, you should refer to this sketch repeatedly. Also write down the group velocity formula.

(b) Plot the computed solutions $v(x,t)$ at time $t = 0.25$ for the initial data $v(x,0) =$

$$\textit{(i)}\ f(x), \qquad \textit{(ii)}\ (-1)^j f(x), \qquad \textit{(iii)}\ \mathrm{Re}\{i^j f(x)\},$$

and explain your results. In particular, explain why much more dispersion is apparent in *(iii)* than in *(i)* or *(ii)*.

(c) Plot the computed solutions at time $t = 0.75$ for initial data $v(x,0) = f(x)$ and left-hand boundary conditions

$$\textit{(i)}\ v_0^{n+1} = v_1^{n+1}, \qquad \textit{(ii)}\ v_0^{n+1} = v_1^n,$$

$$\textit{(iii)}\ v_0^{n+1} = v_2^{n+1}, \qquad \textit{(iv)}\ v_0^{n+1} = -v_1^{n+1} + v_2^{n+1} + v_3^{n+1}.$$

To implement *(iv)*, you will have to make a small modification in your tridiagonal solver.

(d) Repeat *(c)* for the initial data $v(x,0) = g(x) = \max\{1 - 10|x - 1/2|, 0\}$.

(e) On the basis of *(c)* and *(d)*, which boundary conditions would you say appear stable? (Remember, stability is what we need for convergence, and for convergence, the error must decay to zero as $k \to 0$. If you are in doubt, try doubling or halving k to see what the effect on the computed solution is.)

(f) Prove that boundary condition *(i)* is stable or unstable (whichever is correct).

(g) Likewise *(ii)*.

(h) Likewise *(iii)*.

(i) Likewise *(iv)*.

6.4. Absorbing boundary conditions

A recurring problem in scientific computing is the design of **artificial boundaries**. How can one limit a domain numerically, to keep the scale of the computation within reasonable bounds, yet still end up with a solution that approximates the correct result for an unbounded domain?* For example, in the calculation of the transonic flow over an airfoil, what boundary conditions are appropriate at an artificial boundary downstream? In an acoustical scattering problem, what boundary conditions are appropriate at a spherical artificial boundary far away from the scattering object?

Artificial boundary conditions designed to achieve this kind of effect go by many names, such as **absorbing**, **nonreflecting**, **open**, **radiation**, **invisible** or **far-field** boundary conditions. Except in special situations, a perfect artificial boundary cannot be designed even in principle. After all, in the exact solution of the problem being modeled, interactions might occur outside the boundary which then propagate back into the computational domain at a later time. In practice, however, artificial boundaries can often do very well.

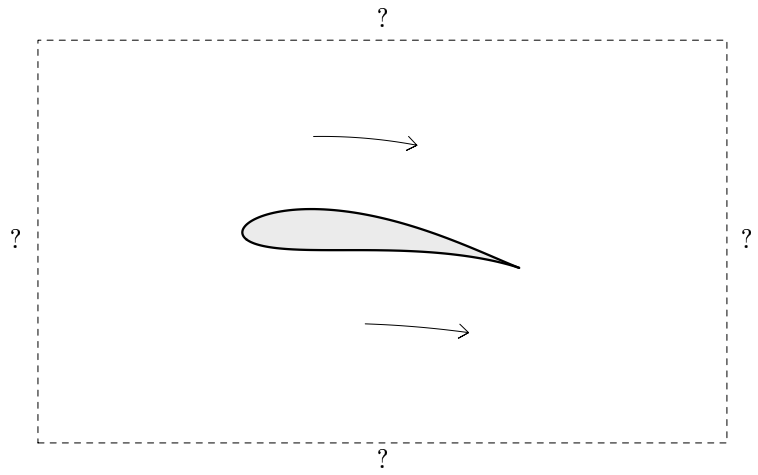


Figure 6.4.1. The problem of artificial boundaries.

There are three general methods of coping with unbounded domains in numerical simulations:

1. Stretched grids; change of variables. By a change of variables, a infinite domain can be mapped into a finite one, which can then be treated by a finite grid. Of course the equation being solved changes in the process. A closely related idea is to stay in the original physical domain, but use a stretched grid that has only finitely many grid

*Experimentalists have the same problem—how can the walls of a wind tunnel be designed to influence the flow as little as possible?

points. Methods of this kind are appealing, but for most problems they are not the best approach.

2. Sponge layers. Another idea is to surround the region of physical interest by a layer of grid points in which the same equations are solved, except with an extra dissipation term added to absorb energy and thereby prevent reflections. This "sponge layer" might have a width of, say, 5 to 50 grid points.

3. One-way equations; matched solutions. A third idea is to devise more sophisticated boundary conditions that allow propagation of energy out of the domain of interest but not into it. In some situations such a boundary condition can be obtained as a discretization of a "one-way equation" of some kind. More generally, one can think of matching the solution in the computational domain to some known outer solution that is valid near infinity, and then design boundary conditions analytically that are consistent with the latter. (See, e.g., papers by H. B. Keller and T. Hagstrom.)

Of these three methods, the first and second have a good deal in common. Both involve padding the computational domain by "wasted" points whose only function is to prevent reflections, and in both cases, the padding must be handled smoothly if the absorption is to be successful. This means that a stretched grid should stretch smoothly, and an artificial dissipation term should be turned on smoothly. Consequently the region of padding must be fairly thick, and hence is potentially expensive, especially in two or three space dimensions. On the other hand these methods are quite general.

The third idea is quite different and more problem-dependent. When appropriate one-way boundary conditions can be devised, their effect may be dramatic. As a rule of thumb, perhaps it is safe to say that effective one-way boundary conditions can usually be found if the boundary is far enough out that the physics in that vicinity is close to linear. Otherwise, some kind of a sponge layer is probably the best idea.

Of course, various combinations of these three ideas have also been considered. See, for example, S. A. Orszag and M. Israeli, *J. Comp. Phys.*, 1981.

The remainder of this section will describe a method for designing one-way boundary conditions for acoustic wave calculations which was developed by Lindman in 1975 (*J. Comp. Phys.*) and by Engquist and Majda in 1977 (*Math. Comp.*) and 1979 (*Comm. Pure & Appl. Math.*). Closely related methods were also devised by Bayliss and Turkel in 1980 (*Comm. Pure & Appl. Math.*).

For the second-order wave equation

$$u_{tt} = u_{xx} + u_{yy}, \tag{6.4.1}$$

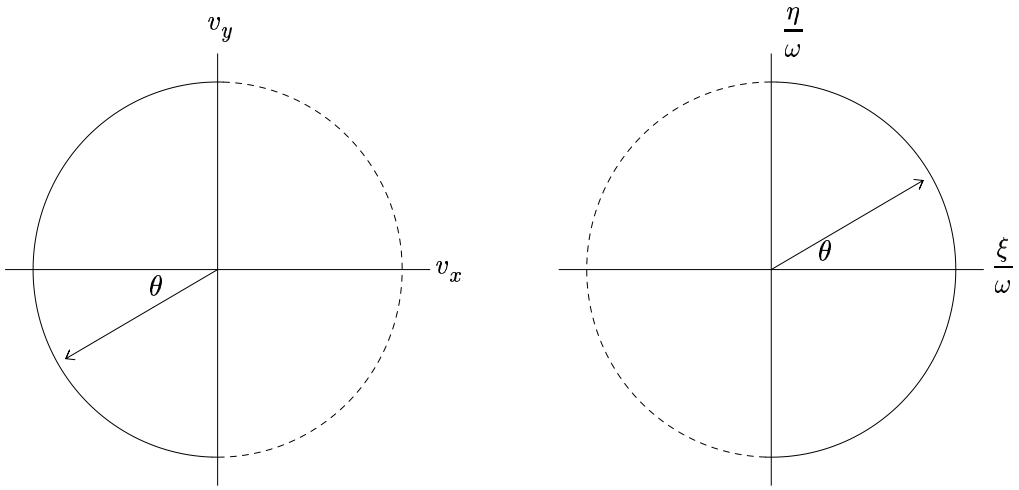
the dispersion relation for wave modes $u(x,y,t) = e^{i(\omega t + \xi x + \eta y)}$ is $\omega^2 = \xi^2 + \eta^2$, or equivalently

$$\boxed{\xi = \pm\omega\sqrt{1-s^2}, \qquad \begin{array}{l}\textit{DISPERSION RELATION}\\ \textit{FOR WAVE EQUATION}\end{array}} \tag{6.4.2}$$

with

$$s = \frac{\eta}{\omega} = \sin\theta \in [-1,1], \qquad \theta \in [-90°, 90°]. \tag{6.4.3}$$

This is the equation of a circle in the $\frac{\xi}{\omega} - \frac{\eta}{\omega}$ plane corresponding to plane waves propagating in all directions. The wave with wave numbers ξ, η has velocity $c = c_g = (-\frac{\xi}{\omega}, -\frac{\eta}{\omega}) = (-\cos\theta, -\sin\theta)$, where θ is the angle counterclockwise from the negative x axis. See Figure 6.4.2.



(a) space *(b)* Fourier space

Figure 6.4.2. Notation for the one-way wave equation.

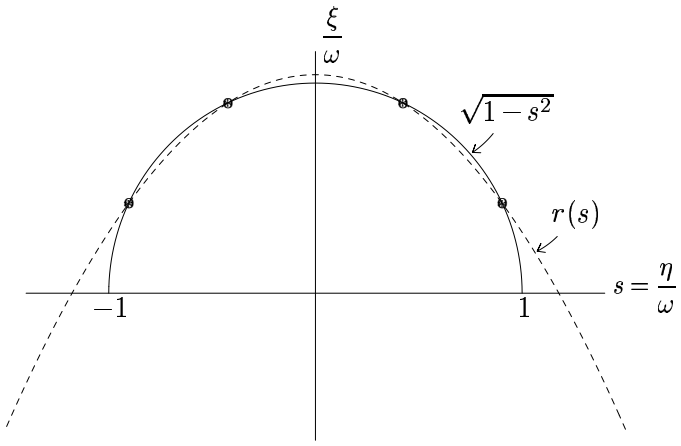


Figure 6.4.3. The approximation problem: $r(s) \approx \sqrt{1-s^2}$.

By taking the plus or minus sign in (6.4.2) only, we can restrict attention to leftgoing ($|\theta| \le 90°$) or rightgoing ($|\theta| \ge 90°$) waves, respectively. For definiteness we shall choose the former course, which is appropriate to a left-hand boundary, and write

$$\boxed{\xi = +\omega\sqrt{1-s^2}. \qquad \begin{array}{c}\textit{DISPERSION RELATION}\\ \textit{FOR IDEAL O.W.W.E.}\end{array}} \tag{6.4.4}$$

See Figure 6.4.2 again.

Because of the square root, (6.4.4) is not the dispersion relation of any partial differential equation, but of a pseudodifferential equation. The idea behind practical one-way wave equations is to replace the square root by a rational function $r(s)$ of type (m,n) for some m

and n, that is, the ratio of a polynomial p_m of degree m and a polynomial q_n of degree n,

$$r(s) = \frac{p_m(s)}{q_n(s)}.$$

Then (6.4.4) becomes

$$\boxed{\xi = \omega r(s) \qquad \begin{array}{c}\textit{DISPERSION RELATION FOR}\\ \textit{APPROXIMATE O.W.W.E.}\end{array}} \tag{6.4.5}$$

for the same range (6.4.4) of s and θ. See Figure 6.4.3. By clearing denominators, we can transform (6.4.5) into a polynomial of degree $\max\{m,n+1\}$ in ω, ξ, and η, and this is the dispersion relation of a true differential equation.

The standard choice of the approximation $r(s) = r_{mn}(s)$ is the **Padé approximant** to $\sqrt{1-s^2}$, which is defined as that rational function of the prescribed type whose Taylor series at $s=0$ matches the Taylor series of $\sqrt{1-s^2}$ to as high an order as possible. Assuming that m and n are even, the order will be

$$r_{mn}(s) - \sqrt{1-s^2} = O(s^{m+n+2}). \tag{6.4.6}$$

For example, the type $(0,0)$ Padé approximant to $\sqrt{1-s^2}$ is $r(s)=1$. Taking this choice in (6.4.5) gives

$$\xi = \omega, \qquad \text{i.e.,} \qquad u_x = u_t. \tag{6.4.7}$$

This simple advection equation is thus a suitable low-order absorbing boundary condition for the wave equation at a left-hand boundary. For a numerical simulation, one would discretize it by a one-sided finite difference formula. In computations (6.4.7) is much better than a Neumann or Dirichlet boundary condition at absorbing outgoing waves.

At the next order, the type $(2,0)$ Padé approximant to $\sqrt{1-s^2}$ is $r(s) = 1-\frac{1}{2}s^2$. Taking this choice in (6.4.5) gives

$$\xi = \omega(1-\tfrac{1}{2}\eta^2/\omega^2),$$

that is,

$$\xi\omega = \omega^2 - \tfrac{1}{2}\eta^2, \qquad \text{i.e.,} \qquad u_{xt} = u_{tt} - \tfrac{1}{2}u_{yy}. \tag{6.4.8}$$

This boundary condition absorbs waves more effectively than (6.4.7), and is the most commonly used absorbing boundary condition of the Engquist-Majda type.

As a higher order example, consider the type $(2,2)$ Padé approximant to $\sqrt{1-s^2}$,

$$r(s) = \frac{1-\frac{3}{4}s^2}{1-\frac{1}{4}s^2}.$$

Then (6.4.5) becomes

$$\xi\left(1-\tfrac{1}{4}\frac{\eta^2}{\omega^2}\right) = \omega\left(1-\tfrac{3}{4}\frac{\eta^2}{\omega^2}\right)$$

or

$$\xi\omega^2 - \tfrac{1}{4}\xi\eta^2 = \omega^3 - \tfrac{3}{4}\omega\eta^2, \qquad \text{i.e.,} \qquad u_{xtt} - \tfrac{1}{4}u_{xyy} = u_{ttt} - \tfrac{3}{4}u_{tyy}. \tag{6.4.9}$$

This third-order boundary condition is harder to implement than (6.4.8), but provides excellent absorption of outgoing waves.

Why didn't we look at the Padé approximation of type $(4,0)$? There are two answers. First, although that approximation is of the the same order of accuracy as the type $(2,2)$ approximation, it leads to a boundary condition of order 4 instead of 3, which is even harder to implement. Second, that boundary condition turns out to be ill-posed and is thus useless in any case (Exercise 6.4.1). In general, it can be shown that the Engquist-Majda boundary conditions are well-posed if and only if $m = n$ or $n+2$—that is, for precisely those approximations $r(s)$ taken from the main diagonal and first superdiagonal in the "Pade table" (L. N. Trefethen & L. Halpern, *Math. Comp. 47* (1986), pp. 421–435).

Figure 6.4.4, taken from Engquist and Majda, illustrates the effectiveness of their absorbing boundary conditions.* The upper-left plot (1A) shows the initial data: a quarter-circular wave radiating out from the upper-right corner. If the boundaries have Neumann boundary conditions, the wave reflects with reflection coefficient 1 at the left-hand boundary (1B) and again at the right-hand boundary (1C). Dirichlet boundary conditions are equally bad, except that the reflection coefficient becomes -1 (1D). Figure 1E shows the type $(0,0)$ absorbing boundary condition (6.4.7), and the reflected wave amplitude immediately cuts to about 5%. In Figure 1F, with the type $(2,0)$ boundary condition (6.4.8), the amplitude is less than 1%.

EXERCISES

▷ *6.4.1. Ill-posed absorbing boundary conditions.*

(a) What is the type $(4,0)$ Padé approximant $r(s)$ to $\sqrt{1-s^2}$?

(b) Find the coefficients of the corresponding absorbing boundary condition for a left-hand boundary.

(c) Show that this boundary condition is ill-posed by finding a mode $u(x,y,t) = e^{i(\omega t+\xi x+\eta y)}$ that satisfies both the wave equation and the boundary condition with $\eta \in \mathbb{R}$, $\mathrm{Im}\,\xi > 0$, $\mathrm{Im}\,\omega < 0$. Explain why the existence of such a mode implies that the initial boundary value problem is ill-posed. In a practical computation, would you expect the ill-posed mode to show up as mild or as explosive?

▷ *6.4.2. Absorbing boundary conditions for oblique incidence.* Sometimes it is advantageous to tune an absorbing boundary condition to absorb not waves that are normally incident at the boundary, but waves at some specified angle (or angles). Use this idea to derive absorbing boundary conditions that are exact for plane waves traveling at 45° in the southwest direction as they hit a left-hand boundary:

(a) Type $(0,0)$, *(b)* Type $(2,0)$.

Don't worry about rigor or about well-posedness. This problem concerns partial differential equations only, not finite difference approximations.

*This figure will appear in the published version of this book only with permission.

Figure 6.4.4. Absorbing boundary conditions for $u_{tt} = u_{xx} + u_{yy}$ (from Engquist & Majda).

Chapter 7.
Fourier spectral methods

Think globally. Act locally.
— BUMPER STICKER (1985)

Finite difference methods, like finite element methods, are based on local representations of functions—usually by low-order polynomials. In contrast, spectral methods make use of global representations, usually by high-order polynomials or Fourier series. Under fortunate circumstances the result is a degree of accuracy that local methods cannot match. For large-scale computations, especially in several space dimensions, this higher accuracy may be most important for permitting a coarser mesh, hence a smaller number of data values to store and operate upon. It also leads to discrete models with little or no artificial dissipation, a particularly valuable feature in high Reynolds number fluid flow calculations, where the small amount of physical dissipation may be easily overwhelmed by any dissipation of the numerical kind. Spectral methods have achieved dramatic successes in this area.

Some of the ideas behind spectral methods have been introduced several times into numerical analysis. One early proponent was Cornelius Lanczos, in the 1950s, who showed the power of Fourier series and Chebyshev polynomials in a variety of problems where they had not been used before. The emphasis was on ordinary differential equations. Lanczos's work has been carried on, especially in Great Britain, by a number of colleagues such as C. W. Clenshaw.

More recently, spectral methods were introduced again by Kreiss and Oliger, Orszag, and others in the 1970s for the purpose of solving the partial differential equations of fluid mechanics. Increasingly they are becoming viewed within some fields as an equal competitor to the better established finite difference and finite element approaches. At present, however, they are less well understood.

Spectral methods fall into various categories, and one distinction often made is between "Galerkin," "tau," and "collocation" (or "pseudospectral") spectral methods. In a word, the first two work with the coefficients of a global expansion, and the latter with its values at points. The discussion in this book is entirely confined to collocation methods, which are probably used the most often, chiefly because they offer the simplest treatment of nonlinear terms.

Spectral methods are affected far more than finite difference methods by the presence of boundaries, which tend to introduce stability problems that are ill-understood and sometimes highly restrictive as regards time step. Indeed, difficulties with boundaries, direct and indirect, are probably the primary reason why spectral methods have not replaced their lower-accuracy competition

in most applications. Chapter 8 considers spectral methods with boundaries, but the present chapter assumes that there are none. This means that the spatial domain is either infinite—a theoretical device, not applicable in practice—or periodic. In those cases where the physical problem naturally inhabits a periodic domain, spectral methods may be strikingly successful. Conspicuous examples are the global circulation models used by meteorologists. Limited-area meteorological codes, since they require boundaries, are often based on finite difference formulas, but as of this writing almost all of the global circulation codes in use— which model flow in the atmosphere of the entire spherical earth— are spectral.

7.1. An example

Spectral methods have been most dramatically successful in problems with periodic geometries. In this section we present two examples of this kind that involve elastic wave propagation. Both are taken from B. Fornberg, "The pseudospectral method: comparisons with finite differences for the elastic wave equation," *Geophysics 52* (1987), 483–501. Details and additional examples can be found in that paper.*

Elastic waves are waves in an elastic medium such as an iron bar, a building, or the earth, and they come in two varieties: "P" waves (pressure or primary), characterized by longitudinal vibrations, and "S" waves (shear or secondary), characterized by transverse vibrations. The partial differential equations of elasticity can be written in various forms, such as a system of two second-order equations involving displacements. For his numerical simulations, Fornberg chose a formulation as a system of five first-order equations.

Figures 7.1.1 and 7.1.2 show the results of calculations for two physical problems. In the first, a P wave propagates uninterruptedly through a periodic, uniform medium. In the second, an oblique P wave oriented at 45° hits a horizontal interface at which the wave speeds abruptly cut in half. The result is reflected and transmitted P and S waves. For this latter example, the actual computation was performed on a domain of twice the size shown — which is a hint of the trouble one may be willing to go to, with spectral methods, to avoid coping explicitly with boundaries.

The figures show that spectral methods may sometimes decisively outperform second-order and fourth-order finite difference methods. In particular, spectral methods are *nondispersive,* and in a wave calculation, that property can be of great importance. In these examples the accuracy achieved by the spectral calculation on a 64×64 grid is not matched by fourth-order finite differences on a 128×128 grid, or by second-order finite differences on a 256×256 grid. The corresponding differences in work and storage are enormous.

Fornberg picked his examples carefully; spectral methods do not always perform so convincingly. Nevertheless, sometimes they are extremely impressive. Although the reasons are not fully understood, their advantages often hold not just for problems involving smooth functions, but even in the presence of discontinuities.

*The figures in this section will appear in the published version of this book only with permission.

(a) Schematic initial and end states

(b) Computational results

Figure 7.1.1. Spectral and finite difference simulations of a P wave propagating through a uniform medium (from Fornberg, 1987).

(a) Schematic initial and end states

(b) Computational results

Figure 7.1.2. Spectral and finite difference simulations of a P wave incident obliquely upon an interface (from Fornberg, 1987).

7.2. Unbounded grids

We shall begin our study of spectral methods by looking at an infinite, unbounded domain. Of course, real computations are not carried out on infinite domains, but this simplified problem contains many of the essential features of more practical spectral methods.

Consider again ℓ_h^2, the set of square-integrable functions $v = \{v_j\}$ on the unbounded regular grid $h\mathbb{Z}$. As mentioned already in §3.3, the foundation of spectral methods is the **spectral differentiation operator** $D : \ell_h^2 \to \ell_h^2$, which can be described in several equivalent ways. One is by means of the Fourier transform:

SPECTRAL DIFFERENTIATION BY THE SEMIDISCRETE FOURIER TRANS.

(1) *Compute* $\hat{v}(\xi)$;
(2) *Multiply by* $i\xi$;
(3) *Inverse transform:*

$$Dv = \mathcal{F}_h^{-1}(i\xi \mathcal{F}_h(v)). \tag{7.2.1}$$

Another is in terms of band-limited interpolation. As described in §2.3, one can think of the interpolant as a Fourier integral of band-limited complex exponentials or, equivalently, as an infinite series of sinc functions:

SPECTRAL DIFFERENTIATION BY SINC FUNCTION INTERPOLATION.

(1) *Interpolate* v *by a sum of sinc functions* $q(x) = \sum_{k=-\infty}^{\infty} v_k S_h(x - x_k)$;
(2) *Differentiate the interpolant at the grid points* x_j*:*

$$(Dv)_j = q'(x_j) \tag{7.2.2}$$

Recall that the sinc function $S_h(x)$, defined by

$$S_h(x) = \frac{\sin(\pi x/h)}{\pi x/h}, \tag{7.2.3}$$

is the unique function in L^2 that interpolates the discrete delta function e_j,

$$e_j = \begin{cases} 1 & j = 0, \\ 0 & j \neq 0, \end{cases} \tag{7.2.4}$$

and that moreover is band-limited in the sense that its Fourier transform has compact support contained in $[-\pi/h, \pi/h]$.

For higher order spectral differentiation, we multiply $\mathcal{F}_h(v)$ by higher powers of $i\xi$, or equivalently, differentiate $q(x)$ more than once.

Why are these two descriptions equivalent? The fundamental reason is that $S_h(x)$ is not just any interpolant to the delta function e, but the *band-limited* interpolant. For a precise argument, note that both processes are obviously linear, and it is not hard to see that both are shift-invariant in the sense that $D(K^m v) = K^m Dv$ for any m. (The shift operator K was defined in (3.2.8).) Since an arbitrary function $v \in \ell_h^2$ can be written as a convolution sum $v_j = \sum_{k=-\infty}^{\infty} v_k e_{j-k}$, it follows that it is enough to prove that the two processes give the same result when applied to the particular function e. That equivalence results from the fact that the Fourier transform of e is the constant function h, whose inverse Fourier transform is in turn precisely $S_h(x)$.

Since spectral differentiation constitutes a linear operation on ℓ_h^2, it can also be viewed as multiplication by a biinfinite Toeplitz matrix:

$$D = \frac{1}{h} \begin{pmatrix} & & \ddots & & & & & & \\ \cdots & -\frac{1}{3} & \frac{1}{2} & -1 & 0 & 1 & -\frac{1}{2} & \frac{1}{3} & \cdots \\ & & & & & & \ddots & & \end{pmatrix}. \tag{7.2.5}$$

As discussed in §3.3, this matrix is the limit of banded Toeplitz matrices corresponding to finite difference differentiation operators of increasing orders of accuracy; see Table 3.3.1 on p. 131. (In this chapter we drop the subscript on the symbol D_∞ used in §3.3.) We shall be careless in this text about the distinction between the operator D and the matrix D that represents it. Another way to express the same thing is to write

$$Dv = a * v, \qquad a = \frac{1}{h^2}(\cdots \tfrac{1}{3} \; -\tfrac{1}{2} \quad 1 \quad 0 \; -1 \quad \tfrac{1}{2} \; -\tfrac{1}{3} \cdots) \tag{7.2.6}$$

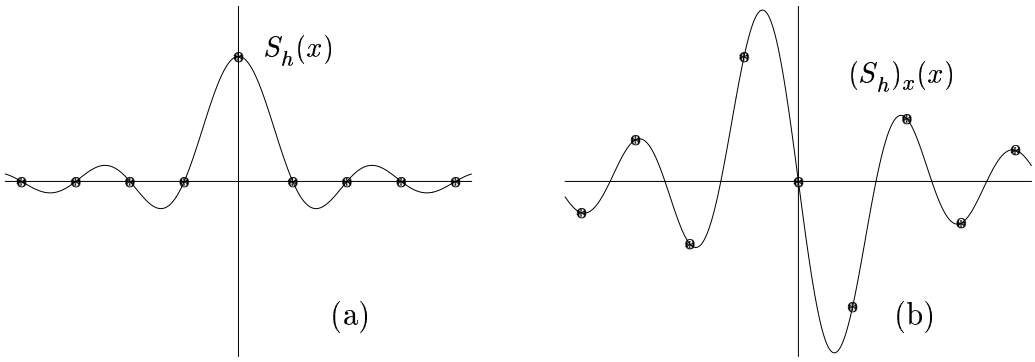


Figure 7.2.1. The sinc function $S_h(x)$ and its derivative $(S_h)_x(x)$.

as in (3.3.16).

The coefficients of (7.2.5)–(7.2.6) can be derived from either the Fourier transform or the sinc function interpretation. Let us begin with the latter. The sinc function has derivative

$$(S_h)_x(x) = \frac{\cos(\pi x/h)}{x} - \frac{\sin(\pi x/h)}{\pi x^2/h}, \tag{7.2.7}$$

with values

$$(S_h)_x(x_j) = \begin{cases} 0 & \text{if } j = 0, \\ \frac{(-1)^j}{jh} & \text{if } j \neq 0 \end{cases} \tag{7.2.8}$$

at the grid points. See Figure 7.2.1. This is precisely the "zeroth column" of D, since that column must by definition contain the values on the grid of the spectral derivative of the delta function.* The other columns, corresponding to delta functions centered at other points x_j, contain the same entries appropriately shifted.

Now let us rederive (7.2.5)–(7.2.6) by means of the Fourier transform. If $Dv = a * v$, then $\widehat{Dv}(\xi) = \hat{a}(\xi)\hat{v}(\xi)$, and for spectral differentiation we want $\hat{a}(\xi) = i\xi$. Therefore by the inverse semidiscrete Fourier transform (2.2.7),

$$a_j = \frac{1}{2\pi} \int_{-\pi/h}^{\pi/h} i\xi e^{i\xi jh} d\xi.$$

For $j = 0$ the integral is 0, giving $a_0 = 0$, while for $j \neq 0$, integration by parts

*Think about this. Make sure you understand why (7.2.8) represents a column rather than a row of D.

yields

$$a_j = \frac{1}{2\pi} i\xi \, \frac{e^{i\xi jh}}{ijh}\bigg|_{-\pi/h}^{\pi/h} - \frac{1}{2\pi}\int_{-\pi/h}^{\pi/h} \frac{ie^{i\xi jh}}{ijh} d\xi.$$

The integral here is again zero, since the integrand is a periodic exponential, and what remains is

$$\begin{aligned} a_j &= \frac{1}{2\pi jh}\left(\frac{\pi}{h}e^{i\pi j} - (-\frac{\pi}{h})e^{-i\pi j}\right) \\ &= \frac{1}{2jh^2}(e^{i\pi j} + e^{-i\pi j}) = \frac{(-1)^j}{jh^2}, \end{aligned} \tag{7.2.9}$$

as in (7.2.6).

The entries of D are suggestive of the Taylor expansion of $\log(1+x)$, and this is not a coincidence. In the notation of the spatial shift operator K of (3.2.8), D can be written

$$D = \frac{1}{h}(K - \tfrac{1}{2}K^2 + \tfrac{1}{3}K^3 - \cdots) - \frac{1}{h}(K^{-1} - \tfrac{1}{2}K^{-2} + \tfrac{1}{3}K^{-3} - \cdots),$$

which corresponds formally to

$$\begin{aligned} D &= \frac{1}{h}\log(1+K) - \frac{1}{h}\log(1+K^{-1}) \\ &= \frac{1}{h}\log\left(\frac{1+K}{1+K^{-1}}\right) = \frac{1}{h}\log K. \end{aligned} \tag{7.2.10}$$

Therefore formally, $e^{hD} = K$, and this makes sense: by integrating the derivative over a distance h, one gets a shift. See the proof of Theorem 1.2.

If $v_j = e^{i\xi jh}$ for some $\xi \in [-\pi/h, \pi/h]$, then $Dv = i\xi v$. Therefore $i\xi$ is an eigenvalue of the operator D.* On the other hand, if v has the same form with $\xi \notin [-\pi/h, \pi/h]$, then ξ will be indistinguishable on the grid from some alias wave number $\xi' \in [-\pi/h, \pi/h]$ with $\xi' = \xi + 2\pi\nu/h$ for some integer ν, and the result will be $Dv = i\xi' v$. In other words in Fourier space, the spatial differentiation operator becomes multiplication by a periodic function, thanks to the discrete grid, and in this sense is only an approximation to the exact differentiation operator for continuous functions. Figure 7.2.2 shows the situation graphically. For band-limited data, however, the spectral differentiation operator is exact, in contrast to finite difference differentiation operators, which are exact only in the limit $\xi \to 0$ (dashed line in the Figure).

*Actually, this is not quite true: by definition, an eigenvector must belong to ℓ^2, and $e^{i\xi jh}$ does not. Strictly speaking, $i\xi$ is in the spectrum of D but is not an eigenvalue. However, this technicality is unimportant for our purposes, and will be ignored in the present draft of this book.

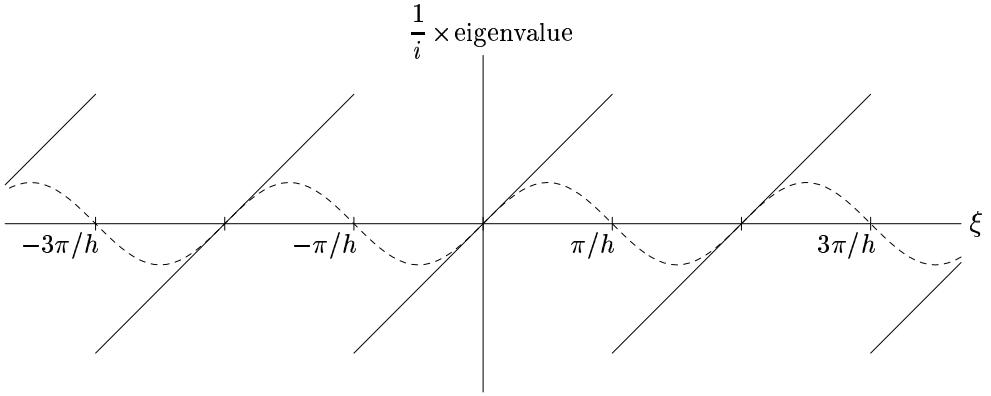
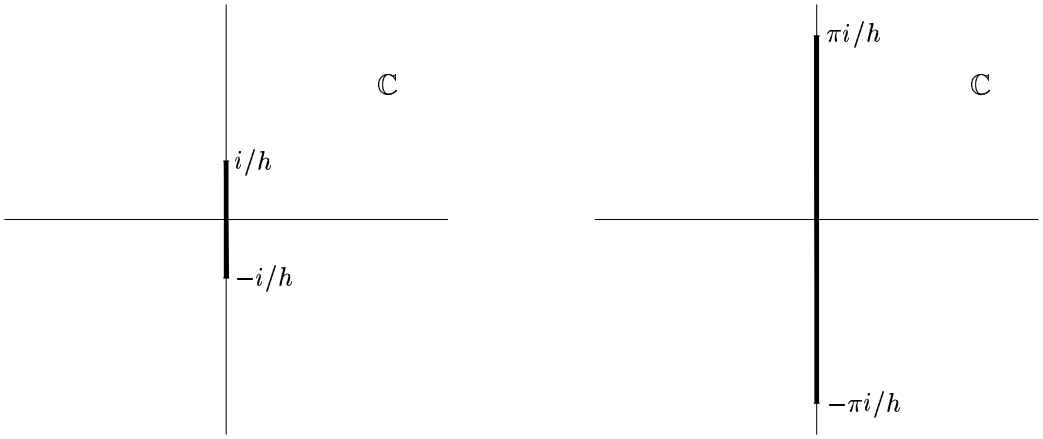


Figure 7.2.2. Eigenvalue of D (divided by i) corresponding to the eigenfunction $e^{i\xi x}$, as a function of ξ. The dashed line shows corresponding eigenvalues for the finite difference operator D_2.



(a) D_2 (finite difference) *(b)* D (spectral)

Figure 7.2.3. Eigenvalues of finite difference and spectral first-order differentiation matrices, as subsets of the complex plane.

Figure 7.2.3 compares the spectrum of D to that of the second-order finite difference operator $D_2 = \delta_0$ of §3.3.

D is a bounded linear operator on ℓ_h^2, with norm

$$\|D\| = \max_{\xi \in [-\pi/h, \pi/h]} |i\xi| = \frac{\pi}{h} \tag{7.2.11}$$

(see §§2.5,3.5). Notice that the norm increases to infinity as the mesh is refined.

This is inevitable, as the differentiation operator for continuous functions is unbounded.

So far we have described the spectral approximation to the first derivative operator $\partial/\partial x$, but it is an easy matter to approximate higher derivatives too. For the second derivative, the coefficients turn out to be

$$D^2 = \frac{1}{h^2}\begin{pmatrix} & & \ddots & & & & & & \\ \cdots & \frac{2}{9} & -\frac{2}{4} & \frac{2}{1} & -\frac{\pi^2}{3} & \frac{2}{1} & -\frac{2}{4} & \frac{2}{9} & \cdots \\ & & & & & & \ddots & & \end{pmatrix}. \tag{7.2.12}$$

To derive the entries of this matrix, one can simply square D; this leads to infinite series to be summed. One can differentiate (7.2.7) a second time. Or one can compute the inverse Fourier transform of $\hat{a}(\xi) = -\xi^2$, as follows. Two integrations by parts are involved, and terms that are zero have been dropped. For $j \neq 0$,

$$\begin{aligned} a_j &= \frac{1}{2\pi}\int_{-\pi/h}^{\pi/h} -\xi^2 e^{i\xi jh}\, d\xi \\ &= \frac{1}{2\pi}\int_{-\pi/h}^{\pi/h} 2\xi \frac{e^{i\xi jh}}{ijh}\, d\xi \\ &= \frac{\xi}{\pi}\,\frac{e^{i\xi jh}}{(ijh)^2}\bigg|_{-\pi/h}^{\pi/h} = -\frac{e^{ij\pi} + e^{-ij\pi}}{j^2h^3} = \frac{2(-1)^{j+1}}{j^2h^3}. \end{aligned} \tag{7.2.13}$$

For $j = 0$ the integral is simply

$$a_0 = -\frac{1}{2\pi}\left(\frac{2\pi^3}{3h^3}\right) = -\frac{\pi^2}{3h^3}.$$

The effect of D^2 on a function $v_j = e^{i\xi jh}$ is to multiply it by the square of the factor associated with D, as illustrated in Figure 7.2.4. Again one has a periodic multiplier that is exactly correct for $\xi \in [-\pi/h, \pi/h]$. The dashed line shows the analogous curve for the standard centered three-point finite

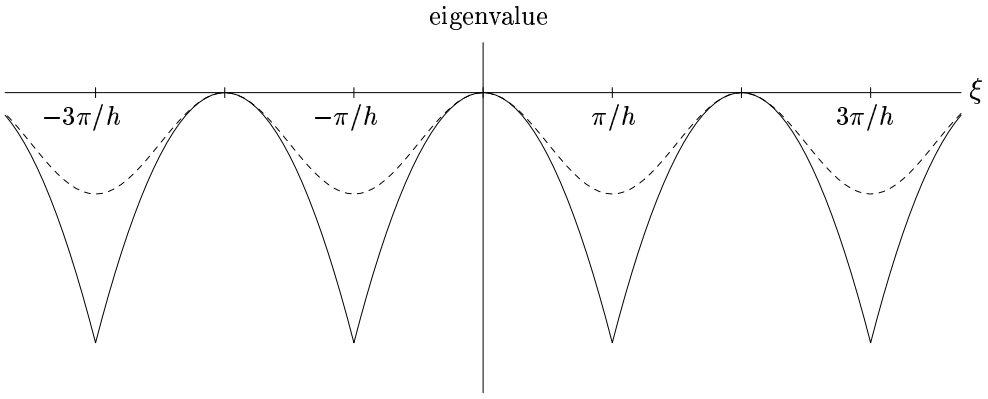
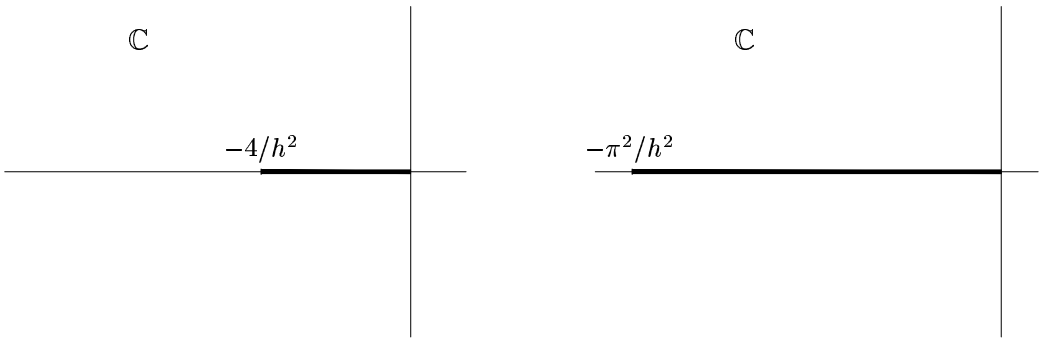


Figure 7.2.4. Eigenvalue of D^2 corresponding to the eigenfunction $e^{i\xi x}$, as a function of ξ. The dashed line shows corresponding eigenvalues for the finite difference operator $\delta_\times$.



(a) $\delta_\times$ (finite difference) *(b)* D^2 (spectral)

Figure 7.2.5. Eigenvalues of finite difference and spectral second-order differentiation matrices, as subsets of the complex plane.

difference operator $\delta_\times$ of §3.2. The spectrum of D^2 must be real, since D is symmetric; it is the interval $[-\pi^2/h^2, 0]$.

The developments of this section are summarized, and generalized to the mth-order case, in the following theorem:

SPECTRAL DIFFERENTIATION ON AN UNBOUNDED REGULAR GRID

Theorem 7.1. *The mth-order spectral differentiation operator D^m is a bounded linear operator on ℓ_h^2 with norm*

$$\|D^m\| = (\frac{\pi}{h})^m. \tag{7.2.14}$$

If m is odd, D^m has the imaginary spectrum $[-i(\pi/h)^m, i(\pi/h)^m]$ and can be represented by an infinite skew-symmetric Toeplitz matrix with entries

$$a_0 = 0, \qquad a_j = h^{-m}(\quad ? \quad) \qquad \textit{for} \quad j \neq 0. \tag{7.2.15}$$

If m is even, D^m has the real spectrum $(-1)^{m/2} \times [0, (\pi/h)^m]$ and can be represented by an infinite symmetric Toeplitz matrix with entries

$$a_0 = \quad ? \quad , \qquad a_j = h^{-m}(\quad ? \quad) \qquad \textit{for} \quad j \neq 0. \tag{7.2.16}$$

The purpose of all of these spectral differentiation matrices is to solve partial differential equations. In a spectral collocation computation this is done in the most straightforward way possible: one discretizes the continuous problem as usual and integrates forward in time by a discrete formula, usually by finite differences.* Spatial derivatives are approximated by the matrix D. This same prescription holds regardless of whether the partial differential equation has variable coefficients or nonlinear terms. For example, to solve $u_t = a(x)u_x$ by spectral collocation, one approximates $a(x)u_x$ at each time step by $a(x_j)Dv$. For $u_t = (u^2)_x$, one uses $D(v^2)$, where v^2 denotes the pointwise square $(v^2)_i = (v_i)^2$. (Alternative discretizations may also be used for better stability properties; see....) This is in contrast to Galerkin or tau spectral methods, in which one adjusts the coefficients in the Fourier expansion of v to satisfy the partial differential equation globally.

*Spectral approximations with respect to time can also sometimes be used; see H. Tal-Ezer, "Spectral methods in time for hyperbolic equations," *SIAM J. Numer. Anal. 23* (1986), pp. 11–26.

EXERCISES

▷ *7.2.1. Coefficients of D^3.* Determine the matrix coefficients of the third-order spectral differentiation matrix D^3. Compare your result with the coefficients of Table 3.3.1.

▷ *7.2.2.*

(a) Compute the integral $\int_{-\infty}^{\infty} S_h(x)dx$ of the sinc function (7.2.3). One could do this by complex contour integration, but instead, be cleverer than that and find the answer by considering the Fourier transform. The argument is quite easy; be sure to state it precisely.

(b) By considering the trapezoid rule for integration (Exercise 2.2.4), explain why the answer above had to come out as it did. (*Hint:* what is the integral of a constant function?)

7.3. Periodic grids

To be implemented in practice, a spectral method requires a bounded domain. In this section we consider the case of a periodic domain—or equivalently, an unbounded domain on which we permit only functions with a fixed periodicity. The underlying mathematics of discrete Fourier transforms was described in §2.4. The next chapter will deal with more general bounded problems.

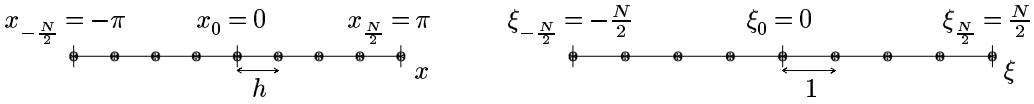


Figure 7.3.1. Space and wave number domains for the discrete Fourier transform.

To repeat some of the material of §2.4, our fundamental spatial domain will now be $[-\pi,\pi)$, as illustrated in Figure 7.3.1. Let N be a positive even integer, set

$$h = \frac{2\pi}{N} \qquad (N \text{ even}), \tag{7.3.1}$$

and define $x_j = jh$ for any j. The grid points in the fundamental domain are

$$x_{-N/2} = -\pi, \ldots, \quad x_0 = 0, \ldots, \quad x_{N/2-1} = \pi - h,$$

and the "invaluable identity" is this:

$$\frac{N}{2} = \frac{\pi}{h}. \tag{7.3.2}$$

The 2-norm $\|\cdot\|$ and space ℓ_N^2 were defined in §2.4, as was the **discrete Fourier transform,**

$$\hat{v}_\xi = (\mathcal{F}_N v)_\xi = h \sum_{j=-N/2}^{N/2-1} e^{-i\xi jh} v_j, \tag{7.3.3}$$

the **inverse discrete Fourier transform,**

$$v_j = (\mathcal{F}_N^{-1} \hat{v})_j = \frac{1}{2\pi} \sum_{\xi=-N/2}^{N/2-1} e^{i\xi jh} \hat{v}_\xi, \tag{7.3.4}$$

and the **discrete convolution,**

$$(v * w)_k = h \sum_{j=-N/2}^{N/2-1} v_{k-j} w_j. \tag{7.3.5}$$

Recall that since the spatial domain is periodic, the set of wave numbers ξ is discrete—namely the set of integers $\mathbb{Z}$—and this is why we have chosen to use ξ itself as a subscript in (7.3.3) and (7.3.4). We take $\xi = -N/2, \ldots, N/2-1$ as our fundamental wave number domain. The properties of the discrete Fourier transform were summarized in Theorems 2.9 and 2.10; recall in particular the convolution formula

$$\widehat{(v*w)}_\xi = \hat{v}_\xi \hat{w}_\xi. \tag{7.3.6}$$

As was described in §2.4, the discrete Fourier transform can be computed with great efficiency by the fast Fourier transform (FFT) algorithm, for which a program was given on p. 104. The discovery of the FFT in 1965 was an impetus to the development of spectral methods for partial differential equations in the 1970's. Curiously, however, practical implementations of spectral methods do not always make use of the FFT, but instead sometimes perform an explicit matrix multiplication. The reason is that in large-scale computations, which typically involve two or three space dimensions, the grid in each dimension may have as few as 32 or 64 points, or even fewer in so-called "spectral element" computations, and these numbers are low enough that the costs of an FFT and of a matrix multiplication may be roughly comparable.

Now we are ready to investigate $D_N : \ell_N^2 \to \ell_N^2$, the spectral differentiation operator for N-periodic functions. As usual, D_N can be described in various ways. One is by means of the discrete Fourier transform:

SPECTRAL DIFFERENTIATION BY THE DISCRETE FOURIER TRANSFORM.

(1) *Compute* $\hat{v}_\xi$*;*

(2) *Multiply by* $i\xi$*, except that* $\hat{v}_{-N/2}$ *is multiplied by* 0*,*

(3) *Inverse transform:*

$$D_N v = \mathcal{F}_N^{-1}(\,0 \text{ for } \xi = -N/2; \quad i\xi\hat{v}_\xi \text{ otherwise}\,). \tag{7.3.8}$$

The special treatment of the value $\hat{v}_{-N/2}$ is required to maintain symmetry, and appears only in spectral differentiation of *odd* order.

Another is in terms of interpolation by a finite series of complex exponentials or, equivalently, periodic sinc functions:

PERIODIC SPECTRAL DIFFERENTIATION BY SINC INTERPOLATION.

(1) *Interpolate* v *by a sum of periodic sinc functions*

$$q(x) = \sum_{k=-N/2}^{N/2-1} v_k S_N(x - x_k);$$

(2) *Differentiate the interpolant at the grid points* x_j*:*

$$(D_N v)_j = q'(x_j). \tag{7.3.9}$$

In the second description we have made use of the **periodic sinc function** on the N-point grid,

$$S_N(x) = \frac{\sin \frac{\pi x}{h}}{\frac{2\pi}{h} \tan \frac{x}{2}}, \tag{7.3.10}$$

which is the unique 2π-periodic function in L^2 that interpolates the discrete delta function e_j on the grid,

$$e_j = \begin{cases} 1 & j = 0, \\ 0 & j = -N/2, \ldots, -1, 1, \ldots, N/2 - 1, \end{cases} \tag{7.3.11}$$

and which is band-limited in the sense that its Fourier transform has compact support contained in $[-\pi/h, \pi/h]$ (and furthermore satisfies $\hat{v}_{-N/2} = \hat{v}_{N/2}$). Compare (7.2.3).

For higher order spectral differentiation on the periodic grid, we multiply $\hat{v}_\xi$ by higher powers of $i\xi$, or equivalently, differentiate $q(x)$ more than once. If the order of differentiation is odd, $\hat{v}_{-N/2}$ is multiplied by the special value 0 to maintain symmetry.

As in the last section, the spectral differentiation process can be viewed as multiplication by a skew-symmetric Toeplitz matrix D_N (compare (7.2.5)):

$$D_N = \begin{pmatrix} 0 & \frac{1}{2}\cot\frac{h}{2} & -\frac{1}{2}\cot\frac{2h}{2} & & & & -\frac{1}{2}\cot\frac{h}{2} \\ & \ddots & & & & & \\ & \frac{1}{2}\cot\frac{2h}{2} & -\frac{1}{2}\cot\frac{h}{2} & 0 & \frac{1}{2}\cot\frac{h}{2} & -\frac{1}{2}\cot\frac{2h}{2} & \\ & & & & \ddots & & \\ \frac{1}{2}\cot\frac{h}{2} & & & & \frac{1}{2}\cot\frac{2h}{2} & -\frac{1}{2}\cot\frac{h}{2} & 0 \end{pmatrix}. \tag{7.3.12}$$

In contrast to the matrix D of (7.2.5), D_N is finite: it applies to vectors $(v_{-N/2},\dots,v_{N/2-1})$, and has dimension $N \times N$. D_N is not only a Toeplitz matrix, but is in fact **circulant**. This means that its entries $(D_N)_{ij}$ "wrap around," depending not merely on $i-j$ but on $(i-j) \pmod N$.*

As in the last section, the entries of (7.3.12) can be derived either by the inverse discrete Fourier transform or by differentiating a sinc function. The latter approach is illustrated in Figure 7.3.2, which shows S_N and S'_N for $N = 16$. Since N is even, symmetry implies that $S'_N(x) = 0$ for $x = \pm\pi$ as well as for $x = 0$. Differentiation yields

$$S'_N(x) = \frac{\cos\frac{\pi x}{h}}{2\tan\frac{x}{2}} - \frac{\sin\frac{\pi x}{h}}{\frac{4\pi}{h}\sin^2\frac{x}{2}}, \tag{7.3.13}$$

and at the grid points the values are

$$S'_N(x_j) = \begin{cases} 0 & \text{if } j = 0, \\ \frac{1}{2}(-1)^j \cot\frac{jh}{2} & \text{if } j \neq 0. \end{cases} \tag{7.3.14}$$

Notice that for $|jh| \ll 1$, these values are approximately the same as in (7.2.8). Thus the (i,j) entry of D_N, as indicated in (7.3.12), is

$$(D_N)_{ij} = \begin{cases} 0 & \text{if } i = j, \\ \frac{1}{2}(-1)^{i+j} \cot(\frac{x_i - x_j}{2}) & \text{if } i \neq j. \end{cases} \tag{7.3.15}$$

* *Any* circulant matrix describes a convolution on a periodic grid, and is equivalent to a pointwise multiplication in Fourier space. In the case of D_N, that multiplication happens to be by the function $i\xi$.

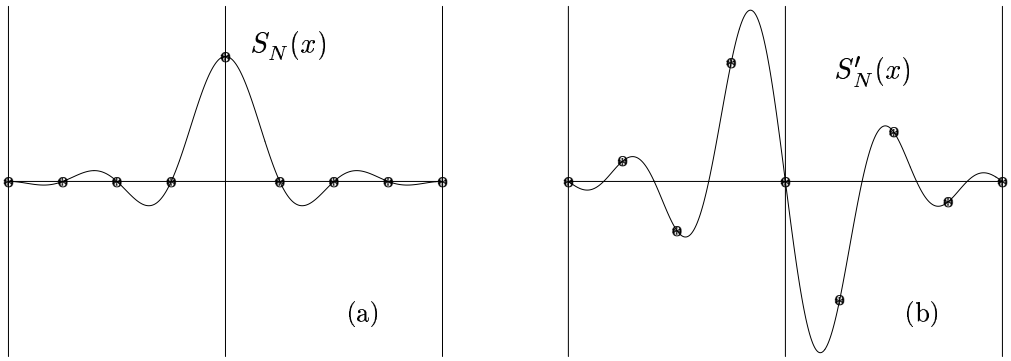


Figure 7.3.2. The periodic sinc function $S_N(x)$ and its derivative $S'_N(x)$.

To derive (7.3.15) by the Fourier transform, one can make use of "summation by parts," the discrete analog of integration by parts. See Exercise 7.3.1.

The eigenvectors of D_N are the vectors $e^{i\xi x}$ with $\xi \in \mathbb{Z}$, and the eigenvalues are the quantities $i\xi$ with $-N/2+1 \le \xi \le N/2-1$.* These are precisely the factors $i\xi$ in the definition of D_N by the Fourier transform formula (7.3.8). The number 0 is actually a double eigenvalue of D_N, corresponding to two distinct eigenvectors, the constant function and the sawtooth.

What about the spectral differentiation operator of second order? Again there are various ways to describe it. This time, because $-\xi^2$ is an even function, no special treatment of $\xi = -N/2$ is required to preserve symmetry in the Fourier transform description:

(1) Compute $\hat{v}_\xi$,
(2) Multiply by $-\xi^2$,
(3) Inverse transform:

$$D_N^{(2)} v = \mathcal{F}_N^{-1}(-\xi^2 \hat{v}_\xi). \tag{7.3.16}$$

The sinc interpolation description follows the usual pattern:

(1) Interpolate v by a sum of periodic sinc functions

$$q(x) = \sum_{k=-N/2}^{N/2-1} v_k S_N(x - x_k);$$

(2) Differentiate the interpolant twice at the grid points x_j,

$$(D_N^{(2)} v)_j = q''(x_j). \tag{7.3.17}$$

The matrix looks like this (compare (7.2.12)):

*Now that D_N is finite, they are truly eigenvalues; there are no technicalities to worry about as in the footnote on p. 241.

$$D_N^{(2)} = \begin{pmatrix} -\frac{\pi^2}{3h^2}-\frac{1}{6} & \frac{1}{2}\csc^2\frac{h}{2} & -\frac{1}{2}\csc^2\frac{2h}{2} & & & & \frac{1}{2}\csc^2\frac{h}{2} \\ & \ddots & & & & & \\ & -\frac{1}{2}\csc^2\frac{2h}{2} & \frac{1}{2}\csc^2\frac{h}{2} & -\frac{\pi^2}{3h^2}-\frac{1}{6} & \frac{1}{2}\csc^2\frac{h}{2} & -\frac{1}{2}\csc^2\frac{2h}{2} & \\ & & & & \ddots & & \\ \frac{1}{2}\csc^2\frac{h}{2} & & & & -\frac{1}{2}\csc^2\frac{2h}{2} & \frac{1}{2}\csc^2\frac{h}{2} & -\frac{\pi^2}{3h^2}-\frac{1}{6} \end{pmatrix} \tag{7.3.18}$$

Note that because $\xi = -N/2$ has been treated differently in the two cases, $D_N^{(2)}$ is not the square of D_N, which is why we have put the superscript in parentheses. In general, the mth-order spectral differentiation operator can be written as a power of $D_N^{(2)}$ if m is even, and as a power of D_N (or as D_N times a power of $D_N^{(2)}$) if m is odd. See Exercise 7.3.2.

Figures 7.3.3–7.3.6 are the analogs of Figures 7.2.2–7.2.5 for a periodic grid.

We summarize and generalize the developments of this section in the following theorem:

SPECTRAL DIFFERENTIATION ON A PERIODIC GRID

Theorem 7.2. *Let N be even. If m is odd, the mth-order spectral differentiation matrix $D_N^{(m)}$ is a skew-symmetric matrix with entries*

$$a_0 = 0, \qquad a_j = (\quad ? \quad) \qquad \text{for} \quad j \neq 0, \tag{7.3.19}$$

eigenvalues $[-i(\frac{\pi}{h}-1)^m, i(\frac{\pi}{h}-1)^m]$, and norm

$$\|D_N^{(m)}\| = \left(\frac{\pi}{h} - 1\right)^m. \tag{7.3.20}$$

If m is even, $D_N^{(m)}$ is a symmetric matrix with entries

$$a_0 = \quad ? \quad , \qquad a_j = (\quad ? \quad) \qquad \text{for} \quad j \neq 0, \tag{7.3.21}$$

eigenvalues $(-1)^{m/2} \times [0, (\frac{\pi}{h})^m]$, and norm

$$\|D_N^{(m)}\| = \left(\frac{\pi}{h}\right)^m. \tag{7.3.22}$$

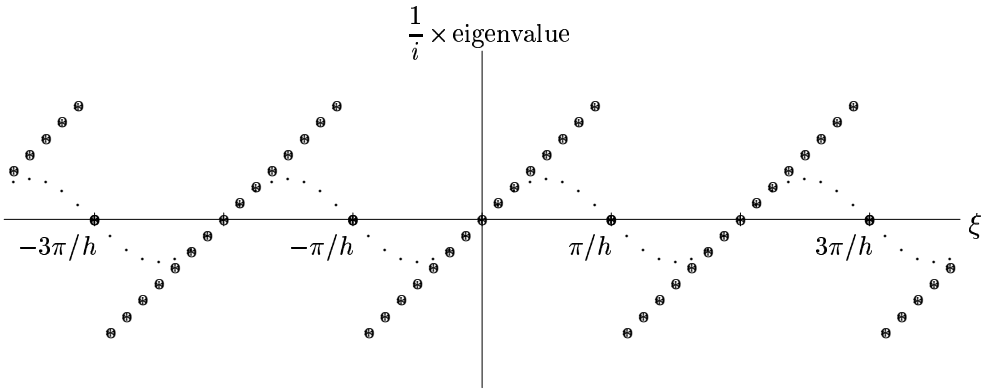
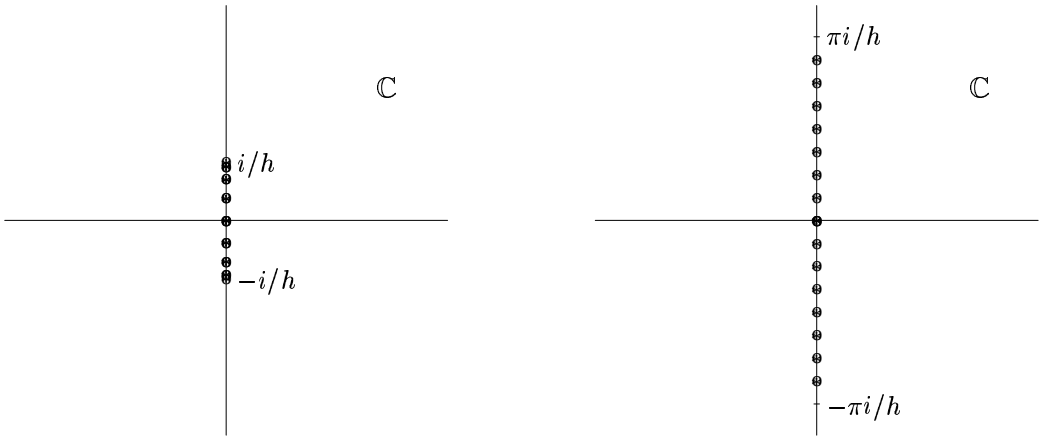


Figure 7.3.3. Eigenvalue of D_N (divided by i) corresponding to the eigenfunction $e^{i\xi x}$, as a function of ξ, for $N = 16$. The smaller dots show corresponding eigenvalues for the finite difference operator δ_0.



(a) δ_0 (finite difference) *(b)* DN (spectral)

Figure 7.3.4. Eigenvalues of finite difference and spectral first-order differentiation matrices on a periodic grid, as subsets of the complex plane, for $N = 16$.

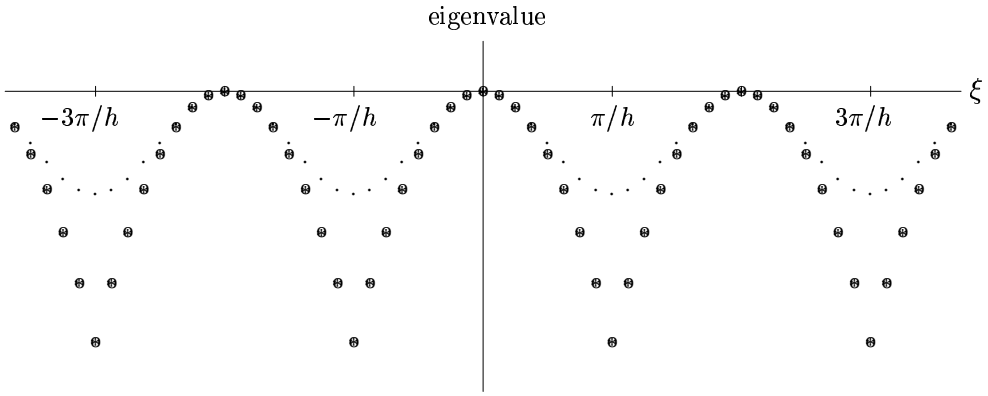
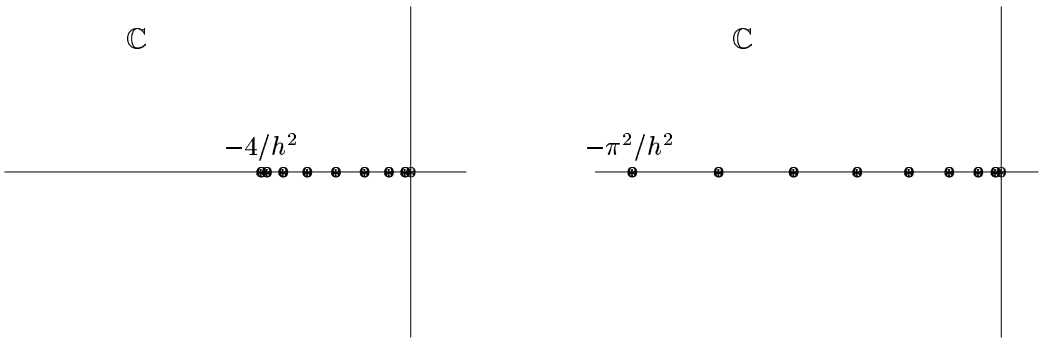


Figure 7.3.5. Eigenvalue of $D_N^{(2)}$ corresponding to the eigenfunction $e^{i\xi x}$, as a function of ξ, for $N = 16$. The smaller dots show corresponding eigenvalues for the finite difference operator $\delta_\times$.



(a) $\delta_\times$ (finite difference) *(b)* D_N^2 (spectral)

Figure 7.3.6. Eigenvalues of second-order finite difference and spectral differentiation matrices on a periodic grid, as subsets of the complex plane, for $N = 16$.

EXERCISES

▷ *7.3.1. Fourier transform derivation of D_N.* [Not yet written.]

▷ *7.3.2. $D_N^{(2)} \neq D_N^2$.* For most values of N, the matrix D_N^2 would serve as quite a good discrete second-order differentiation operator, but as mentioned in the text, it is not identical to $D_N^{(2)}$.

(a) Determine D_N, D_N^2, and $D_N^{(2)}$ for $N = 2$, and confirm that the latter two are not equal.

(b) Explain the result of *(a)* by considering sinc interpolation as in Figure 7.3.2.

(c) Explain it again by considering Fourier transforms as in Figure 7.3.3 and 7.3.5.

(d) Give an exact formula for the eigenvalues of $D_N^{(J)}$ for arbitrary $J \geq 0$.

▶ *7.3.3. Spectral differentiation.*

Making use of a program for computing the FFT (in Matlab such a program is built-in; in Fortran one can use the program of p. 102), write a program DERIV that computes the mth-order spectral derivative of an N-point data sequence v representing a function defined on $[-\pi,\pi]$ or $[0,2\pi]$:

N: length of sequence (power of 2) (input)
m: order of derivative (integer ≥ 0) (input)
v: sequence to be differentiated (real sequence of length N) (input)
w: mth spectral derivative of v (real sequence of length N) (output)

Although a general FFT code deals with complex sequences, make v and w real variables in your program, since most applications concern real variables. Allow m to be any integer $m \geq 0$, and make sure to treat the distinction properly between even and odd values of m.

Test DERIV with $N = 32$ and $N = 64$ on the functions

$$u_1(x) = \exp(\sin 3x) \qquad \text{and} \qquad u_2(x) = |\sin x|^3$$

for the values $m = 1, 2$, and hand in two 2×2 tables—one for each function—of the resulting errors in the discrete ∞-norm. Put a star * where appropriate, and explain your results. Plot the computed derivatives with $m = 1$.

▶ *7.3.4. Spectral integration.* Modify DERIV to accept negative as well as positive values of m. For $m < 0$, DERIV should return a scalar representing the definite integral of v over one period, together with a function w representing the $|m|$th indefinite integral. Explore the behavior of DERIV with various v and $m < 0$.

▶ *7.3.5. Hamming window.* Write a program FILTER that takes a sequence v and smooths it by transforming to $\hat{v}$, multiplying the transform by the "Hamming window",

$$\hat{w}_k = (0.54 + 0.46 \cos \frac{2\pi k}{N}) \hat{v}_k,$$

and inverse transforming. Apply FILTER to the function $|\sin x|$ and hand in a plot of the result.

-

▷ *8.3.1.* Fourier transform derivation of D_N. [Not yet written.]

▷ *8.3.2.* $D_N^{(2)} \neq D_N^2$. For most values of N, the matrix D_N^2 would serve as quite a good discrete second-order differentiation operator, but as mentioned in the text, it is not identical to $D_N^{(2)}$.

*(a)*Determine D_N, D_N^2, and $D_N^{(2)}$ for $N=2$, and confirm that the latter two are not equal.

*(b)*Explain the result of *(a)*by considering sinc interpolation as in Figure 8.3.3.

*(c)*Explain it again by considering Fourier transforms as in Figures 8.3.4 and 8.3.6.

▶ *8.3.3.* Spectral differentiation. Type the program FFT of Figure 8.3.2 into your computer and experiment with it until you are confident you understand how to compute both a discrete Fourier transform and an inverse discrete Fourier transform. For example, try as input a sine wave and a sinc function, and make sure the output you get is what you expect. Then make sure you can get the input back again by inversion.

Making use of FFT, write a program DERIV(N,m,v,w) which returns the mth-order spectral derivative of the N-point data sequence v representing a function defined on $[-\pi,\pi]$ or $[0,2\pi]$:

N: length of sequence (power of 2) (input)
m: order of derivative (integer ≥ 0) (input)
v: sequence to be differentiated (real sequence of length N) (input)
w: mth spectral derivative of v (real sequence of length N) (output)

Although FFT deals with complex sequences, make v and w real variables in your program, since most applications concern real variables. Allow m to be any integer $m \geq 0$, and make sure to treat the distinction properly between even and odd values of m.

Test DERIV with $N=32$ and $N=64$ on the functions

$$u_1(x) = \exp(\sin^3 x) \qquad \text{and} \qquad u_2(x) = |\sin x|^3$$

for the values $m=1,2$, and hand in two 2×2 tables—one for each function—of the resulting errors in the discrete ∞-norm. Explain your results. If possible, plot the computed derivatives with $m=1$.

▶ *8.3.4.* Spectral integration. Modify DERIV to accept negative as well as positive values of m. For $m<0$, DERIV should return a scalar representing the definite integral of v over one period, together with a function w representing the $|m|$th indefinite integral. Explore the behavior of DERIV with various v and $m<0$.

▶ *8.3.5.* Hamming window. Write a program FILTER(N,v,w) which takes a sequence v and smooths it by transforming to $\hat{v}$, multiplying the transform by the "Hamming window",

$$\hat{w}_k = (.54 + .46\cos\frac{2\pi k}{N})\hat{v}_k,$$

and inverse transforming. Apply FILTER to the function $|\sin x|$ and hand in the result. A plot would be nice.

7.4. Stability

[Just a few results so far. The finished section will be more substantial.]

Spectral methods are commonly applied to time-dependent problems according to the "method of lines" prescription of §3.3: first the problem is discretized with respect to space, and then the resulting system of ordinary differential equations is solved by a finite difference method in time. As usual, we can investigate the eigenvalue stability of this process by examining under what conditions the eigenvalues of the spectral differentiation operator are contained in the stability region of the time discretization formula. The separate question of stability in the sense of the Lax Equivalence Theorem is rather different, and involves some subtleties that were not present with finite difference methods; these issues are deferred to the next section.

In §§7.2,7.3 we have introduced two families of spectral differentiation matrices: D and its powers D^m for an infinite grid, and D_N and its higher-order analogs $D_N^{(m)}$ (not exactly powers) for a periodic grid. The spectra of all of these matrices lie in the closed left half of the complex plane, and that is the same region that comes up in the definition of A-stability. We conclude that if any equation

$$u_t = \frac{\partial^m u}{\partial x^m} \tag{7.4.1}$$

is modeled on a regular grid by spectral differentiation in space and an A-stable formula in time, the result is eigenvalue stable, regardless of the time step.

By Theorem 1.13, an A-stable linear multistep formula must be implicit. For a model problem as simple as (7.4.1), the system of equations involved in the implicit formula can be solved quickly by the FFT, but in more realistic problems this is often not true. Since spectral differentiation matrices are dense (unlike finite difference differentiation matrices), the implementation of implicit formulas can be a formidable problem. Therefore it is desirable to look for explicit alternatives.

On a regular grid, satisfactory explicit alternatives exist. For example, suppose we solve $u_t = u_x$ by spectral differentiation in space and the midpoint formula (1.2.6) in time. The stability region for the midpoint formula is the complex interval $[-i/k, i/k]$. From Theorem 7.1 or Figure 7.2.3, we conclude that the time-stability restriction is $\pi/h \le 1/k$, i.e.

$$k \le \frac{h}{\pi}. \tag{7.4.2}$$

This is stricter by a factor π than the time-stability restriction $k \le h$ for the leap frog formula, which is based on second-order finite difference differentiation. The explanation goes back to the fact that the sawtooth curve in Figure 7.2.2 is π times taller than the dashed one.

On a periodic grid, Theorem 7.2 or Figure 7.3.4 loosens (7.4.2) slightly to

$$k \le \frac{h}{\pi - h} = \frac{2}{N-2}. \tag{7.4.3}$$

Other explicit time-discretization formulas can also be used with $u_t = u_x$, so long as their stability regions include a neighborhood of the imaginary axis near the origin. Figure

1.7.4 reveals that this is true, for example, for the Adams-Bashforth formulas of orders 3–6. The answer to Exercise 1.7.2(b) can readily be converted to the exact stability bound for the 3rd-order Adams-Bashforth discretization.

For $u_t = u_{xx}$, the eigenvalues of D or D_N become real and negative, so we need a stability region that contains a segment of the negative real axis. Thus the midpoint rule will be unstable. On the other hand the Euler formula, whose stability region was drawn in Figure 1.7.3 and again in Figure 1.7.4, leads to the stability restriction $\pi^2/h^2 \le 2/k$, i.e.

$$k \le \frac{2h^2}{\pi^2}. \tag{7.4.4}$$

On an infinite grid this is $\pi^2/4$ stricter than the stability restriction for the finite difference forward Euler formula considered in Example 4.4.3.* (The cusped curve in Figure 7.2.4 is $\pi^2/4$ times deeper than the dashed one.) By Theorem 7.2, exactly the same restriction is also valid for a periodic grid:

$$k \le \frac{2h^2}{\pi^2} = \frac{8}{N^2}. \tag{7.4.5}$$

As another example, the answer to Exercise 1.7.2(a) can be converted to the exact stability bound for the 3rd-order Adams-Bashforth discretization of $u_t = u_{xx}$.

In general, spectral methods on a periodic grid tend to have stability restrictions that are stricter by a constant factor than their finite difference counterparts. (This is opposite to what the CFL condition might suggest: the numerical domain of dependence of a spectral method is unbounded, so there is no CFL stability limit.) The constant factor is usually not much of a problem in practice, for spectral methods permit larger values of h than finite difference methods in the first place, because of their high order of spatial accuracy. In other words, relatively small time steps k are needed anyway to avoid large time-discretization errors.

EXERCISES

▶ *7.4.1. A simple spectral calculation.*

Write a program to solve $u_t = u_x$ on $[-\pi, \pi]$ with periodic boundary conditions by the pseudospectral method. The program should use the midpoint time integration formula and spatial differentiation obtained from the program DERIV of Exercise 7.3.3.

(a) Run the program up to $t = 2\pi$ with $k = h/4$, $N = 8$ and $N = 16$, and initial data $v^0 = f(x)$, $v^1 = f(x+k)$, with both

$$f_1(x) = \cos^2 x \qquad \text{and} \qquad f_2(x) = \begin{cases} \cos^2 x & \text{for } |x| \le \pi/2 \,(\mathrm{mod}\, 2\pi), \\ 0 & \text{otherwise.} \end{cases}$$

List the four ℓ_N^∞ errors you obtain at $t = 2\pi$. Plot the computed solutions $v(x, 2\pi)$ if possible.

(b) Rerun the program with $k = h/2$ and list the same four errors as before.

(c) Explain the results of *(a)* and *(b)*. What order of accuracy is observed? How might it be improved?

*Why is the ratio not π^2? Because $\delta_\times = \delta_0(h/2)^2$, not $\delta_0(h)^2$.

▶ *7.4.2. Spectral calculation with filtering.*

(a) Modify the program above so that instead of computing v^n at each step with DERIV alone, it uses DERIV followed by the program FILTER of Exercise 7.3.5. Take $k = h/4$ again and print and plot the same results as in the last problem. Are the errors smaller than they were without FILTER?

(b) Run the program again in DERIV/FILTER mode with $k = h/2$. Print and plot the same results as in *(c)*.

(c) What is the exact theoretical stability restriction on k for the DERIV/FILTER method? You may consider the limit $N = \infty$ for simplicity.

Note. Filtering is an important idea in spectral methods, but this simple linear problem is not a good example of a problem where filtering is needed.

▶ *7.4.3. Inviscid Burgers' equation.*

Write a program to solve $u_t = (u^2)_x$ on $[0, 2\pi]$ with periodic boundary conditions by the pseudospectral method. The program should use forward Euler time integration formula and spatial differentiation implemented with the program DERIV.

(a) Run the program up to $t = 2\pi$ with $k = h/8$, $N = 32$, and initial data $v^0 = 0.3 + 0.08 \sin x$. Plot the computed results (superimposed on a single plot) at times $t = 0, \pi/4, \pi/2, \ldots, 2\pi$.

(b) Explain the results of part *(a)* as well as you can.

(c) (Optional.) Can you find a way to improve the calculation?

Chapter 8.
Chebyshev spectral methods

This chapter discusses spectral methods for domains with boundaries. The effect of boundaries in spectral calculations is great, for they often introduce stability conditions that are both highly restrictive and difficult to analyze. Thus for a first-order partial differential equation solved on an N-point spatial grid by an explicit time-integration formula, a spectral method typically requires $k = O(N^{-2})$ for stability, in contrast to $k = O(N^{-1})$ for finite differences. For a second-order equation the disparity worsens to $O(N^{-4})$ vs. $O(N^{-2})$. To make matters worse, the matrices involved are usually nonnormal, and often very far from normal, so they are difficult to analyze as well as troublesome in practice.

Spectral methods on bounded domains typically employ grids consisting of zeros or extrema of Chebyshev polynomials ("Chebyshev points"), zeros or extrema of Legendre polynomials ("Legendre points"), or some other set of points related to a family or orthogonal polynomials. Chebyshev grids have the advantage that the FFT is available for an $O(N \log N)$ implementation of the differentiation process, and they also have slight advantages connected their ability to approximate functions. Legendre grids have various theoretical and practical advantages because of their connection with Gauss quadrature. At this point one cannot say which choice will win in the long run, but in this book, in keeping with out emphasis on Fourier analysis, most of the discussion is of Chebyshev grids.

Since explicit spectral methods are sometimes troublesome, implicit spectral calculations are increasingly popular. Spectral differentiation matrices are dense and ill-conditioned, however, so solving the associated systems of equations is not a trivial matter, even in one space dimension. Currently popular methods for solving these systems include preconditioned iterative methods and multigrid methods. These techniques are discussed briefly in §8.7.

8.1. Polynomial interpolation

Spectral methods arise from the fundamental problem of approximation of a function by interpolation on an interval. Multidimensional domains of a rectilinear shape are treated as products of simple intervals, and more complicated geometries are sometimes divided into rectilinear pieces.* In this section, therefore, we restrict our attention to the fundamental interval $[-1,1]$. The question to be considered is, what kinds of interpolants, in what sets of points, are likely to be effective?

Let $N \geq 1$ be an integer, even or odd, and let $x_0,\ldots,x_N$ or sometimes $x_1,\ldots,x_N$ be a set of distinct points in $[-1,1]$. For definiteness let the numbering be in reverse order:

$$1 \quad \geq \quad x_0 > x_1 > \cdots > x_{N-1} > x_N \quad \geq \quad -1. \tag{8.1.1}$$

The following are some grids that are often considered:

$$\textbf{Equispaced points:} \quad x_j = 1 - \frac{2j}{N} \quad (0 \leq j \leq N),$$

$$\textbf{Chebyshev zero points:} \quad x_j = \cos\frac{(j-1/2)\pi}{N} \quad (1 \leq j \leq N),$$

$$\textbf{Chebyshev extreme points:} \quad x_j = \cos\frac{j\pi}{N} \quad (0 \leq j \leq N),$$

$$\textbf{Legendre zero points:} \quad x_j = j\,\text{th zero of } P_N \quad (1 \leq j \leq N),$$

$$\textbf{Legendre extreme points:} \quad x_j = j\,\text{th extremum of } P_N \quad (0 \leq j \leq N),$$

where P_N is the Legendre polynomial of degree N. Chebyshev zeros and extreme points can also be described as zeros and extrema of Chebyshev polynomials T_N (more on these in §8.3). Chebyshev and Legendre zero points are also called Gauss-Chebyshev and Gauss-Legendre points, respectively, and Chebyshev and Legendre extreme points are also called Gauss-Lobatto-Chebyshev and Gauss-Lobatto-Legendre points, respectively. (These names originate in the field of numerical quadrature.)

*Such subdivision methods have been developed independently by I. Babushka and colleagues for structures problems, who call them **"p" finite element methods**, and by A. Patera and colleagues for fluids problems, who call them **spectral element methods.**

It is easy to remember how Chebyshev points are defined: they are the projections onto the interval $[-1,1]$ of equally-spaced points (roots of unity) along the unit circle $|z|=1$ in the complex plane:

Figure 8.1.1. Chebyshev extreme points ($N=8$).

To the eye, Legendre points look much the same, although there is no elementary geometrical definition. Figure 8.1.2 illustrates the similarity:

(a) $N=5$

(b) $N=25$

Figure 8.1.2. Legendre vs. Chebyshev zeros.

As $N\to\infty$, equispaced points are distributed with density

$$\mu(x)=\frac{N}{2} \qquad \text{Equally spaced,} \tag{8.1.2}$$

and Legendre or Chebyshev points—either zeros or extrema—have density

$$\mu(x) = \frac{N}{\pi\sqrt{1-x^2}} \qquad \text{Legendre, Chebyshev.} \tag{8.1.3}$$

Indeed, the density function (8.1.3) applies to point sets associated with any Jacobi polynomials, of which Legendre and Chebyshev polynomials are special cases.

Why is it a good idea to base spectral methods upon Chebyshev, Legendre, and other irregular grids? We shall answer this question by addressing a second, more fundamental question: why is it a good idea to interpolate a function $f(x)$ defined on $[-1,1]$ by a polynomial $p_N(x)$ rather than a trigonometric polynomial, and why is it a good idea to use Chebyshev or Legendre points rather than equally spaced points?

[The remainder of this section is just a sketch... details to be supplied later.]

PHENOMENA

Trigonometric interpolation in equispaced points suffers from the **Gibbs phenomenon**, due to Michelson and Gibbs at the turn of the twentieth century. $\|f - p_N\| = O(1)$ as $N \to \infty$, even if f is analytic. One can try to get around the Gibbs phenomenon by various tricks such as doubling the domain and reflecting, but the price is high.

Polynomial interpolation in equally spaced points suffers from the **Runge phenomenon**, due to Meray and Runge (Figure 8.1.3). $\|f - p_N\| = O(2^N)$—much worse!

Polynomial interpolation in Legendre or Chebyshev points: $\|f - p_N\| = O(\text{constant}^{-N})$ if f is analytic (for some constant greater than 1). Even if f is quite rough the errors will still go to zero provided f is, say, Lipschitz continuous.

Figure 8.1.3. The Runge phenomenon.

FIRST EXPLANATION—EQUIPOTENTIAL CURVES

Think of the limiting point distribution $\mu(x)$, above, as a charge density distribution; a charge at position x is associated with a potential $\log|z-x|$. Look at the equipotential curves of the resulting potential function $\phi(z) = \int_{-1}^{1} \mu(x) \log|z-x|\, dx$.

CONVERGENCE OF POLYNOMIAL INTERPOLANTS

Theorem 8.1.

In general, $\|f - p_N\| \to 0$ as $N \to \infty$ in the largest region bounded by an equipotential curve in which f is analytic. In particular:

For Chebyshev or Legendre points, or any other type of Gauss-Jacobi points, convergence is guaranteed if f is analytic on $[-1,1]$.

For equally spaced points, convergence is guaranteed if f is analytic in a particular lens-shaped region containing $(-1,1)$ (Figure 8.1.4).

Figure 8.1.4. Equipotential curves.

SECOND EXPLANATION—LEBESGUE CONSTANTS

Definition of **Lebesgue constant**:

$$\Lambda_N = \|I_N\|_\infty,$$

where I_N is the interpolation operator $I_N : f \mapsto p_N$. A small Lebesgue constant means that the interpolation process is not much worse than best approximation:

$$\|f - p_N\| \le (\Lambda_N + 1)\|f - p_N^*\|, \tag{8.1.1}$$

where p_N^* is the best (minimax, equiripple) approximation.

LEBESGUE CONSTANTS

Theorem 8.2.

Equispaced points: $\Lambda_N \sim 2^N/eN \log N$.

Legendre points: $\Lambda_N \sim const\sqrt{N}$.

Chebyshev points: $\Lambda_N \sim const \log N$.

THIRD EXPLANATION—NUMBER OF POINTS PER WAVELENGTH

Consider approximation of, say, $f_N(x) = \cos \alpha N x$ as $N \to \infty$. Thus f_N changes but the number of points per wavelength remains constant. Will the error $\|f_N - p_N\|$ go to zero? The answer to this question tells us something about the ability of various kinds of spectral methods to resolve data.

POINTS PER WAVELENGTH

Theorem 8.3.

Equispaced points: convergence if there are at least 6 *points per wavelength.*

Chebyshev points: convergence if there are at least π *points per wavelength on average.*

We have to say "on average" because the grid is nonuniform. In fact, it is $\pi/2$ times less dense in the middle than the equally spaced grid with the same number of points N (see (8.1.2) and (8.1.3)). Thus the second part of the theorem says that we need at least 2 points per wavelength in the center of the grid—the familiar Nyquist limit. See Figure 8.1.5. The first part of the theorem is mathematically valid, but of little value in practice because of rounding errors.

(a) Equally spaced points

(b) Chebyshev points

Figure 8.1.5. Error as a function of N in interpolation of $\cos \alpha N x$, with α, hence the number of points per wavelength, held fixed.

8.2. Chebyshev differentiation matrices

[Just a sketch]

From now on "Chebyshev points" means Chebyshev extreme points.

Multiplication by the first-order Chebyshev differentiation matrix D_N transforms a vector of data at the Chebyshev points into approximate derivatives at those points:

$$D_N \begin{bmatrix} v_0 \\ \vdots \\ v_N \end{bmatrix} = \begin{bmatrix} w_0 \\ \vdots \\ w_N \end{bmatrix}.$$

As usual, the implicit definition of D_N is as follows:

CHEBYSHEV SPECTRAL DIFFERENTIATION BY POLYNOMIAL INTERPOLATION.

(1) *Interpolate* v *by a polynomial* $q(x) = q_N(x)$;

(2) *Differentiate the interpolant at the grid points* x_j:

$$w_j = (D_N v)_j = q'(x_j). \tag{8.2.1}$$

Higher-order differentiation matrices are defined analogously. From this definition it is easy to work out the entries of D_N in special cases. For $N = 1$:

$$\mathbf{x} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}, \qquad D_1 = \begin{bmatrix} \frac{1}{2} & -\frac{1}{2} \\ \frac{1}{2} & -\frac{1}{2} \end{bmatrix}.$$

For $N = 2$:

$$\mathbf{x} = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix}, \qquad D_2 = \begin{bmatrix} \frac{3}{2} & -2 & \frac{1}{2} \\ \frac{1}{2} & 0 & -\frac{1}{2} \\ -\frac{1}{2} & 2 & -\frac{3}{2} \end{bmatrix}.$$

For $N=3$:

$$\mathbf{x}=\begin{bmatrix} 1 \\ \frac{1}{2} \\ -\frac{1}{2} \\ -1 \end{bmatrix}, \qquad D_3=\begin{bmatrix} \frac{19}{6} & -4 & \frac{4}{3} & -\frac{1}{2} \\ 1 & -\frac{1}{3} & -1 & \frac{1}{3} \\ -\frac{1}{3} & 1 & \frac{1}{3} & -1 \\ \frac{1}{2} & -\frac{4}{3} & 4 & -\frac{19}{6} \end{bmatrix}.$$

These three examples illustrate an important fact, mentioned in the introduction to this chapter: Chebyshev spectral differentiation matrices are in general not symmetric or skew-symmetric. A more general statement is that they are not normal.* This is why stability analysis is difficult for spectral methods. The reason they are not normal is that unlike finite difference differentiation, spectral differentiation is not a translation-invariant process, but depends instead on the same global interpolant at all points x_j.

The general formula for D_N is as follows. First, define

$$c_i = \begin{cases} 2 & \text{for } i=0 \text{ or } N, \\ 1 & \text{for } 1 \le i \le N-1, \end{cases} \tag{8.2.2}$$

and of course analogously for c_j. Then:

CHEBYSHEV SPECTRAL DIFFERENTIATION

Theorem 8.4. *Let $N \ge 1$ be any integer. The first-order spectral differentiation matrix D_N has entries*

$$(D_N)_{00} = \frac{2N^2+1}{6}, \qquad (D_N)_{NN} = -\frac{2N^2+1}{6},$$

$$(D_N)_{jj} = \frac{-x_j}{2(1-x_j^2)} \quad \text{for } 1 \le j \le N-1,$$

$$(D_N)_{ij} = \frac{c_i}{c_j}\frac{(-1)^{i+j}}{x_i - x_j} \quad \text{for } i \ne j.$$

Analogous formulas for D_N^2 can be found in Peyret (1986), Ehrenstein & Peyret [ref?] and in Zang, Streett, and Hussaini, ICASE Report 89-13, 1989. See also Canuto, Hussaini, Quarteroni & Zang.

*Recall that a normal matrix A is one that satisfies $AA^T = A^TA$. Equivalently, A possesses an orthogonal set of eigenvectors, which implies many desirable properties such as $\rho(A^n) = \|A^n\| = \|A\|^n$ for any n.

A note of caution: D_N is rarely used in exactly the form described in Theorem 8.4, for boundary conditions will modify it slightly, and these depend on the problem.

EXERCISES

▷ *8.2.1.* Prove that for any N, D_N is nilpotent: $D_N^n = 0$ for a sufficiently high integer n.

8.3. Chebyshev differentiation by the FFT

Polynomial interpolation in Chebyshev points is equivalent to trigonometric interpolation in equally spaced points, and hence can be carried out by the FFT. The algorithm described below has the optimal order $O(N \log N)$,* but we do not worry about achieving the optimal constant factor. For more practical discussions, see Appendix B of the book by Canuto, et al., and also P. N. Swarztrauber, "Symmetric FFTs," *Math. Comp. 47* (1986), 323–346. Valuable additional references are the book *The Chebyshev Polynomials* by Rivlin and Chapter 13 of P. Henrici, *Applied and Computational Complex Analysis*, 1986.

Consider three independent variables $\theta \in \mathbb{R}$, $x \in [-1,1]$, and $z \in S$, where S is the complex unit circle $\{z : |z| = 1\}$. They are related as follows:

$$z = e^{i\theta}, \qquad x = \mathrm{Re}\, z = \tfrac{1}{2}(z + z^{-1}) = \cos\theta, \tag{8.3.1}$$

which implies

$$\frac{dx}{d\theta} = -\sin\theta = -\sqrt{1-x^2}. \tag{8.3.2}$$

See Figure 8.3.1. Note that there are two conjugate values $z \in S$ for each $x \in (-1,1)$, and an infinite number of possible choices of θ.

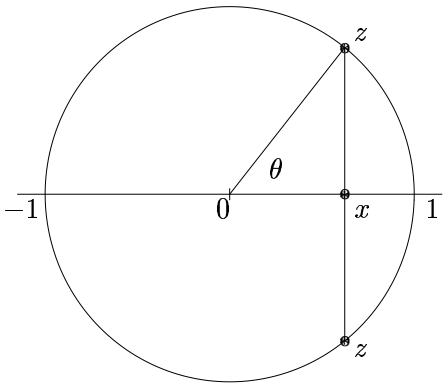


Figure 8.3.1. z, x, and θ.

*optimal, that is, so far as anyone knows as of 1994.

In generalization of the fact that the real part of z is x, the real part of z^n $(n \geq 0)$ is $T_n(x)$, the **Chebyshev polynomial** of degree n. This statement can be taken as a definition of Chebyshev polynomials:

$$T_n(x) = \mathrm{Re}\, z^n = \tfrac{1}{2}(z^n + z^{-n}) = \cos n\theta, \tag{8.3.3}$$

where x and z and θ are, as always, implicitly related by (8.3.1).* It is clear that (8.3.3) defines $T_n(x)$ to be *some* function of x, but it is not obvious that the function is a polynomial. However, a calculation of the first few cases makes it clear what is going on:

$$\begin{aligned} T_0(x) &= \tfrac{1}{2}(z^0 + z^{-0}) = 1, \\ T_1(x) &= \tfrac{1}{2}(z^1 + z^{-1}) = x, \\ T_2(x) &= \tfrac{1}{2}(z^2 + z^{-2}) = \tfrac{1}{2}(z^1 + z^{-1})^2 - 1 = 2x^2 - 1, \\ T_3(x) &= \tfrac{1}{2}(z^3 + z^{-3}) = \tfrac{1}{2}(z^1 + z^{-1})^3 - \tfrac{3}{2}(z^1 + z^{-1}) = 4x^3 - 3x. \end{aligned} \tag{8.3.4}$$

In general, the Chebyshev polynomials are related by the three-term recurrence relation

$$\begin{aligned} T_{n+1}(x) &= \tfrac{1}{2}(z^{n+1} + z^{-n-1}) \\ &= \tfrac{1}{2}(z^1 + z^{-1})(z^n + z^{-n}) - \tfrac{1}{2}(z^{n-1} + z^{-n+1}) \\ &= 2x T_n(x) - T_{n-1}(x). \end{aligned} \tag{8.3.5}$$

By (8.3.2) and (8.3.3), the derivative of $T_n(x)$ is

$$T_n'(x) = -n \sin n\theta \frac{d\theta}{dx} = \frac{n \sin n\theta}{\sin \theta}. \tag{8.3.6}$$

Thus just as x, z, and θ are equivalent, so are $T_n(x)$, z^n, and $\cos n\theta$. By taking linear combinations, we obtain three equivalent kinds of polynomials. A **trigonometric polynomial** $q(\theta)$ of degree N is a 2π-periodic sum of complex exponentials in θ (or equivalently, sines and cosines). Assuming that $q(\theta)$ is an even function of θ, it can be written

$$q(\theta) = \tfrac{1}{2} \sum_{n=0}^{N} a_n (e^{in\theta} + e^{-in\theta}) = \sum_{n=0}^{N} a_n \cos n\theta. \tag{8.3.7}$$

A **Laurent polynomial** $q(z)$ of degree N is a sum of negative and positive powers of z up to degree N. Assuming $q(z) = q(\bar{z})$ for $z \in S$, it can be written

$$q(z) = \tfrac{1}{2} \sum_{n=0}^{N} a_n (z^n + z^{-n}). \tag{8.3.8}$$

An **algebraic polynomial** $q(x)$ of degree N is a polynomial in x of the usual kind, and we can express it as a linear combination of Chebyshev polynomials:

$$q(x) = \sum_{n=0}^{N} a_n T_n(x). \tag{8.3.9}$$

*Equivalently, the Chebyshev polynomials can be defined as a system of polynomials orthogonal on $[-1, 1]$ with respect to the weight function $(1 - x^2)^{-1/2}$.

The use of the same coefficients a_n in (8.3.7)–(8.3.9) is no accident, for all three of the polynomials above are identical:

$$q(\theta) = q(z) = q(x), \tag{8.3.10}$$

where again, x and z and θ are implicitly related by (8.3.1). For this reason we hope to be forgiven the sloppy use of the same letter q in all three cases.

Finally, for any integer $N \geq 1$, we define regular grids in the three variables as follows:

$$\theta_j = \frac{j\pi}{N}, \qquad z_j = e^{i\theta_j}, \qquad x_j = \mathrm{Re}\, z_j = \tfrac{1}{2}(z_j + z_j^{-1}) = \cos\theta_j \tag{8.3.11}$$

for $0 \leq j \leq N$. The points $\{x_j\}$ and $\{z_j\}$ were illustrated already in Figure 8.1.1. And now we are ready to state the algorithm for Chebyshev differentiation by the FFT.

ALGORITHM FOR CHEBYSHEV DIFFERENTIATION

1. *Given data* $\{v_j\}$ *defined at the Chebyshev points* $\{x_j\}$, $0 \le j \le N$, *think of the same data as being defined at the equally spaced points* $\{\theta_j\}$ *in* $[0,\pi]$.

2. *(FFT) Find the coefficients* $\{a_n\}$ *of the trigonometric polynomial*
$$q(\theta) = \sum_{n=0}^{N} a_n \cos n\theta \tag{8.3.12}$$
that interpolates $\{v_j\}$ *at* $\{\theta_j\}$.

3. *(FFT) Compute the derivative*
$$\frac{dq}{d\theta} = -\sum_{n=0}^{N} n\, a_n \sin n\theta. \tag{8.3.13}$$

4. *Change variables to obtain the derivative with respect to* x:
$$\frac{dq}{dx} = \frac{dq}{d\theta}\frac{d\theta}{dx} = \sum_{n=0}^{N} \frac{n\, a_n \sin n\theta}{\sin\theta} = \sum_{n=0}^{N} \frac{n\, a_n \sin n\theta}{\sqrt{1-x^2}}. \tag{8.3.14}$$
At $x = \pm 1$, *i.e.* $\theta = 0, \pi$, *L'Hopital's rule gives the special values*
$$\frac{dq}{dx}(\pm 1) = \sum_{n=0}^{N} (\pm 1)^n n^2\, a_n \tag{8.3.15}$$

5. *Evaluate the result at the Chebyshev points:*
$$w_j = \frac{dq}{dx}(x_j). \tag{8.3.16}$$

Note that by (8.3.3), equation (8.3.12) can be interpreted as a linear combination of Chebyshev polynomials, and by (8.3.6), equation (8.3.14) is the corresponding linear combination of derivatives.* But of course the algorithmic content of the description above relates to the θ variable, for in Steps 2 and 3, we have performed Fourier spectral differentiation exactly as in §7.3: discrete Fourier transform, multiply by $i\xi$, inverse discrete Fourier transform. Only the use of sines and cosines rather than complex exponentials, and of n instead of ξ, has disguised the process somewhat.

*or of Chebyshev polynomials $U_n(x)$ of the *second kind*.

EXERCISES

▶ *8.3.1. Fourier and Chebyshev spectral differentiation.*

Write four brief, elegant Matlab programs for first-order spectral differentiation:

FDERIVM, CDERIVM: construct differentiation matrices;

FDERIV, CDERIV: differentiate via FFT.

In the Fourier case, there are N equally spaced points $x_{-N/2},\dots,x_{N/2-1}$ (N even) in $[-\pi,\pi]$, and no boundary conditions. In the Chebyshev case, there are N Chebyshev points $x_1,\dots,x_N$ in $[-1,1)$ (N arbitrary), with a zero boundary condition at $x=1$. The effect of this boundary condition is that one removes the first row and first column from D_N, leading to a square matrix of dimension N instead of $N+1$.

You do not have to worry about computational efficiency (such as using an FFT of length N rather than $2N$ in the Chebyshev case), but you are welcome to worry about it if you like.

Experiment with your programs to make sure they differentiate successfully. Of course, the matrices can be used to check the FFT programs.

(a) Turn in a plot showing the function $u(x)=\cos(x/2)$ and its derivative computed by FDERIV, for $N=32$. Discuss the results.

(b) Turn in a plot showing the function $u(x)=\cos(\pi x/2)$ and its derivative computed by CDERIV, again for $N=32$. Discuss the results.

(c) Plot the eigenvalues of D_N for Fourier and Chebyshev spectral differentiation with $N=8$, 16, 32, 64.

8.5. Stability

This section is not yet written. What follows is a copy of a paper of mine from K. W. Morton and M. J. Baines, eds., *Numerical Methods for Fluid Dynamics III*, Clarendon Press, Oxford, 1988.

Because of stability problems like those described in this paper, more and more attention is currently being devoted to implicit time-stepping methods for spectral computations. The associated linear algebra problems are generally solved by preconditioned matrix iterations, sometimes including a multigrid iteration.

This paper was written before I was using the terminology of pseudospectra. I would now summarize Section 5 of this paper by saying that although the spectrum of the Legendre spectral differentiation matrix is of size $\Theta(N)$ as $N \to \infty$, the pseudospectra are of size $\Theta(N^2)$ for any $\epsilon > 0$. The connection of pseudospectra with stability of the method of lines was discussed in Sections 4.5–4.7.

8.6. Some review problems

EXERCISES

▷ *8.6.1. TRUE or FALSE?* Give each answer together with at most two or three sentences of explanation. The best possible explanation is a proof, a counterexample, or the citation of a theorem in the text from which the answer follows. If you can't do quite that well, try at least to give a convincing reason why the answer you have chosen is the right one. In some cases a well-thought-out sketch will suffice.

(a) The Fourier transform of $f(x) = \exp(-x^4)$ has compact support.

(b) When you multiply a matrix by a vector on the right, i.e. Ax, the result is a linear combination of the columns of that matrix.

(c) If an ODE initial-value problem with a smooth solution is solved by the fourth-order Adams-Bashforth formula with step size k, and the missing starting values v^1, v^2, v^3 are obtained by taking Euler steps with some step size k', then in general we will need $k' = O(k^4)$ to maintain overall fourth-order accuracy.

(d) If a consistent finite difference model of a well-posed linear initial-value problem violates the CFL condition, it must be unstable.

(e) If you Fourier transform a function $u \in L^2$ four times in a row, you end up with u again, times a constant factor.

(f) If the function $f(x) = (x^2 - 2x + 26/25)^{-1}$ is interpolated by a polynomial $q_N(x)$ in N equally spaced points of $[-1,1]$, then $\|f - q_N\|_\infty \to 0$ as $N \to \infty$.

(g) $e^x = O(xe^{x/2})$ as $x \to \infty$.

(h) If a stable finite-difference approximation to $u_t = u_x$ with real coefficients has order of accuracy 3, then the formula must be dissipative.

(i) If

$$A = \begin{pmatrix} \frac{1}{2} & 1 \\ \frac{1}{2} & \frac{1}{2} \end{pmatrix},$$

then $\|A^n\| < C \ \ \forall n$ for some constant $C < \infty$.

(j) If the equation $u_t = -100A^2u$ is solved by the fourth-order Adams-Moulton formula, where $u(x,t)$ is a 2-vector and A is the matrix above, then $k = 0.01$ is a sufficiently small time step to ensure time-stability.

(k) Let $u_t = u_{xx}$ on $[-\pi, \pi]$, with periodic boundary conditions, be solved by Fourier pseudospectral differentiation in x coupled with a fourth-order Runge-Kutta

formula in t. For $N = 32$, $k = 0.01$ is a sufficiently small time step to ensure time-stability.

(l) The ODE initial-value problem $u_t = f(u,t) = \cos^2 u$, $u(0) = 1$, $0 \le t \le 100$, is well-posed.

(m) In exact arithmetic and with exact starting values, the numerical approximations computed by the linear multistep formula

$$v^{n+3} = \tfrac{1}{3}(v^{n+2} + v^{n+1} + v^n) + \tfrac{2}{3}k(f^{n+2} + f^{n+1} + f^n)$$

are guaranteed to converge to the unique solution of a well-posed initial-value problem in the limit $k \to 0$.

(n) If computers did not make rounding errors, we would not need to study stability.

(o) The solution at time $t = 1$ to $u_t = u_x + u_{xx}$ ($x \in \mathbb{R}$, initial data $u(x,0) = f(x)$) is the same as what you would get by first diffusing the data $f(x)$ according to the equation $u_t = u_{xx}$, then translating the result leftward by one unit according to the equation $u_t = u_x$.

(p) The discrete Fourier transform of a three-dimensional periodic set of data on an $N \times N \times N$ grid can be computed on a serial computer in $O(N^3 \log N)$ operations.

(q) The addition of numerical dissipation may sometimes increase the stability limit of a finite difference formula without affecting the order of accuracy.

(r) For a nondissipative semidiscrete finite-difference model (i.e., discrete space but continuous time), phase velocity as well as group velocity is a well-defined quantity.

(s) $v_0^{n+1} = v_4^n$ is a stable left-hand boundary condition for use with the leap frog model of $u_t = u_x$ with $k/h = 0.5$.

(t) If a finite difference model of a partial differential equation is stable with $k/h = \lambda_0$ for some $\lambda_0 > 0$, then it is stable with $k/h = \lambda$ for any $\lambda \le \lambda_0$.

(u) To solve the system of equations that results from a standard second-order discretization of Laplace's equation on an $N \times N \times N$ grid in three dimensions by the obvious method of banded Gaussian elimination, without any clever tricks, requires $\Theta(N^7)$ operations on a serial computer.

(v) If $u(x,t)$ is a solution to $u_t = iu_{xx}$ for $x \in \mathbb{R}$, then the 2-norm $\|u(\cdot,t)\|$ is independent of t.

(w) In a method of lines discretization of a well-posed linear IVP, having the appropriate eigenvalues fit in the appropriate stability region is sufficient but not necessary for Lax-stability.

(x) Suppose a signal that's band-limited to frequencies in the range $[-40\text{kHz}, 40\text{kHz}]$ is sampled 60,000 times a second, i.e., fast enough to resolve frequencies in the range $[-30\text{kHz}, 30\text{kHz}]$. Then although some aliasing will occur, the information in the range $[-20\text{kHz}, 20\text{kHz}]$ remains uncorrupted.

8.7. Two final problems

EXERCISES

▶ *8.7.1. Equipotential curves.* Write a short and elegant Matlab program to plot equipotential curves in the plane corresponding to a vector of point charges (interpolation points) $x_1,\dots,x_N$. Your program should simply sample $N^{-1}\sum \log|z-x_j|$ on a grid, then produce a contour plot of the result. (See `meshdom` and `contour`.) Turn in beautiful plots corresponding to (a) 6 equispaced points, (b) 6 Chebyshev points, (c) 30 equispaced points, (d) 30 Chebyshev points. By all means play around with 3D graphics, convergence and divergence of associated interpolation processes, or other amusements if you're in the mood.

▶ *8.7.2. Fun with Chebyshev spectral methods.* The starting point of this problem is the Chebyshev differentiation matrix of Exercise 8.3.1. It will be easiest to use a program like CDERIVM from that exercise, which works with an explicit matrix rather than the FFT. Be careful with boundary conditions; you will want to square the $(N+1)\times(N+1)$ matrix first before stripping off any rows or columns.

(a) Poisson equation in 1D. The function $u(x)=(1-x^2)e^x$ satisfies $u(\pm1)=0$ and has second derivative $u''(x)=-(1+4x+x^2)e^x$. Thus it is the solution to the boundary value problem

$$u_{xx}=-(1+4x+x^2)e^x, \quad x\in[-1,1], \quad u(\pm1)=0. \tag{1}$$

Write a little Matlab program to solve (1) by a Chebyshev spectral method and produce a plot of the computed discrete solution values ($N+1$ discrete points in $[-1,1]$) superimposed upon exact solution (a curve). Turn in the plot for $N=6$ and a table of the errors $u_{\text{computed}}(0)-u_{\text{exact}}(0)$ for $N=2,4,6,8$. What can you say about the rate of convergence?

(b) Poisson equation in 2D. Similarly, the function $u(x,y)=(1-x^2)(1-y^2)\cos(x+y)$ is the solution to the boundary value problem

$$u_{xx}+u_{yy}=\texttt{<sorry, illegible!>}, \quad x,y\in[-1,1], \quad u(x,\pm1)=u(\pm1,y)=0. \tag{2}$$

Write a Matlab program to solve (2) by a Chebyshev spectral method involving a grid of $(N-1)^2$ interior points. You may find that the Matlab command KRON comes in handy for this purpose. You don't have to produce a plot of the computed solution, but do turn in a table of $u_{\text{computed}}(0,0)-u_{\text{exact}}(0,0)$ for $N=2,4,6,8$. How does the rate of convergence look?

(c) Heat equation in 1D. Back to 1D now. Suppose you have the problem

$$u_t=u_{xx}, \quad u(\pm1,t)=0, \quad u(x,0)=(1-x^2)e^x. \tag{3}$$

At what time t_c does $\max_{x\in[-1,1]} u(x,t)$ first fall below 1? Figure out the answer to at least 2 digits of relative precision. Then describe what you would do if I asked for 12 digits.

General References

D. A. Anderson, J. C. Tannehill, and R. H. Pletcher, *Computational Fluid Mechanics and Heat Transfer,* McGraw-Hill, 1984.

J. P. Boyd, *Chebyshev & Fourier Spectral Methods,* Springer-Verlag, 1989.

J. C. Butcher, *The Numerical Analysis of Ordinary Differential Equations,* Wiley, 1987.

C. Canuto, M. Y. Hussaini, A. Quarteroni, and T. A. Zang, *Spectral Methods in Fluid Dynamics,* Springer, 1987.

R. Courant and D. Hilbert, *Methods of Mathematical Physics,* v. I and II, Wiley-Interscience, 1953.

C. W. Gear, *Numerical Initial Value Problems in Ordinary Differential Equations,* Prentice-Hall, 1971.

W. Hackbusch, *Multi-grid Methods and Applications,* Springer-Verlag, 1985.

W. Hackbusch and U. Trottenberg, *Multigrid Methods,* Lect. Notes in Math., v. 960, Springer-Verlag, 1982.

G. H. Golub and C. F. Van Loan, *Matrix Computations,* Johns Hopkins Univ. Press, 1983.

E. Hairer, S. P. Nørsett, and G. Wanner, *Solving Ordinary Differential Equations I,* Springer-Verlag, 1987.

F. John, *Partial Differential Equations,* Springer-Verlag, 1978.

H. Kreiss and J. Oliger, *Methods for the Approximate Solution of Time Dependent Problems,* Global Atm. Res. Prog. Publ. 10, GARP, 1973.

J. D. Lambert, *Computational Methods in Ordinary Differential Equations,* Wiley, 1973.

R. J. LeVeque, *Numerical Methods for Conservation Laws,* Birkhäuser-Verlag, to appear.

A. R. Mitchell and D. F. Griffiths, *Computational Methods in Partial Differential Equations,* Wiley, 1980.

P. M. Morse and H. Feshbach, *Methods of Mathematical Physics,* v. I and II, McGraw-Hill, 1953.

R. Peyret and T. D. Taylor, *Computational Methods for Fluid Flow,* Springer-Verlag, 1983.

R. D. Richtmyer and K. W. Morton, *Difference Methods for Initial-Value Problems,* 2nd ed., Wiley-Interscience, 1967.

G. A. Sod, *Numerical Methods in Fluid Dynamics,* Cambridge Univ. Press, 1985.

Journals

ACM Transactions on Mathematical Software

AIAA Journal

BIT

Communications on Pure and Applied Mathematics

IMA Journal of Numerical Analysis

Journal of Computational Physics

Journal of Computational and Applied Mathematics

Journal of Scientific Computing

Mathematics of Computation

Numerische Mathematik

SIAM Journal on Numerical Analysis

SIAM Journal on Scientific and Statistical Computing

SIAM Review

References for Chapter 1

The following list comprises most of the the standard books on the numerical solution of ordinary differential equations. Henrici, Stetter, and Dekker & Verwer represent the theoretical end of the spectrum, Shampine & Gordon the practical. The pamphlet by Sand and Østerby is a delightful compendium of plots of stability regions. Coddington & Levinson and Bender & Orszag present the mathematical theory of ordinary differential equations from the pure and applied points of view, respectively. Lambert and Gear are perhaps the best general introductions to numerical methods for o.d.e.'s, while Butcher's book is a more detailed treatise containing 100 pages of references up to 1982. The books by Keller and by Ascher, Mattheij & Russell treat boundary-value o.d.e. problems (not covered in this book). For perspective, historical notes, and sense of humor one cannot do better than Hairer, Nørsett and Wanner.

U. M. Ascher, R. M. M. Mattheij, R. D. Russell, *Numerical Solution of Boundary Value Problems for Ordinary Differential Equations,* Prentice-Hall, 1988.

C. M. Bender and S. A. Orszag, *Advanced Mathematical Methods for Scientists and Engineers,* McGraw-Hill, 1978.

J. C. Butcher, *The Numerical Analysis of Ordinary Differential Equations,* Wiley, 1987.

G. D. Byrne and A. C. Hindmarsh, "Stiff ODE solvers: A review of current and coming attractions," *J. Comp. Phys. 70* (1987), 1–62.

E. A. Coddington and N. Levinson, *Theory of Ordinary Differential Equations,* McGraw-Hill, 1955.

K. Dekker and J. G. Verwer, *Stability of Runge-Kutta Methods for Stiff Nonlinear Differential Equations,* North-Holland, 1984.

C. W. Gear, *Numerical Initial Value Problems in Ordinary Differential Equations,* Prentice-Hall, 1971.

E. Hairer, S. P. Nørsett, and G. Wanner, *Solving Ordinary Differential Equations I,* Springer, 1987.

P. Henrici, *Discrete Variable Methods in Ordinary Differential Equations,* Wiley, 1962.

H. B. Keller, *Numerical Solution of Two Point Boundary Value Problems,* SIAM, 1976.

J. D. Lambert, *Computational Methods in Ordinary Differential Equations,* Wiley, 1973.

L. Lapidus and J. H. Seinfeld, *Numerical Solution of Ordinary Differential Equations,* Academic Press, 1971.

J. Sand and O. Østerby, *Regions of Absolute Stability,* Rep. PB-102, Comp. Sci. Dept., Aarhus Univ., Denmark, 1979.

L. F. Shampine and C. W. Gear, "A user's view of solving stiff ordinary differential equations," *SIAM Review 21* (1979), 1–17.

L. F. Shampine and M. K. Gordon, *Computer Solution of Ordinary Differential Equations,* W. H. Freeman, 1975.

H. J. Stetter, *Analysis of Discretization Methods for Ordinary Differential Equations,* Springer-Verlag, 1973.

References for Chapter 2

Mathematical treatments of Fourier analysis appear in many books of analysis, partial differential equations, and applications, as well as in books devoted to Fourier analysis itself. Fourier analysis is also discussed in books in many other fields, especially digital signal processing. The following references include a sampling of various kinds. For rigorous theory, the books by Katznelson and by Stein and Weiss are both excellent and readable; the classic is by Zygmund. For discrete FFT's as they appear in complex analysis, the best source is Henrici.

Bracewell

Brigham

P. L. Butzer and R. J. Nessel, *Fourier Analysis and Approximation,* 1976.

Churchill

H. Dym and H. P McKean, *Fourier Series and Integrals,* Academic Press, 1972.

G. Edwards, *Fourier Series,* 1979.

P. Henrici, *Applied and Computational Complex Analysis, III,* Wiley, 1986 (chap. 13).

Y. Katznelson, *An Introduction to Harmonic Analysis,* Dover, 1976.

A. V. Oppenheim and R. W. Schafer, *Digital Signal Processing*, Prentice-Hall, 1975.

W. H. Press, et al., *Numerical Recipes,* Cambridge Univ. Press, 1986.

W. Rudin, *Real and Complex Analysis,* McGraw-Hill, 1974.

E. M. Stein and G. Weiss, *Introuction to Fourier Analysis on Euclidean Spaces,* Princeton Univ. Press, 1971.

I. N. Sneddon, *Fourier Transforms,* McGraw-Hill, 1951.

P. N. Swarztrauber, "Symmetric FFT's," *Math. Comput. 47,* 323–346.

A. Zygmund, *Trigonometric Series,* Cambridge University Press, 1959.

References for Chapter 5

P. Brenner, V. Thomée, and L. Wahlbin, *Besov Spaces and Applications to Difference Methods for Initial Value Problems,* Springer Lect. Notes in Math. v. 434, 1975.

R. C. Y. Chin and G. W. Hedstrom, "A dispersion analysis for difference schemes: tables of generalized Airy functions," *Math. Comp. 32* (1978), 1163–1170.

M. B. Giles and W. T. Thompkins, Jr., "Propagation and stability of wavelike solutions of finite difference equations with variable coefficients," *J. Comp. Phys. 58* (1985), 349–360.

J. Lighthill, *Waves in Fluids,* Cambridge University Press, 1978.

Shokin, book on modified equations, etc.

L. N. Trefethen, "Group velocity in finite difference schemes," *SIAM Review 24* (1982), 113–136.

L. N. Trefethen, "Dispersion, dissipation, and stability," in D. F. Griffiths and G. A. Watson, eds., *Numerical Analysis,* Longman, 1986.

R. Vichnevetsky and J. B. Bowles, *Fourier Analysis of Numerical Approximations of Hyperbolic Equations,* SIAM, 1982.

R. F. Warming and B. J. Hyett, "The modified equation approach to the stability and accuracy analysis of finite-difference methods," *J. Comp. Phys. 14* (1974), 159–179.

G. B. Whitham, *Linear and Nonlinear Waves,* Wiley-Interscience, 1974.

References for Chapter 6

Four basic papers on well-posedness and stability for hyperbolic IBVP's:

H.-O. Kreiss, "Initial boundary value problems for hyperbolic systems," *Comm. Pure and Appl. Math. 23* (1970), 277–298. *[Well-posedness for hyperbolic IBVP's.]*

R. L. Higdon, "Initial-boundary value problems for linear hyperbolic systems," *SIAM Review 28* (1986), 177–217. *[Wave propagation interpretation of above.]*

B. Gustafsson, H.-O. Kreiss, and A. Sundström, "Stability theory of difference approximations for initial boundary value problems. II", *Math. Comput. 26* (1972), 649–686. *[Stability for hyperbolic finite difference models.]*

L. N. Trefethen, "Instability of difference models for hyperbolic initial boundary value problems," *Commun. Pure and Appl. Math. 37* (1984), 329–367. *[Wave propagation interpretation of above.]*

Additional references:

B. Engquist and A. Majda, "Absorbing boundary conditions for the numerical simulation of waves," *Math. Comp. 31* (1977), 629–651.

M. Golberg and E. Tadmor, "Convenient stability criteria for difference approximations of hyperbolic initial-boundary value problems. II," *Math. Comp. 48* (1987), 503–520.

G. A. Sod, *Numerical Methods in Fluid Dynamics,* Cambridge University Press, 1985.

References for Chapter 8

C. Canuto, M. Y. Hussaini, A. Quarteroni, and T. A. Zang, *Spectral Methods in Fluid Dynamics*, Springer-Verlag, 1988.

B. Fornberg, "On a Fourier method for the integration of hyperbolic equations," *SIAM J. Numer. Anal. 12* (1977), 509–528.

D. Gottlieb and S. A. Orszag, *Numerical Analysis of Spectral Methods: Theory and Applications*, SIAM, Philadelphia, 1977.

R. G. Voigt, D. Gottlieb, and M. Y. Hussaini, *Spectral Methods for Partial Differential Equations*, SIAM, Philadelphia, 1984.

References for Chapter 9

C. Canuto, M. Y. Hussaini, A. Quarteroni, and T. A. Zang, *Spectral Methods in Fluid Dynamics,* Springer-Verlag, 1988.

D. Gottlieb and S. A. Orszag, *Numerical Analysis of Spectral Methods: Theory and Applications,* SIAM, Philadelphia, 1977.

T. J. Rivlin, *The Chebyshev Polynomials,* Wiley, 1974.

R. G. Voigt, D. Gottlieb, and M. Y. Hussaini, *Spectral Methods for Partial Differential Equations,* SIAM, Philadelphia, 1984.

References for Chapter 8

C. Canuto, M. Y. Hussaini, A. Quarteroni, and T. A. Zang, *Spectral Methods in Fluid Dynamics*, Springer-Verlag, 1988.

B. Fornberg, "On a Fourier method for the integration of hyperbolic equations," *SIAM J. Numer. Anal. 12* (1977), 509–528.

D. Gottlieb and S. A. Orszag, *Numerical Analysis of Spectral Methods: Theory and Applications*, SIAM, Philadelphia, 1977.

R. G. Voigt, D. Gottlieb, and M. Y. Hussaini, *Spectral Methods for Partial Differential Equations*, SIAM, Philadelphia, 1984.