

# Chapter 1

## Introduction

### 1.1 Problems Leading to Partial Differential Equations

Many problems in science and engineering are modeled and analyzed using systems of partial differential equations. Realistic models involve multi-dimensional behavior, non-linearity, and irregular boundaries which are too complicated to be solved by analytic techniques. Numerical techniques provide useful approximations and now provide the dominant approach to addressing partial differential equations. Let's begin by examining examples of physical situations that involve partial differential equations.

*Example 1.1.1.* Consider the problem of determining the temperature  $T$  in a long prismatical rod of length  $L$  for times  $t > 0$  given that its temperature is known at the instant that the experiment is started ( $t = 0$ ) and that some information, *e.g.*, the temperature or the heat flux, is prescribed at the boundaries  $x = 0, L$  (Figure 1.1.1). The temperature in the rod may be determined as a function of the spatial coordinate  $x$  and the time  $t$  by solving the one-dimensional heat conduction equation

$$\rho c \frac{\partial T}{\partial t} = \frac{\partial}{\partial x} (k \frac{\partial T}{\partial x}), \quad 0 < x < L, \quad t > 0, \quad (1.1.1a)$$

subject to the initial conditions

$$T(x, 0) = T^0(x), \quad 0 \leq x \leq L, \quad (1.1.1b)$$

and appropriate boundary data, such as

$$T(0, t) = T_L, \quad \frac{\partial T(L, t)}{\partial x} = 0. \quad (1.1.1c)$$

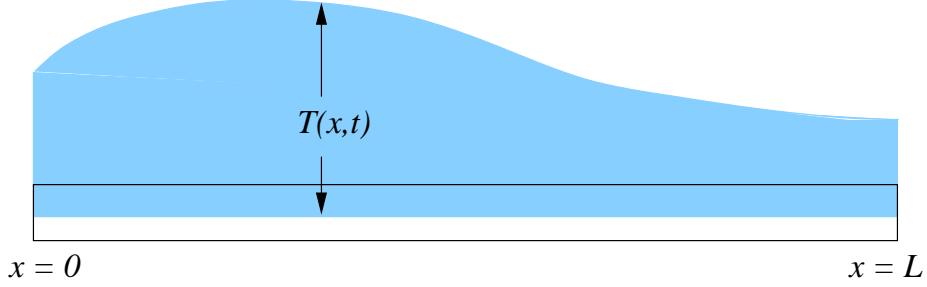


Figure 1.1.1: Geometry of the prismatical rod of Example 1.1.1.

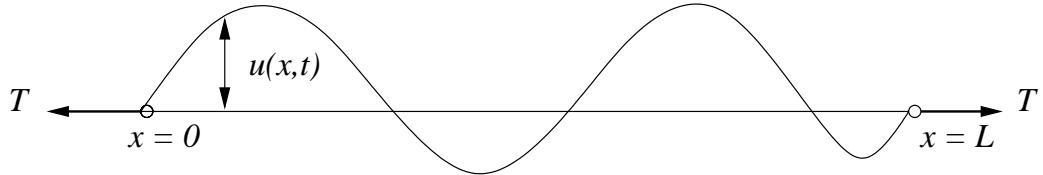


Figure 1.1.2: Geometry of the taut string of Example 1.1.2.

The parameters  $\rho$ ,  $c$ , and  $k$  are, respectively, the density, specific heat, and conductivity of the rod. Equation (1.1.1c) implies that the left end of the rod is maintained at the temperature  $T_L$  while the right end is insulated.

*Example 1.1.2.* The problem of finding the deflection  $u(x, t)$  of a taut string of length  $L$ , as shown in Figure 1.1.2, given its initial deflection and velocity is governed by

$$\rho \frac{\partial^2 u}{\partial t^2} = T \frac{\partial^2 u}{\partial x^2}, \quad 0 < x < L, \quad t > 0, \quad (1.1.2a)$$

$$u(x, 0) = u^0(x), \quad \frac{\partial u(x, 0)}{\partial t} = v^0(x), \quad 0 \leq x \leq L, \quad (1.1.2b)$$

$$u(0, t) = 0, \quad u(L, t) = 0, \quad t > 0. \quad (1.1.2c)$$

Here,  $T$  is the tension in the string,  $\rho$  is the string's density per unit length,  $u^0(x)$  is the initial deflection, and  $v^0(x)$  is the initial velocity.

*Example 1.1.3.* The problem of determining the stresses in a prismatical shaft that is subjected to a torque  $T$  (Figure 1.1.3) is governed by

$$\nabla^2 \psi \equiv \frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} = -2, \quad (x, y) \in \Omega, \quad (1.1.3a)$$

$$\psi = 0, \quad (x, y) \in \partial\Omega, \quad (1.1.3b)$$

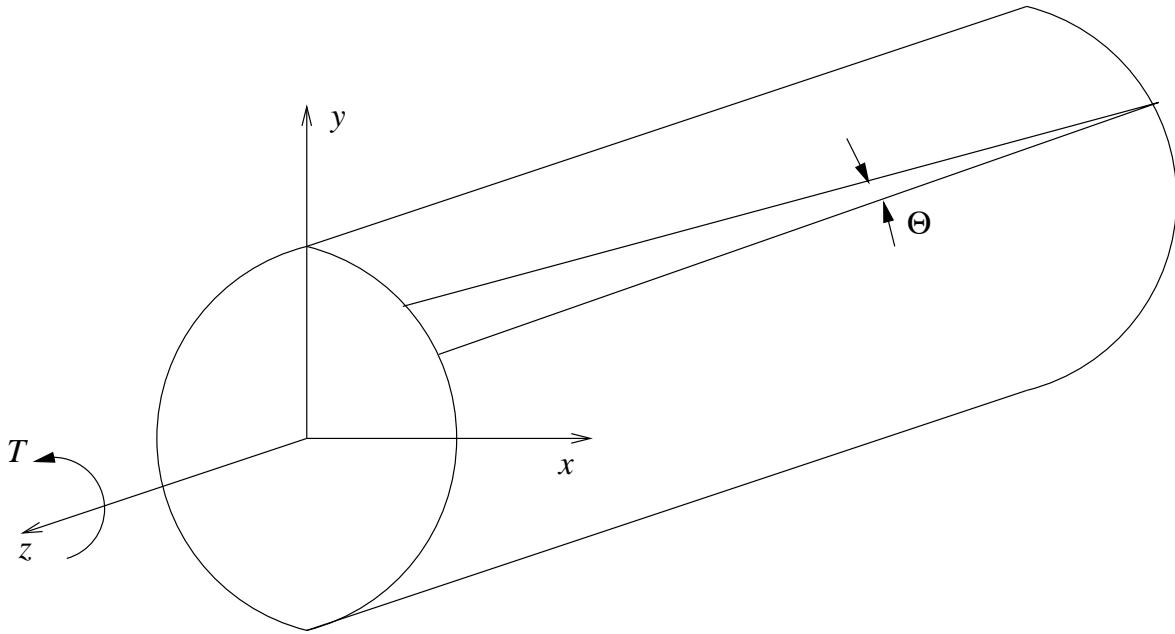


Figure 1.1.3: Geometry of the prismatical rod of Example 1.1.3.3.

$$T = 2G\Theta \int \int_{\Omega} \psi d\omega, \quad \tau_{zx} = 2G\Theta \frac{\partial \psi}{\partial x}, \quad \tau_{yz} = -2G\Theta \frac{\partial \psi}{\partial y}. \quad (1.1.3c)$$

The variable  $\psi(x, y)$  is a stress function. Once it has been determined from (1.1.3a,b), the angle of twist per unit length  $\Theta$  and the components of the shear stress  $\tau_{zx}$  and  $\tau_{yz}$  may be determined from (1.1.3c). The parameter  $G$  is the shear modulus of the rod,  $\Omega$  is the cross-sectional area of the rod, and  $\partial\Omega$  denotes its boundary. Observe that this problem is in equilibrium and, unlike those of Examples 1.1.1 and 1.1.2, it does not evolve in time.

*Example 1.1.4.* The two-dimensional flow of a compressible inviscid fluid is governed by the Euler equations

$$\rho_t + m_x + n_y = 0, \quad (1.1.4a)$$

$$m_t + \left(\frac{m^2}{\rho} + p\right)_x + \left(\frac{mn}{\rho}\right)_y = 0, \quad (1.1.4b)$$

$$n_t + \left(\frac{mn}{\rho}\right)_x + \left(\frac{n^2}{\rho} + p\right)_y = 0, \quad (1.1.4c)$$

$$e_t + [(e + p)\frac{m}{\rho}]_x + [(e + p)\frac{n}{\rho}]_y = 0. \quad (1.1.4d)$$

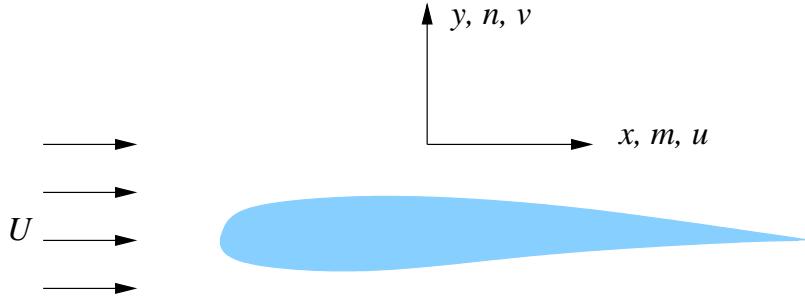


Figure 1.1.4: Typical flow region for Example 1.1.4.

Subscripts denote partial differentiation and  $\rho$ ,  $m$  and  $n$ ,  $e$ , and  $p$  are the fluid density, the  $x$  and  $y$  components of the momentum per unit volume, the total energy per unit volume, and the pressure, respectively. The  $x$  and  $y$  components of the velocity vector,  $u$  and  $v$ , are related to the components of the momentum vector by

$$m = \rho u, \quad n = \rho v. \quad (1.1.5)$$

Finally, the pressure must be related to  $\rho$ ,  $m$ ,  $n$ , and  $e$  by an equation of state, which, for a perfect gas, is

$$p = (\gamma - 1)[e - (m^2 + n^2)/2\rho], \quad (1.1.6)$$

where  $\gamma$  is a constant. Equations (1.1.4a), (1.1.4b), (1.1.4c), and (1.1.4d) express the physical conditions that the mass, the  $x$  component of momentum, the  $y$  component of momentum, and the energy of the fluid, respectively, are conserved during its motion. A typical problem involving the flow around an airfoil is shown in Figure 1.1.4. The boundary conditions for this example would state that the momentum vector must be tangent to the airfoil and that flow conditions far from the airfoil are uniform and prescribed.

## 1.2 Second-Order Partial Differential Equations

Mathematicians classify partial differential systems according to the order of their highest derivatives, the number of independent variables, and the number of dependent variables. Other properties, such as linearity, are also used for characterization. Thus, Examples 1.1.1, 1.1.2, and 1.1.3 are second-order, linear, scalar problems in two variables while Example 1.1.4 is a first-order, nonlinear, vector system in the three variables  $x$ ,  $y$ , and

*t.* We'll consider first-order problems in Section 1.3 and examine second-order problems here. In particular, we'll focus on a linear, scalar, partial differential equation in two variables having the form

$$\mathcal{L}[u] \equiv au_{xx} + 2bu_{xy} + cu_{yy} + du_x + eu_y + fu = g, \quad (1.2.1)$$

where  $a, b, \dots, g$  are continuous functions of  $x$  and  $y$  and subscripts denote partial differentiation, *e.g.*,  $u_x \equiv \partial u / \partial x$ .

Such second-order equations are classified into three types depending on the roots

$$\lambda_1, \lambda_2 = \frac{-b \pm \sqrt{b^2 - ac}}{a} \quad (1.2.2a)$$

of the characteristic equation

$$a\lambda^2 + 2b\lambda + c = 0. \quad (1.2.2b)$$

In particular, (1.2.1) is

- *hyperbolic* if  $\lambda_1, \lambda_2$  are real and distinct, *i.e.*, if  $b^2 - ac > 0$ ,
- *elliptic* if  $\lambda_1, \lambda_2$  are complex, *i.e.*, if  $b^2 - ac < 0$ , and
- *parabolic* if  $\lambda_1 = \lambda_2$ , *i.e.*, if  $b^2 - ac = 0$ .

*Example 1.2.1.* A problem can be of a different type in different spatial regions. The equation

$$xu_{xx} + u_{yy} = 0, \quad (1.2.3)$$

has  $a = x$ ,  $b = 0$ , and  $c = 1$ ; thus,  $b^2 - ac = -x$  and (1.2.3) is hyperbolic in the left-half plane ( $x < 0$ ), elliptic in the right-half plane ( $x > 0$ ), and parabolic on the line  $x = 0$ .

There are canonical partial differential equations of each type that are important for theoretical reasons and when developing practical numerical methods. The canonical hyperbolic equation is (1.2.1) with  $a = 1$ ,  $c = -1$ ,  $b = d = e = f = g = 0$ , or

$$u_{xx} - u_{yy} = 0. \quad (1.2.4)$$

This equation is called the *one-dimensional wave equation*. Usually one of the independent variables, say  $y$ , corresponds to time. Initial and/or boundary conditions have to be specified for the solution to be uniquely determined. An *initial value* or *Cauchy* problem for the wave equation consists of finding  $u$  satisfying (1.2.4) on  $y > 0$ ,  $-\infty < x < \infty$ , given the initial conditions

$$u(x, 0) = \phi(x), \quad u_y(x, 0) = \psi(x). \quad (1.2.5a)$$

An *initial-boundary value* problem for the wave equation consists of determining  $u$  satisfying (1.2.4) on  $y > 0$ ,  $0 < x < 1$ , given the initial conditions (1.2.5a) and the boundary conditions

$$\alpha_0 u(0, t) + \beta_0 u_x(0, t) = \gamma_0, \quad \alpha_1 u(1, t) + \beta_1 u_x(1, t) = \gamma_1. \quad (1.2.5b)$$

A boundary condition (1.2.5b) is called *Dirichlet* if  $\beta_j = 0$  for  $j = 0$  or 1; it is called *Neumann* if  $\alpha_j = 0$ ; and it is called *Robin* if  $\alpha_j$  and  $\beta_j$  are nonzero.

The canonical parabolic problem is the *one-dimensional heat conduction equation* which has  $a = 1$ ,  $e = -1$ ,  $b = c = d = f = g = 0$  in (1.2.1), or

$$u_{xx} - u_y = 0. \quad (1.2.6)$$

Again,  $y$  frequently corresponds to time. The initial value problem for the heat equation is to find  $u$  satisfying (1.2.6) on  $y > 0$ ,  $-\infty < x < \infty$ , given the initial condition

$$u(x, 0) = \phi(x). \quad (1.2.7a)$$

An initial-boundary value problem for the heat equation is to find  $u$  satisfying (1.2.6) on  $y > 0$ ,  $0 < x < 1$ , given the initial condition (1.2.7a) and the boundary conditions

$$\alpha_0 u(0, t) + \beta_0 u_x(0, t) = \gamma_0, \quad \alpha_1 u(1, t) + \beta_1 u_x(1, t) = \gamma_1. \quad (1.2.7b)$$

The canonical elliptic problem is the two-dimensional *Poisson equation* where  $a = 1$ ,  $c = 1$ ,  $b = d = e = f = 0$  in (1.2.1) to produce

$$u_{xx} + u_{yy} = g(x, y). \quad (1.2.8)$$

If the function  $g(x, y) \equiv 0$  then (1.2.8) is called *Laplace's equation*. There are no well posed (stable) initial or initial-boundary value problems for Poisson's or Laplace's equation. There are only boundary value problems, which consist of determining  $u(x, y)$  for  $(x, y) \in \Omega$  satisfying (1.2.8) given

$$\alpha u + \beta u_{\mathbf{n}} = \gamma, \quad (x, y) \in \partial\Omega, \quad (1.2.9)$$

where  $\mathbf{n}$  is a unit outward normal vector to  $\partial\Omega$  and  $\alpha$ ,  $\beta$ , and  $\gamma$  are functions of  $x$  and  $y$ . In analogy with the boundary conditions for the wave and heat equations, if  $\beta \equiv 0$  the problem is called a Dirichlet problem, if  $\alpha \equiv 0$  the problem is called a Neumann problem, and if neither  $\alpha$  nor  $\beta$  are zero the problem is called a Robin problem.

Exact solutions to each of the canonical problems are known in certain circumstances and the classical technique for obtaining them is by using Fourier series. Fourier series will also play a key role in analyzing finite difference approximations of the canonical problems, so let us illustrate Fourier's method for a simple heat conduction problem. We're not going to delve into this subject, so consult an elementary differential equations text such as Boyce and DiPrima [1] for additional information.

*Example 1.2.2.* Consider the Dirichlet problem for the heat conduction equation

$$u_t = u_{xx}, \quad 0 < x < 1, \quad t > 0, \quad (1.2.10a)$$

$$u(x, 0) = \phi(x), \quad 0 \leq x \leq 1, \quad (1.2.10b)$$

$$u(0, t) = u(1, t) = 0. \quad (1.2.10c)$$

(As noted,  $y$  in (1.2.6) frequently corresponds to time. In order to emphasize this, we have replaced the symbol  $y$  with a  $t$ .)

The homogeneous boundary conditions suggest the use of the Fourier sine series

$$u(x, t) = \sum_{k=1}^{\infty} a_k(t) \sin k\pi x, \quad (1.2.11)$$

which satisfies (1.2.10c) for all  $k > 0$ . Substitution of (1.2.11) into (1.2.10a) yields

$$\sum_{k=1}^{\infty} [\dot{a}_k + (k\pi)^2 a_k] \sin k\pi x = 0,$$

where a superimposed dot denotes time differentiation. The term in brackets must vanish if the above system is to be satisfied for all values of  $k$  and  $x$ ; thus,

$$\dot{a}_k + (k\pi)^2 a_k = 0, \quad \forall k > 0,$$

or

$$a_k(t) = a_k^0 e^{-k^2\pi^2 t}, \quad \forall k > 0.$$

Using this result in (1.2.11) gives

$$u(x, t) = \sum_{k=1}^{\infty} a_k^0 e^{-k^2\pi^2 t} \sin k\pi x. \quad (1.2.12a)$$

The constants  $a_k^0$ ,  $k > 0$ , are determined from the initial conditions (1.2.10b); thus, evaluating (1.2.12a) at  $t = 0$  yields

$$u(x, 0) = \phi(x) = \sum_{k=1}^{\infty} a_k^0 \sin k\pi x$$

Multiplying the above equation by  $\sin l\pi x$ , integrating on  $(0, 1)$ , interchanging limiting processes, and using the orthogonality properties of the sine function yields

$$\int_0^1 \phi(x) \sin l\pi x dx = \sum_{k=1}^{\infty} a_k^0 \int_0^1 \sin k\pi x \sin l\pi x dx = \frac{a_l^0}{2} \begin{cases} 1, & \text{if } k = l \\ 0, & \text{otherwise} \end{cases}.$$

Thus,

$$a_k^0 = 2 \int_0^1 \phi(x) \sin k\pi x dx, \quad \forall k > 0. \quad (1.2.12b)$$

Even without a specific initial function  $\phi(x)$  we are able to discern several properties of the solution. For example, assuming that (1.2.12a) converges, we see that (i)  $u(x, t)$  is a smooth function of  $x$  and  $t$  even if  $\phi(x)$  isn't and (ii)  $u(x, t)$  is a decreasing function of  $t$ . Incidentally, (1.2.12) converges in the  $\mathcal{L}^2$  norm (*cf.* (1.2.14)) under rather general conditions [3].

To be more specific, suppose that

$$\phi(x) = \begin{cases} 2x, & 0 \leq x \leq 1/2 \\ 2(1-x), & 1/2 \leq x \leq 1 \end{cases}.$$

Then, (1.2.12b) implies

$$a_k^0 = 2 \left[ \int_0^{1/2} 2x \sin k\pi x dx + \int_{1/2}^1 2(1-x) \sin k\pi x dx \right] = \frac{8}{(k\pi)^2}$$

and (1.2.12a) becomes

$$u(x, t) = \sum_{k=1}^{\infty} \frac{8}{(k\pi)^2} e^{-k^2\pi^2 t} \sin k\pi x. \quad (1.2.13)$$

With this initial data, (1.2.13) is formally converging as  $O(1/k^2)$ . In general, the Fourier series converges as  $O(1/k^{\sigma+2})$  if  $\phi(x) \in C^\sigma$ . Thus, even (some) discontinuous data leads to convergent Fourier series [3].

It will not be possible to obtain Fourier series or other explicit solutions to practical (variable-coefficient or nonlinear) problems. Nevertheless, we can obtain information about the solution without much effort. Let us, for example, obtain an estimate of the behavior of the solution in the  $\mathcal{L}^2$  norm, which, for (1.2.10), is defined as

$$\|u(\cdot, t)\|_2 \equiv \left[ \int_0^1 u(x, t)^2 dx \right]^{1/2}. \quad (1.2.14)$$

Multiplying (1.2.10a) by  $u$  and integrating on  $(0, 1)$ , we find

$$\int_0^1 uu_t dx = \int_0^1 uu_{xx} dx.$$

This may be written as

$$\frac{1}{2} \frac{d}{dt} \int_0^1 u^2 dx = \int_0^1 (uu_x)_x dx - \int_0^1 u_x^2 dx$$

or, using (1.2.14)

$$\frac{1}{2} \frac{d}{dt} \|u(\cdot, t)\|_2^2 = uu_x|_0^1 - \int_0^1 u_x^2 dx.$$

Using the boundary conditions (1.2.10c)

$$\frac{d}{dt} \|u(\cdot, t)\|_2^2 = -2 \int_0^1 u_x^2 dx.$$

Since the integral on the right is non-positive, the  $\mathcal{L}^2$  norm of the solution is a decreasing function of  $t$ . The decrease is due to the presence of the  $u_{xx}$  term in (1.2.10a), which we call *dissipative*.

## Problems

- ([1], Section 10.7.) Find the Fourier-series solution of Laplace's equation

$$u_{xx} + u_{yy} = 0$$

on the rectangle  $0 < x < a$ ,  $0 < y < b$ , satisfying the boundary conditions

$$u(0, y) = 0, \quad u(a, y) = 0, \quad 0 < y < b,$$

$$u(x, 0) = 0, \quad u(x, b) = g(x), \quad 0 < x < a.$$

Additionally, find the solution in the particular case when

$$g(x) = \begin{cases} x, & 0 < x \leq a/2 \\ a - x, & a/2 < x < a \end{cases}.$$

2. ([5], Section 8.) Find where the following partial differential equations are hyperbolic, parabolic, and elliptic:

$$x \frac{\partial^2 u}{\partial x^2} + y \frac{\partial^2 u}{\partial x \partial y} + \frac{\partial^2 u}{\partial y^2} = 0,$$

$$x^2 \frac{\partial^2 u}{\partial y^2} - \frac{\partial^2 u}{\partial x^2} + u = 0,$$

$$x \frac{\partial^2 u}{\partial x^2} + 2 \frac{\partial^2 u}{\partial x \partial y} + y \frac{\partial^2 u}{\partial y^2} + \frac{\partial u}{\partial x} = 0.$$

3. Under suitable assumptions [2] the equations of two-dimensional, compressible, steady, inviscid flow ((1.1.4) with all time derivatives set to zero) can be reduced to the simpler system

$$(a^2 - u^2)u_x - uv(u_y + v_x) + (a^2 - v^2)v_y = 0,$$

$$v_x - u_y = 0.$$

Once again,  $u$  and  $v$  are the Cartesian coordinates of the velocity vector and  $a$  is the speed of sound

$$(a/a_0)^2 = 1 - \frac{\gamma - 1}{2} \frac{u^2 + v^2}{a_0^2},$$

with  $a_0$  being the speed of sound when  $u = v = 0$  and  $\gamma > 1$  being a constant.

Introducing a potential function  $\phi(x, y)$  such that

$$u = \phi_x, \quad v = \phi_y$$

we satisfy the second partial differential equation and “reduce” the first to the *transonic full-potential equation*

$$(a^2 - \phi_x^2)\phi_{xx} - 2\phi_x\phi_y\phi_{xy} + (a^2 - \phi_y^2)\phi_{yy} = 0.$$

Although this is a complicated nonlinear second-order differential equation, it can still be classified as hyperbolic, parabolic, or elliptic. However, with nonlinear equations, the type may depend on the unknown solution. Determine the type of the transonic full-potential equation. Express your answer in terms of the Mach number

$$M^2 = \frac{u^2 + v^2}{a^2} = \frac{\phi_x^2 + \phi_y^2}{a^2}.$$

4. Show that the  $\mathcal{L}^2$  norm of the solution of Burgers' equation

$$u_t + uu_x = \sigma u_{xx}, \quad 0 < x < 1, \quad t > 0,$$

where  $\sigma$  is a positive constant, is a decreasing function of time when the initial and boundary conditions are

$$u(x, 0) = \phi(x), \quad 0 \leq x \leq 1,$$

$$u(0, t) = u(1, t) = 0, \quad t > 0.$$

### 1.3 Hyperbolic Conservation Laws: Characteristics, Shock Waves, and Rankine-Hugoniot Conditions

A conservation laws states that the total amount of some quantity remains unchanged during the evolution of the solution according to the partial differential equation. In physical processes without dissipation, these quantities might be the total mass, momentum, and energy. In this introductory chapter, let us confine our attention to conservation laws in one space dimension which typically have the form

$$\frac{d}{dt} \int_{\alpha}^{\beta} \mathbf{u} dx = -\mathbf{f}(\mathbf{u})|_{\alpha}^{\beta} = -\mathbf{f}(\mathbf{u}(\beta, t)) + \mathbf{f}(\mathbf{u}(\alpha, t)), \quad (1.3.1)$$

where

$$\mathbf{u}(x, t) = \begin{bmatrix} u_1(x, t) \\ u_2(x, t) \\ \vdots \\ u_m(x, t) \end{bmatrix}, \quad \mathbf{f}(\mathbf{u}) = \begin{bmatrix} f_1(\mathbf{u}) \\ f_2(\mathbf{u}) \\ \vdots \\ f_m(\mathbf{u}) \end{bmatrix}. \quad (1.3.2)$$

are density and flux vectors, respectively. Equation (1.3.1) expresses the fact that the rate of change of  $\mathbf{u}$  within the “volume”  $\alpha \leq x \leq \beta$  is equal to the change in its flux through the boundaries  $x = \alpha, \beta$ .

If  $\mathbf{f}$  and  $\mathbf{u}$  are smooth functions, then (1.3.1) can be written as

$$\int_{\alpha}^{\beta} [\mathbf{u}_t + \mathbf{f}(\mathbf{u})_x] dx = 0.$$

Since this result should hold for all possible “control volumes”  $(\alpha, \beta)$ , the integrand must vanish, *i.e.*,

$$\mathbf{u}_t + \mathbf{f}(\mathbf{u})_x = 0. \quad (1.3.3)$$

*Example 1.3.1.* The Euler equations for one-dimensional compressible inviscid flows may be obtained by setting all  $y$  derivatives and  $n$  to zero in equations (1.1.4) to obtain

$$\rho_t + m_x = 0, \quad (1.3.4a)$$

$$m_t + \left( \frac{m^2}{\rho} + p \right)_x = 0, \quad (1.3.4b)$$

$$e_t + [(e + p) \frac{m}{\rho}]_x = 0. \quad (1.3.4c)$$

The pressure is determined by an equation of state which we take in the rather general form  $p = p(\rho, m, e)$ . Recall that equations (1.3.4a), (1.3.4b), and (1.3.4c) express the facts that the mass, momentum, and energy of the fluid are neither created nor destroyed and are, hence, *conserved*. We readily see that the system (1.3.4) has the form of (1.3.3) with

$$\mathbf{u} = \begin{bmatrix} \rho \\ m \\ e \end{bmatrix}, \quad \mathbf{f}(\mathbf{u}) = \begin{bmatrix} m \\ m^2/\rho + p \\ (e + p)m/\rho \end{bmatrix}. \quad (1.3.5)$$

*Example 1.3.2.* The deflection of a taut string has the form

$$u_{tt} = c^2 u_{xx}, \quad (1.3.6)$$

where  $c^2 = T/\rho$  with  $T$  being the tension and  $\rho$  being the linear density of the string (cf. (1.1.2)).

Equation (1.3.6) can be written as a first-order system in a variety of ways. Perhaps the most common approach is to let

$$u_1 = u_t, \quad u_2 = cu_x. \quad (1.3.7)$$

Differentiating with respect to  $t$  while using (1.3.6) and (1.3.7) yields

$$(u_1)_t = u_{tt} = c^2 u_{xx} = c(u_2)_x, \quad (u_2)_t = cu_{xt} = cu_{tx} = c(u_1)_x.$$

Thus, the one-dimensional wave equation has the form of (1.3.3) with

$$\mathbf{u} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}, \quad \mathbf{f}(\mathbf{u}) = \begin{bmatrix} -cu_2 \\ -cu_1 \end{bmatrix}.$$

For the remainder of this introductory section, we'll confine our attention to the scalar conservation law

$$u_t + f(u)_x = 0 \quad (1.3.8)$$

and return to vector systems of conservation laws in Chapter 6. To begin, let

$$a(u) = \frac{df(u)}{du}, \quad (1.3.9a)$$

and write (1.3.8) in the *convective form*

$$u_t + a(u)u_x = 0. \quad (1.3.9b)$$

Regard  $u(x, t)$  as a function of two variables and compute its total (directional) derivative

$$du = u_t dt + u_x dx = (u_t + \frac{dx}{dt}u_x)dt. \quad (1.3.10)$$

Let us evaluate (1.3.10) in the direction given by

$$\frac{dx}{dt} = a(u). \quad (1.3.11)$$

Then, using (1.3.9b) we see that

$$du = (u_t + au_x)dt = 0.$$

Thus,  $u(x, t)$  is constant along the curve  $dx/dt = a$ . This curve is called a *characteristic* of the differential equation (1.3.8). Characteristics can be used to construct solutions of initial or initial boundary value problems. Let us begin with an initial value problem for (1.3.8) on  $-\infty < x < \infty$ ,  $t > 0$ , with the initial condition

$$u(x, 0) = \phi(x), \quad -\infty < x < \infty. \quad (1.3.12)$$

Since  $u$  is constant along the characteristic curves, it must have the same value that it had initially. Thus,  $u = \phi(x_0) \equiv \phi_0$  along the characteristic that passes through  $(x_0, 0)$ . This characteristic satisfies the ordinary initial value problem

$$\frac{dx}{dt} = a(\phi_0) \equiv a_0, \quad t > 0, \quad x(0) = x_0. \quad (1.3.13)$$

Integrating, we see that the characteristic is the straight line

$$x = x_0 + a_0 t. \quad (1.3.14)$$

This procedure can be repeated to trace other characteristics and thereby construct the solution. The solution of (1.3.8) is implicitly given by the formula

$$u(x, t) = \phi(x - a(\phi_0)t); \quad (1.3.15)$$

however, this formula is rather obscure. Let us clarify the situation with some examples.

*Example 1.3.3.* The simplest case occurs when  $a$  is a constant and  $f(u) = au$ . All of the characteristics are parallel straight lines with slope  $1/a$ . The solution of the initial value problem (1.3.8, 1.3.12) is  $u(x, t) = \phi(x - at)$  and is, as shown in Figure 1.3.1, a wave that maintains its shape and travels with speed  $a$ .

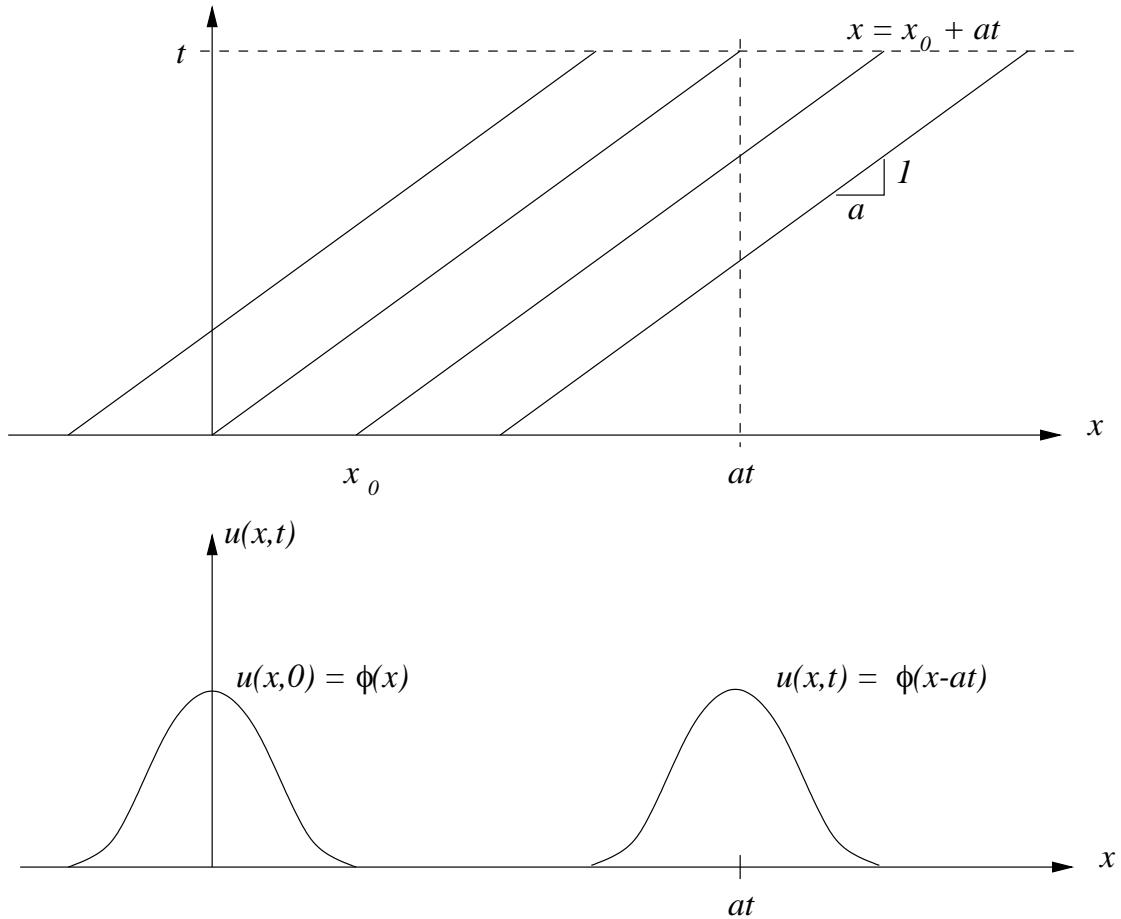


Figure 1.3.1: Characteristic curves and solution of the initial value problem (1.3.8, 1.3.12) when  $a$  is a constant.

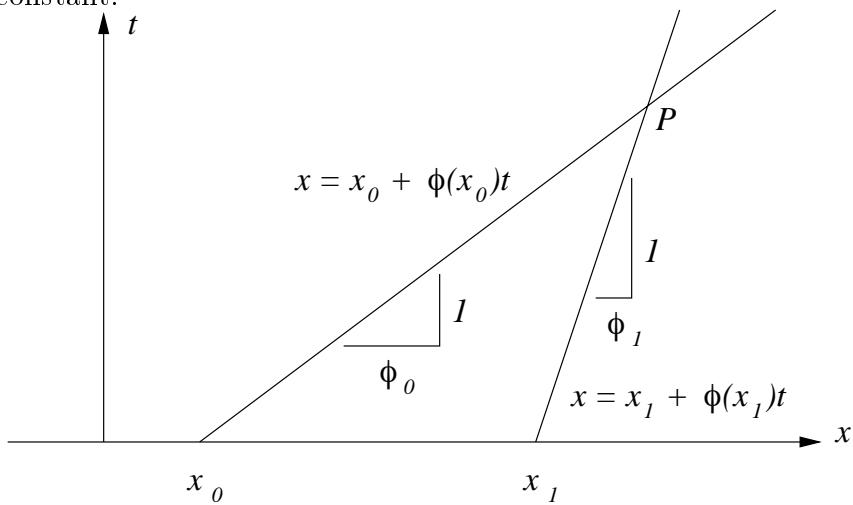


Figure 1.3.2: Characteristic curves for two initial points  $x_0$  and  $x_1$  for Burgers' equation. The characteristics intersect at a point  $P$ .

*Example 1.3.4.* Setting  $a(u) = u$  and  $f(u) = u^2/2$  in (1.3.8, 1.3.9a) yields the inviscid Burgers' equation

$$u_t + \frac{1}{2}(u^2)_x = 0. \quad (1.3.16)$$

Again, consider an initial value problem having the initial condition (1.3.12), so the characteristic is given by (1.3.14) with  $a_0 = u(x_0, 0) = \phi(x_0)$ , *i.e.*,

$$x = x_0 + \phi(x_0)t. \quad (1.3.17)$$

The characteristics are straight lines, but they are not parallel as they were in Example 1.3.3. Their slope depends on the value of the initial data; thus, the characteristic passing through the point  $(x_0, 0)$  has slope  $1/\phi(x_0)$ . The fact that the characteristics are not parallel introduces a difficulty that was not present in the linear problem of Example 1.3.3. Consider characteristics passing through  $(x_0, 0)$  and  $(x_1, 0)$  and suppose that  $\phi(x_0) > \phi(x_1)$  for  $x_1 > x_0$ . Since the slope of the characteristic passing through  $(x_0, 0)$  is less than the slope of the one passing through  $(x_1, 0)$ , the two characteristics will intersect at a point, say,  $P$  as shown in Figure 1.3.2. The solution would appear to be multivalued at points such as  $P$ .

In order to help clarify matters, let's examine the specific choice of  $\phi$  given by Lax [4]

$$\phi(x) = \begin{cases} 1, & \text{if } x < 0 \\ 1-x, & \text{if } 0 \leq x < 1 \\ 0, & \text{if } 1 \leq x \end{cases}. \quad (1.3.18)$$

Using (1.3.17), we see that the characteristic passing through the point  $(x_0, 0)$  satisfies

$$x = \begin{cases} x_0 + t, & \text{if } x_0 < 0 \\ x_0 + (1 - x_0)t, & \text{if } 0 \leq x < 1 \\ x_0, & \text{if } 1 \leq x \end{cases}. \quad (1.3.19)$$

Several characteristics are shown in Figure 1.3.3. The characteristics first intersect at  $t = 1$ . After that, the solution would presumably be multivalued, as shown in Figure 1.3.4.

It's, of course, quite possible for multivalued solutions to exist; however, (*i*) they are not observed in physical situations and (*ii*) they do not satisfy (1.3.8) in any classical sense. Discontinuous solutions are often observed in nature once characteristics of the

corresponding conservation law model have intersected. They also do not satisfy (1.3.8), but they might satisfy the integral form of the conservation law (1.3.1). We examine the simplest case when two classical solutions satisfying (1.3.8) are separated by a single smooth curve  $x = \xi(t)$  across which  $u(x, t)$  is discontinuous. For each  $t > 0$  we assume that  $\alpha < \xi(t) < \beta$  and let superscripts - and + denote conditions immediately to the left and right, respectively, of  $x = \xi(t)$ . Then, using (1.3.1), we have

$$\frac{d}{dt} \int_{\alpha}^{\beta} u dx = \frac{d}{dt} \left[ \int_{\alpha}^{\xi^-} u dx + \int_{\xi^+}^{\beta} u dx \right] = -f(u)|_{\alpha}^{\beta}$$

or, differentiating the integrals

$$\int_{\alpha}^{\xi^-} u_t dx + u^- \dot{\xi}^- + \int_{\xi^+}^{\beta} u_t dx - u^+ \dot{\xi}^+ = -f(u)|_{\alpha}^{\beta}.$$

The solution on either side of the discontinuity was assumed to be smooth, so (1.3.8) holds in  $(\alpha, \xi^-)$  and  $(\xi^+, \beta)$  and can be used to replace the integrals. Additionally, since  $\xi$  is smooth,  $\dot{\xi}^- = \dot{\xi}^+ = \dot{\xi}$ . Thus, we have

$$-f(u)|_{\alpha}^{\xi^-} + u^- \dot{\xi} - f(u)|_{\xi^+}^{\beta} - u^+ \dot{\xi} = -f(u)|_{\alpha}^{\beta},$$

or

$$\dot{\xi}(u^+ - u^-) = f(u^+) - f(u^-). \quad (1.3.20)$$

Let

$$[q] \equiv q^+ - q^- \quad (1.3.21)$$

denote the jump in a quantity  $q$  and write (1.3.20) as

$$[u]\dot{\xi} = [f(u)]. \quad (1.3.22)$$

Equation (1.3.22) is called the *Rankine-Hugoniot jump condition* and the discontinuity is called a *shock wave*. We can use the Rankine-Hugoniot condition to find a discontinuous solution of Example 1.3.4.

*Example 1.3.5.* For  $t < 1$ , the discontinuous solution of (1.3.16, 1.3.18) is as given in Example 1.3.4. For  $t \geq 1$ , we hypothesize the existence of a single shock wave, passing

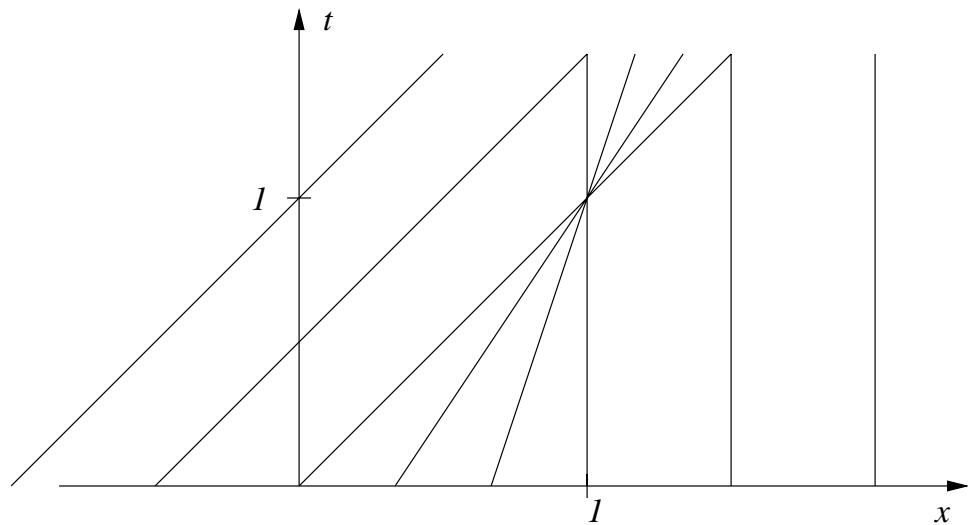


Figure 1.3.3: Characteristics for Burgers' equation (1.3.16) with initial data given by (1.3.18).

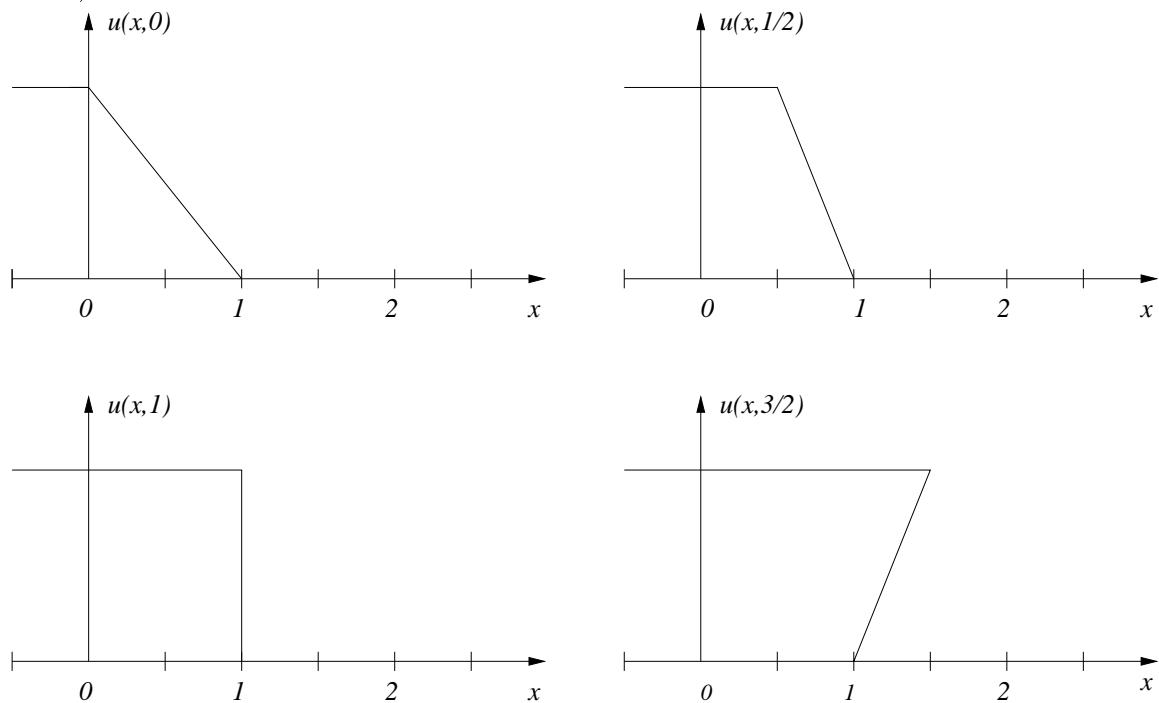


Figure 1.3.4: Multivalued solution of Burgers' equation (1.3.16) with initial data given by (1.3.18). The solution  $u(x, t)$  is shown as a function of  $x$  for  $t = 0, 1/2, 1$ , and  $3/2$ .

through  $(1, 1)$  in the  $(x, t)$ -plane. As shown in Figure 1.3.5, the solution of Example 1.3.4 can be used to infer that  $u^- = 1$  and  $u^+ = 0$ . Thus,  $f(u^-) = (u^-)^2/2 = 1/2$  and  $f(u^+) = (u^+)^2/2 = 0$ . Using (1.3.22), the velocity of the shock wave is

$$\dot{\xi} = \frac{1}{2}.$$

Integrating, we find the shock location as

$$\xi = \frac{1}{2}t + c.$$

Since the shock passes through  $(1, 1)$ , the constant of integration  $c = 1/2$ , and

$$\xi = \frac{1}{2}(t + 1). \quad (1.3.23)$$

The characteristics and shock wave are shown in Figure 1.3.5 and the solution  $u(x, t)$  is shown as a function of  $x$  for several times in Figure 1.3.6.

Let us consider another problem for Burgers' equation with different initial conditions that will illustrate another structure that arises in the solution of nonlinear hyperbolic systems.

*Example 1.3.6.* Consider Burgers' equation (1.3.16) subject to the initial conditions

$$\phi(x) = \begin{cases} 0, & \text{if } x < 0 \\ x, & \text{if } 0 \leq x < 1 \\ 1, & \text{if } 1 \leq x \end{cases}. \quad (1.3.24)$$

Using (1.3.17) and (1.3.24), we see that the characteristic passing through  $(x_0, 0)$  satisfies

$$x = \begin{cases} x_0, & \text{if } x < 0 \\ x_0(1+t), & \text{if } 0 \leq x < 1 \\ x_0 + t, & \text{if } 1 \leq x \end{cases}. \quad (1.3.25)$$

These characteristics, shown in Figure 1.3.7, may be used to verify that the solution, shown in Figure 1.3.8, is continuous. Additional considerations and difficulties with nonlinear hyperbolic systems are discussed in Lax [4].

Thus far, we have only considered initial value problems for (1.3.8). Initial-boundary problems are also possible; however, boundary conditions cannot be prescribed arbitrarily

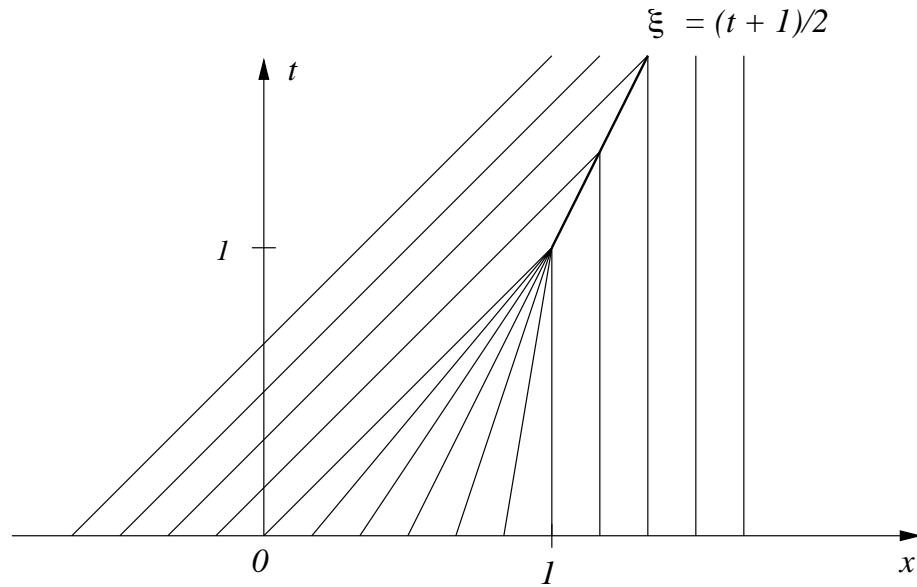


Figure 1.3.5: Characteristics and shock discontinuity for Example 1.3.5.

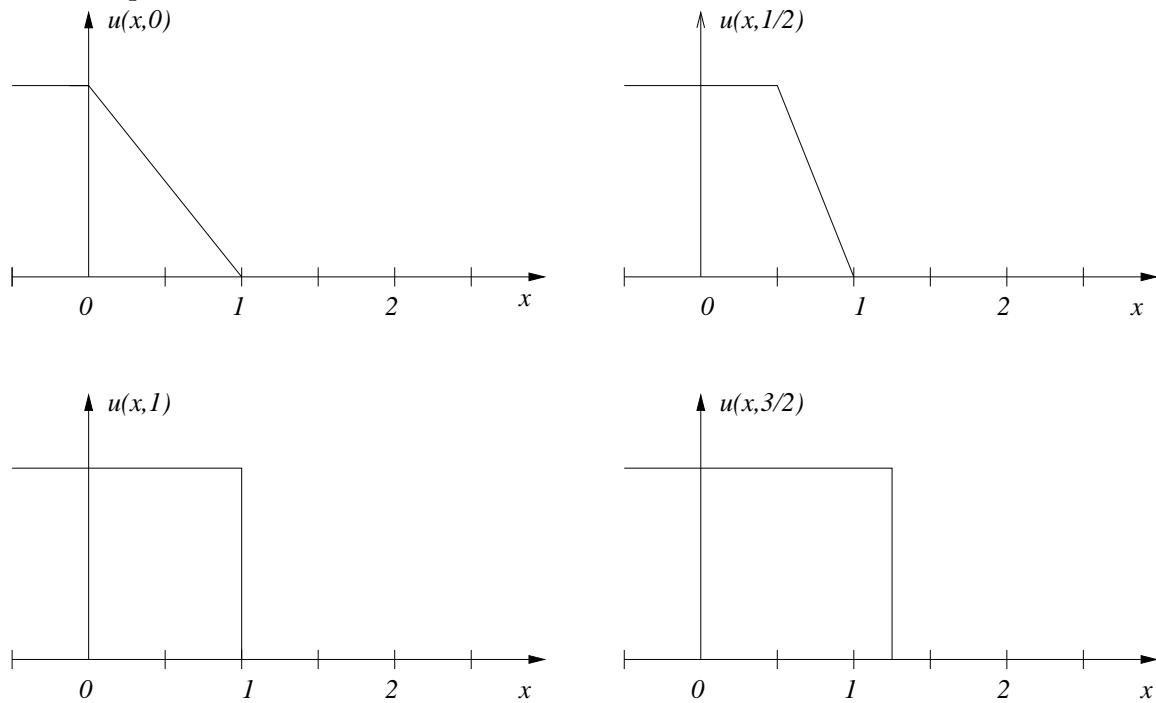


Figure 1.3.6: Solution  $u(x,t)$  of Example 1.3.5 as a function of  $x$  at  $t = 0, 1/2, 1$ , and  $3/2$ . The solution is discontinuous for  $t > 1$ .

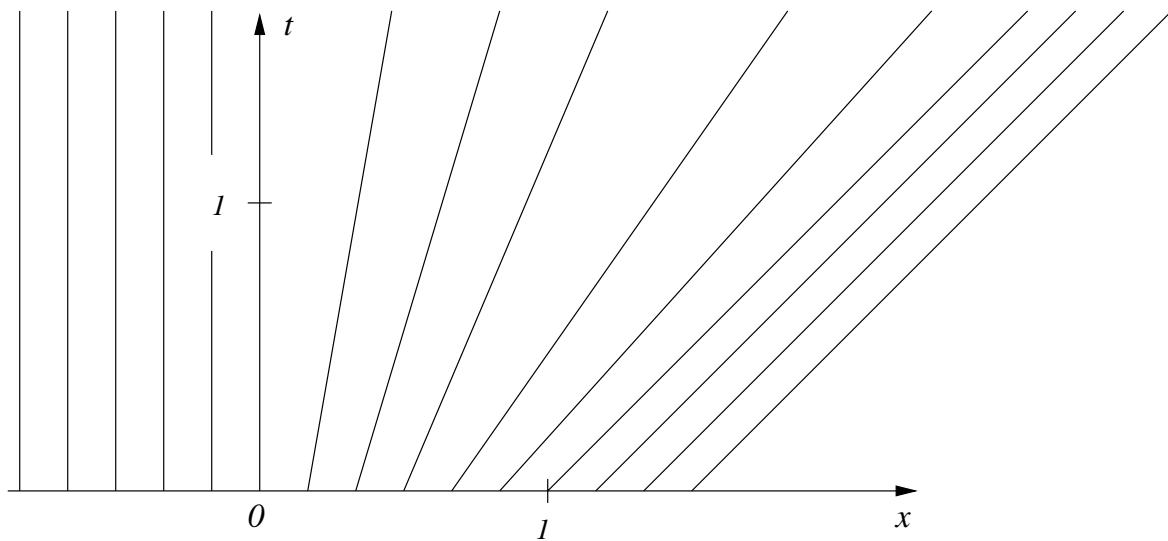
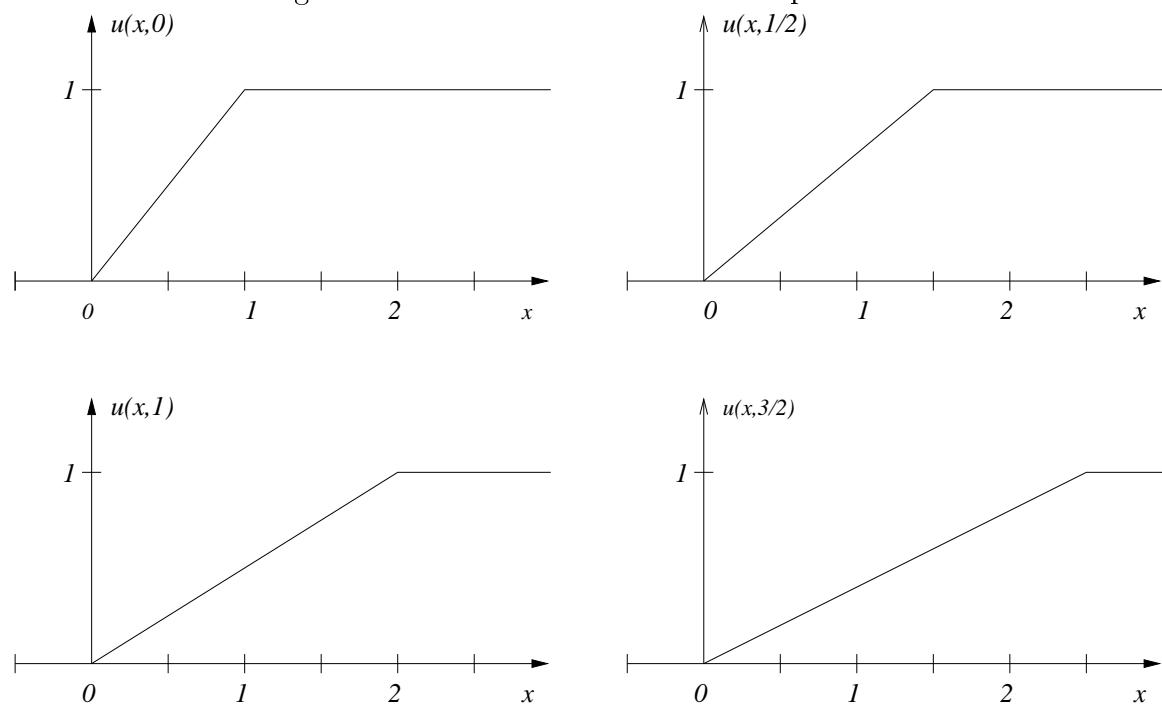


Figure 1.3.7: Characteristics for Example 1.3.6.

Figure 1.3.8: Solution  $u(x, t)$  of Example 1.3.6 as a function of  $x$  at  $t = 0, 1/2, 1$ , and  $3/2$ .

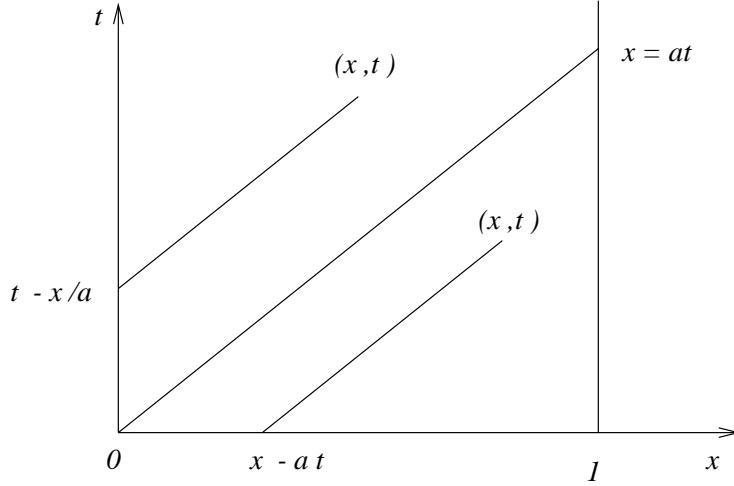


Figure 1.3.9: Characteristics for the linear initial-boundary value problem (1.3.26a).

and are determined by the characteristic directions (1.3.11). Let us begin with the linear problem

$$u_t + au_x = 0, \quad 0 < x < 1, \quad t > 0, \quad (1.3.26a)$$

$$u(x, 0) = \phi(x), \quad 0 \leq x \leq 1. \quad (1.3.26b)$$

Suppose that  $a > 0$ . As shown in Figure 1.3.9, the solution at a point  $(x_1, t_1)$  where  $x_1 > at_1$  is determined by the initial condition at  $(x_1 - at_1, 0)$ . On the other hand, the solution at a point  $(x_2, t_2)$  where  $x_2 < at_2$  is not determined by the initial condition and requires a boundary value to be prescribed at  $(0, t_2 - x_2/a)$ . Therefore, the proper boundary condition to apply is

$$u(0, t) = \psi(t), \quad t > 0. \quad (1.3.26c)$$

The solution of the initial-boundary value problem (1.3.26a) is given by

$$u(x, t) = \begin{cases} \psi(t - x/a), & \text{if } 0 < x < at \\ \phi(x - at), & \text{if } at \leq x < 1 \end{cases}. \quad (1.3.27)$$

There are no boundary conditions at  $x = 1$ . However, if  $a < 0$ , the situation would be reversed and boundary conditions would be needed at  $x = 1$  and not at  $x = 0$ .

Determining the proper boundary conditions for nonlinear problems can be quite complicated. For example, consider solving (1.3.9b) on  $0 < x < 1, t > 0$  with  $a(u(0, t)) >$

0 and  $a(u(1,t)) < 0$ . As shown on the left of Figure 1.3.10, such a problem would need boundary conditions at both  $x = 0$  and  $x = 1$ . On the other hand, a problem with  $a(u(0,t)) < 0$  and  $a(u(1,t)) > 0$  (Figure 1.3.10, right) would not require any boundary conditions.

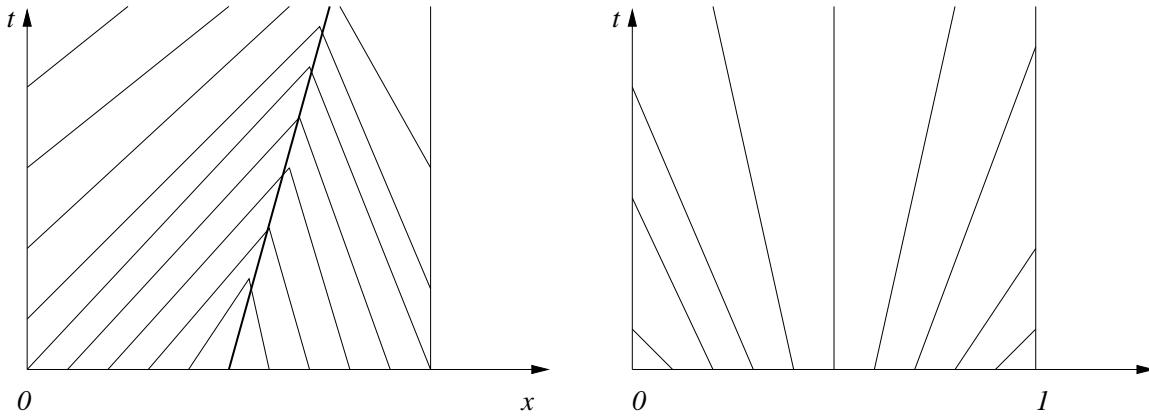


Figure 1.3.10: Characteristics for a nonlinear initial-boundary value problem with  $a(u(0,t)) > 0$  and  $a(u(1,t)) < 0$  (left) and with  $a(u(0,t)) < 0$  and  $a(u(1,t)) > 0$  (right).

## Problems

1. A *Riemann problem* is a Cauchy (initial value) problem with piecewise constant initial data. Consider the Riemann problem for the inviscid Burgers' equation

$$u_t + \frac{1}{2}(u^2)_x = 0$$

$$u(x, 0) = \begin{cases} u_L, & \text{if } x < 0 \\ u_R, & \text{if } x \geq 0 \end{cases} .$$

where  $u_L$  and  $u_R$  are constants. Solve this problem in the two cases when  $u_L > u_R$  and when  $u_L < u_R$ . Sketch several characteristic curves and sketch the solution as a function of  $x$  at a few times. As a hint, you may want to first solve a problem with continuous initial data, *e.g.*,

$$u(x, 0) = \begin{cases} u_L, & \text{if } x < -\epsilon \\ \frac{u_L}{2\epsilon}(\epsilon - x) + \frac{u_R}{2\epsilon}(\epsilon + x), & \text{if } -\epsilon < x \leq \epsilon \\ u_R, & \text{if } \epsilon < x \end{cases} ,$$

and take the limit as  $\epsilon \rightarrow 0$ . You may also consult Lax [4].

2. Use the method of characteristics to construct a solution of the inhomogeneous initial value problem

$$u_t + au_x = b(x), \quad -\infty < x < \infty, \quad t > 0,$$

$$u(x, 0) = \phi(x), \quad -\infty < x < \infty,$$

where  $a$  is a constant and  $b(x)$  is smooth.

3. A scalar conservation law having cylindrical symmetry has the form

$$\frac{d}{dt} \int_{\alpha}^{\beta} u r dr = -f(u)r|_{\alpha}^{\beta},$$

where  $r$  is a radial coordinate. If solutions are smooth this may be written as the first-order hyperbolic partial differential equation

$$u_t + \frac{1}{r}[f(u)r]_r = 0.$$

Solve an initial-boundary value problem for this equation on  $r > 0$ ,  $t > 0$  when  $f(u) = au$  with  $a$  a positive constant. Assume the initial conditions prescribe

$$u(r, 0) = \phi(r), \quad r > 0,$$

where  $\phi(r)$  is smooth. What boundary conditions, if any, must be prescribed?

4. Consider a linear vector systems of  $m$  conservation laws of the form

$$\mathbf{u}_t + \mathbf{A}\mathbf{u}_x = 0$$

where  $\mathbf{A}$  is a constant  $m \times m$  matrix. Let  $\mathbf{P}$  be the  $m \times m$  matrix that reduces  $\mathbf{A}$  to the canonical form

$$\mathbf{P}^{-1}\mathbf{A}\mathbf{P} = \mathbf{\Lambda}.$$

If  $\mathbf{A}$  has  $m$  distinct real eigenvalues then  $\mathbf{\Lambda}$  is the diagonal matrix

$$\mathbf{\Lambda} = \begin{bmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_m \end{bmatrix}$$

and the partial differential system is called *hyperbolic*. (Additionally, if  $\mathbf{A}$  has  $m$  complex eigenvalues the system is called *elliptic*. These extend the notions of hyperbolicity and ellipticity that were introduced in Section 1.2.)

4.1. Assume that the given partial differential system is hyperbolic and introduce the transformation

$$\mathbf{u} = \mathbf{P}\mathbf{v}.$$

Show that  $\mathbf{v}$  satisfies

$$\mathbf{v}_t + \Lambda\mathbf{v}_x = \mathbf{0}.$$

Thus, the transformation has uncoupled the system (except possibly for the initial and/or boundary conditions). It now has the scalar form

$$(v_i)_t + \lambda_i(v_i)_x = 0, \quad i = 1, 2, \dots, m.$$

This system may be analyzed using characteristics in the same manner as for the scalar case.

4.2. Consider the wave equation written as a first-order system of two equations with

$$\mathbf{A} = \begin{bmatrix} 0 & -c \\ -c & 0 \end{bmatrix}.$$

Find  $\mathbf{P}$ ,  $\Lambda$ , and the diagonal system satisfied by  $\mathbf{v}$ . Determine the characteristics and, hence,  $\mathbf{v}$ . Knowing  $\mathbf{v}$ , determine  $\mathbf{u}(x, t)$  such that it satisfies the initial data

$$\mathbf{u}(x, 0) = \begin{bmatrix} u_1^0(x) \\ u_2^0(x) \end{bmatrix}.$$



# Bibliography

- [1] W.E. Boyce and R.C. DiPrima (contibuter). *Elementary Differential Equations and Boundary Value Problems*. John Wiley and Sons, New York, sixth edition, 1996.
- [2] P.R. Garabedian. *Partial Differential Equations*. John Wiley and Sons, New York, 1964.
- [3] D. Gottlieb and S.A. Orszag. *Numerical Analysis of Spectral Methods: Theory and Applications*. Regional Conference Series in Applied Mathematics, No. 26. SIAM, Philadelphia, 1977.
- [4] P.D. Lax. *Hyperbolic Systems of Conservation Laws and the Mathematical Theory of Shock Waves*. Regional Conference Series in Applied Mathematics, No. 11. SIAM, Philadelphia, 1973.
- [5] H.F. Weinberger. *A First Course in Partial Differential Equations*. Ginn-Blaisdell, Waltham, 1965.

# Chapter 2

## Introduction to Finite Difference Methods

### 2.1 Constructing Difference Operators

Our initial study will involve the solution of time-dependent partial differential equations in one space variable. We'll begin by introducing some elementary finite difference operators and examining their basic properties. Following this, in Sections 2.2 and 2.3, respectively, we'll use these difference operators to solve simple wave propagation and heat conduction problems. With this plan, let us partition the upper half of the  $(x, t)$ -plane into uniform cells of size  $\Delta x \times \Delta t$  as shown in Figure 2.1.1. The grid intersection with the Cartesian coordinates  $x_j = j\Delta x$  and  $t_n = n\Delta t$  is denoted as  $(j, n)$  and the restriction of the solution of the partial differential equation to the grid is denoted as

$$u_j^n \equiv u(j\Delta x, n\Delta t). \quad (2.1.1)$$

Finite difference approximations of  $u_j^n$  are denoted as  $U_j^n$  and are obtained by replacing derivatives of  $u$  at  $x_j$  and  $t_n$  with algebraic expressions involving  $u$  at points neighboring  $(x_j, t_n)$ . Difference approximations may be constructed in a variety of ways, but the use of Taylor's formula is probably the simplest for our present purposes. To begin, let us use Taylor's formula to express  $u_{j+1}^n$  in terms of  $u_j^n$  and its derivatives as

$$\begin{aligned} u_{j+1}^n &= u_j^n + (\frac{\partial u}{\partial x})_j^n \Delta x + \frac{1}{2!} (\frac{\partial^2 u}{\partial x^2})_j^n \Delta x^2 + \dots + \frac{1}{k!} (\frac{\partial^k u}{\partial x^k})_j^n \Delta x^k \\ &\quad + \frac{1}{(k+1)!} (\frac{\partial^{k+1} u}{\partial x^{k+1}})_{j+\xi}^n \Delta x^{k+1}. \end{aligned} \quad (2.1.2)$$

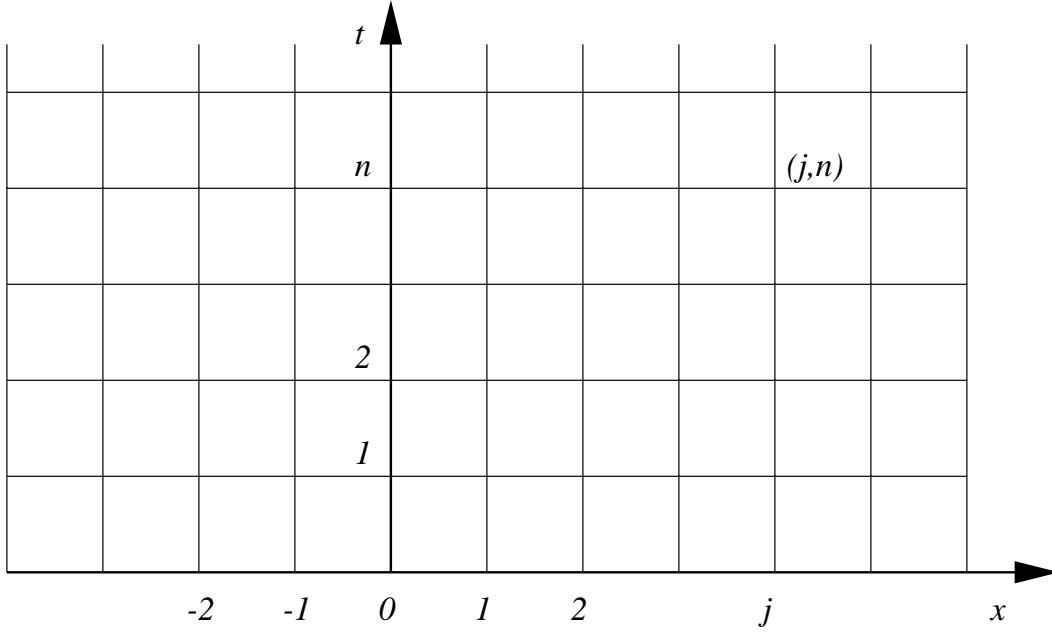


Figure 2.1.1: A partition of the upper half of the  $(x, t)$ -plane into uniform cells of size  $\Delta x \times \Delta t$ .

The last term in (2.1.2), the remainder, involves the evaluation of  $\partial u^{k+1} / \partial x^{k+1}$  at  $x = (j + \xi)\Delta x$  and  $n\Delta t$ , with  $\xi$  an unknown point on  $(0, 1)$ .

As an example, suppose that we retain the first two terms ( $k = 1$ ) in (2.1.2) and solve for  $(\partial u / \partial x)_j^n$  to obtain

$$(u_x)_j^n = \frac{u_{j+1}^n - u_j^n}{\Delta x} - \frac{1}{2}(u_{xx})_{j+\xi}^n \Delta x. \quad (2.1.3)$$

Neglecting the remainder term, we get the formula for the *first forward finite difference* approximation of  $u_x$  as

$$(U_x)_j^n = \frac{U_{j+1}^n - U_j^n}{\Delta x}. \quad (2.1.4a)$$

The neglected remainder term in (2.1.3)

$$\tau_j^n \equiv (u_x)_j^n - \frac{u_{j+1}^n - u_j^n}{\Delta x} = -\frac{1}{2}(u_{xx})_{j+\xi}^n \Delta x, \quad 0 < \xi < 1, \quad (2.1.4b)$$

is called the *local discretization error* or *local truncation error*.

Backward finite difference approximations can be developed by expanding  $u_{j-1}^n$  in a

Taylor's series about  $(x_j, t_n)$  to obtain

$$\begin{aligned} u_{j-1}^n &= u_j^n - \left(\frac{\partial u}{\partial x}\right)_j^n \Delta x + \frac{1}{2!} \left(\frac{\partial^2 u}{\partial x^2}\right)_j^n \Delta x^2 + \dots + \frac{(-1)^k}{k!} \left(\frac{\partial^k u}{\partial x^k}\right)_j^n \Delta x^k \\ &\quad + \frac{(-1)^{k+1}}{(k+1)!} \left(\frac{\partial^{k+1} u}{\partial x^{k+1}}\right)_{j-\eta}^n \Delta x^{k+1}, \quad 0 < \eta < 1. \end{aligned} \quad (2.1.5)$$

Retaining the first two terms in (2.1.5) and neglecting the remainder gives the *first backward finite difference* approximation of  $(u_x)_j^n$  as

$$(U_x)_j^n = \frac{U_j^n - U_{j-1}^n}{\Delta x}. \quad (2.1.6a)$$

The local discretization error is again obtained from the remainder term as

$$\tau_j^n = \frac{1}{2} (u_{xx})_{j-\eta}^n \Delta x, \quad 0 < \eta < 1. \quad (2.1.6b)$$

The approximations of  $(u_x)_j^n$  given by (2.1.4) and (2.1.6) are directional. A *centered finite difference* approximation of the first derivative can be obtained by retaining the first three terms ( $k = 2$ ) in (2.1.2) and (2.1.5) and subtracting the resulting expressions to get

$$(U_x)_j^n = \frac{U_{j+1}^n - U_{j-1}^n}{2\Delta x}. \quad (2.1.7a)$$

The local discretization error is obtained from the remainder terms as

$$\tau_j^n = -\frac{1}{6} (u_{xxx})_{j+\zeta}^n \Delta x^2, \quad -1 < \zeta < 1. \quad (2.1.7b)$$

The discretization error of the centered formula (2.1.7) is  $O(\Delta x^2)$  while those of the forward and backward formulas (2.1.4) and (2.1.6) are  $O(\Delta x)$ . Since the centered formula converge at a faster rate than either of the two directional formulas, it would normally be preferred; however, we shall see examples (Section 2.2) where this is not the case.

Obviously, Taylor's series can also be used to construct approximations of time derivatives. The first forward difference approximation of  $u_t$  at  $(x_j, t_n)$  is

$$(U_t)_j^n = \frac{U_j^{n+1} - U_j^n}{\Delta t}. \quad (2.1.8a)$$

The local discretization error is

$$\tau_j^n = -\frac{1}{2} (u_{tt})_j^{n+\theta} \Delta t, \quad 0 < \theta < 1. \quad (2.1.8b)$$

The same approach can be used to construct approximations of higher derivatives. For example, a centered difference approximation of  $(u_{xx})_j^n$  can be obtained by retaining the first four terms in (2.1.2) and (2.1.5) and adding the resulting expressions to get

$$(U_{xx})_j^n = \frac{U_{j+1}^n - 2U_j^n + U_{j-1}^n}{\Delta x^2}. \quad (2.1.9a)$$

The discretization error of this approximation is

$$\tau_j^n = -\frac{1}{12}(u_{xxxx})_{j+\xi}^n \Delta x^2, \quad -1 < \xi < 1, \quad (2.1.9b)$$

where the  $\xi$  of (2.1.9) is a generic symbol and has no relation to the  $\xi$  used in (2.1.3) or (2.1.4b).

Centered differences only have a higher order of accuracy than forward or backward differences on uniform grids. To see this, consider three points  $x_{j-1}$ ,  $x_j$ , and  $x_{j+1}$  of a nonuniform grid as shown in Figure 2.1.2. Let  $\Delta x_L \equiv x_j - x_{j-1}$  and  $\Delta x_R \equiv x_{j+1} - x_j$  and construct the Taylor's series expansions

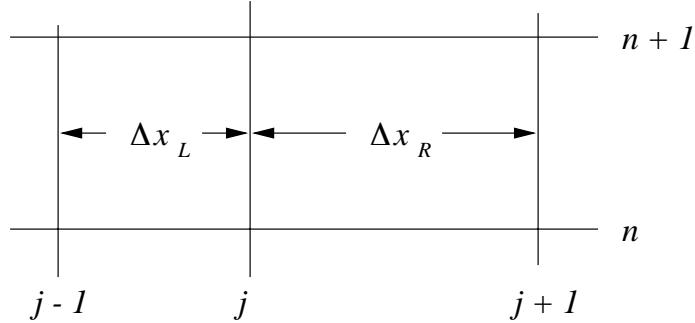


Figure 2.1.2: Two neighboring intervals of a nonuniform grid.

$$u_{j+1}^n = u_j^n + \Delta x_R (u_x)_j^n + \frac{1}{2} \Delta x_R^2 (u_{xx})_j^n + O(\Delta x_R^3), \quad (2.1.10a)$$

$$u_{j-1}^n = u_j^n - \Delta x_L (u_x)_j^n + \frac{1}{2} \Delta x_L^2 (u_{xx})_j^n + O(\Delta x_L^3). \quad (2.1.10b)$$

Divide (2.1.10a) by  $\Delta x_R$ , divide (2.1.10b) by  $\Delta x_L$ , and subtract the results to get

$$(u_x)_j^n = \frac{1}{2} \left[ \frac{u_{j+1}^n - u_j^n}{\Delta x_R} + \frac{u_j^n - u_{j-1}^n}{\Delta x_L} \right] - \frac{1}{4} (\Delta x_R - \Delta x_L) (u_{xx})_j^n + O(\Delta x^2). \quad (2.1.11a)$$

where  $\Delta x \equiv \max(\Delta x_L, \Delta x_R)$ . Let us also divide (2.1.10a) by  $\Delta x_R$ , divide (2.1.10b) by  $\Delta x_L$ , and add the results to get

$$(u_{xx})_j^n = \frac{2}{\Delta x_R + \Delta x_L} \left[ \frac{u_{j+1}^n - u_j^n}{\Delta x_R} - \frac{u_j^n - u_{j-1}^n}{\Delta x_L} \right] - \frac{1}{6} (\Delta x_R - \Delta x_L) (u_{xxx})_j^n + O(\Delta x^2). \quad (2.1.11b)$$

The approximations of  $(u_x)_j^n$  and  $(u_{xx})_j^n$  that are obtained by retaining the first terms of (2.1.11a) and (2.1.11b) are only accurate to  $O(\Delta x)$ . If  $\Delta x_L = \Delta x_R$ , the  $u_{xx}$  terms in (2.1.11a) and the  $u_{xxx}$  terms in (2.1.11b) cancel and the accuracy of both formulas is  $O(\Delta x^2)$ .

It will be convenient to have a shorthand operator notation for the finite difference operators in the same way that such notation is used for derivatives. The notation shown in Table 2.1.1 is relatively standard and will be used throughout these notes. For simplicity, we have suppressed the time dependence and only show spatial difference operators in Table 2.1.1. Thus, we have assumed that  $u$  is a function of  $x$  only with  $u_j \equiv u(x_j)$ . Temporal difference operators are defined in analogous fashion. Some examples follow.

Operator	Symbol	Definition
Forward Difference	$\Delta$	$\Delta u_j \equiv u_{j+1} - u_j$
Backward Difference	$\nabla$	$\nabla u_j \equiv u_j - u_{j-1}$
Central Difference	$\delta$	$\delta u_j \equiv u_{j+1/2} - u_{j-1/2}$
Average	$\mu$	$\mu u_j \equiv (u_{j+1/2} + u_{j-1/2})/2$
Shift	$E$	$E u_j \equiv u_{j+1}$
Derivative	$D$	$D u_j \equiv (u_x)_j$

Table 2.1.1: Definition of finite difference operators.

*Example 2.1.1.* The centered difference formula (2.1.7a) can be expressed in terms of the central difference and averaging operators  $\delta$  and  $\mu$ . Observe that

$$\mu \delta u_j = \mu(u_{j+1/2} - u_{j-1/2}) = \frac{1}{2}(u_{j+1} - u_{j-1}).$$

Thus,

$$\frac{\mu \delta u_j}{\Delta x} = \frac{u_{j+1} - u_{j-1}}{2\Delta x}.$$

*Example 2.1.2.* An operator appearing to a positive integral power is iterated; thus,

$$\delta^2 u_j = \delta(u_{j+1/2} - u_{j-1/2}) = u_{j+1} - 2u_j + u_{j-1}.$$

Thus, the centered second difference approximation (2.1.9a) of the second derivative can be written as

$$(u_{xx})_j \approx \frac{\delta^2 u_j}{\Delta x^2}.$$

*Example 2.1.3.* Let us expand  $u_{j+1}$  in a Taylor's series about  $x_j$  to obtain

$$u_{j+1} = u_j + \Delta x(u_x)_j + \frac{1}{2}\Delta x^2(u_{xx})_j + \dots,$$

or, using the derivative operator  $D$  defined in Table 2.1.1,

$$u_{j+1} = [1 + \Delta x D + \frac{1}{2}\Delta x^2 D^2 + \dots]u_j.$$

We may use the Taylor's series expansion of the exponential function and the shift operator of Table 2.1.1 to write this in the short-hand form

$$E u_j = u_{j+1} = e^{\Delta x D} u_j.$$

We can, thus, infer the identity between the shift, exponential, and derivative operators

$$E = e^{\Delta x D}. \quad (2.1.12)$$

Additional relationships can be obtained by noting that  $\Delta u_j = (E - 1)u_j$ , which implies that  $\Delta = E - 1$  or  $E = 1 + \Delta$ . Treating the operators in (2.1.12) as algebraic quantities, we find

$$\Delta x D = \ln E = \ln(1 + \Delta) = \Delta - \frac{1}{2}\Delta^2 + \frac{1}{3}\Delta^3 - \dots, \quad (2.1.13a)$$

where the series expansion of  $\ln(1+x)$ ,  $|x| < 1$ , has been used. A similar relation in terms of the backward difference operator can be constructed by noting that  $\nabla = 1 - E^{-1}$ ; thus,

$$\Delta x D = \ln E = -\ln(1 - \nabla) = \nabla + \frac{1}{2}\nabla^2 + \frac{1}{3}\nabla^3 + \dots, \quad (2.1.13b)$$

These identities can be used to derive high-order finite difference approximations of the first derivative. For example, retaining the first two terms in (2.1.13a)

$$\Delta x D u_j \approx [\Delta - \frac{1}{2}\Delta^2]u_j,$$

or

$$\Delta x D u_j \approx [(u_{j+1} - u_j) - \frac{1}{2}(u_{j+2} - 2u_{j+1} + u_j)],$$

or

$$D u_j \approx \frac{-u_{j+2} + 4u_{j+1} - 3u_j}{2\Delta x}.$$

This formula can be verified as an  $O(\Delta x^2)$  approximation of  $(u_x)_j$ .

The formal manipulations used in Examples 2.1.1 - 2.1.3 have to be verified as being rigorous. Estimates of local discretization errors must also be obtained. Nevertheless, using the formal operators of Table 2.1.1 provides a simple way of developing high-order finite difference approximations.

### Problems

- High-order centered difference approximations can be constructed by manipulating identities involving the central difference operator ([3], Chapter 1)

$$\delta u_j \equiv u_{j+1/2} - u_{j-1/2}.$$

- 1.1. Use Taylor's series expansions of a function  $u(x)$  on a uniform mesh of spacing  $\Delta x$  to show that

$$u_{j+1/2} = e^{\frac{\Delta x}{2}D} u_j, \quad u_{j-1/2} = e^{-\frac{\Delta x}{2}D} u_j$$

where  $D u_j \equiv u'(j\Delta x)$ . Use the definition of the central difference operator to infer

$$\delta = 2 \sinh \frac{\Delta x}{2} D,$$

or, inverting,

$$\Delta x D = 2 \sinh^{-1} \frac{\delta}{2} = \delta - \frac{1}{2^2 3!} \delta^3 + \frac{3^2}{2^4 5!} \delta^5 - \dots .$$

This relationship can be used to construct central difference approximations of  $u_x$ .

- 1.2. Use the result of Question 1.1 to show that

$$\Delta x^2 D^2 u_j = \Delta x^2 (u_{xx})_j = [\delta^2 - \frac{1}{12} \delta^4 + \frac{1}{90} \delta^6 + \dots] u_j.$$

Truncating this relationship gives higher-order centered difference approximations of the second derivative.

## 2.2 Simple Difference Schemes for a Kinematic Wave Equation

We have developed more than enough finite difference formulas to begin solving some simple problems. Let us begin with the kinematic wave propagation problem

$$u_t + a(u)u_x = 0, \quad -\infty < x < \infty, \quad t > 0, \quad (2.2.1a)$$

$$u(x, 0) = \phi(x), \quad -\infty < x < \infty, \quad (2.2.1b)$$

where the wave speed  $a(u)$  is a real function of  $u$ . We have neglected boundary conditions for our initial study, so, in order to have a finite spatial domain, we will require that  $\phi(x)$  either have compact support

$$\phi(x) \equiv 0, \quad |x| > X,$$

or be periodic

$$\phi(x + X) = \phi(x),$$

where  $X$  is a positive constant.

Perhaps the simplest strategy for solving (2.2.1) is to approximate both time and spatial derivatives by first-forward differences. As in Section 2.1, let us cover the half-plane  $t > 0$  by a uniform rectangular space-time mesh having cells of size  $\Delta x \times \Delta t$  (Figure 2.1.1), evaluate (2.2.1a) at  $(j\Delta x, n\Delta t)$ , and use the forward difference approximations (2.1.3, 2.1.8) to obtain

$$(u_t)_j^n + a(u_j^n)(u_x)_j^n = \frac{u_j^{n+1} - u_j^n}{\Delta t} - \frac{\Delta t}{2}(u_{tt})_j^{n+\theta} + a(u_j^n)\left[\frac{u_{j+1}^n - u_j^n}{\Delta x} - \frac{\Delta x}{2}(u_{xx})_{j+\xi}^n\right] = 0. \quad (2.2.2a)$$

Neglecting the second-order derivative terms in the local discretization errors, we obtain the finite difference equation

$$\frac{U_j^{n+1} - U_j^n}{\Delta t} + a(U_j^n) \frac{U_{j+1}^n - U_j^n}{\Delta x} = 0.$$

Solving for  $U_j^{n+1}$ , we obtain

$$U_j^{n+1} = (1 + \alpha_j^n)U_j^n - \alpha_j^n U_{j+1}^n, \quad (2.2.2b)$$

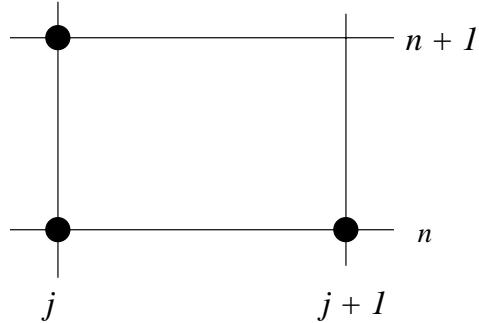


Figure 2.2.1: Computational stencil of the forward time-forward space finite difference scheme (2.2.2b).

where

$$\alpha_j^n \equiv a(U_j^n) \frac{\Delta t}{\Delta x}. \quad (2.2.3)$$

The parameter  $\alpha_j^n$  is called the *Courant number*.

The finite difference equation (2.2.2b) involves three points as indicated in the stencil of Figure 2.2.1. It is easy to solve using the prescribed initial data (2.2.1b). Knowing  $U_j^0 = \phi(j\Delta x)$  for all  $j$ , we calculate  $U_j^1$  for all  $j$  of interest. Then, knowing  $U_j^1$  for all  $j$ , we repeat the process to obtain  $U_j^2$ , etc.

Forward differences are appropriate for approximating time derivatives in this type of marching procedure, but it seems reasonable to also consider backward differences or centered differences for approximating the spatial derivatives in (2.2.1a). Thus, using (2.1.6, 2.1.8) in (2.2.1a) we obtain the forward time-backward space difference scheme

$$U_j^{n+1} = (1 - \alpha_j^n)U_j^n + \alpha_j^n U_{j-1}^n. \quad (2.2.4)$$

Using (2.1.7, 2.1.8) in (2.2.1a) yields the forward time-centered space difference scheme

$$U_j^{n+1} = U_j^n - \frac{\alpha_j^n}{2}(U_{j+1}^n - U_{j-1}^n). \quad (2.2.5)$$

These two schemes have the computational stencils shown in Figure 2.2.2. They are used in exactly the same way as the forward time-forward space scheme (2.2.2b). Each scheme has about the same computational complexity.

The question to ask at this juncture is whether or not there are any significant differences between the three schemes (2.2.2b), (2.2.4), and (2.2.5). We have not yet studied

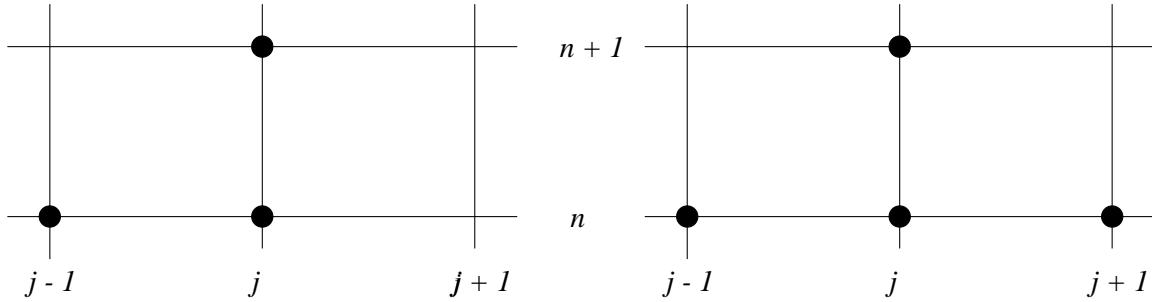


Figure 2.2.2: Computational stencils of the forward time-backward space scheme (2.2.4) (left) and the forward time-centered space scheme (2.2.5) (right).

their discretization errors; however, based on the analyses of Section 2.1, we might expect that solutions obtained by (2.2.5) have a higher order of spatial accuracy than those obtained by either (2.2.2b) or (2.2.4) (*cf.* (2.1.4b), 2.1.6b, 2.1.7b). This would be enough to abandon schemes (2.2.2b) and (2.2.4), if it were the only difference between the methods. Let's apply the methods to two simple examples.

*Example 2.2.1.* Consider (2.2.1) with  $a = 1$  and

$$\phi(x) = \begin{cases} x, & \text{if } x \leq 0 \\ 0, & \text{if } x > 0 \end{cases}.$$

The solution of this problem is easily obtained by the method of characteristics (Section 1.3) as

$$u(x, t) = \phi(x - t),$$

which is a sloping ramp moving in the positive  $x$  direction with unit speed.

Let us, rather arbitrarily, choose  $\Delta x = 1/10$  and  $\Delta t = 1/20$  and solve this problem for several spatial locations and a few time levels by the forward time-forward space and forward time-backward space finite difference schemes (2.2.2b) and (2.2.4), respectively. The Courant number  $\alpha_j^n = 1/2$  in each case. The results are shown in Tables 2.2.1 and 2.2.2 and Figure 2.2.3.

The solution obtained by the forward time-forward space scheme (2.2.2b) bears little resemblance to the exact solution. It is oscillatory and growing as the time level increases. The solution obtained by the forward time-backward space difference scheme (2.2.4) appears to be a good approximation of the exact solution. We will be more precise about the accuracy in Section 2.3, for now, we seek an explanation for the catastrophic failure

$n$	$j$									
	-4	-3	-2	-1	0	1	2	3	4	
0	-0.4	-0.3	-0.2	-0.1	0.0	0.0	0.0	0.0	0.0	
1	-0.45	-0.35	-0.25	-0.15	0.0	0.0	0.0	0.0	0.0	
2	-0.5	-0.4	-0.3	-0.225	0.0	0.0	0.0	0.0	0.0	
3	-0.55	-0.45	-0.338	-0.338	0.0	0.0	0.0	0.0	0.0	
4	-0.6	-0.506	-0.338	-0.506	0.0	0.0	0.0	0.0	0.0	
5	-0.649	-0.591	-0.253	-0.759	0.0	0.0	0.0	0.0	0.0	

Table 2.2.1: Solution of Example 2.2.1 using the forward time-forward space scheme (2.2.2b).

$n$	$j$									
	-4	-3	-2	-1	0	1	2	3	4	
0	-0.4	-0.3	-0.2	-0.1	0.0	0.0	0.0	0.0	0.0	
1	-0.45	-0.35	-0.25	-0.15	-0.05	0.0	0.0	0.0	0.0	
2	-0.5	-0.4	-0.3	-0.2	-0.1	-0.025	0.0	0.0	0.0	
3	-0.55	-0.45	-0.35	-0.25	-0.15	-0.063	-0.013	0.0	0.0	
4	-0.6	-0.5	-0.4	-0.3	-0.2	-0.106	-0.038	-0.006	0.0	
5	-0.65	-0.55	-0.45	-0.35	-0.25	-0.153	-0.072	-0.022	-0.003	

Table 2.2.2: Solution of Example 2.2.1 using the forward time-backward space scheme (2.2.4).

of the forward time-forward space scheme (2.2.2b). To rule out the possibility that the problems could be due to a lack of smoothness in the data and to further study the performance of these two schemes, let us study a second example.

*Example 2.2.2.* Consider (2.2.1) with  $a = 1$  and

$$\phi(x) = \sin x;$$

hence, the exact solution is the traveling sinusoidal wave

$$u(x, t) = \sin(x - t).$$

We'll solve this initial value problem on  $0 \leq x \leq 2\pi$  using  $\Delta x = 2\pi/16$  and  $\Delta t = 2\pi/32$ , so  $\alpha_j^n = 1/2$ . Solutions of the forward time-forward space (2.2.2b) and forward time-backward space (2.2.4) schemes are shown in Figure 2.2.4. The forward space scheme appears to be working; however, its solution is increasing in amplitude. Oscillations are not present on this time scale but would appear had we computed for longer times. (The

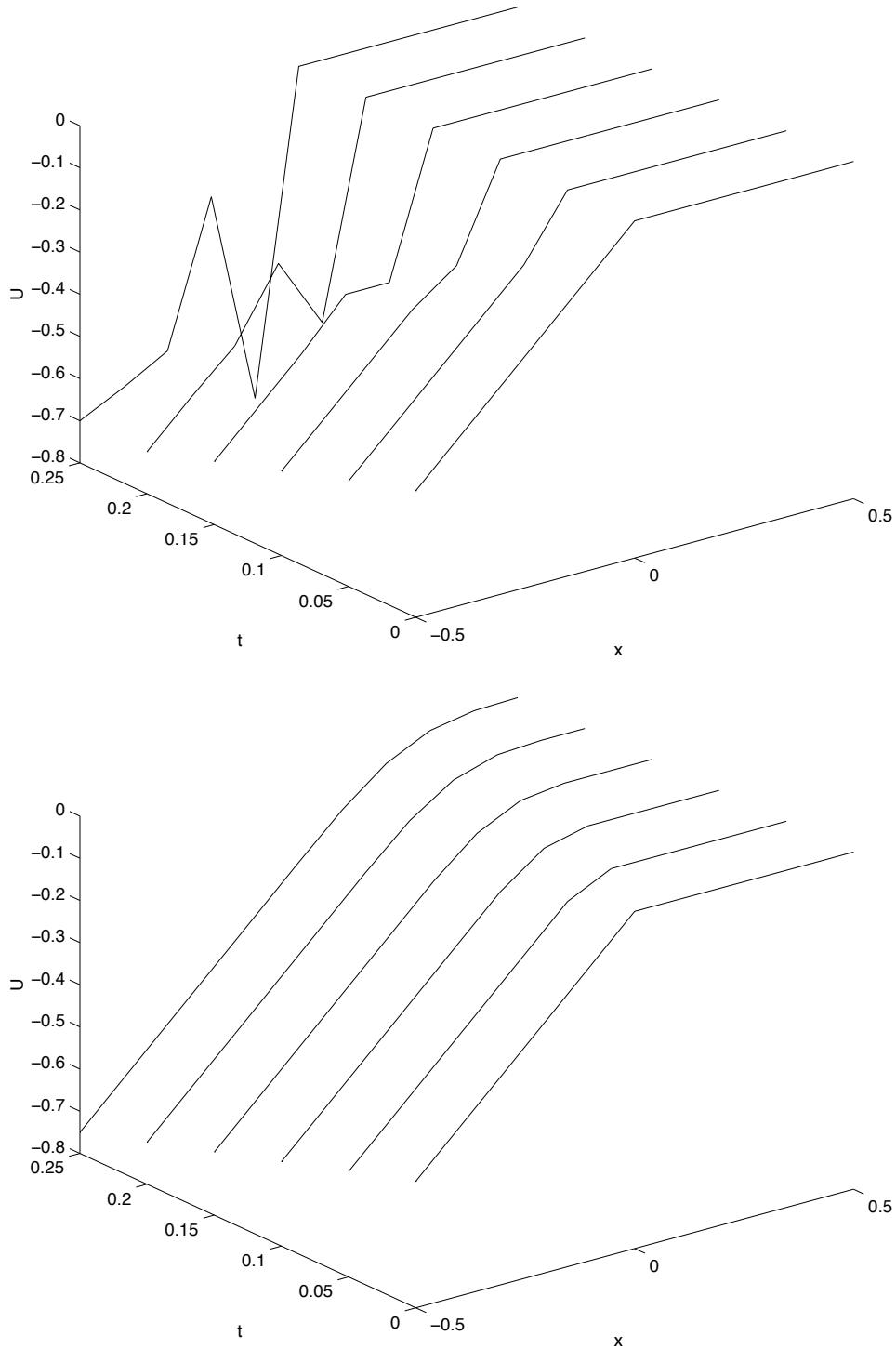


Figure 2.2.3: Solutions of Example 2.2.1 obtained by the forward time-forward space scheme (2.2.2b) (top) and forward time-backward space scheme (2.2.4) (bottom). Each solution has a Courant number of  $1/2$ .

oscillations take longer to develop with smoother initial data.) Once again, the backward space scheme is producing a reasonable approximation of the exact solution.

Let us postpone treatment of the forward time-centered space scheme (2.2.5) and seek to understand the difficulty with scheme (2.2.2b). Thus, consider (2.2.1) when  $a$  is a positive constant. The solution of (2.2.1) at a point  $(x, t)$  is determined by the initial data at the point  $(x - at, 0)$  (Figure 2.2.5).

**Definition 2.2.1.** *The domain of dependence of a point  $(x, t)$  for the initial value problem (2.2.1) is the set of all points at  $t = 0$  that determine the solution at  $(x, t)$ .*

*Remark 1.* Definition 2.2.1 is particular to (2.2.1). It will have to be amended to account for inhomogeneous equations, vector systems, and initial-boundary value problems.

As noted, the domain of dependence of the point  $(x, t)$  for the initial value problem (2.2.1) is the single point  $(x - at, 0)$ . In a similar manner, let us use Definition 2.2.1 to identify domains of dependence of the finite difference schemes (2.2.2b) and (2.2.4). The solution of the forward time-backward space scheme (2.2.4) at a point  $(j\Delta x, n\Delta t)$  is determined by the initial data on the interval  $[(j-n)\Delta x, j\Delta x]$  at  $t = 0$  (Figure 2.2.5). Thus, following Definition 2.2.1, we'll call this interval the domain of dependence of the point  $(j\Delta x, n\Delta t)$  for the difference scheme (2.2.4). The domain of dependence of the forward time-forward space scheme (2.2.2b), however, is the interval  $[j\Delta x, (j+n)\Delta x]$ , which cannot possibly be correct (Figure 2.2.5). This scheme does not use the correct initial data to determine the solution at  $(j\Delta x, n\Delta t)$ . These simple arguments lead to the famous Courant, Friedrichs, Lewy Theorem [2, 1] which we'll state in the context of (2.2.1).

**Theorem 2.2.1. (Courant, Friedrichs, Lewy).** *A necessary condition for the convergence of the solution of a finite-difference approximation to the solution of (2.2.1) for arbitrary initial data is that the domain of dependence of the finite-difference approximation contain the domain of dependence of the partial differential equation (2.2.1).*

*Remark 2.* We'll give a formal definition of convergence in the next section; however, informally, convergence implies that the solution of the difference scheme approaches the

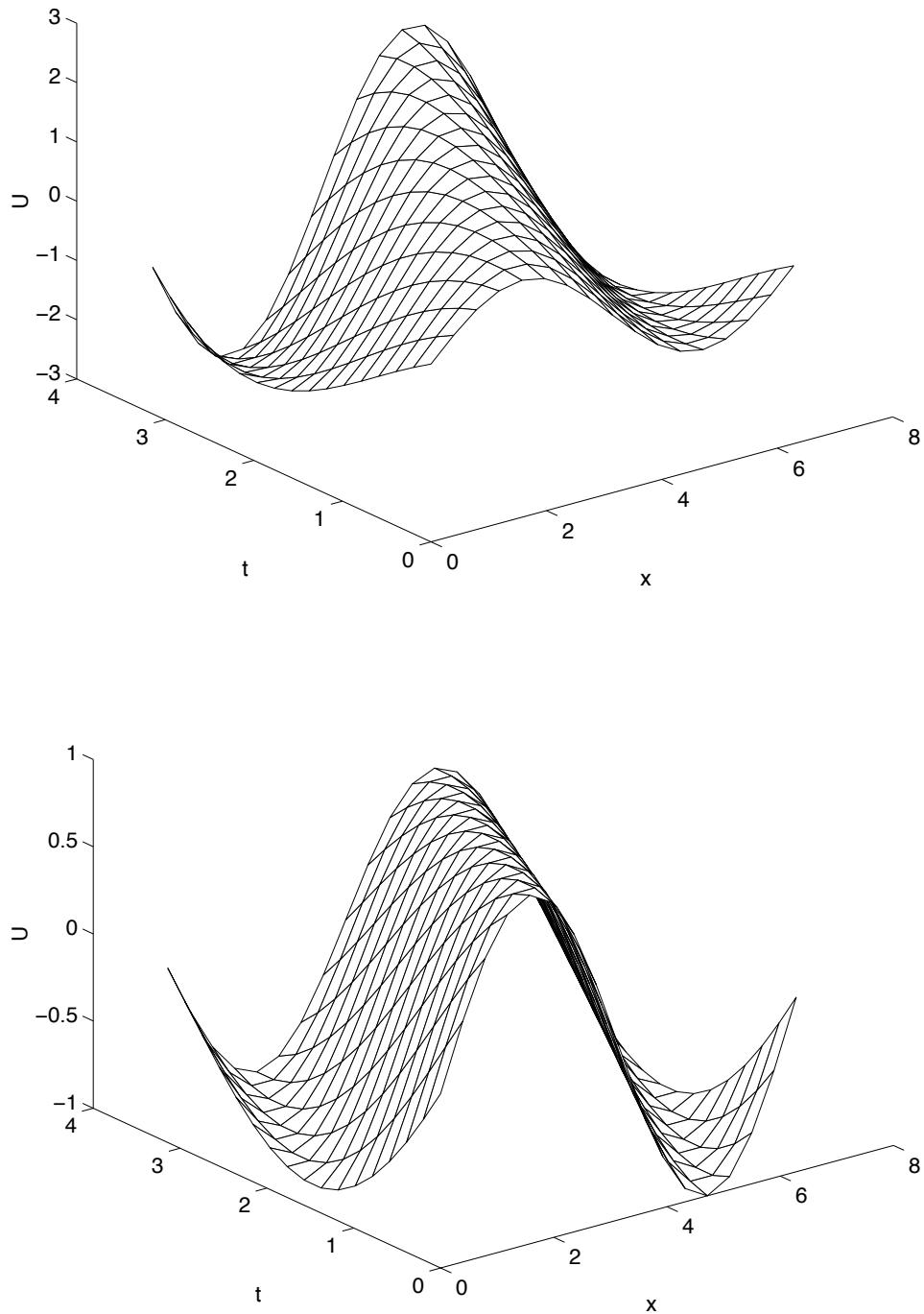


Figure 2.2.4: Solutions of Example 2.2.2 obtained by the forward time-forward space scheme (2.2.2b) (top) and forward time-backward space scheme (2.2.4) (bottom). Each solution has a Courant number of  $1/2$ .

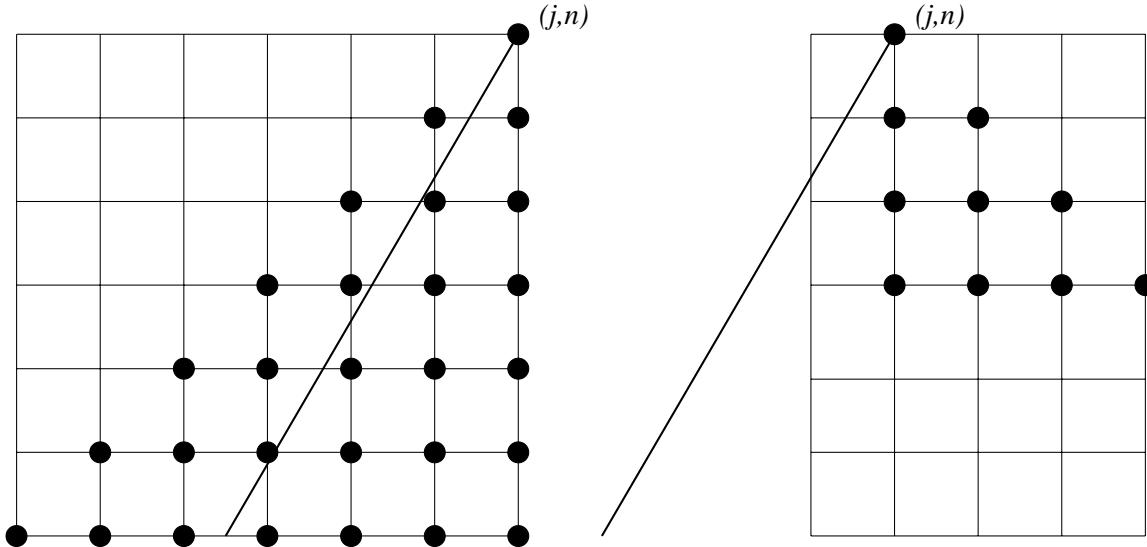


Figure 2.2.5: Domains of dependence of the solution of (2.2.1), the solution of (2.2.4) (left), and the solution of (2.2.2b) (right) for a point  $(j\Delta x, n\Delta t)$ .

solution of the partial differential equation as the space and time steps are reduced.

*Remark 3.* Theorem 2.2.1 implies that (2.2.2b) is useless when  $a$  is positive and that (2.2.4) is useless when  $a$  is negative. Thus, we should maintain  $0 \leq a \leq \Delta x / \Delta t$  for (2.2.4) and  $-\Delta x / \Delta t \leq a \leq 0$  for (2.2.2b). Stated in terms of the Courant number, we should maintain  $0 \leq \alpha_j^n \leq 1$  for (2.2.4) and  $-1 \leq \alpha_j^n \leq 0$  for (2.2.2b).

*Remark 4.* The Courant, Friedrichs, Lewy Theorem is applicable under more general circumstances than stated here. It is, furthermore, usual to state it as a stability condition rather than one on the convergence of a difference scheme. We'll consider other forms of the theorem in Chapter 6.

*Proof.* We use a straight forward contradiction argument. Since the difference scheme is required to converge for *all* initial conditions, consider space-time points where the two domains of dependence are disjoint and choose initial data that is nonzero on the domain of dependence of the partial differential equation and zero on the domain of dependence of the difference scheme. The solution of the difference scheme must be trivial at these points, whereas the solution of the partial differential equation at the same points will be nonzero. Refining the mesh so that the two domains of dependence remain disjoint does not alter this conclusion; hence, the solution of the difference equation cannot possibly

converge to that of the partial differential equation. Therefore, the domain of dependence of the difference scheme must contain that of the partial differential equation.  $\square$

### Problems

1. Suppose  $a$  is a positive constant. The scheme (2.2.4) only depends on the Courant number  $\alpha_j^n$  which, for Examples 2.2.1 and 2.2.2, is the constant  $a\Delta t/\Delta x$ . Is there a particular choice of the Courant number on  $(0, 1]$  that produces more accurate solutions than others? Answer the same question for (2.2.2b) when  $a < 0$ .
2. What restrictions, if any, should be placed on the Courant number  $\alpha_j^n$  for the forward time-centered space scheme (2.2.5) to satisfy the Courant, Friedrichs, Lewy theorem? We must, of course, study the behavior of (2.2.5); however, prior to doing this in Chapter 3, experiment by applying the method to the problems of Examples 2.2.1 and 2.2.2. Use the same mesh spacings as the earlier examples.

## 2.3 A Simple Difference Scheme for the Heat Equation

Let us determine whether similar or different phenomena occur when solving a simple initial-boundary value problem for the heat conduction equation. In particular, consider

$$u_t = \sigma u_{xx}, \quad 0 < x < 1, \quad t > 0, \quad (2.3.1a)$$

$$u(x, 0) = \phi(x), \quad 0 \leq x \leq 1, \quad (2.3.1b)$$

$$u(0, t) = u(1, t) = 0, \quad (2.3.1c)$$

where the diffusivity  $\sigma$  is positive.

In order to construct finite-difference approximations of (2.3.1): (i) introduce a uniform grid of spacing  $\Delta x = 1/J$ ,  $J > 0$ , and  $\Delta t$  on the strip  $(0, 1) \times (t > 0)$  (Figure 2.3.1); (ii) evaluate (2.3.1a) at the mesh point  $(j\Delta x, n\Delta t)$ ; and (iii) replace the partial derivatives by forward time (2.1.8) and centered space (2.1.9) differences to obtain

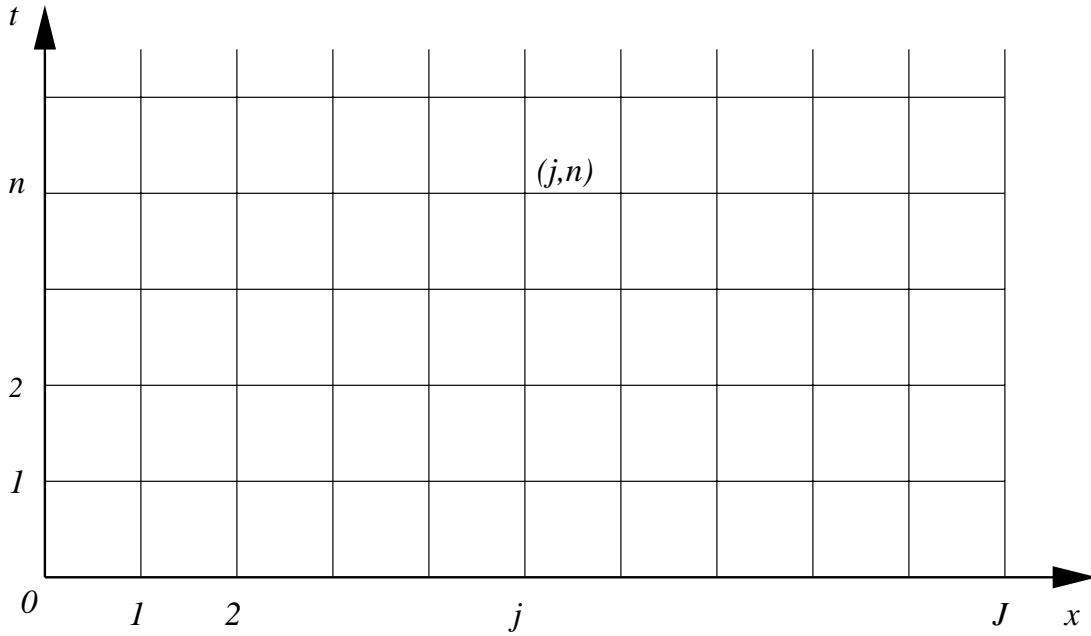


Figure 2.3.1: Computational grid used for the finite difference solution of the heat conduction problem (2.3.1).

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} - \frac{\Delta t}{2}(u_{tt})_j^{n+\theta} = \sigma \left[ \frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{\Delta x^2} - \frac{\Delta x^2}{12}(u_{xxxx})_{j+\xi}^n \right], \quad (2.3.2a)$$

where  $-1 < \xi < 1$ ,  $0 < \theta < 1$ . (With second spatial derivatives in (2.3.1) there seems little point in using forward or backward spatial derivatives and we have not done so.)

Neglecting the discretization error terms, we get the finite difference scheme

$$\frac{U_j^{n+1} - U_j^n}{\Delta t} = \sigma \frac{U_{j+1}^n - 2U_j^n + U_{j-1}^n}{\Delta x^2} \quad (2.3.2b)$$

or, solving for  $U_j^{n+1}$ ,

$$U_j^{n+1} = rU_{j-1}^n + (1 - 2r)U_j^n + rU_{j+1}^n, \quad (2.3.3a)$$

where

$$r = \sigma \frac{\Delta t}{\Delta x^2}. \quad (2.3.3b)$$

The initial and boundary conditions (2.3.1b, 2.3.1c) are

$$U_j^0 = \phi(j\Delta x), \quad j = 0, 1, \dots, J, \quad (2.3.3c)$$

$$U_0^n = U_J^n = 0, \quad n > 0. \quad (2.3.3d)$$

The computational stencil for (2.3.3a) is shown in Figure 2.3.2. The parameter  $r$  is analogous to the Courant number (2.2.3) for the kinematic wave equation (2.2.1).

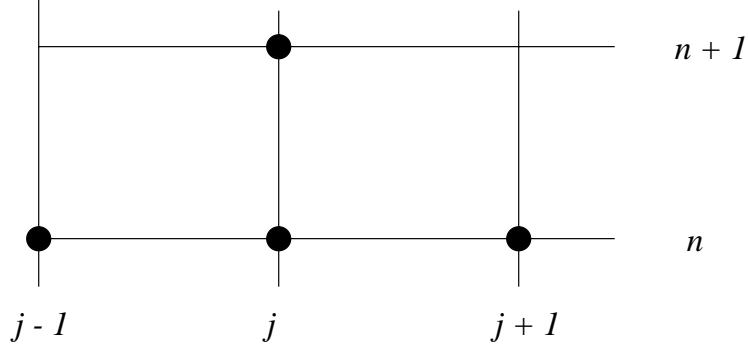


Figure 2.3.2: Computational stencil of the forward time-centered space scheme (2.3.3a) for the heat conduction equation (2.3.1a).

The solution of (2.3.3a) is obtained in the same manner as the finite difference solutions of (2.2.2b) and (2.2.4) for the kinematic wave equation (2.2.1). Thus, using the initial data (2.3.3c), we calculate a solution  $U_j^1$  at the interior mesh points,  $j = 1, 2, \dots, J - 1$ , of time level 1 using (2.3.3a) with  $j$  ranging from 1 to  $J - 1$ . The boundary conditions (2.3.3d) with  $n = 1$  determine  $U_0^1$  and  $U_J^1$ . Knowing the discrete solution at time level 1, we proceed to determine it at time level 2, *etc.* in the same manner.

*Example 2.3.1.* Let us use the forward time-centered space scheme (2.3.3a) to obtain an approximate solution of (2.3.1) with  $\sigma = 1$  and the initial data

$$\phi(x) = \begin{cases} 2x, & \text{if } 0 \leq x < 1/2 \\ 2(1-x), & \text{if } 1/2 < x \leq 1 \end{cases}.$$

In Section 1.2, we saw that the exact Fourier series solution of this problem is

$$u(x, t) = \sum_{k=1}^{\infty} \frac{8}{(k\pi)^2} e^{-k^2\pi^2 t} \sin k\pi x.$$

We'll solve this problem with  $\Delta x = 0.1$  ( $J = 10$ ) and either  $\Delta t = 0.001$  or  $0.01$ . Again, the selection of these parameters is rather arbitrary. Using (2.3.3b) with  $\Delta t = 0.001$  and  $\Delta x = 0.1$  gives  $r = 0.1$ . Similarly, with  $\Delta t = 0.01$  we find  $r = 1$ . The two solutions are

$t$	$n$	$x = 0$	0.1	0.2	0.3	0.4	0.5	0.6
		$j = 0$	1	2	3	4	5	6
0.0	0	0.0	0.2	0.4	0.6	0.8	1.0	0.8
0.001	1	0.0	0.2	0.4	0.6	0.8	0.96	0.8
0.002	2	0.0	0.2	0.4	0.6	0.796	0.928	0.796
0.003	3	0.0	0.2	0.4	0.600	0.790	0.901	0.790
0.004	4	0.0	0.2	0.4	0.599	0.782	0.879	0.782
0.005	5	0.0	0.2	0.400	0.597	0.773	0.860	0.773
0.006	6	0.0	0.2	0.400	0.595	0.764	0.842	0.764
0.007	7	0.0	0.200	0.399	0.592	0.755	0.827	0.755
0.008	8	0.0	0.200	0.399	0.589	0.746	0.813	0.746
0.009	9	0.0	0.200	0.398	0.586	0.737	0.799	0.737
0.010	10	0.0	0.200	0.397	0.582	0.728	0.787	0.728

Table 2.3.1: Solution of Example 2.3.1 using the forward time-centered space scheme (2.3.3a) with  $r = 0.1$ .

$t$	$n$	$x = 0$	0.1	0.2	0.3	0.4	0.5	0.6
		$j = 0$	1	2	3	4	5	6
0.0	0	0.0	0.2	0.4	0.6	0.8	1.0	0.8
0.01	1	0.0	0.2	0.4	0.6	0.8	0.6	0.8
0.02	2	0.0	0.2	0.4	0.6	0.4	1.0	0.4
0.03	3	0.0	0.2	0.4	0.2	1.2	-0.2	1.2
0.04	4	0.0	0.2	0.0	1.4	-1.2	2.6	-1.2

Table 2.3.2: Solution of Example 2.3.1 using the forward time-centered space scheme (2.3.3a) with  $r = 1$ .

$t$	$n$	$x = 0.3(j = 3)$	$x = 0.5(j = 5)$
		$ u_3^n - U_3^n $	$ u_5^n - U_5^n $
0.005	5	0.0008	0.023
0.01	10	0.004	0.016
0.1	100	0.011	0.012

Table 2.3.3: Errors at  $x = 0.3$  ( $j = 3$ ) and  $x = 0.5$  ( $j = 5$ ) for the solution of Example 2.3.1 using the forward time-centered space scheme (2.3.3a) with  $r = 0.1$ .

shown in Tables 2.3.1 and 2.3.2 for a few time steps. Errors of the solution with  $r = 0.1$  at  $x = 0.3$  and 0.5 are presented in Table 2.3.3 for a few times. The solutions are also shown in Figure 2.3.3.

As shown in Tables 2.3.1 and 2.3.3, the solution of (2.3.3a) with  $r = 0.1$  is producing a reasonable approximation of the exact solution. The larger errors at  $x = 0.5$  for

small times are due to the discontinuity in the initial data there. When  $r = 1$ , the finite difference solution bears little resemblance to the exact solution. As with the forward time-forward space scheme (2.2.2b) for the kinematic wave equation (2.2.1), it is oscillatory and increasing in amplitude. Apparently, some restriction must be placed on  $r$ ; however, the explanation for this restriction is not as simple as it was with (2.2.1). We'll discuss it in the next chapter.

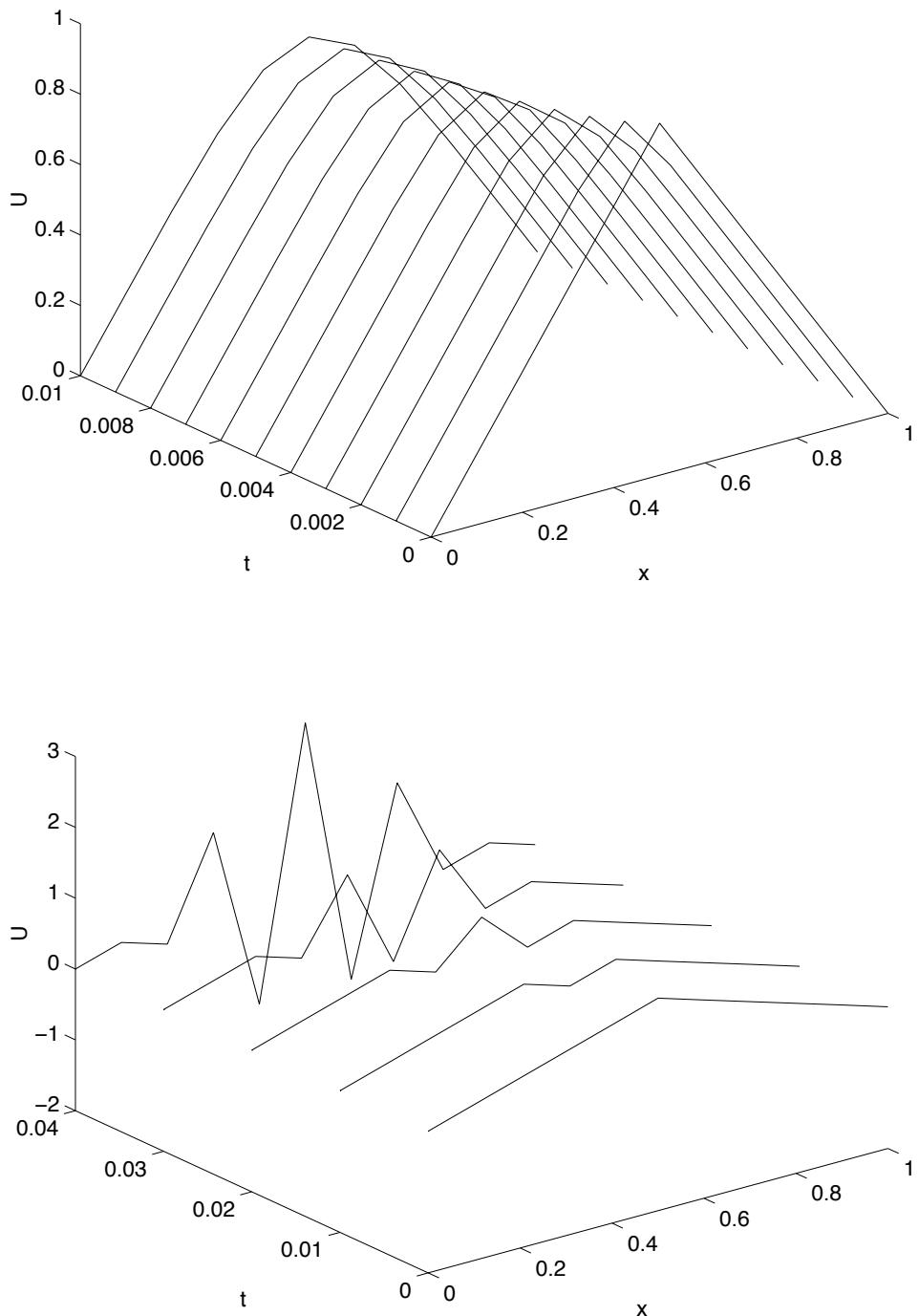


Figure 2.3.3: Solutions of Example 2.3.1 obtained by the forward time-centered space scheme (2.3.3a) with  $r = 0.1$  (top) and  $r = 1$  (bottom).



# Bibliography

- [1] R. Courant, K.O. Friedrichs, and H. Lewy. On the partial differential equations of mathematical physics. Technical Report NYO-7689, New York University, New York, 1956. Translated by P. Fox.
- [2] R. Courant, K.O. Friedrichs, and H. Lewy. Über die partiellen differenzengleichungen der mathematischen physik. *Mathematische Annalen*, 100:32–74, 1956. Also see [1].
- [3] L. Lapidus and G.F. Pinder. *Numerical Solution of Partial Differential Equations in Science and Engineering*. Wiley-Interscience, New York, 1982.

# Chapter 3

## Basic Theoretical Concepts for Time-Dependent Problems

### 3.1 Consistency, convergence, and stability

Three fundamental properties that every finite difference approximation of a partial differential equation should possess are *consistency*, *convergence*, and *stability*. Roughly speaking, consistency implies that the finite difference equation is a good approximation of the partial differential equation, convergence implies that the solution of the difference equation approaches the solution of the partial differential equation as the computational mesh is refined, and stability implies that the solution of the difference equation is not too sensitive to small perturbations in, say, the initial data. These properties are often difficult to verify for realistic problems, but they can be explained and illustrated quite easily using difference schemes for some simple model problems, such as (2.2.2b) for the kinematic wave equation (2.2.1) and (2.3.3) for the one-dimensional heat conduction equation (2.3.1).

**Definition 3.1.1.** Consider a differential equation  $\mathcal{P}u = 0$  and a finite difference approximation of it  $\mathcal{P}_\Delta U_j^n = 0$ . Let  $v(x, t)$  be any smooth function, then the local discretization or local truncation error is

$$\tau_j^n = \mathcal{P}v(j\Delta x, n\Delta t) - \mathcal{P}_\Delta v(j\Delta x, n\Delta t). \quad (3.1.1)$$

*Remark 1.*  $\mathcal{P}$  is a differential operator, e.g.,

$$\mathcal{P}v \equiv v_t - \sigma v_{xx}$$

for the heat conduction equation (2.3.1a). Similarly,  $\mathcal{P}_\Delta$  is a finite difference operator; thus, the forward time-centered space difference operator (2.3.2b) for the heat conduction operator is

$$\mathcal{P}_\Delta v \equiv \frac{v_j^{n+1} - v_j^n}{\Delta t} - \sigma \frac{v_{j+1}^n - 2v_j^n + v_{j-1}^n}{\Delta x^2}.$$

*Remark 2.* The function  $v$  is often regarded as the exact solution  $u$  of the partial differential equation. This is convenient but not necessary; thus,  $v$  may be any smooth function. This gives us a way of defining the local discretization error even when the solution of the partial differential equation has singularities.

*Example 3.1.1.* The local discretization error of the forward time-centered space difference approximation (2.3.2b) for the heat conduction equation (2.3.1a) is

$$\tau_j^n = (v_t - \sigma v_{xx})_j^n - \left( \frac{v_j^{n+1} - v_j^n}{\Delta t} - \sigma \frac{v_{j+1}^n - 2v_j^n + v_{j-1}^n}{\Delta x^2} \right)$$

or

$$\tau_j^n = (v_t)_j^n - \frac{v_j^{n+1} - v_j^n}{\Delta t} - \sigma \left[ (v_{xx})_j^n - \frac{v_{j+1}^n - 2v_j^n + v_{j-1}^n}{\Delta x^2} \right]. \quad (3.1.2a)$$

Using the Taylor's series (2.1.8, 2.1.9)

$$\tau_j^n = -\frac{\Delta t}{2}(v_{tt})_j^{n+\theta} + \frac{\sigma \Delta x^2}{12}(v_{xxxx})_{j+\xi}^n. \quad (3.1.2b)$$

If  $v$  is the solution  $u$  of the partial differential equation

$$\tau_j^n = -\frac{\Delta t}{2}(u_{tt})_j^{n+\theta} + \frac{\sigma \Delta x^2}{12}(u_{xxxx})_{j+\xi}^n. \quad (3.1.2c)$$

The local error is often used in place of the local discretization error.

**Definition 3.1.2.** *The local error is the difference  $u_j^{n+1} - U_j^{n+1}$  between solutions of the partial differential equation and its finite difference approximation assuming that no errors were committed prior to time level  $n + 1$ .*

*Example 3.1.2.* The forward time-centered space difference approximation (2.3.2b) of the heat conduction equation (2.3.1a) is

$$\frac{U_j^{n+1} - U_j^n}{\Delta t} - \sigma \frac{U_{j+1}^n - 2U_j^n + U_{j-1}^n}{\Delta x^2} = 0.$$

According to Definition 3.1.2, we commit no errors prior to time level  $n+1$ ; thus,  $U_j^n = u_j^n$ ,  $j = 0, 1, \dots, J$ , and

$$\frac{U_j^{n+1} - u_j^n}{\Delta t} - \sigma \frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{\Delta x^2} = 0.$$

Using (3.1.2a) with  $v$  replaced by  $u$  and the heat conduction equation (2.3.1a) reveals

$$\tau_j^n = -\frac{u_j^{n+1} - u_j^n}{\Delta t} + \sigma \frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{\Delta x^2}.$$

Combining the previous two relations, we obtain

$$u_j^{n+1} - U_j^{n+1} = -\Delta t \tau_j^n. \quad (3.1.3)$$

Thus, the local error is the negative of the product of the local discretization error and the time step. This relationship between local and local discretization errors generally holds for explicit finite difference schemes. Explicit difference schemes are the type that we have been discussing. The solution at time level  $n+1$  is explicitly obtained from known solution values at previous time levels. The alternative, an implicit finite difference scheme, couples other unknowns at time level  $n+1$  to  $U_j^n$ . Such schemes will be defined in Chapter 4.

**Definition 3.1.3.** A finite difference scheme  $\mathcal{P}_\Delta U_j^n = 0$  is consistent with a partial differential equation  $\mathcal{P}u = 0$  if the local discretization error tends to zero as  $\Delta x \rightarrow 0$  and  $\Delta t \rightarrow 0$ .

*Remark 3.* An equivalent definition expressed in terms of the local error would be that a difference scheme is consistent if the local error tends to zero at least to  $O(\Delta t)$  as  $\Delta x \rightarrow 0$  and  $\Delta t \rightarrow 0$ .

*Example 3.1.3.* Using (3.1.2b), we see that the forward time-centered space scheme for the heat conduction equation (2.3.1a) is consistent.

Let's emphasize the time dependence of the partial differential equation by switching notation from  $\mathcal{P}u = 0$  to

$$u_t = \mathcal{L}u, \quad (3.1.4)$$

where  $\mathcal{L}$  is a spatial differential operator, e.g.,  $\mathcal{L}u \equiv \sigma u_{xx}$  for the heat conduction equation (2.3.1a) and  $\mathcal{L}u \equiv -au_x$  for the kinematic wave equation (2.2.1a).

The solution  $u$  of the partial differential equation can be regarded as being an element of a function space. We'll not be too precise about the nature of such function spaces at the moment, but simply note that there is a positive scalar, called a norm, that is associated with many function spaces and which is used to measure the “size” of member functions and the “distances” between them.

**Definition 3.1.4.** *The norm  $\|u\|$  of a function  $u$  is a scalar that satisfies:*

1.  $\|u\| \geq 0$  and  $\|u\| = 0$  if and only if  $u = 0$ ,
2.  $\|\alpha u\| = |\alpha| \|u\|$  for any constant  $\alpha$ , and
3.  $\|u + v\| \leq \|u\| + \|v\|$ .

*Remark 4.* Condition 2 is called the condition of homogeneity and Condition 3 is the triangular inequality.

The two norms that are most suited to our purposes are the *maximum norm*

$$\|u\|_\infty \equiv \max_{x \in \Omega} |u(x, t)|, \quad (3.1.5a)$$

and the *Euclidean* or  $\mathcal{L}^2$  *norm*

$$\|u\|_2 \equiv [\int_{\Omega} u(x, t)^2 dx]^{1/2}. \quad (3.1.5b)$$

In both instances,  $\Omega$  is the spatial domain of the partial differential equation. As noted in Section 1.2, norms often give us useful estimates of the growth or decay of solutions with time.

While solutions  $u$  of the partial differential equation are regarded as elements of a function space, we may think of finite difference solutions as elements of a linear vector space. To this end, we'll collect all of the unknowns at a given time level  $n$  into a vector  $\mathbf{U}^n$ . For example, the unknowns at the time level  $n$  for the forward time-centered space scheme (2.3.2b) for the heat-conduction problem (2.3.1) are  $U_1^n, U_2^n, \dots, U_{J-1}^n$ ; hence, we may define the vector

$$\mathbf{U}^n \equiv [U_1^n, U_2^n, \dots, U_{J-1}^n]^T. \quad (3.1.6a)$$

The superscript  $T$  denotes transposition. Similarly, the unknowns at a time level  $n$  for a problem that is periodic in  $x$  on an interval  $(0, X)$  might be  $U_0^n, U_1^n, \dots, U_{J-1}^n$  and we would introduce the vector

$$\mathbf{U}^n \equiv [U_0^n, U_1^n, \dots, U_{J-1}^n]^T. \quad (3.1.6b)$$

Using this notation, we'll write the finite difference approximation as a matrix equation of the form

$$\mathbf{U}^{n+1} = \mathbf{L}_\Delta \mathbf{U}^n. \quad (3.1.7a)$$

An inhomogeneous difference equation would have the form

$$\mathbf{U}^{n+1} = \mathbf{L}_\Delta \mathbf{U}^n + \mathbf{f}^n. \quad (3.1.7b)$$

where the vector  $\mathbf{f}^n$  is independent of  $\mathbf{U}^n$ . We'll ignore this complication at present.

*Example 3.1.4.* The matrix form of the forward time-centered space difference scheme (2.3.3) for the heat-conduction problem (2.3.1) is

$$\begin{bmatrix} U_1^{n+1} \\ U_2^{n+1} \\ \vdots \\ U_{J-1}^{n+1} \end{bmatrix} = \begin{bmatrix} 1 - 2r & r & & & \\ r & 1 - 2r & r & & \\ & r & 1 - 2r & r & \\ & & & \ddots & \\ & & & r & 1 - 2r \end{bmatrix} \begin{bmatrix} U_1^n \\ U_2^n \\ \vdots \\ U_{J-1}^n \end{bmatrix}. \quad (3.1.8)$$

Elements not shown in the above matrix are zero; thus,  $\mathbf{L}_\Delta$  is a  $J - 1 \times J - 1$  tridiagonal matrix.

You may recall the definition of a linear vector space [4].

**Definition 3.1.5.**  $V$  is a linear vector space if

1.  $\mathbf{U}, \mathbf{V} \in V$  then  $\mathbf{U} + \mathbf{V} \in V$  and
2.  $\alpha$  is a scalar constant and  $\mathbf{U} \in V$  then  $\alpha\mathbf{U} \in V$ .

The properties of a norm on a vector space are identical to those for a function space, but we repeat them for completeness.

**Definition 3.1.6.** The norm  $\|\mathbf{U}\|$  of a vector  $\mathbf{U}$  is a scalar that satisfies:

1.  $\|\mathbf{U}\| \geq 0$  and  $\|\mathbf{U}\| = 0$  if and only if  $U = 0$ ,
2.  $\|\alpha\mathbf{U}\| = |\alpha|\|\mathbf{U}\|$  for any constant  $\alpha$ , and
3.  $\|\mathbf{U} + \mathbf{V}\| \leq \|\mathbf{U}\| + \|\mathbf{V}\|$ .

Again, the two norms that most suit our purposes are the maximum norm

$$\|\mathbf{U}^n\|_\infty \equiv \max_j |U_j^n|, \quad (3.1.9a)$$

and the Euclidean or  $\mathcal{L}^2$  norm

$$\|\mathbf{U}^n\|_2 \equiv [\sum_j (U_j^n)^2]^{1/2}. \quad (3.1.9b)$$

The range on  $j$  is over all components of the vector  $\mathbf{U}^n$ .

Matrix norms provide estimates of the “size” of the discrete operator  $\mathbf{L}_\Delta$  and the growth or decay of  $\mathbf{U}^n$  with  $n$ . They are typically defined by their effect on vector norms.

**Definition 3.1.7.** *The norm  $\|\mathbf{A}\|$  of a matrix  $\mathbf{A}$  induced by a vector norm is the scalar*

$$\|\mathbf{A}\| \equiv \max_{\|\mathbf{z}\| \neq 0} \frac{\|\mathbf{Az}\|}{\|\mathbf{z}\|} = \max_{\|\mathbf{z}\|=1} \|\mathbf{Az}\|. \quad (3.1.10)$$

The matrix norm satisfies the properties of a vector norm as well as

$$\|\mathbf{AB}\| \leq \|\mathbf{A}\| \|\mathbf{B}\|, \quad (3.1.11a)$$

for any matrices  $\mathbf{A}$  and  $\mathbf{B}$ .

Using (3.1.10), we immediately see that

$$\|\mathbf{Az}\| \leq \|\mathbf{A}\| \|\mathbf{z}\|. \quad (3.1.11b)$$

This inequality also follows from (3.1.11a) with  $\mathbf{B}$  replaced by the vector  $\mathbf{z}$ .

The maximum and Euclidean norms of a matrix are

$$\|\mathbf{A}\|_\infty = \max_i \sum_j |a_{ij}|, \quad \|\mathbf{A}\|_2 = \max_j |\lambda_j(\mathbf{A}^T \mathbf{A})|^{1/2}, \quad (3.1.12)$$

where  $\lambda_j$  is an eigenvalue of the matrix  $\mathbf{A}^T \mathbf{A}$ . Additional matrix norms and properties of norms appear in Strikwerda [5], Appendix A.

*Example 3.1.5.* The maximum norm of the matrix  $\mathbf{L}_\Delta$  of (3.1.8) is  $\|\mathbf{L}_\Delta\|_\infty = |1 - 2r| + 2r$ . If  $r \leq 1/2$  then  $(1 - 2r) > 0$  and  $\|\mathbf{L}_\Delta\|_\infty = 1$ .

With these preliminary considerations established, we define the concepts of convergence and stability for finite difference schemes.

**Definition 3.1.8.** *A finite difference approximation converges to the solution of a partial differential equation on  $0 < t \leq T$  in a particular vector norm if*

$$\|\mathbf{u}^n - \mathbf{U}^n\| \rightarrow 0, \quad n \rightarrow \infty, \quad \Delta x \rightarrow 0, \quad \Delta t \rightarrow 0, \quad n\Delta t \leq T. \quad (3.1.13)$$

*Remark 5.* The vector  $\mathbf{u}^n$  is the restriction of the continuous solution  $u(x, t_n)$  to the mesh.

*Remark 6.* When applying Definition 3.1.8, visualize a sequence of computations performed on  $0 < t \leq T$  using finer-and-finer meshes. Convergence implies that the discrete and continuous solutions approach each other for  $t \in (0, T]$  in a particular vector norm as the mesh spacing decreases.

**Definition 3.1.9.** *Let  $\mathbf{U}^n$  and  $\mathbf{V}^n$  satisfy homogeneous finite-difference initial value problems with different initial conditions, i.e.,*

$$\mathbf{U}^{n+1} = \mathbf{L}_\Delta \mathbf{U}^n, \quad \mathbf{U}^0 = \phi,$$

$$\mathbf{V}^{n+1} = \mathbf{L}_\Delta \mathbf{V}^n, \quad \mathbf{V}^0 = \psi.$$

*A finite difference scheme is stable if there exists a positive constant  $C$ , independent of the mesh spacing and initial data, such that*

$$\|\mathbf{U}^n - \mathbf{V}^n\| \leq C \|\mathbf{U}^0 - \mathbf{V}^0\|, \quad n \rightarrow \infty, \quad \Delta x \rightarrow 0, \quad \Delta t \rightarrow 0, \quad n\Delta t \leq T. \quad (3.1.14a)$$

*Remark 7.* The requirement that  $C$  be independent of  $\Delta x$ ,  $\Delta t$ ,  $\mathbf{U}^0$ , and  $\mathbf{V}^0$  implies that the bound expressed in (3.1.14a) is uniform.

*Remark 8.* Definition 3.1.9, like Definition 3.1.8, can be thought of in relation to a sequence of calculations performed on finer-and-finer meshes.

*Remark 9.* This concept of stability implies that initial bounded differences between two solutions remain bounded for finite times when the mesh spacing is sufficiently small. Nothing in Definition 3.1.9 implies that  $C \leq 1$ ; hence, contrary to notions of stability that arise in physics and engineering, some growth of the difference between solutions is permitted.

Our most successful stability analyses will occur when  $\mathbf{L}_\Delta$  is linear. In this case, the matrix  $\mathbf{L}_\Delta$  is independent of  $\mathbf{U}^n$  and  $\mathbf{V}^n$  and we have

$$\mathbf{U}^{n+1} - \mathbf{V}^{n+1} = \mathbf{L}_\Delta(\mathbf{U}^n - \mathbf{V}^n), \quad \mathbf{U}^0 - \mathbf{V}^0 = \boldsymbol{\phi} - \boldsymbol{\psi}.$$

Replacing  $\mathbf{U}^n - \mathbf{V}^n$  by  $\mathbf{U}^n$ , we see that stability can be defined without introducing a perturbation. This alternate definition of stability is used throughout numerical analysis (*cf., e.g.*, Strikwerda [5], Section 1.5) and we repeat it here for completeness.

**Definition 3.1.10.** *A finite difference scheme (3.1.7) for a homogeneous initial value problem is stable if there exists a positive constant  $C$ , independent of the mesh spacing and initial data, such that*

$$\|\mathbf{U}^n\| \leq C\|\mathbf{U}^0\|, \quad n \rightarrow \infty, \quad \Delta x \rightarrow 0, \quad \Delta t \rightarrow 0, \quad n\Delta t \leq T. \quad (3.1.14b)$$

*Remark 10.* Definitions 3.1.9 and 3.1.10 are equivalent for linear homogeneous initial value problems; however, both forms are used for nonlinear problems where they are not equivalent.

*Remark 11.* The notion of stability expressed by Definition 3.1.10 implies a bound on the growth of the solution and, as such, is similar to the concept of a “well posed” partial differential equation.

Let us use these definitions to establish the convergence or stability of some of the finite difference schemes that were introduced in Chapter 2.

*Example 3.1.6.* We show that the solution of the forward time-centered space scheme (2.3.3) for the heat-conduction problem (2.3.1) converges in the maximum norm when  $r \leq 1/2$ .

Using (2.3.3a) and (3.1.2a) with  $v$  replaced by  $u$ , the finite difference and partial differential equation solutions satisfy

$$U_j^{n+1} = rU_{j-1}^n + (1 - 2r)U_j^n + rU_{j+1}^n,$$

$$u_j^{n+1} = ru_{j-1}^n + (1 - 2r)u_j^n + ru_{j+1}^n + \Delta t \tau_j^n, \quad j = 1, 2, \dots, J-1.$$

Letting  $e_j^n = u_j^n - U_j^n$  and subtracting

$$e_j^{n+1} = re_{j-1}^n + (1 - 2r)e_j^n + re_{j+1}^n + \Delta t \tau_j^n, \quad j = 1, 2, \dots, J-1.$$

Taking the absolute value and using the triangular inequality

$$|e_j^{n+1}| \leq |r||e_{j-1}^n| + |1 - 2r||e_j^n| + |r||e_{j+1}^n| + \Delta t |\tau_j^n|, \quad j = 1, 2, \dots, J-1.$$

Replacing the error terms on the right side of the above expression by their maximum values  $\|\mathbf{e}^n\|_\infty$  and  $\|\boldsymbol{\tau}^n\|_\infty$ , we obtain

$$|e_j^{n+1}| \leq (|r| + |1 - 2r| + |r|)\|\mathbf{e}^n\|_\infty + \Delta t \|\boldsymbol{\tau}^n\|_\infty, \quad j = 1, 2, \dots, J-1.$$

We know that  $r > 0$ . If, in addition,  $r \leq 1/2$ , then  $1 - 2r \geq 0$  and the absolute value signs can be removed from the terms involving  $r$  to obtain

$$|e_j^{n+1}| \leq \|\mathbf{e}^n\|_\infty + \Delta t \|\boldsymbol{\tau}^n\|_\infty, \quad j = 1, 2, \dots, J-1.$$

Since this result is valid for  $j \in [1, J-1]$ , it holds for the particular value of  $j$  where the left side attains its maximum value. Thus, we can replace the left side absolute value by its maximum norm to obtain

$$\|\mathbf{e}^{n+1}\|_\infty \leq \|\mathbf{e}^n\|_\infty + \Delta t \|\boldsymbol{\tau}^n\|_\infty.$$

Iterating the inequality over  $n$

$$\|\mathbf{e}^{n+1}\|_\infty \leq \|\mathbf{e}^{n-1}\|_\infty + \Delta t(\|\boldsymbol{\tau}^n\|_\infty + \|\boldsymbol{\tau}^{n-1}\|_\infty) \leq \dots$$

Thus,

$$\|\mathbf{e}^n\|_\infty \leq \|\mathbf{e}^0\|_\infty + n\Delta t \tau,$$

where

$$\tau = \max_{0 < k < n-1} \|\boldsymbol{\tau}^k\|_\infty.$$

Neglecting round off errors, solutions of the partial differential and difference equations both satisfy the same initial conditions, so  $\|\mathbf{e}^0\|_\infty = 0$ . Additionally, if  $T$  is the total time of interest,  $n\Delta t \leq T$  for all combinations of  $n$  and  $\Delta t$ . Thus,

$$\|\mathbf{e}^n\|_\infty \leq T\tau.$$

Using (3.1.2c) we bound  $\tau$  as

$$\tau \leq \frac{\Delta t}{2}K + \frac{\sigma\Delta x^2}{12}M,$$

where

$$K = \max_{0 \leq t \leq T, 0 \leq x \leq 1} |u_{tt}|, \quad M = \max_{0 \leq t \leq T, 0 \leq x \leq 1} |u_{xxxx}|.$$

Thus,

$$\|\mathbf{e}^n\|_\infty \leq T\left(\frac{\Delta t}{2}K + \frac{\sigma\Delta x^2}{12}M\right).$$

Letting  $\Delta x$  and  $\Delta t$  approach zero, we see that  $\|\mathbf{e}^n\|_\infty \rightarrow 0$ ; hence, the forward time-centered space scheme (2.3.3) converges to the solution of the heat-conduction problem (2.3.1) in the maximum norm when  $r \leq 1/2$ .

*Example 3.1.7.* We show that the forward time-backward space scheme (2.2.4)

$$U_j^{n+1} = (1 - \alpha_j^n)U_j^n + \alpha_j^n U_{j-1}^n.$$

for the kinematic wave equation (2.2.1) is stable in the maximum norm when the Courant number  $\alpha_j^n \equiv a(U_j^n)\Delta t/\Delta x \in (0, 1]$ . (Recall that this condition must be imposed to satisfy the Courant, Friedrichs, Lewy Theorem 2.2.1.)

We'll use the stability definition (3.1.10) and begin by taking the absolute value of the difference scheme and using the triangular inequality to obtain

$$|U_j^{n+1}| \leq |1 - \alpha_j^n||U_j^n| + |\alpha_j^n||U_{j-1}^n|.$$

Since  $0 < \alpha_j^n \leq 1$  by the Courant, Friedrichs, Lewy Theorem, we can remove the absolute value signs from the terms involving  $\alpha_j^n$  to get

$$|U_j^{n+1}| \leq (1 - \alpha_j^n)|U_j^n| + \alpha_j^n|U_{j-1}^n|.$$

Replacing the solution values on the right side by their maximum values

$$|U_j^n| \leq \|\mathbf{U}^n\|_\infty, \quad \forall j.$$

As in Example 3.1.6, the inequality must hold for the value of  $j$  that maximizes its left side; thus,

$$\|\mathbf{U}^{n+1}\|_\infty \leq \|\mathbf{U}^n\|_\infty.$$

This inequality holds for all  $n$  and may be iterated to yield

$$\|\mathbf{U}^n\|_\infty \leq \|\mathbf{U}^0\|_\infty,$$

which establishes stability in the sense of (3.1.14b).

In the two previous examples, we analyzed the convergence and stability of finite difference schemes using very similar arguments. These arguments can be generalized further to obtain a “maximum principle.”

**Theorem 3.1.1.** *A sufficient condition for stability of the one-level finite difference scheme*

$$U_j^{n+1} = \sum_{|s| \leq S} c_s U_{j+s}^n$$

*in the maximum norm is that all coefficients  $c_s$ ,  $|s| \leq S$ , be positive and add to unity.*

*Proof.* The arguments follow those used in Examples 3.1.6 and 3.1.7. □

*Remark 12.* A one-level difference scheme is one that only involves solutions at time levels  $n$  and  $n + 1$ . A multilevel difference scheme might involve time levels prior to  $n$  as well.

Typically, we not only want to know that a given scheme converges but the rate at which the numerical and exact solutions approach each other. This prompts the notion of order of accuracy.

**Definition 3.1.11.** *A consistent finite-difference approximation of a partial differential equation is of order  $p$  in time and  $q$  in space if*

$$\tau_j^n = O(\Delta t^p) + O(\Delta x^q). \quad (3.1.15)$$

## Problems

- Using (3.1.2b), we easily see that the forward time-centered space scheme (3.2.3a) for the heat conduction equation is first order in time and second order in space. Show that this scheme has an  $O(\Delta x^2)$  local discretization error when  $r = \sigma \Delta t / \Delta x^2 = 1/6$ .

2. Show that the Euclidean norm of a  $J \times J$  matrix  $\mathbf{A}$  given by (3.1.12) follows from from the definition of a matrix norm (3.1.10) upon use of the Euclidean vector norm (3.1.9b).
3. In Example 3.1.7, we used the maximum principle (Theorem 3.1.1) to establish stability of the forward time-backward space scheme (2.2.4). It can likewise be used to establish stability in the maximum norm of the forward time-forward space scheme (2.2.2b) when  $-1 \leq \alpha_j^n \leq 0$ . Can it be applied to the forward time-centered space scheme (2.2.5)?

## 3.2 Stability Analysis Using a Discrete Fourier Series

A discrete Fourier series can be used to analyze the stability of constant coefficient finite-difference problems with periodic initial data in much the same way that an infinite Fourier series can be used to solve constant coefficient partial differential equations. This method was introduced by John von Neumann and is called *von Neumann stability analysis*.

Thus, suppose that the solution of a finite difference problem is periodic in  $j$  with period  $J$  and express its solution as the discrete complex Fourier series

$$U_j^n = \sum_{k=0}^{J-1} A_k^n \omega_j^k, \quad j = 0, 1, \dots, J-1, \quad (3.2.1a)$$

where

$$\omega_j \equiv e^{2\pi i j / J}. \quad (3.2.1b)$$

The complex form of the Fourier series is much more convenient for our present purposes than the more common representation in terms of sines and cosines. In this form, the solution  $U_j^n$ ,  $j = 0, 1, \dots, J-1$ , is an approximation of the solution of a partial differential equation that is periodic in  $x$  with period  $2\pi$ . The mesh spacing that is inferred by (3.2.1a) is  $\Delta x = 2\pi/J$ .

The discrete Fourier series has many properties in common with the infinite Fourier series or the continuous Fourier transform (*cf.* Gottlieb and Orszag [1] or Strikwerda [5], Section 2.1). For example, the discrete Fourier series satisfies the orthogonality relation

$$\sum_{j=0}^{J-1} \omega_j^k \bar{\omega}_j^l = \begin{cases} J, & \text{if } k \equiv l \pmod{J} \\ 0, & \text{otherwise} \end{cases}, \quad (3.2.2)$$

where  $k \equiv l \pmod{J}$  means that  $k$  and  $l$  differ by an integral multiple of  $J$  and a superimposed bar denotes a complex conjugate, *e.g.*,  $\bar{\omega}_j = e^{-2\pi i j/J}$ . The relationship (3.2.2) is easily established using properties of the complex roots of unity.

Given the solution  $U_j^n$ ,  $j = 0, 1, \dots, J-1$ , we can find the Fourier coefficients  $A_k^n$ ,  $k = 0, 1, \dots, J-1$ , by inverting the discrete Fourier series using the orthogonality relation (3.2.2). Thus, multiplying (3.2.1a) by  $\bar{\omega}_j^l$  and summing over  $j$  yields

$$\sum_{j=0}^{J-1} U_j^n \bar{\omega}_j^l = \sum_{j=0}^{J-1} \bar{\omega}_j^l \sum_{k=0}^{J-1} A_k^n \omega_j^k.$$

Interchanging the order of the summations on the right side and using (3.2.2) gives

$$\sum_{j=0}^{J-1} U_j^n \bar{\omega}_j^l = \sum_{k=0}^{J-1} A_k^n \sum_{j=0}^{J-1} \omega_j^k \bar{\omega}_j^l = J A_l^n.$$

Thus,

$$A_l^n = \frac{1}{J} \sum_{j=0}^{J-1} U_j^n \bar{\omega}_j^l. \quad (3.2.3)$$

The inverse (3.2.3) of the discrete Fourier series (3.2.1a) is called the *discrete Fourier transform*.

Another property of discrete Fourier series and transforms that follows directly from the orthogonality condition (3.2.2) is the discrete form of Parseval's relation (*cf.* Problem 1 at the end of this section)

$$\|\mathbf{U}^n\|_2^2 = J \|\mathbf{A}^n\|_2^2, \quad (3.2.4a)$$

where, for complex quantities,

$$\|\mathbf{U}^n\|_2^2 \equiv \sum_{j=0}^{J-1} U_j^n \bar{U}_j^n, \quad \|\mathbf{A}^n\|_2^2 \equiv \sum_{k=0}^{J-1} A_k^n \bar{A}_k^n. \quad (3.2.4b)$$

Given the link between physical and Fourier coefficients expressed by Parseval's relation (3.2.4a), it is natural to use von Neumann's stability analysis in the Euclidean ( $\mathcal{L}^2$ ) norm. Let us illustrate the technique with some examples.

*Example 3.2.1.* We use von Neumann's approach to show that the forward time-backward space finite difference scheme (2.2.4) is stable in  $\mathcal{L}^2$  when  $\alpha_j^n$  is a positive constant, say,  $\alpha \in (0, 1]$  and periodic initial data is applied. In this case, (2.2.4) becomes

$$U_j^{n+1} = (1 - \alpha)U_j^n + \alpha U_{j-1}^n.$$

Substitute (3.2.1a) into the above expression to obtain

$$\sum_{k=0}^{J-1} A_k^{n+1} \omega_j^k = \sum_{k=0}^{J-1} [(1 - \alpha)A_k^n \omega_j^k + \alpha A_k^n \omega_{j-1}^k].$$

According to (3.2.1b),  $\omega_{j-1} = \omega_j e^{-2\pi i/J}$ ; thus,

$$\sum_{k=0}^{J-1} [A_k^{n+1} - (1 - \alpha)A_k^n - \alpha A_k^n e^{-2\pi i k/J}] \omega_j^k = 0.$$

Using the orthogonality relation (3.2.2) and the discrete Fourier transform (3.2.3), we infer that the (bracketed) coefficient of  $\omega_j^k$  must vanish for each  $k$ , *i.e.*,

$$A_k^{n+1} - [(1 - \alpha) + \alpha e^{-2\pi i k/J}] A_k^n = 0.$$

Solving for  $A_k^{n+1}$

$$A_k^{n+1} = M_k A_k^n, \quad M_k = (1 - \alpha) + \alpha e^{-2\pi i k/J}.$$

The *amplification factor*  $M_k$  gives the growth or decay of the  $k$ th Fourier mode in one time step.

The above recurrence can be iterated to obtain

$$A_k^n = (M_k)^n A_k^0,$$

where the Fourier coefficient  $A_k^0$ ,  $k = 0, 1, \dots, J - 1$ , can be determined from the prescribed initial data upon use of (3.2.3). Explicit determination of  $A_k^0$  is rarely necessary since we seek to establish stability independently of the initial data.

We can use Eulers identity

$$e^{i\phi} = \cos \phi + i \sin \phi$$

to determine the magnitude of the amplification factor as

$$|M_k|^2 = M_k \bar{M}_k = (1 - \alpha + \alpha \cos \frac{2\pi k}{J})^2 + (\alpha \sin \frac{2\pi k}{J})^2$$

or

$$|M_k|^2 = 1 - 2\alpha(1 - \alpha)(1 - \cos \frac{2\pi k}{J}).$$

The half-angle formula

$$\sin^2 \phi/2 = (1 - \cos \phi)/2$$

can be used to simplify the above relation to

$$|M_k|^2 = 1 - 4\alpha(1 - \alpha) \sin^2 \frac{\pi k}{J}.$$

We now see than the initial Fourier mode  $A_k^0$  will grow or decay depending on whether  $|M_k|$  is greater than or less than unity. In this example, no Fourier mode grows since  $0 \leq 4\alpha(1 - \alpha) \leq 1$  when  $0 < \alpha \leq 1$ .

Using (3.2.1a), we see that the finite difference solution satisfies

$$U_j^n = \sum_{k=0}^{J-1} (M_k)^n A_k^0 \omega_j^k.$$

Stability in the Euclidean norm follows by applying Parseval's relation (3.2.4a) to the above equation to obtain

$$\|\mathbf{U}^n\|_2^2 = J \sum_{k=0}^{J-1} |M_k|^{2n} |A_k^0|^2.$$

Since  $|M_k| \leq 1$  for all  $k$  when  $0 < \alpha \leq 1$ , we can use Parseval's relation once more to find

$$\|\mathbf{U}^n\|_2^2 \leq J \sum_{k=0}^{J-1} |A_k^0|^2 = \|\mathbf{U}^0\|_2^2.$$

This establishes the  $\mathcal{L}^2$  stability of (2.2.4) when the Courant number is a constant  $\alpha \in (0, 1]$ .

*Example 3.2.2.* Let us, at long last, use von Neumann's method to examine the  $\mathcal{L}^2$  stability of the forward time-centered space scheme (2.2.5) for the kinematic wave

equation (2.2.1). Necessarily, we assume that  $\alpha_j^n$  is a constant  $\alpha$  and the initial data is periodic in  $j$  with period  $J$  so that (2.2.5) becomes

$$U_j^{n+1} = U_j^n - \frac{\alpha}{2}(U_{j+1}^n - U_{j-1}^n).$$

Expanding  $U_j^n$  in a discrete Fourier series (3.2.1a), we find

$$\sum_{k=0}^{J-1} [A_k^{n+1} - A_k^n + \frac{\alpha}{2}(e^{2\pi ik/J} - e^{-2\pi ik/J}) A_k^n] \omega_j^k = 0.$$

Once again, the orthogonality relation (3.2.2) may be used to infer that

$$A_k^n = (M_k)^n A_k^0,$$

where

$$M_k = 1 - \frac{\alpha}{2}(e^{2\pi ik/J} - e^{-2\pi ik/J}) = 1 - i\alpha \sin \frac{2\pi k}{J}.$$

The magnitude of the amplification factor is

$$|M_k|^2 = 1 + \alpha^2 \sin^2 \frac{2\pi k}{J}.$$

There is no possibility of restricting  $\alpha$  so that  $|M_k| \leq 1$  for all  $k$ ; thus, most Fourier modes will grow. As bad as this appears, (2.2.5) is still stable according to either (3.1.14a) or (3.1.14b). To demonstrate this, observe that

$$|M_k|^2 \leq 1 + \alpha^2.$$

The following inequality is useful in many situations, so we'll record it as a lemma.

**Lemma 3.2.1.** *For all real  $z$*

$$1 + z \leq e^z. \quad (3.2.5)$$

*Proof.* Using Taylor's formula for the exponential function

$$e^z = 1 + z + \frac{1}{2}e^\xi z^2 \geq 1 + z.$$

□

In the present case, (3.2.5) can be used to bound the amplification factor as

$$|M_k|^2 \leq e^{\alpha^2}.$$

Taking the  $n$ th power and using the definition of the Courant number (2.2.3)

$$|M_k|^{2n} \leq e^{na^2\Delta t^2/\Delta x^2} = e^{(n\Delta t)(a^2\Delta t/\Delta x^2)}.$$

As usual, introduce  $T$  so that  $n\Delta t \leq T$ . Also, let us agree to select meshes having temporal and spatial steps satisfying  $a^2\Delta t/\Delta x^2 = \beta$ , where  $\beta$  is a constant. Then

$$|M_k|^{2n} \leq e^{T\beta} = C^2.$$

Using Parseval's relation (3.2.4a) and following the steps used in Example 3.2.1, we find

$$\|\mathbf{U}^n\|_2^2 = J \sum_{k=0}^{J-1} |M_k|^{2n} |A_k^0|^2 \leq C^2 \|\mathbf{U}^0\|_2^2.$$

We have found a constant  $C$  satisfying

$$\|\mathbf{U}^n\|_2 \leq C \|\mathbf{U}^0\|_2;$$

hence, (2.2.5) is stable in  $\mathcal{L}^2$ . The constant  $C$  is greater than unity, so that most Fourier modes will grow. In fact, if  $T$  and  $\beta$  are not small enough, initial disturbances will grow to such an extent that they dominate the computation. Additionally, we are forced to restrict the time step  $\Delta t \leq \beta\Delta x^2/a^2$ , while the directional methods (2.2.2b, 4) only require  $\Delta t \leq \Delta x/a$ . This leads us to conclude that, while technically stable, the forward time-centered space scheme (2.2.5) is not a practical method to apply to the kinematic wave equation (2.2.1).

When applying von Neumann's method we will always find a solution of the form

$$U_j^n = \sum_{k=0}^{J-1} (M_k)^n A_k^0 \omega_j^k.$$

In Example 3.2.1, we found that  $|M_k| \leq 1$  so that initial perturbations did not grow and the forward time-backward space scheme was stable in  $\mathcal{L}^2$ . In Example 3.2.2,  $|M_k| > 1$  and perturbations grew, but were bounded for finite times  $T$ . While the (forward time-centered space) scheme was stable, we saw that it was impractical and, thus, may ask

whether or not we should keep  $|M_k| \leq 1$  as a practical matter. The answer to this question depends on the behavior of the problem under investigation. If the solution of the partial differential equation is unstable and, hence, growing in time, then some growth of initial perturbations can and should be tolerated. However, if the solution of the partial differential equation does not grow, then either  $|M_k|$  must be bounded by unity or we must be willing to compute for sufficiently small times so that the bound  $C$  on the growth of  $\|\mathbf{U}^n\|$  remains small. There are only a few instances where the latter option will be desirable; thus, we should typically maintain  $|M_k| \leq 1$  whenever solutions of the partial differential equation under study do not grow.

In those cases where  $|M_k|$  may exceed unity, we may ask the amount by which it may do so. This leads to the von Neumann condition.

**Definition 3.2.1.** *A finite difference scheme satisfies the von Neumann condition if there exists a positive constant  $c$  that is independent of  $\Delta t$ ,  $\Delta x$ , and  $k$  which satisfies*

$$|M_k| \leq 1 + c\Delta t, \quad \forall \Delta t \leq \Delta t^*, \quad \Delta x \leq \Delta x^*. \quad (3.2.6)$$

Finite difference schemes that satisfy the von Neumann condition are stable and conversely as expressed by the following theorem.

**Theorem 3.2.1.** *A constant coefficient scalar one-level difference scheme is stable in the Euclidean norm if and only if it satisfies the von Neumann condition.*

*Proof.* Suppose the von Neumann condition is satisfied. We follow the arguments used in Example 3.2.2 and use Parseval's relation (3.2.4a) to obtain

$$\|\mathbf{U}^n\|_2^2 = J \sum_{k=0}^{J-1} |M_k|^{2n} |A_k^0|^2 \leq (1 + c\Delta t)^{2n} \|\mathbf{U}^0\|_2^2.$$

Using (3.2.5), we have

$$\|\mathbf{U}^n\|_2^2 \leq e^{2cn\Delta t} \|\mathbf{U}^0\|_2^2 \leq e^{2cT} \|\mathbf{U}^0\|_2^2,$$

where,  $n\Delta t \leq T$ , with  $T$  being the total time of interest. Choosing  $C^2 = e^{cT}$  establishes stability according to (3.1.14b).

To prove the converse, we suppose that there is a value of  $k = k^*$  such that

$$|M_{k^*}| > (1 + c\Delta t), \quad \forall c.$$

Further suppose that initial data is selected so that  $A_{k^*}^0 \neq 0$  and  $A_k^0 = 0$ ,  $k \neq k^*$ . Thus, the initial condition is

$$U_j^0 = A_{k^*}^0 \omega_j^{k^*}.$$

The solution after  $n$  time steps is

$$U_j^n = (M_{k^*})^n A_{k^*}^0 \omega_j^{k^*} = (M_{k^*})^n U_j^0.$$

Taking the Euclidean norm

$$\|\mathbf{U}^n\|_2^2 = |M_{k^*}|^{2n} \|\mathbf{U}^0\|_2^2 > (1 + c\Delta t)^{2n} \|\mathbf{U}^0\|_2^2, \quad \forall c.$$

Since this must hold for any and all values of  $c$ ,  $\|\mathbf{U}^n\|_2$  must be unbounded and, hence, the finite difference method is unstable.  $\square$

The von Neumann condition is a necessary condition for stability in much more general situations than stated by Theorem 3.2.1. Sufficiency has also been verified under less restrictive conditions [2, 3].

*Example 3.2.3.* It may seem like we are restricted to using the directional derivatives when solving the kinematic wave equation (2.2.1); however, the Lax-Friedrichs scheme uses centered space difference with a forward-averaged time difference to obtain the approximation

$$\frac{U_j^{n+1} - (U_{j+1}^n + U_{j-1}^n)/2}{\Delta t} + a_j^n \frac{U_{j+1}^n - U_{j-1}^n}{2\Delta x} = 0$$

or

$$U_j^{n+1} = \frac{1 + \alpha_j^n}{2} U_{j-1}^n + \frac{1 - \alpha_j^n}{2} U_{j+1}^n. \quad (3.2.7)$$

The computational stencil for this scheme is shown in Figure 3.2.1. This spatially-centered difference scheme is clearly stable in the maximum norm by the Maximum Principle as long as the Courant, Friedrichs, Lewy Theorem is satisfied, *i.e.*, as long as  $|\alpha_j^n| \leq 1$ . It is also stable in  $\mathcal{L}^2$  under similar conditions (*cf.* Problems 2 and 3 at the end of this section).

## Problems

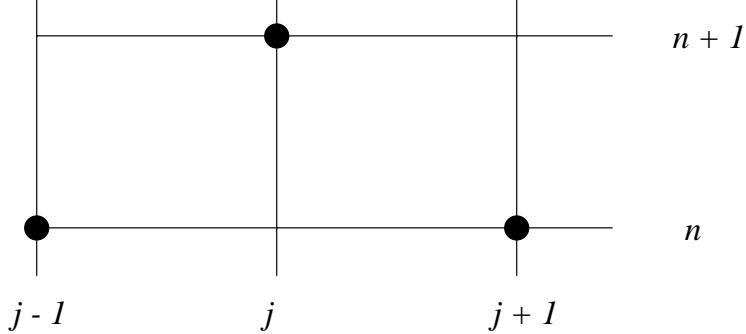


Figure 3.2.1: Computational stencil for the Lax-Friedrichs scheme (3.2.7).

1. Let  $U_j^n$ ,  $j = 0, 1, \dots, J - 1$ , and  $A_k^n$ ,  $k = 0, 1, \dots, J - 1$ , be discrete Fourier pairs according to (3.2.1a). Derive the discrete form of Parseval's relation (3.2.4a).
2. Consider the Lax-Friedrichs difference scheme (3.2.7) for the constant-coefficient kinematic wave equation

$$u_t + au_x = 0.$$

- 2.1. Calculate the leading terms in the local discretization error for this scheme. Is the Lax-Friedrichs scheme consistent for all choices of mesh spacings? Explain.
- 2.2. Use a von Neumann stability analysis to show that the Lax-Friedrichs scheme is stable in  $\mathcal{L}^2$  whenever the Courant, Friedrichs, Lewy Theorem is satisfied
3. Consider finite-difference schemes for the constant-coefficient kinematic wave equation of Problem 2 that have the form

$$U_j^{n+1} = U_j^n - \frac{\alpha}{2}(U_{j+1}^n - U_{j-1}^n) + \frac{\beta}{2}(U_{j+1}^n - 2U_j^n + U_{j-1}^n) \quad (3.2.8a)$$

with

$$\alpha = a \frac{\Delta t}{\Delta x}, \quad \beta = \alpha z. \quad (3.2.8b)$$

The parameter  $\alpha$  is the Courant number and  $\beta$  and  $z$  are dissipation factors. Some common schemes having this form for specific choices of  $z$  and  $\beta$  appear in Table 3.2.1. The function  $sgn(x) = x/|x|$  and the Lax-Wendroff scheme is described in Section 3.3. Find the region in the  $(\alpha, \beta)$ -plane where the amplification factor associated with the scheme (3.2.8) does not exceed unity in magnitude. You may,

Method	$z$	$\beta$
Centered	0	0
Lax-Wendroff	$\alpha$	$\alpha^2$
Upwind	$\text{sgn}\alpha$	$\alpha \text{sgn}\alpha$
Lax-Friedrichs	$1/\alpha$	1

Table 3.2.1: Values of the dissipation factors for schemes of the form (3.2.8).

if you want, confine your analysis to positive values of  $\alpha$ . This region of parameter space is often called a “region of absolutely stability.” Present a graph of the absolute stability region. The specific methods presented in the table correspond to curves in the  $(\alpha, \beta)$ -plane. Superimpose these curves on your stability diagram. Comment on the stability properties of the various methods.

### 3.3 Matrix Stability Analysis

The tools that are currently available to us for performing stability analyses have severe limitations. The maximum principle (Theorem 3.1.1) requires finite difference schemes to have positive coefficients that sum to unity and the von Neumann method requires constant coefficient linear problems with periodic initial data. Matrix methods can be used to analyze more general linear problems. With the goal of describing these methods, consider a linear finite difference scheme in the form of (3.1.7). For simplicity, assume that the difference operator  $\mathbf{L}_\Delta$  is independent of the time level  $n$  and iterate this (3.1.7) to obtain

$$\mathbf{U}^n = (\mathbf{L}_\Delta)^n \mathbf{U}^0.$$

(If  $\mathbf{L}_\Delta$  depended on  $n$ , we would get a product of the operators at each time level.)

Taking a norm

$$\|\mathbf{U}^n\| = \|(\mathbf{L}_\Delta)^n \mathbf{U}^0\| \leq \|(\mathbf{L}_\Delta)^n\| \|\mathbf{U}^0\|.$$

The finite difference scheme (3.1.7) is stable according to (3.1.14b) if and only if there exists a constant  $C$  such that

$$\|(\mathbf{L}_\Delta)^n\| \leq C, \quad n \rightarrow \infty, \quad \Delta x, \Delta t \rightarrow 0, \quad n\Delta t \leq T. \quad (3.3.1)$$

Using matrix stability methods, we try to limit the growth of  $\|\mathbf{L}_\Delta\|$ . Before beginning, we review some material about matrix eigenvalue problems. If  $\mathbf{x}^i$  is an eigenvector of  $\mathbf{L}_\Delta$  corresponding to the eigenvalue  $\lambda_i$ , then

$$\mathbf{L}_\Delta \mathbf{x}^i = \lambda_i \mathbf{x}^i, \quad i = 1, 2, \dots, N, \quad (3.3.2)$$

where  $N$  is the dimension of  $\mathbf{L}_\Delta$ .

**Definition 3.3.1.** *The spectral radius  $\rho(\mathbf{L}_\Delta)$  of  $\mathbf{L}_\Delta$  is the modulus of its largest eigenvalue, i.e.,*

$$\rho(\mathbf{L}_\Delta) \equiv \max_{1 \leq i \leq N} |\lambda_i(\mathbf{L}_\Delta)|. \quad (3.3.3)$$

The spectral radius provides a lower bound to any matrix norm.

**Lemma 3.3.1.** *Let  $\rho(\mathbf{L}_\Delta)$  and  $\|\mathbf{L}_\Delta\|$  be the spectral radius and any vector-induced matrix norm of  $\mathbf{L}_\Delta$ , then*

$$\rho(\mathbf{L}_\Delta) \leq \|\mathbf{L}_\Delta\|. \quad (3.3.4)$$

*Proof.* Take a norm of (3.3.2)

$$\|\mathbf{L}_\Delta \mathbf{x}^i\| = |\lambda_i| \|\mathbf{x}^i\|, \quad i = 1, 2, \dots, N,$$

and use (3.1.10, 3.1.11)

$$|\lambda_i| = \frac{\|\mathbf{L}_\Delta \mathbf{x}^i\|}{\|\mathbf{x}^i\|} \leq \|\mathbf{L}_\Delta\|, \quad i = 1, 2, \dots, N.$$

The result (3.3.4) follows since this relation holds for the value of  $i$  corresponding to the maximum eigenvalue.  $\square$

Since the eigenvalues of  $(\mathbf{L}_\Delta)^n$  are  $(\lambda_i)^n$ , we can use (3.3.4) and (3.1.11) to obtain

$$\rho^n(\mathbf{L}_\Delta) \leq \|(\mathbf{L}_\Delta)^n\| \leq \|\mathbf{L}_\Delta\|^n. \quad (3.3.5)$$

The left and right sides of (3.3.4) lead, respectively, to necessary and sufficient stability conditions.

**Theorem 3.3.1.** (*The von Neumann necessary stability condition.*) A necessary condition for the stability of the linear homogeneous finite difference scheme (3.1.7) is that there exist a constant  $c$ , independent  $\Delta x$  and  $\Delta t$ , satisfying

$$\rho(\mathbf{L}_\Delta) \leq 1 + c\Delta t. \quad (3.3.6)$$

*Proof.* Using (3.3.5), we see that  $\rho^n(\mathbf{L}_\Delta)$  must be bounded otherwise  $\|(\mathbf{L}_\Delta)^n\|$  couldn't be. Thus, a necessary condition for the stability of (3.1.7) is that there exist a constant  $C$  such that  $\rho^n(\mathbf{L}_\Delta) \leq C$  for all  $n\Delta t \leq T$  as  $\Delta x, \Delta t \rightarrow 0$ . Taking an  $n$ th root and using the maximal value of  $n$  gives

$$\rho(\mathbf{L}_\Delta) \leq C^{1/n} \leq C^{\Delta t/T}.$$

When  $\Delta t$  is sufficiently small, Taylor's formula can be used to find a value of  $c$  so that  $C^{\Delta t/T}$  is bounded by  $1 + c\Delta t$ .  $\square$

**Theorem 3.3.2.** A sufficient condition for the stability of the linear finite difference scheme (3.1.7) is that there exist a constant  $c$ , independent of  $\Delta x$  and  $\Delta t$ , such that

$$\|\mathbf{L}_\Delta\| \leq 1 + c\Delta t. \quad (3.3.7)$$

*Proof.* The proof follows the arguments used in Theorem 3.2.1.  $\square$

*Remark 1.* The von Neumann condition (3.3.6) is also sufficient for stability in many cases. For example, it is sufficient for stability when  $\mathbf{L}_\Delta$  is either symmetric or similar to a symmetric matrix [2].

*Example 3.3.1.* In Example 3.1.4 we showed that  $\mathbf{L}_\Delta$  for the forward time-centered space scheme (2.3.3) for the heat conduction problem (2.3.1) satisfies

$$\mathbf{L}_\Delta = \begin{bmatrix} 1 - 2r & r & & & \\ r & 1 - 2r & r & & \\ & r & 1 - 2r & r & \\ & & & \ddots & \\ & & & r & 1 - 2r \end{bmatrix}.$$

If  $r \leq 1/2$ , then  $\|\mathbf{L}_\Delta\|_\infty = 1$ ; therefore, from Theorem 3.3.2, this scheme is stable in the maximum norm.

*Example 3.3.2.* Consider the finite difference scheme

$$U_j^{n+1} = c_{-1}U_{j-1}^n + c_1U_{j+1}^n$$

with periodic initial data in  $j$  with period  $J$ . (This problem is similar to Exercise 1.5 of Strikwerda [5].) The Lax-Friedrichs scheme (3.2.7) for the kinematic wave equation (2.2.1) has this form with

$$c_{-1} = \frac{1+\alpha}{2}, \quad c_1 = \frac{1-\alpha}{2},$$

where, for simplicity, we have assumed the Courant number  $\alpha$  to be constant.

The difference scheme has the form of (3.1.7) with

$$\mathbf{L}_\Delta = \begin{bmatrix} 0 & c_1 & c_{-1} \\ c_{-1} & 0 & c_1 \\ & \ddots & \\ c_1 & c_{-1} & 0 \end{bmatrix}, \quad \mathbf{U}^n = \begin{bmatrix} U_0^n \\ U_1^n \\ \vdots \\ U_{J-1}^n \end{bmatrix}.$$

If  $|c_{-1}| + |c_1| \leq 1$  then  $\|\mathbf{L}_\Delta\|_\infty \leq 1$  and the scheme is stable in the maximum norm. This is the case for the Lax-Friedrichs scheme (even for nonlinear problems) when  $|\alpha| \leq 1$ .

We can analyze the stability of finite difference schemes using the basic definitions (3.1.14a, 3.1.14b). Nonlinear problems can be analyzed in this manner; thus, in some sense, it is the most powerful analytical technique at our disposal, but it is also the most difficult one to apply. Let us try an example.

*Example 3.3.3.* The Lax-Wendroff scheme for the kinematic wave equation

$$u_t + au_x = 0$$

is obtained from the Taylor's series in time

$$u_j^{n+1} = u_j^n + \Delta t(u_t)_j^n + \frac{\Delta t^2}{2}(u_{tt})_j^n + \dots$$

Time derivatives are replaced by spatial derivatives that are obtained from the partial differential equation. For simplicity, let us suppose that  $a$  is a constant, then

$$u_t = -au_x, \quad u_{tt} = -au_{xt} = -a(u_t)_x = a^2u_{xx}, \quad \dots$$

(In fact, the Lax-Wendroff scheme is rather difficult to implement when  $\alpha$  is not constant. We'll discuss an alternative in Chapter 6 that is superior to this approach in this case.)

The Taylor's series is truncated and the spatial derivatives are approximated by centered differences (2.1.7, 2.1.9). Retaining  $O(\Delta t^2)$  terms leads to the second-order Lax-Wendroff scheme

$$U_j^{n+1} = U_j^n - \frac{\alpha}{2}(U_{j+1}^n - U_{j-1}^n) + \frac{\alpha^2}{2}(U_{j+1}^n - 2U_j^n + U_{j-1}^n)$$

or

$$U_j^{n+1} = \frac{\alpha(\alpha+1)}{2}U_{j-1}^n + (1-\alpha^2)U_j^n + \frac{\alpha(\alpha-1)}{2}U_{j+1}^n. \quad (3.3.8)$$

where, as usual,  $\alpha$  is the Courant number. Taylor's series arguments immediately show that the local error is  $O(\Delta t^2) + O(\Delta x^2)$ .

The coefficients of this Lax-Wendroff scheme are not all positive. For example, if  $0 < \alpha < 1$  then the coefficient of  $U_{j+1}^n$  will be negative. Likewise, the coefficient of  $U_{j-1}^n$  is negative when  $\alpha$  is negative. Thus, Theorem 3.1.1 cannot be used to establish stability in the maximum norm. As expected, the Lax-Wendroff scheme gives the exact solution of the kinematic wave equation when  $|\alpha| = 1$ .

We'll establish the stability of a periodic initial value problem in the Euclidean norm when  $|\alpha| \leq 1$  using definition (3.1.14b). The von Neumann method could also be used in this case (*cf.* Problem 3.2.3). To begin, we square (3.3.8) to obtain

$$(U_j^{n+1})^2 = [\frac{\alpha(\alpha+1)}{2}U_{j-1}^n + (1-\alpha^2)U_j^n + \frac{\alpha(\alpha-1)}{2}U_{j+1}^n]^2.$$

If  $|\alpha| \leq 1$  then

$$\frac{\alpha^2(1-\alpha^2)}{4}[U_{j-1}^n - 2U_j^n + U_{j+1}^n]^2 \geq 0.$$

Add this term to the right side of the previous expression and sum over a period to get

$$\begin{aligned} \sum_{j=0}^{J-1} (U_j^{n+1})^2 &\leq \sum_{j=0}^{J-1} [\frac{\alpha^2(1+\alpha)}{2}(U_{j-1}^n)^2 + (1-\alpha^2)(U_j^n)^2 + \frac{\alpha^2(1-\alpha)}{2}(U_{j+1}^n)^2 \\ &\quad - \alpha(1-\alpha^2)(U_{j+1}^n U_j^n - U_j^n U_{j-1}^n)]. \end{aligned}$$

This expression can be simplified by reindexing the summations, *e.g.*,

$$\sum_{j=0}^{J-1} (U_{j-1}^n)^2 = \sum_{k=-1}^{J-2} (U_k^n)^2 = \sum_{k=0}^{J-1} (U_k^n)^2 + (U_{-1}^n)^2 - (U_{J-1}^n)^2.$$

The solution is periodic so  $U_{-1}^n = U_{J-1}^n$ ; hence,

$$\sum_{j=0}^{J-1} (U_{j-1}^n)^2 = \sum_{k=0}^{J-1} (U_k^n)^2.$$

Similar reindexing of other terms yields

$$\sum_{j=0}^{J-1} (U_{j+1}^n)^2 = \sum_{k=0}^{J-1} (U_k^n)^2, \quad \sum_{j=0}^{J-1} U_j^n U_{j-1}^n = \sum_{k=0}^{J-1} U_{k+1}^n U_k^n.$$

Thus, the summation simplifies to

$$\sum_{j=0}^{J-1} (U_j^{n+1})^2 \leq \sum_{j=0}^{J-1} \left[ \frac{\alpha^2(1+\alpha)}{2} + (1-\alpha^2) + \frac{\alpha^2(1-\alpha)}{2} \right] (U_j^n)^2 = \sum_{j=0}^{J-1} (U_j^n)^2.$$

Therefore,  $\|\mathbf{U}^{n+1}\|_2 \leq \|\mathbf{U}^n\|_2$  and the Lax-Wendroff scheme is stable when the Courant, Friedrichs, Lewy Theorem ( $|\alpha| \leq 1$ ) is satisfied

Centered schemes like the Lax-Wendroff (3.3.8) and Lax-Friedrichs (3.2.7) methods may require artificial boundary conditions for initial-boundary value problems. For instance, suppose the kinematic wave equation with a positive wave speed  $a$  is to be solved on  $0 < x < 1$ . This problem only requires a boundary condition at  $x = 0$  (Section 1.3). A numerical solution at the right-most point  $j = J$ , however, cannot be computed by either the Lax-Wendroff or Lax-Friedrichs schemes. Another method is needed to compute  $U_J^n$ . As we'll learn in Chapter 6, a poor choice could affect the stability of the method.

## Problems

1. Write a computer program for the difference scheme (3.2.8). Implement it with  $\beta = \alpha^2$ , which corresponds to the Lax-Wendroff scheme. Assume that the initial data

$$u(x, 0) = \phi(x)$$

is periodic in  $x$  with period 2. The problem should be solved on  $-1 \leq x \leq 1$ ,  $0 < t \leq T$ . Use  $J (= 2/\Delta x)$ ,  $\alpha$ , and  $N (= T/\Delta t)$  as input parameters.

- 1.1. Execute your program when  $a = 1$  and  $\phi(x)$  has the form

$$\phi(x) = \begin{cases} \Delta x + x, & \text{if } -\Delta x \leq x < 0 \\ \Delta x - x, & \text{if } 0 \leq x < \Delta x \\ 0, & \text{elsewhere for } x \in (-1, 1) \end{cases}.$$

1.2. This data is an attempt to simulate the effects of a small error introduced by, say, round off. Execute your program for about 20 time steps with  $J = 10$  and  $\alpha = 0.5, 0.999, 1.1$  (more if you like). Plot the numerical and exact solutions as functions of  $x$  for a few times. Comment on the solutions for each value of  $\alpha$ . Which choice of  $\alpha$  would be preferred if the initial data actually did correspond to a rounding error?

1.3. Solve a problem with  $a = 1$  and

$$\phi(x) = \begin{cases} -2(1+x), & \text{if } -1 \leq x < -1/2 \\ 2x, & \text{if } -1/2 \leq x < 1/2 \\ 2(1-x), & \text{if } 1/2 \leq x < 1 \end{cases}.$$

Compute solutions on  $0 < t \leq 2$  using  $J = 10, 20, 40$  and  $\alpha = 1/2, 1$ . (Values of  $N$  follow from the Courant number.) Compare the accuracy of the solutions at  $t = 2$ . Tabulate errors at  $t = 2$  in the maximum and  $L^2$  norms and estimate the order of convergence of the numerical to the exact solution. Plot the solutions as a function of  $x$  at  $t = 2$ .

1.4. Repeat Part 3 using either the upwind or Lax-Friedrichs schemes. Compare results with those obtained using the Lax-Wendroff scheme.

## 3.4 The Lax Equivalence Theorem

In Section 3.3, we learned that every difference scheme should be consistent, convergent, and stable. The Lax Equivalence Theorem expresses a relationship between these three properties.

**Theorem 3.4.1.** (*Lax Equivalence Theorem*). *Given a properly posed linear initial value problem and a consistent finite difference approximation of it, then stability is necessary and sufficient for convergence.*

*Proof.* We will prove that a consistent and stable difference scheme is convergent. The proof that unstable schemes do not converge is more involved and we'll defer it to Chapter 6. The entire proof appears in Richtmyer and Morton [3] and Strikwerda [5], Chapter 10.

Let  $\mathbf{U}^n$  be the solution of the difference equation (3.1.7) and let  $\boldsymbol{\tau}^n$  be the local discretization error. Then

$$\mathbf{u}^{n+1} = \mathbf{L}_\Delta \mathbf{u}^n + \Delta t \boldsymbol{\tau}^n.$$

Let  $\mathbf{e}^n \equiv \mathbf{u}^n - \mathbf{U}^n$  be the global discretization error and subtract (3.1.7) to obtain

$$\mathbf{e}^{n+1} = \mathbf{L}_\Delta \mathbf{e}^n + \Delta t \boldsymbol{\tau}^n.$$

For simplicity, assume that  $\mathbf{L}_\Delta$  is independent of  $n$  and iterate the above relation to obtain

$$\mathbf{e}^n = (\mathbf{L}_\Delta)^n \mathbf{e}^0 + \Delta t [\mathbf{L}_\Delta^{n-1} \boldsymbol{\tau}^0 + \mathbf{L}_\Delta^{n-2} \boldsymbol{\tau}^1 + \dots + \boldsymbol{\tau}^{n-1}].$$

Taking a norm and noting that  $\mathbf{e}^0 = 0$  in the absence of round-off errors,

$$\|\mathbf{e}^n\| \leq \Delta t [\|\mathbf{L}_\Delta^{n-1}\| \|\boldsymbol{\tau}^0\| + \|\mathbf{L}_\Delta^{n-2}\| \|\boldsymbol{\tau}^1\| + \dots + \|\boldsymbol{\tau}^{n-1}\|]. \quad (3.4.1)$$

The difference scheme is consistent, so we can make  $\boldsymbol{\tau}^k$  arbitrarily small by making  $\Delta x$  and  $\Delta t$  sufficiently small. Let us choose a total time of interest  $T$  and an  $\epsilon > 0$  such that  $\|\boldsymbol{\tau}^k\| \leq \epsilon$  for all  $k\Delta t \leq T$  whenever  $\Delta x, \Delta t < \delta$ . Furthermore, the scheme is stable, so there exists a constant  $C$  such that  $\|\mathbf{L}_\Delta^k\| \leq C$  for all  $k\Delta t \leq T$ . With (3.4.1), these considerations imply that

$$\|\mathbf{e}^n\| \leq n\Delta t C\epsilon.$$

If  $n\Delta t \leq T$

$$\|\mathbf{e}^n\| = \|\mathbf{u}^n - \mathbf{U}^n\| \leq TC\epsilon,$$

and the scheme converges.  $\square$

*Remark 1.* The Lax Equivalence Theorem provides the link between stability and convergence that, perhaps, we expected based on the examples of Chapter 2.

*Remark 2.* In practice, convergence is difficult to establish, while consistency and stability are less so. Consistency merely requires the use of Taylor's series expansions and stability can be proven, at least, in simplified situations using von Neumann's approach. Thus, Lax's theorem provides very useful information.

# Bibliography

- [1] D. Gottlieb and S.A. Orszag. *Numerical Analysis of Spectral Methods: Theory and Applications*. Regional Conference Series in Applied Mathematics, No. 26. SIAM, Philadelphia, 1977.
- [2] A.R. Mitchell and D.F. Griffiths. *The Finite Difference Method in Partial Differential Equations*. John Wiley and Sons, Chichester, 1980.
- [3] R.D. Richtmyer and K.W. Morton. *Difference Methods for Initial Value Problems*. John Wiley and Sons, New York, second edition, 1967.
- [4] G. Strang. *Linear Algebra and its Applications*. Academic Press, New York, third edition, 1988.
- [5] J.C. Strikwerda. *Finite Difference Schemes and Partial Differential Equations*. Chapman and Hall, Pacific Grove, 1989.

# Chapter 4

## Difference Methods for Parabolic Partial Differential Equations

### 4.1 Implicit Difference Methods

In this chapter, we focus on parabolic problems of which the now familiar one-dimensional heat conduction problem

$$u_t = \sigma u_{xx}, \quad 0 < x < 1, \quad t > 0, \quad (4.1.1a)$$

$$u(x, 0) = \phi(x), \quad 0 \leq x \leq 1, \quad (4.1.1b)$$

$$u(0, t) = f(t), \quad u(1, t) = g(t), \quad t > 0, \quad (4.1.1c)$$

is a representative example. Recall that the positive constant  $\sigma (= k/\rho c)$  is the diffusivity of the material. In Chapters 2 and 3, we used forward differences to approximate temporal derivatives and centered differences to approximate spatial derivatives on a uniform mesh of spacing  $\Delta x$  by  $\Delta t$  to obtain the scheme (2.3.3), which we reproduce here for convenience

$$U_j^{n+1} = rU_{j-1}^n + (1 - 2r)U_j^n + rU_{j+1}^n, \quad j = 1, 2, \dots, J-1, \quad n > 0, \quad (4.1.2a)$$

$$U_j^0 = \phi(j\Delta x), \quad j = 0, 1, \dots, J, \quad (4.1.2b)$$

$$U_0^n = f(n\Delta t), \quad U_J^n = g(n\Delta t), \quad (4.1.2c)$$

$$r = \frac{\sigma \Delta t}{\Delta x^2}. \quad (4.1.3)$$

With homogeneous boundary conditions ( $f(t) = g(t) = 0$ ), we found that keeping  $r \leq 1/2$  was sufficient for the convergence and stability of (4.1.2). The nontrivial Dirichlet boundary conditions (4.1.2c) introduce no complications in the computational scheme; however, their effect on stability will have to be investigated.

The main difficulty with (4.1.2) is the stability restriction  $r \leq 1/2$ , which places a severe limitation on the choice of the time step  $\Delta t$ . Additionally, recall (*cf.* (3.1.2)) that the local discretization error is  $O(\Delta t) + O(\Delta x^2)$  for arbitrary values of  $r$  (but, *cf.* Problem 1 at the end of this section). A scheme with a higher-order temporal error would be preferable. However, before developing better difference schemes, let us try to understand the reasons for the stability restriction. Our explanation follows Mitchell and Griffiths [15]

Domains of dependence aren't appropriate for parabolic partial differential equations; however, as shown in Figure 4.1.1, we can calculate a domain of dependence of a point  $P$  with coordinates  $(j\Delta x, n\Delta t)$  for the difference scheme (4.1.2). The solution  $U_j^n$  at  $P$  is determined without using the boundary conditions at points  $Q$  and  $R$ . Thus, the finite difference scheme is behaving like a scheme for a hyperbolic partial differential equation having two real characteristics. From the theory of partial differential equations [10], we know that parabolic equations have only one real characteristic and that the solution at a point such as  $P$  cannot be determined without knowing the boundary data at points  $Q$  and  $R$ .

Let

$$\theta = \tan^{-1} \frac{\Delta x}{\Delta t} = \tan^{-1} \frac{\sigma}{r \Delta x}.$$

The angle  $\theta \rightarrow \pi/2$  (Figure 4.1.1) as the mesh is refined with  $r$  fixed. Thus, the boundary conditions at  $Q$  and  $R$  do enter into the scheme in the limit as  $\Delta x$  and  $\Delta t$  tend to zero with  $r$  fixed. Furthermore, for small values of  $r$ ,  $\theta$  will be sufficiently close to  $\pi/2$  so that boundary values at points close to  $Q$  and  $R$  will be used in obtaining solutions of (4.1.2). As  $r$  increases, the boundary conditions that determine the solution at  $P$  are further removed from  $Q$  and  $R$ , thereby, affecting the stability of the scheme.

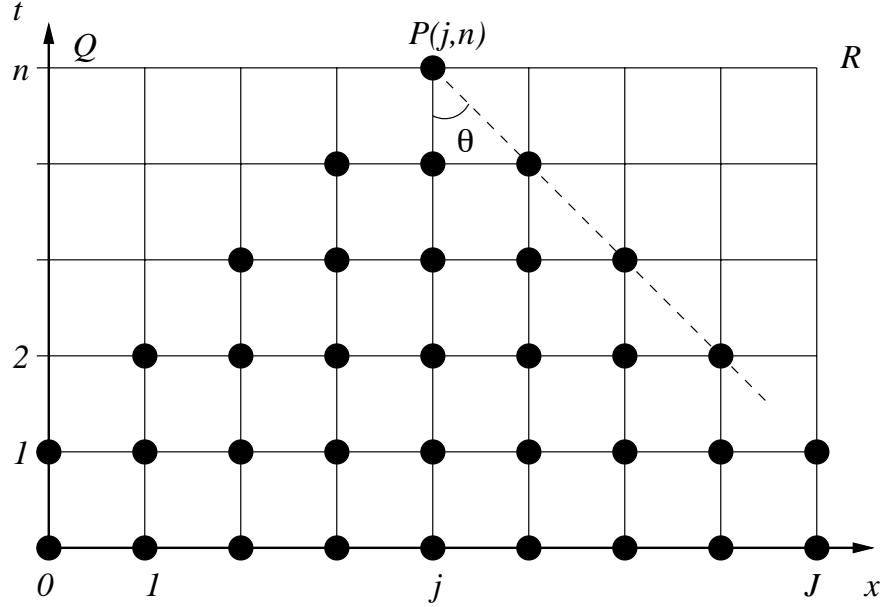


Figure 4.1.1: “Domain of dependence” of a point  $P$  for the forward time-centered space difference scheme (4.1.2). The solution at  $P$  can be calculated without knowledge of the boundary conditions at  $Q$  and  $R$ .

These arguments imply that stability restrictions may be relaxed by developing schemes that include boundary conditions at  $Q$  and  $R$  when computing the solution at a point such as  $P$  of Figure 4.1.1. Such schemes are called *implicit* because the solution at a point such as  $P$  will only be determined implicitly in terms of other unknowns at the same time level.

Perhaps, the simplest implicit finite difference scheme for the heat conduction equation (4.1.1a) is obtained by using backward temporal differences and centered spatial differences, *i.e.*,

$$\frac{U_j^{n+1} - U_j^n}{\Delta t} = \sigma \frac{U_{j-1}^{n+1} - 2U_j^{n+1} + U_{j+1}^{n+1}}{\Delta x^2}. \quad (4.1.4)$$

The computational stencil of this scheme, called the *backward Euler method*, is shown in Figure 4.1.2. The scheme is clearly implicit because determination of  $U_j^{n+1}$  depends on knowing its neighboring values  $U_{j-1}^{n+1}$  and  $U_{j+1}^{n+1}$ . We must develop a solution procedure for (4.1.4); however, from the computational stencil, we see that the solution at time level  $n + 1$  will have to involve a knowledge of the boundary data at  $x = 0$  and  $x = 1$  at time  $(n + 1)\Delta t$ .

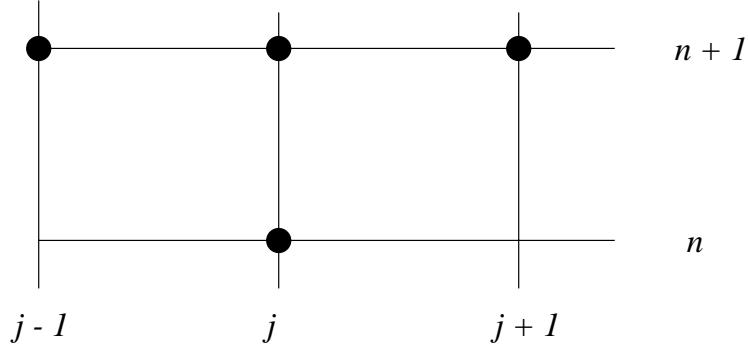


Figure 4.1.2: Computational stencil for the backward Euler difference scheme (4.1.4) for (4.1.1a).

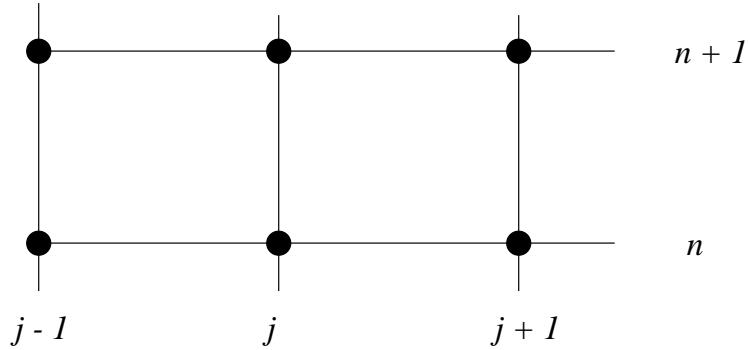


Figure 4.1.3: Computational stencil for the averaged time-centered space difference scheme (4.1.5) for (4.1.1a) when  $\theta \in (0, 1)$ .

Before analyzing (4.1.4), let us consider a more general scheme for (4.1.1a) that is obtained by using a weighted average of forward and backward time differences with centered spatial differences, *i.e.*,

$$\frac{U_j^{n+1} - U_j^n}{\Delta t} = \sigma \frac{\theta \delta^2 U_j^{n+1} + (1 - \theta) \delta^2 U_j^n}{\Delta x^2}. \quad (4.1.5)$$

Recall (Section 2.1) that the central difference operator  $\delta$  satisfies

$$\delta U_j^n = U_{j+1/2}^n - U_{j-1/2}^n; \quad (4.1.6a)$$

thus,

$$\delta^2 U_j^n = U_{j+1}^n - 2U_j^n + U_{j-1}^n. \quad (4.1.6b)$$

The parameter  $\theta$  is a weighting factor selected between zero and unity. The computational stencil for (4.1.5) when  $0 < \theta < 1$  is shown in Figure 4.1.3. Several schemes are contained within this weighted scheme. In fact,

- if  $\theta = 0$ , (4.1.5) is the explicit scheme (4.1.2),
- if  $\theta = 1$ , (4.1.5) is the backward Euler scheme (4.1.4), and
- if  $\theta = 1/2$ , (4.1.5) is an implicit scheme called the *Crank-Nicolson method*.

We need a method of solving the initial-boundary value problem (4.1.1) when using (4.1.5) with  $\theta \neq 0$ . As usual, assume that the solution  $U_j^n$ ,  $j = 0, 1, \dots, J$ , is known and that we want to determine  $U_j^{n+1}$ ,  $j = 0, 1, \dots, J$ . Using the boundary conditions (4.1.1c) we have

$$U_0^{n+1} = f^{n+1}, \quad U_J^{n+1} = g^{n+1}, \quad (4.1.7a)$$

where  $f^n \equiv f(n\Delta t)$ , etc. Using (4.1.6b) to evaluate the differences in (4.1.5) and writing the result at all interior mesh points  $j = 1, 2, \dots, J - 1$  gives

$$\begin{aligned} -r\theta U_{j-1}^{n+1} + (1 + 2r\theta)U_j^{n+1} - r\theta U_{j+1}^{n+1} &= r(1 - \theta)U_{j-1}^n + \\ [1 - 2r(1 - \theta)]U_j^n + r(1 - \theta)U_{j+1}^n, & \quad j = 1, 2, \dots, J - 1. \end{aligned} \quad (4.1.7b)$$

If we regard  $U_0^{n+1}$  and  $U_J^{n+1}$  as being known quantities, then (4.1.7b) is a linear tridiagonal system of  $J - 1$  equations for the  $J - 1$  unknowns  $U_j^{n+1}$ ,  $j = 1, 2, \dots, J - 1$ . It is convenient to write (4.1.7b) in vector form, so let

$$\mathbf{U}^n = [U_1^n, U_2^n, \dots, U_{J-1}^n]^T \quad (4.1.8)$$

be the vector of unknowns at time level  $n$ . Then, write (4.1.7b) as

$$[\mathbf{I} - r\theta \mathbf{C}] \mathbf{U}^{n+1} = [\mathbf{I} + r(1 - \theta) \mathbf{C}] \mathbf{U}^n + r \mathbf{f}^n, \quad (4.1.9a)$$

where

$$\mathbf{C} = \begin{bmatrix} -2 & 1 & & & \\ 1 & -2 & 1 & & \\ & \ddots & & \ddots & \\ & & 1 & -2 & \end{bmatrix}, \quad \mathbf{f}^n = \begin{bmatrix} \theta f^{n+1} + (1 - \theta) f^n \\ 0 \\ \vdots \\ 0 \\ \theta g^{n+1} + (1 - \theta) g^n \end{bmatrix}. \quad (4.1.9b)$$

Tridiagonal systems, such as (4.1.9), arise frequently in numerical analysis and it is important to have a fast algorithm for solving them. Fortunately, a special case of Gaussian elimination suffices. Let us describe the method for a slightly more general problem than (4.1.9); thus, consider a procedure for solving linear systems of the form

$$\mathbf{AX} = \mathbf{F}, \quad (4.1.10a)$$

where  $\mathbf{F}$  and  $\mathbf{X}$  are  $N$ -dimensional vectors and  $\mathbf{A}$  is an  $N \times N$  tridiagonal matrix having the form

$$\mathbf{A} = \begin{bmatrix} a_1 & c_1 & & & \\ b_2 & a_2 & c_2 & & \\ & \ddots & & \ddots & \\ & & b_{N-1} & a_{N-1} & c_{N-1} \\ & & & b_N & a_N \end{bmatrix}. \quad (4.1.10b)$$

In the present case,  $N = J - 1$ ,

$$a_j = 1 + 2r\theta, \quad j = 1, 2, \dots, J - 1, \quad (4.1.11a)$$

$$b_j = -r\theta, \quad j = 2, 3, \dots, J - 1, \quad (4.1.11b)$$

$$c_j = -r\theta, \quad j = 1, 2, \dots, J - 2, \quad (4.1.11c)$$

and

$$F_j = r(1 - \theta)U_{j-1}^n + [1 - 2r(1 - \theta)]U_j^n + r(1 - \theta)U_{j+1}^n + r\delta_{j,1}[\theta f^{n+1} + (1 - \theta)f^n] +$$

$$r\delta_{j,J-1}[\theta g^{n+1} + (1 - \theta)g^n], \quad j = 1, 2, \dots, J - 1. \quad (4.1.11d)$$

The quantity  $\delta_{j,k}$  is the Kronecker delta which is unity when  $j = k$  and zero otherwise.

We assume that pivoting is not necessary (which is often the case when solving parabolic problems by finite difference or finite element methods) and factor  $\mathbf{A}$  as

$$\mathbf{A} = \mathbf{LU}. \quad (4.1.12a)$$

Here,  $\mathbf{L}$  and  $\mathbf{U}$  are lower and upper bidiagonal matrices having the form

$$\mathbf{L} = \begin{bmatrix} 1 & & & \\ l_2 & 1 & & \\ & l_3 & 1 & \\ & & \ddots & \\ & & & l_N & 1 \end{bmatrix}, \quad \mathbf{U} = \begin{bmatrix} u_1 & v_1 & & & \\ & u_2 & v_2 & & \\ & & \ddots & & \\ & & & u_{N-1} & v_{N-1} \\ & & & & u_N \end{bmatrix}. \quad (4.1.12b)$$

Once the coefficients  $l_j$ ,  $j = 2, 3, \dots, N$ ,  $u_j$ ,  $j = 1, 2, \dots, N$ , and  $v_j$ ,  $j = 1, 2, \dots, N - 1$ , have been determined, the system (4.1.10a) may easily be solved by forward and backward substitution. Thus, using (4.1.12a) in (4.1.10a) gives

$$\mathbf{LUX} = \mathbf{F}. \quad (4.1.13a)$$

Let

$$\mathbf{UX} = \mathbf{Y}, \quad (4.1.13b)$$

then,

$$\mathbf{LY} = \mathbf{F}. \quad (4.1.13c)$$

The system (4.1.13c) may be solved for  $\mathbf{Y}$  by forward substitution and, once  $\mathbf{Y}$  has been determined, the system (4.1.13b) may be solved for  $\mathbf{X}$  by backward substitution.

The coefficients of  $\mathbf{L}$  and  $\mathbf{U}$  are determined by a direct computation. Thus, substituting (4.1.10b) and (4.1.12b) into (4.1.12a), we find

$$u_1 = a_1, \quad (4.1.14a)$$

$$l_j = b_j/u_{j-1}, \quad u_j = a_j - l_j c_{j-1}, \quad j = 2, 3, \dots, N, \quad (4.1.14b)$$

$$v_j = c_j, \quad j = 2, 3, \dots, N. \quad (4.1.14c)$$

Having determined  $l_j$ ,  $j = 2, 3, \dots, N$ ,  $u_j$ ,  $j = 1, 2, \dots, N$ , and  $v_j$ ,  $j = 1, 2, \dots, N - 1$ , we can use (4.1.12b) in (4.1.13c, 4.1.13c) to find

$$Y_1 = F_1, \quad Y_j = F_j - l_j Y_{j-1}, \quad j = 2, 3, \dots, N, \quad (4.1.15a)$$

$$X_N = y_N/u_N, \quad X_j = (Y_j - v_j X_{j+1})/u_j, \quad j = N-1, N-2, \dots, 1. \quad (4.1.15b)$$

The procedure (4.1.14, 4.1.15) is called the *tridiagonal algorithm*. It can be implemented in a space-efficient manner by overwriting  $a_j$ ,  $b_j$ ,  $c_j$ , and  $F_j$  with  $u_j$ ,  $l_j$ ,  $v_j$ , and  $x_j$  (Figure 4.1.4). It requires about 2 divisions, 3 multiplications, and 3 additions or subtractions for each  $j$  (other than  $j = 1$  or  $N$ ). On some computers, division is much slower than multiplication. In this case, we can trade 1 division for 2 multiplications by storing  $1/u_j$  instead of  $u_j$ .

```
procedure tridi( $N, \mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{F}$ )
```

{ Factorization }

```
for  $j = 2$  to  $N$  do
   $b_j = b_j/a_{j-1}$ 
   $a_j = a_j - b_j c_{j-1}$ 
end for
```

{ Forward and backward substitution }

```
for  $j = 2$  to  $N$  do
   $F_j = F_j - b_j F_{j-1}$ 
end for
 $F_N = F_N/a_N$ 
for  $j = N-1$  downto 1 do
   $F_j := (F_j - c_j F_{j+1})/a_j$ 
end for
```

Figure 4.1.4: Tridiagonal algorithm for solving (4.1.10).

Let us examine the stability of the weighted average scheme (4.1.7b) using the Von Neumann method. As necessary, we assume that the initial conditions and solution are periodic in  $x$  and write the solution as the discrete Fourier series (3.2.1), *i.e.*,

$$U_j^n = \sum_{k=0}^{J-1} A_k^n \omega_j^k, \quad \omega_j = e^{2\pi i j/J}. \quad (4.1.16)$$

Substituting (4.1.16) into (4.1.7b)

$$\sum_{k=0}^{J-1} \{A_k^{n+1}[-r\theta e^{-2\pi ik/J} + 1 + 2r\theta - r\theta e^{2\pi ik/J}] -$$

$$A_k^n[r(1-\theta)e^{-2\pi ik/J} + 1 - 2r(1-\theta) + r(1-\theta)e^{2\pi ik/J}] \omega_j^k = 0.$$

Using the orthogonality property (3.2.2) and Euler's identity, we find

$$A_k^{n+1} = M_k A_k^n,$$

where the amplification factor  $M_k$  satisfies

$$M_k = 1 - \frac{2r(1 - \cos 2k\pi/J)}{1 + 2r\theta(1 - \cos 2k\pi/J)}.$$

Using the half angle formula

$$M_k = 1 - \frac{4r \sin^2 k\pi/J}{1 + 4r\theta \sin^2 k\pi/J}. \quad (4.1.17)$$

As usual, we can solve the first-order difference relation for  $A_k^n$  to get

$$A_k^n = (M_k)^n A_k^0. \quad (4.1.18)$$

Since solutions of periodic initial value problems for (4.1.1) are decaying, let us demand that no Fourier mode grow; thus,  $|M_k| \leq 1$  for all  $k$ . Since  $M_k$  is real and  $r$  and  $\theta$  are non-negative, this implies

$$-1 \leq 1 - \frac{4r \sin^2 k\pi/J}{1 + 4r\theta \sin^2 k\pi/J} \leq 1.$$

The right-hand inequality is always satisfied; hence, it remains to show that

$$\frac{4r \sin^2 k\pi/J}{1 + 4r\theta \sin^2 k\pi/J} \leq 2,$$

or

$$2r(1 - 2\theta) \sin^2 \frac{k\pi}{J} \leq 1.$$

Since this must be satisfied for all  $k$ , it suffices to restrict

$$2r(1 - 2\theta) \leq 1. \quad (4.1.19)$$

(This restriction is also necessary since  $\sin k\pi/J$  will be unity for some choices of  $k$  and  $J$ .)

If  $1/2 \leq \theta \leq 1$ , the term  $1 - 2\theta$  will be non-positive, and (4.1.19) will be satisfied for all values of  $r > 0$ . Thus, (4.1.7b) is stable for all choices of  $\Delta t$  and  $\Delta x$ . We call such a scheme *unconditionally stable*. If  $0 \leq \theta < 1/2$ , then  $|M_k| \leq 1$  when

$$r \leq \frac{1}{2(1 - 2\theta)}. \quad (4.1.20)$$

As in Section 3.2, we may use (4.1.18) in (4.1.16) to find the solution of (4.1.7b) as

$$U_j^n = \sum_{k=0}^{J-1} (M_k)^n A_k^0 \omega_j^k. \quad (4.1.21)$$

When  $|M_k| \leq 1$ , we use Parseval's relation (3.2.4) to show that  $\|\mathbf{U}^n\|_2 \leq \|\mathbf{U}^0\|_2$ .

In Chapter 3, we noted that it was essential to maintain  $|M_k| \leq 1$  whenever the solution of the partial differential equation does not increase in time. For any particular problem, initial conditions may be prescribed such that  $A_{k^*}^0 = 0$  whenever  $|M_{k^*}| > 1$ . It would appear that  $|M_{k^*}|$  could exceed unity in this case; however, round off errors will excite this mode and the growth of  $|M_{k^*}|$  will eventually dominate the solution. These arguments motivate the need for an alternate stability definition.

**Definition 4.1.1.** *A finite difference scheme*

$$\mathbf{U}^{n+1} = \mathbf{L}_\Delta \mathbf{U}^n \quad (4.1.22a)$$

*is absolutely stable for a given mesh (of size  $\Delta x$  and  $\Delta t$ ) if*

$$\|\mathbf{U}^n\| \leq \|\mathbf{U}^0\|, \quad n > 0. \quad (4.1.22b)$$

*Remark 1.* This definition parallels (3.1.14b). A similar one can be stated in the terminology of (3.1.14a).

*Remark 2.* The basic stability definitions (3.1.14a,b) are used in the limit as  $\Delta x, \Delta t \rightarrow 0$ , whereas the notion of absolute stability applies at a finite mesh spacing ( $\Delta x \times \Delta t$ ).

*Example 4.1.1.* In Chapter 2, we showed that the forward time-centered space scheme (4.1.2) is absolutely stable in  $\mathcal{L}^\infty$  and  $\mathcal{L}^2$  for  $r \leq 1/2$  when homogeneous boundary conditions ( $f(t) = g(t) = 0$ ) are prescribed.

**Definition 4.1.2.** Let the difference scheme (4.1.22a) depend on a certain minimal number of parameters involving the mesh spacing. Its region of absolute stability is the region of parameter space where it is absolutely stable.

**Definition 4.1.3.** A finite difference scheme (4.1.22a) is unconditionally stable if it is absolutely stable for all choices of mesh spacing  $\Delta x$  and  $\Delta t$ .

*Example 4.1.2.* The weighted average scheme (4.1.7b) depends on two parameters  $r$  and  $\theta$ . We have shown that (4.1.7b) is unconditionally stable in  $\mathcal{L}^2$  for a periodic initial value problem when  $\theta \in [1/2, 1]$ . It is absolutely stable in the regions  $\{\theta \in [1/2, 1], r > 0\}$  and  $\{\theta \in [0, 1/2), r \leq 1/2(1 - 2\theta)\}$ .

Let us conclude our discussion of the weighted average scheme (4.1.7b) by examining the local discretization error

$$\tau_j^n \equiv (u_t - \sigma u_{xx})_j^n - \frac{u_j^{n+1} - u_j^n}{\Delta t} - \sigma \frac{\theta \delta^2 u_j^{n+1} + (1 - \theta) \delta^2 u_j^n}{\Delta x^2}. \quad (4.1.23a)$$

Using a Taylor's series expansion about  $(j\Delta x, n\Delta t)$ , we find

$$\tau_j^n = -\Delta t \left( \frac{1}{2} - \theta \right) (u_{tt})_j^n + \frac{\sigma \Delta x^2}{12} (u_{xxxx})_j^n + O(\Delta t^2) + O(\Delta x^4). \quad (4.1.23b)$$

Thus, the local discretization error is  $O(\Delta t) + O(\Delta x^2)$ , unless  $\theta = 1/2$ , where the accuracy is  $O(\Delta t^2) + O(\Delta x^2)$ . The higher-order accuracy of the Crank-Nicolson scheme relative to other choices of  $\theta$  is due to the centering of the finite difference scheme about  $(j, n + 1/2)$  on a uniform mesh.

### Problems

1. The local discretization error of the explicit forward time-centered space scheme (4.1.2) satisfies (*cf.* (3.1.2))

$$\tau_j^n = -\frac{\Delta t}{2} (u_{tt})_j^n + \frac{\sigma \Delta x^2}{12} (u_{xxxx})_j^n + O(\Delta t^2) + O(\Delta x^4).$$

Thus, the local discretization error is  $O(\Delta t) + O(\Delta x^2)$  for arbitrary values of  $r$ . Is there a particular choice of  $r$  for which the discretization error is of higher order?

2. Use (4.1.23b) and the heat equation (4.1.1a) to show that the value  $\theta = 1/2 - \Delta x^2/12\sigma\Delta t$  gives a weighted average scheme with an  $O(\Delta t^2) + O(\Delta x^4)$  local discretization error. Show that the absolute stability restriction (4.1.20) is satisfied for this choice of  $\theta$ .

## 4.2 Neumann Boundary Conditions

Thus far, we have considered problems with either periodic or Dirichlet boundary conditions. Let us now investigate problems having Neumann boundary conditions, *e.g.*,

$$u_t = \sigma u_{xx}, \quad 0 < x < 1, \quad t > 0, \quad (4.2.1a)$$

$$u(x, 0) = \phi(x), \quad 0 \leq x \leq 1, \quad (4.2.1b)$$

$$u(0, t) = f(t), \quad (4.2.1c)$$

$$u_x(1, t) = g(t), \quad t > 0, \quad (4.2.1d)$$

which has a Neumann condition at  $x = 1$ .

It is possible to use backward or forward differences to discretize a Neumann boundary condition at the right or left end of the domain, respectively; however, it is generally preferable to handle such boundary conditions by introducing a fictitious external grid line and using higher-order centered differences. For the problem at hand, construct a uniform grid having cell spacing  $\Delta x = 1/J$  by  $\Delta t$  on  $0 \leq x \leq 1$ ,  $t > 0$  and introduce the fictitious grid line  $x = (J + 1)\Delta x$  as shown in Figure 4.2.1. With the extended problem domain, we may approximate the derivative in the boundary condition (4.2.1d) using the usual centered difference formula

$$(u_x)_J^n = \frac{u_{J+1}^n - u_{J-1}^n}{2\Delta x} + O(\Delta x^2). \quad (4.2.2)$$

Neglecting the local discretization error in (4.2.2) and substituting the result into (4.2.1d), we obtain an approximation of the Neumann condition at  $(j\Delta x, n\Delta t)$  as

$$U_{J+1}^n = U_{J-1}^n + 2\Delta x g^n. \quad (4.2.3)$$

Since centered differences (4.2.2) were used, (4.2.3) is an  $O(\Delta x^2)$  approximation of the boundary condition (4.2.1d). This is consistent with the order of the centered finite-difference approximations (2.1.9) that were used to discretize the partial differential equation (4.2.1a).

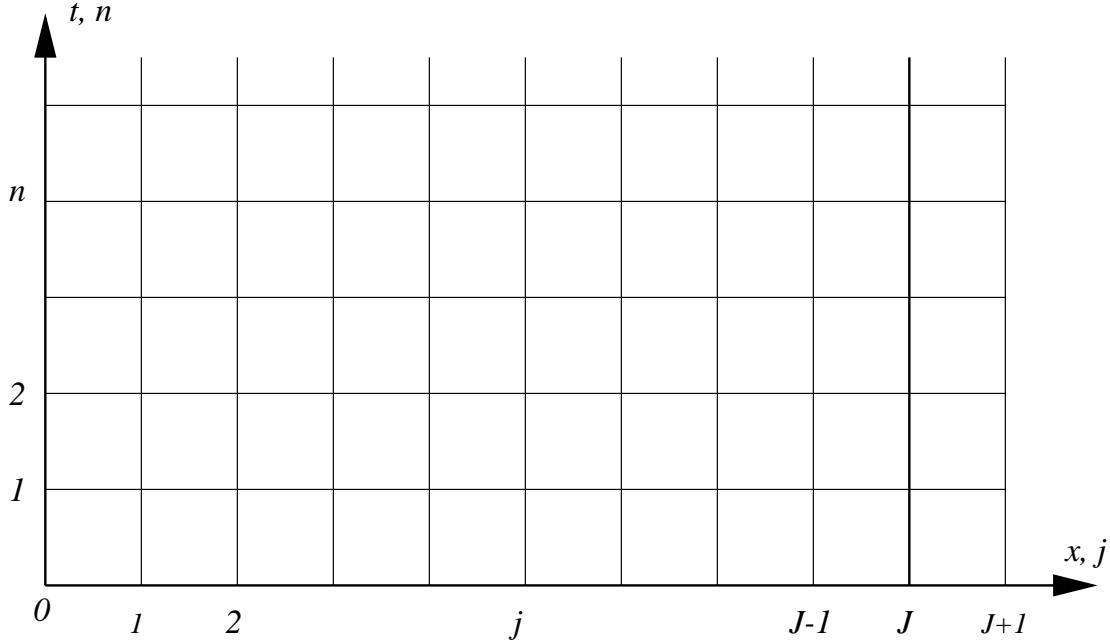


Figure 4.2.1: Computational grid showing fictitious grid line at  $x = (J + 1)\Delta x$  to handle a Neumann boundary condition.

The discrete problem on  $n\Delta t < t \leq (n + 1)\Delta t$  is obtained by combining (4.2.3) with the difference approximation of the partial differential equation, *e.g.*, (4.1.7b), for  $j = 1, 2, \dots, J$ . This gives a system of  $J + 1$  equations for the  $J + 1$  unknowns  $U_j^{n+1}$ ,  $j = 1, 2, \dots, J + 1$ . It is often easy to eliminate the fictitious unknown  $U_{J+1}^{n+1}$  from the system. For example, suppose that we are discretizing (4.2.1a) using the weighted average scheme (4.1.7b), then writing (4.1.7b) at  $j = J$  gives

$$\begin{aligned} -r\theta U_{J-1}^{n+1} + (1 + 2r\theta)U_J^{n+1} - r\theta U_{J+1}^{n+1} &= r(1 - \theta)U_{J-1}^n + \\ [1 - 2r(1 - \theta)]U_J^n + r(1 - \theta)U_{J+1}^n. \end{aligned}$$

Now, eliminate  $U_{J+1}^n$  and  $U_{J+1}^{n+1}$  using (4.2.3) to get

$$\begin{aligned} -2r\theta U_{J-1}^{n+1} + (1 + 2r\theta)U_J^{n+1} &= 2r(1 - \theta)U_{J-1}^n + [1 - 2r(1 - \theta)]U_J^n \\ + 2r(1 - \theta)\Delta x g^n + 2r\theta\Delta x g^{n+1}. \end{aligned} \tag{4.2.4}$$

When (4.2.4) is combined with (4.1.7b) at the interior points  $j = 1, 2, \dots, J - 1$ , we obtain a system of  $J$  equations for the  $J$  unknowns  $U_j^{n+1}$ ,  $j = 1, 2, \dots, J$ . This system is tridiagonal and may be solved using the tridiagonal algorithm (4.1.14, 4.1.15).

*Example 4.2.1.* Let us analyze the stability of the weighted average scheme (4.1.7b, 4.2.4) for solving the heat conduction problem (4.2.1) with  $f(t) = g(t) = 0$  in order to determine if a homogeneous Neumann boundary condition alters the stability characteristics of (4.1.7). We will use matrix methods since, in principle, the von Neumann method ignores boundary conditions. (As with the Fourier series solution of partial differential equations, however, certain homogeneous boundary conditions may be treated without difficulty by periodic extension of the domain and initial data.)

Using (4.1.7b) and (4.2.4) with  $f(t) = g(t) = 0$  yields the linear system

$$[\mathbf{I} - r\theta\mathbf{C}]\mathbf{U}^{n+1} = [\mathbf{I} + r(1-\theta)\mathbf{C}]\mathbf{U}^n, \quad (4.2.5a)$$

$$\mathbf{U}^n = \begin{bmatrix} U_1^n \\ U_2^n \\ \vdots \\ U_J^n \end{bmatrix}, \quad \mathbf{C} = \begin{bmatrix} -2 & 1 & & \\ 1 & -2 & 1 & \\ & \ddots & & \\ & 1 & -2 & 1 \\ & & 2 & -2 \end{bmatrix}. \quad (4.2.5b)$$

Equation (4.2.5a) may be written in the standard form

$$\mathbf{U}^{n+1} = \mathbf{L}_\Delta \mathbf{U}^n$$

with

$$\mathbf{L}_\Delta = (\mathbf{I} - r\theta\mathbf{C})^{-1}[\mathbf{I} + r(1-\theta)\mathbf{C}].$$

We'll seek to bound the spectral radius of  $\mathbf{L}_\Delta$ . Thus, consider the eigenvalue problem

$$(\mathbf{L}_\Delta - \lambda\mathbf{I})\mathbf{z} = \mathbf{0},$$

or

$$[\mathbf{I} + r(1-\theta)\mathbf{C} - \lambda(\mathbf{I} - r\theta\mathbf{C})]\mathbf{z} = \mathbf{0},$$

or

$$[r(1-\theta + \lambda\theta)\mathbf{C} - (\lambda - 1)\mathbf{I}]\mathbf{z} = \mathbf{0}.$$

Write the above equation as

$$(\mathbf{C} - \mu\mathbf{I})\mathbf{z} = \mathbf{0},$$

where

$$\mu = \frac{\lambda - 1}{r(1 - \theta + \lambda\theta)}.$$

Thus,  $\mu$  is an eigenvalue of  $\mathbf{C}$  and

$$\lambda = \frac{1 + \mu r(1 - \theta)}{1 - \mu r\theta} = 1 + \frac{\mu r}{1 - \mu r\theta}.$$

The eigenvalues and eigenvectors of many tridiagonal systems are known and, in particular, those of  $\mathbf{C}$  are

$$z_j^k = \sin \frac{k\pi j}{2J}, \quad j = 1, 2, \dots, J, \quad \mu_k = -4 \sin^2 \frac{k\pi}{4J}, \quad k = 2m - 1, \\ m = 0, 1, \dots, J - 1.$$

(We'll indicate how to obtain these results in Chapter 9.)

Knowing  $\mu_k$ , we calculate  $\lambda_k$  as

$$\lambda_k = 1 - \frac{4r \sin^2 k\pi/4J}{1 + 4r\theta \sin^2 k\pi/4J}.$$

This is identical to the result (4.1.17) that we obtained for the amplification factor of the homogeneous Dirichlet problem (4.1.7) using the von Neumann method; thus, the homogeneous Neumann condition has not altered the basic stability of the scheme.

The necessary stability condition that  $\rho(\mathbf{L}_\Delta) \leq 1$  is also sufficient in this case since  $\mathbf{L}_\Delta$  is similar to a symmetric matrix (*cf.* Mitchell and Griffiths [15] or Section 3.3). In order to see this, let

$$\mathbf{D} = \begin{bmatrix} 1 & & & & \\ & 1 & & & \\ & & \ddots & & \\ & & & 1 & \\ & & & & \sqrt{2} \end{bmatrix}.$$

Then

$$\hat{\mathbf{C}} = \mathbf{D}^{-1} \mathbf{C} \mathbf{D} = \begin{bmatrix} -2 & 1 & & & \\ 1 & -2 & 1 & & \\ & & \ddots & & \\ & & & 1 & -2 & \sqrt{2} \\ & & & & \sqrt{2} & -2 \end{bmatrix}.$$

is symmetric. Now let

$$\hat{\mathbf{L}}_\Delta = \mathbf{D}^{-1} \mathbf{L}_\Delta \mathbf{D} = \mathbf{D}^{-1} (\mathbf{I} - r\theta \mathbf{C})^{-1} [\mathbf{I} + r(1 - \theta) \mathbf{C}] \mathbf{D}$$

or

$$\begin{aligned}\hat{\mathbf{L}}_\Delta &= \mathbf{D}^{-1}(\mathbf{I} - r\theta\mathbf{C})^{-1}\mathbf{D}\mathbf{D}^{-1}[\mathbf{I} + r(1-\theta)\mathbf{C}]\mathbf{D} \\ &= [\mathbf{D}^{-1}(\mathbf{I} - r\theta\mathbf{C})\mathbf{D}]^{-1}[\mathbf{D}^{-1}(\mathbf{I} + r(1-\theta)\mathbf{C})\mathbf{D}].\end{aligned}$$

Using the similarity of  $\hat{\mathbf{C}}$  and  $\mathbf{C}$ ,

$$\hat{\mathbf{L}}_\Delta = (\mathbf{I} - r\theta\hat{\mathbf{C}})^{-1}[\mathbf{I} + r(1-\theta)\hat{\mathbf{C}}].$$

Taking the transpose of  $\hat{\mathbf{L}}_\Delta$  and utilizing the symmetry of  $\hat{\mathbf{C}}$ , we find

$$\hat{\mathbf{L}}_\Delta^T = (\mathbf{I} + r(1-\theta)\hat{\mathbf{C}})^T(\mathbf{I} - r\theta\hat{\mathbf{C}})^{-T} = (\mathbf{I} + r(1-\theta)\hat{\mathbf{C}})(\mathbf{I} - r\theta\hat{\mathbf{C}})^{-1}.$$

Now,

$$[(\mathbf{I} + r(1-\theta)\hat{\mathbf{C}})(\mathbf{I} - r\theta\hat{\mathbf{C}})]^{-1}(\mathbf{I} + r(1-\theta)\hat{\mathbf{C}})(\mathbf{I} - r\theta\hat{\mathbf{C}}) = \mathbf{I};$$

however,  $\mathbf{I} + r(1-\theta)\hat{\mathbf{C}}$  and  $\mathbf{I} - r\theta\hat{\mathbf{C}}$  commute, so

$$[(\mathbf{I} - r\theta\hat{\mathbf{C}})(\mathbf{I} + r(1-\theta)\hat{\mathbf{C}})]^{-1}(\mathbf{I} + r(1-\theta)\hat{\mathbf{C}})(\mathbf{I} - r\theta\hat{\mathbf{C}}) = \mathbf{I}$$

or

$$(\mathbf{I} + r(1-\theta)\hat{\mathbf{C}})^{-1}(\mathbf{I} - r\theta\hat{\mathbf{C}})^{-1}(\mathbf{I} + r(1-\theta)\hat{\mathbf{C}})(\mathbf{I} - r\theta\hat{\mathbf{C}}) = \mathbf{I}.$$

Thus,

$$(\mathbf{I} - r\theta\hat{\mathbf{C}})^{-1}(\mathbf{I} + r(1-\theta)\hat{\mathbf{C}}) = (\mathbf{I} + r(1-\theta)\hat{\mathbf{C}})(\mathbf{I} - r\theta\hat{\mathbf{C}})^{-1},$$

and  $\hat{\mathbf{L}}_\Delta = \hat{\mathbf{L}}_\Delta^T$ .

### 4.3 Multilevel Schemes and the Method of Lines

The one-level schemes that we have been studying only use data at  $t_n$  to obtain a solution at  $t_{n+1}$ . Multilevel schemes employ solutions at  $t_n$  and prior times to compute a solution at  $t_{n+1}$ . Aside from difficulties in starting them, they provide a mechanism for increasing accuracy without much additional computation. Multistep schemes are widely used to solve ordinary differential equations [12] and we expect them to be of similar value when solving partial differential equations. Let's again begin with the model heat conduction

equation (4.1.1a) and replace the time derivative by a centered difference approximation to obtain

$$(u_t)_j^n = \frac{u_j^{n+1} - u_j^{n-1}}{2\Delta t} + O(\Delta t^2). \quad (4.3.1)$$

Substituting (4.3.1) and the centered difference approximation (2.1.9) for the second spatial derivative into (4.1.1a) yields

$$(u_t)_j^n - \sigma(u_{xx})_j^n = \frac{u_j^{n+1} - u_j^{n-1}}{2\Delta t} - \sigma \frac{u_{j-1}^n - 2u_j^n + u_{j+1}^n}{\Delta x^2} + O(\Delta t^2) + O(\Delta x^2).$$

Neglecting the local discretization error and solving for  $U_j^{n+1}$  gives

$$U_j^{n+1} = U_j^{n-1} + 2r(U_{j-1}^n - 2U_j^n + U_{j+1}^n). \quad (4.3.2)$$

This centered time-centered space scheme is often called the *leap frog method*. Its stencil, shown on the left of Figure 4.3.1, indicates that data is needed at the two time levels  $n$  and  $n - 1$  in order to compute a solution at time level  $n + 1$ . Since initial data is only prescribed at time level 0, the leap frog method is not self starting. An independent method is needed to compute a solution at time level 1. Although the leap frog scheme involves solutions at three time levels, we'll call it a *two-level method* because it spans the two time intervals  $(t_{n-1}, t_n)$  and  $(t_n, t_{n+1})$ .

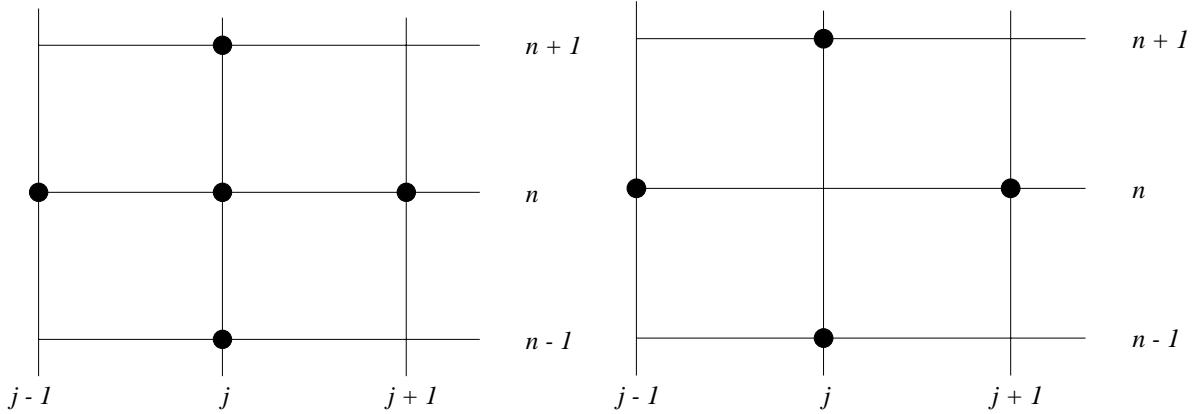


Figure 4.3.1: Computational stencils of the leap frog scheme (4.3.2) (left) and the Du Fort-Frankel scheme (4.3.8) (right).

Before discussing starting techniques, let us examine the stability of (4.3.2) using the von Neumann method. Thus, assume periodic initial data on  $0 \leq x < 2\pi$ , represent the

solution of (4.3.2) as the discrete Fourier series (4.1.16), substitute (4.1.16) into (4.3.2), and use the orthogonality condition (3.2.2) to obtain

$$A_k^{n+1} = A_k^{n-1} + 2r(e^{-2\pi ik/J} - 2 + e^{2\pi ik/J})A_k^n$$

or

$$A_k^{n+1} = A_k^{n-1} - 4r\mu_k A_k^n \quad (4.3.3a)$$

where

$$\mu_k = 1 - \frac{e^{-2\pi ik/J} + e^{2\pi ik/J}}{2} = 1 - \cos \frac{2k\pi}{J} = 2 \sin^2 \frac{k\pi}{J}. \quad (4.3.3b)$$

The difference equation (4.3.3a) is second order as opposed to the first-order schemes that we have been studying. Let us try, however, to obtain a solution having the same form as the first-order difference equations. Thus, assume

$$A_k^n = (M_k)^n c_k, \quad (4.3.4)$$

where  $M_k$  is, as usual, the amplification factor and  $c_k$  is a constant. Substituting (4.3.4) into (4.3.3a)

$$(M_k)^{n+1} = (M_k)^{n-1} - 4r\mu_k(M_k)^n.$$

Dividing by the common factor of  $(M_k)^{n-1}$  yields the quadratic equation

$$(M_k)^2 + 4r\mu_k M_k - 1 = 0,$$

which has the solution

$$M_k^\pm = -2r\mu_k \pm \sqrt{1 + (2r\mu_k)^2}. \quad (4.3.5)$$

The two amplification factors correspond to the two independent solutions of the second-order linear difference equation (4.3.3a). In parallel with the solution of linear ordinary differential equations, the general solution of (4.3.3a) is the linear combination

$$A_k^n = c_k^+(M_k^+)^n + c_k^-(M_k^-)^n. \quad (4.3.6)$$

of the two solutions. The constants  $c_k^\pm$  can be determined from the initial condition prescribing  $A_k^0$  and the independent starting method that is used to compute  $A_k^1$ . However, it is usually not necessary to explicitly determine  $c_k^\pm$ .

It is a bit difficult to ascertain the properties of the two amplification factors  $M_k^\pm$  from (4.3.5), so let us study their behavior when  $0 < r\mu_k \ll 1$ . Thus, expanding the square root term in a Taylor's series

$$M_k^+ = 1 - 2r\mu_k + O(r^2\mu_k^2), \quad M_k^- = -[1 + 2r\mu_k + O(r^2\mu_k^2)].$$

Substituting this into (4.3.6)

$$A_k^n = c_k^+[1 - 2r\mu_k + O(r^2\mu_k^2)]^n + c_k^-(-1)^n[1 + 2r\mu_k + O(r^2\mu_k^2)]^n. \quad (4.3.7)$$

The first term in (4.3.7) is an approximation of the solution of the heat conduction problem. It is decaying with increasing  $n$  when  $r\mu_k$  is sufficiently small. The second term in (4.3.7) increases in amplitude and oscillates as  $n$  increases. The growth of the solution can be restricted by maintaining  $r$  at a very small value; however, there is no value of  $r > 0$  for which the leap frog scheme (4.3.2) is absolutely stable. The second growing solution is called *parasitic* because, if  $n$  and/or  $r$  are large enough, it will grow to dominate the correct solution. Parasitic solutions will always be present when a higher-order difference scheme is used to approximate a lower-order differential equation. In order to be useful, the parasitic solutions of such multilevel schemes should be bounded well below the solution that is approximating the partial differential equation.

The Du Fort and Frankel [7] scheme

$$\frac{U_j^{n+1} - U_j^{n-1}}{2\Delta t} = \sigma \frac{U_{j-1}^n - (U_j^{n+1} + U_j^{n-1}) + U_{j+1}^n}{\Delta x^2}$$

or

$$(1 + 2r)U_j^{n+1} = (1 - 2r)U_j^{n-1} + 2r(U_{j-1}^n + U_{j+1}^n) \quad (4.3.8)$$

uses centered time differences with an averaged centered spatial difference approximation (Figure 4.3.1, right).

A von Neumann stability analysis reveals that the du Fort-Frankel scheme (4.3.8) is unconditionally stable in  $\mathcal{L}^2$  (*cf.* Problem 1 at the end of this section). This result seems

to be at odds with the discussion in Section 4.1 which implied that stability restrictions were necessary for explicit finite-difference approximations of the heat equation. The answer to the dilemma in this case is linked to a loss of consistency unless  $\Delta t \rightarrow 0$  at a faster rate than  $\Delta x$  (again, *cf.* Problem 1 at the end of this section). Should the computational mesh be refined with  $\Delta t/\Delta x = \gamma$  (a constant), then the du Fort-Frankel scheme (4.3.8) is consistent with the hyperbolic equation

$$u_t - \sigma u_{xx} + \sigma \gamma^2 u_{tt} = 0.$$

Thus, the time step of the du Fort-Frankel scheme must still be restricted to be  $O(\Delta x^2)$ ; however, now for reasons of consistency rather than for stability.

Multistep schemes can be started with a one-step method. We must be careful to preserve accuracy of the multistep scheme when the one-step method has a lower order. Suppose, for example, that the time step of a second-order multistep method is  $\Delta t_M$ , then the time step of the first-order starting method should be  $O(\Delta t_M^2)$  in order to keep the local errors in balance. This, however, is not the case when the forward time-centered space scheme (4.1.2) is used as a starting method for the du Fort-Frankel (4.3.8) scheme. Although the order of (4.1.2) is  $O(\Delta t) + O(\Delta x^2)$  and that of (4.3.8) is  $O(\Delta t^2) + O(\Delta x^2)$ , both schemes restrict  $\Delta t$  to  $O(\Delta x^2)$ . Thus, both schemes have  $O(\Delta x^2)$  accuracy.

### 4.3.1 Matrix Stability Analysis

When using matrix methods to analyze the stability of a multilevel scheme it is convenient to write the scheme as an equivalent one-level scheme. This is easily done and we'll illustrate it for the du Fort-Frankel scheme (4.3.8). For simplicity, consider an initial-boundary value problem for (4.3.8) with homogeneous Dirichlet boundary conditions, then the vector form of (4.3.8) is

$$\mathbf{U}^{n+1} = \mathbf{A}\mathbf{U}^{n-1} + \mathbf{B}\mathbf{U}^n, \quad (4.3.9a)$$

where

$$\mathbf{A} = \frac{1-2r}{1+2r} \mathbf{I}, \quad \mathbf{B} = \frac{2r}{1+2r} \begin{bmatrix} 0 & 1 & & \\ 1 & 0 & 1 & \\ & & \ddots & \\ & & & 1 & 0 \end{bmatrix}, \quad \mathbf{U}^n = \begin{bmatrix} U_1^n \\ U_2^n \\ \vdots \\ U_{J-1}^n \end{bmatrix}. \quad (4.3.9b)$$

Rewrite (4.3.9a) as

$$\begin{bmatrix} \mathbf{U}^{n+1} \\ \mathbf{U}^n \end{bmatrix} = \begin{bmatrix} \mathbf{B} & \mathbf{A} \\ \mathbf{I} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{U}^n \\ \mathbf{U}^{n-1} \end{bmatrix}.$$

In this form, the multilevel du Fort-Frankel scheme looks like the one-level scheme

$$\mathbf{W}^{n+1} = \mathbf{L}_\Delta \mathbf{W}^n, \quad (4.3.9c)$$

with

$$\mathbf{W}^n = \begin{bmatrix} \mathbf{U}^n \\ \mathbf{U}^{n-1} \end{bmatrix}, \quad \mathbf{L}_\Delta = \begin{bmatrix} \mathbf{B} & \mathbf{A} \\ \mathbf{A} & \mathbf{0} \end{bmatrix} \quad (4.3.9d)$$

Thus, matrix stability analysis and techniques described in Section 3.3 are directly applicable to multilevel difference methods using the matrix  $\mathbf{L}_\Delta$  of (4.3.9d).

### 4.3.2 The Method of Lines

Excellent software is available for solving ordinary differential equations. Modern multistep methods use codes containing either Adams or backward difference formulas [5, 12, 13] to integrate the system in time. Adams methods are preferred for nonstiff problems while backward difference methods are useful for stiff problems. In order to use the ordinary differential equations software to solve partial differential equations of the form

$$u_t = \mathcal{L}u, \quad (4.3.10a)$$

we introduce a spatial grid (as shown in Figure 4.3.2) and replace all of the spatial derivatives appearing in the operator  $\mathcal{L}$  by finite difference approximations to obtain

$$\dot{\mathbf{V}} = \mathbf{L}_{\Delta x} \mathbf{V}, \quad (4.3.10b)$$

where  $\mathbf{L}_{\Delta x}$  is the discrete approximation of  $\mathcal{L}$  and  $(\cdot)$  denotes time differentiation. The elements  $V_j(t)$  of  $\mathbf{V}(t)$  are approximations of  $u(x_j, t)$ . Let us illustrate the idea for a heat conduction problem.

*Example 4.3.1.* The spatial operator for the heat conduction equation (4.1.1a) is  $\mathcal{L}u = \sigma u_{xx}$ . Discretizing the second spatial derivative using centered differences gives

$$\dot{V}_j(t) = \sigma \frac{V_{j-1}(t) - 2V_j(t) + V_{j+1}(t)}{\Delta x^2}.$$

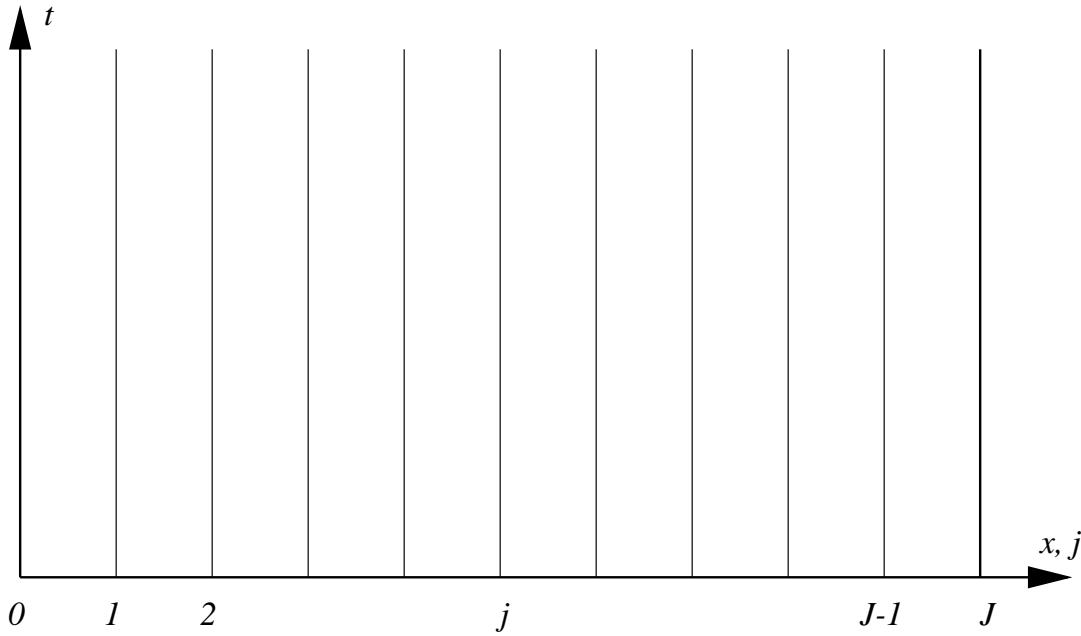


Figure 4.3.2: Spatial discretization to be used with the method of lines.

Consider a problem with homogeneous Dirichlet boundary conditions and let  $\mathbf{V}(t) = [V_1(t), V_2(t), \dots, V_{j-1}(t)]^T$ , then the matrix form of this problem is

$$\mathbf{L}_{\Delta x} = \frac{\sigma}{\Delta x^2} \begin{bmatrix} -2 & 1 & & & \\ 1 & -2 & 1 & & \\ & & \ddots & & \\ & & & 1 & -2 \end{bmatrix}.$$

The above arguments and example emphasize that the partial differential equation has been “reduced,” by spatial discretization, to a system of ordinary differential equations. The semi-discrete system (4.3.10b) can be integrated in time by most ordinary differential equations software. This software automatically selects time steps to satisfy a prescribed (local) temporal error tolerance and to maintain stability. Most multistep codes also adjust the order of accuracy of the formulas in order to improve performance. A user of the software would only have to provide a temporal error tolerance, initial values  $\mathbf{V}(0)$ , and a procedure for defining the spatially-discrete operator  $\mathbf{L}_{\Delta x}\mathbf{V}$ .

For heat conduction problems, the small divisor ( $\Delta x^2$ ) present in  $\mathbf{L}_{\Delta x}$  suggests that the ordinary differential equations (4.3.10b) will be stiff. These notes are not the appropriate place to initiate an extended discussion of “stiffness.” Let us simply state that stiff systems have more stringent stability restrictions than nonstiff systems. The accepted

$S$	1	2	3	4	5	6
$\beta_0$	1	2/3	6/11	12/25	60/137	60/147
$\alpha_0$	1	4/3	18/11	48/25	300/137	360/147
$\alpha_1$		-1/3	-9/11	-36/25	-300/137	-450/147
$\alpha_2$			2/11	16/25	200/137	400/147
$\alpha_3$				-3/25	-75/137	225/147
$\alpha_4$					12/137	72/147
$\alpha_5$						-10/147

Table 4.3.1: Coefficients of backward difference formulas (4.3.11) [11].

remedy is to use implicit ordinary differential equations software and the backward difference codes seem to be adequate for most problems [5, 11, 13].

The simplest backward difference method is the implicit or backward Euler method

$$\mathbf{V}^{n+1} = \mathbf{V}^n + \Delta t \dot{\mathbf{V}}^{n+1}.$$

Use of this method with the centered space differences of Example 4.3.1 yields the backward Euler method (4.1.4), *i.e.*,

$$V_j^{n+1} = V_j^n + \sigma \Delta t \left( \frac{V_{j-1}^{n+1} - 2V_j^{n+1} + V_{j+1}^{n+1}}{\Delta x^2} \right).$$

Higher-order backward difference formulas have the form

$$\mathbf{V}^{n+1} = \sum_{s=0}^{S-1} \alpha_s \mathbf{V}^{n-s} + \beta_0 \Delta t \dot{\mathbf{V}}^{n+1} \quad (4.3.11)$$

where the coefficients  $\alpha_s$ ,  $s = 0, 1, \dots, S$ , and  $\beta$  appear in Table 4.3.1 for methods of orders  $S = 1$  through 6 [11]. Use of the second-order formula with the centered spatial differences of Example 4.3.1 gives the scheme

$$V_j^{n+1} = \frac{4}{3} V_j^n - \frac{1}{3} V_j^{n-1} + \frac{2}{3} \sigma \Delta t \left( \frac{V_{j-1}^{n+1} - 2V_j^{n+1} + V_{j+1}^{n+1}}{\Delta x^2} \right).$$

Berzins and Furzeland [6] developed a system called *SPRINT* that utilizes method of lines integration with a variety of spatial discretization schemes. Modern method of lines software can also adjust the spatial mesh and order to satisfy prescribed spatial

accuracy criteria in an optimal manner. Adjerid and Flaherty [2, 3] describe a method that automatically moves the mesh in time while adding and deleting computational cells. Adjerid *et al.* [4] and Flaherty and Moore [9] additionally vary the spatial order of the method to achieve further gains in efficiency. Roughly speaking, they utilize high-order methods in regions where the solution is smooth and low-order methods with fine meshes near singularities. Adjerid *et al.* [1] consider multi-dimensional problems with automatic mesh refinement, method-order variation, and mesh motion for arbitrarily shaped regions. Recent developments are summarized in Fairweather and Gladwell [8]. Many of the articles in these proceedings describe mesh motion techniques that concentrate the mesh in regions of high solution activity. The following example illustrates such a procedure.

*Example 4.3.2.* Adjerid and Flaherty [3] consider the Buckley-Leverett system

$$u_t + f(u)_x = \epsilon u_{xx}, \quad 0 < x < 1, \quad t > 0, \quad (4.3.12a)$$

$$u(x, 0) = \frac{1}{10x + 1}, \quad 0 \leq x \leq 1, \quad (4.3.12b)$$

$$u(0, t) = 1, \quad u_x(1, t) = 0, \quad t > 0, \quad (4.3.12c)$$

$$f(u) = \frac{u^2}{u^2 + a(1 - u^2)}. \quad (4.3.12d)$$

This model has been used to describe the simultaneous flow of two immiscible fluids through a porous medium neglecting capillary pressure and gravitational forces. The solution  $u(x, t)$  represents the concentration of one of the species,  $\epsilon$  is a viscosity parameter taken as  $10^{-3}$  and  $a = 1/2$ .

Adjerid and Flaherty [3] solved this problem on a moving mesh using piecewise linear finite element approximations. In this case, the finite element technique yields approximately the same discrete system as centered differences. The problem is nonlinear and centered-difference approximations of such systems is discussed in Section 4.4. Despite these uncertainties, let us concentrate on their mesh moving scheme

$$\dot{x}_j - \dot{x}_{j-1} = -\lambda(W_j - \bar{W}), \quad j = 1, 2, \dots, J-1, \quad (4.3.13a)$$

where a superimposed dot denotes time differentiation,  $x_j(t)$  is the position of the  $j$ th mesh point at time  $t$ ,  $\lambda > 0$  is a parameter to be chosen,  $W_j$  is a mesh motion indicator on the subinterval  $(x_{j-1}, x_j)$ , and  $\bar{W}$  is the average of  $W_j$ ,  $j = 1, 2, \dots, J$ .

If  $W_j > \bar{W}$ , the right side of (4.3.13a) is negative and the mesh points  $x_j$  and  $x_{j-1}$  move closer to each other. The opposite is true when  $W_j < \bar{W}$ . Neglecting the “boundary conditions”

$$x_0 = 0, \quad x_J = 1, \quad (4.3.13b)$$

the nodes tend to an “equidistributing mesh” where  $W_j = \bar{W}$ ,  $j = 1, 2, \dots, J$ , as  $t \rightarrow \infty$ . The parameter  $\lambda$  controls the relaxation time to equidistribution. Large values of  $\lambda$  produce shorter relaxation times but introduce stiffness into the system (4.3.13a). Smaller values of  $\lambda$  produce a less stiff system that may not be able to follow some rapid dynamic phenomena. Adjerid and Flaherty [3] discuss an automatic procedure for selecting  $\lambda$  that balances stiffness and responsiveness.

The motion indicator is a solution-dependent parameter that should be large where the mesh should be fine and small where it should be coarse. It could be chosen proportional to a solution derivative, *e.g.*,

$$W_j(t_n) = \max_{x \in (x_{j-1}, x_j)} |u_x(x, t_n)| \approx \frac{|U_j^n - U_{j-1}^n|}{x_j(t_n) - x_{j-1}(t_n)}.$$

Choosing  $W_j$  proportional to the second derivative of  $u$  is also popular and, in most cases, preferable. When using second-order schemes, the gradient could be large while the actual discretization error is small. A large second derivative is a better indicator of large error. Adjerid and Flaherty [2] used an estimate of the spatial discretization error as a mesh motion indicator. In particular, the results shown in Figure 4.3.3 were obtained with

$$W_j(t) = \int_{x_{j-1}}^{x_j} |E(x, t)| dx$$

where  $E(x, t)$  is an estimate of the spatial discretization error. We will discuss methods for obtaining such errors later in this section.

The differential equations (4.3.13a) for the mesh motion can be solved simultaneously with the spatially discrete version of (4.3.12) using the ordinary differential equations soft-

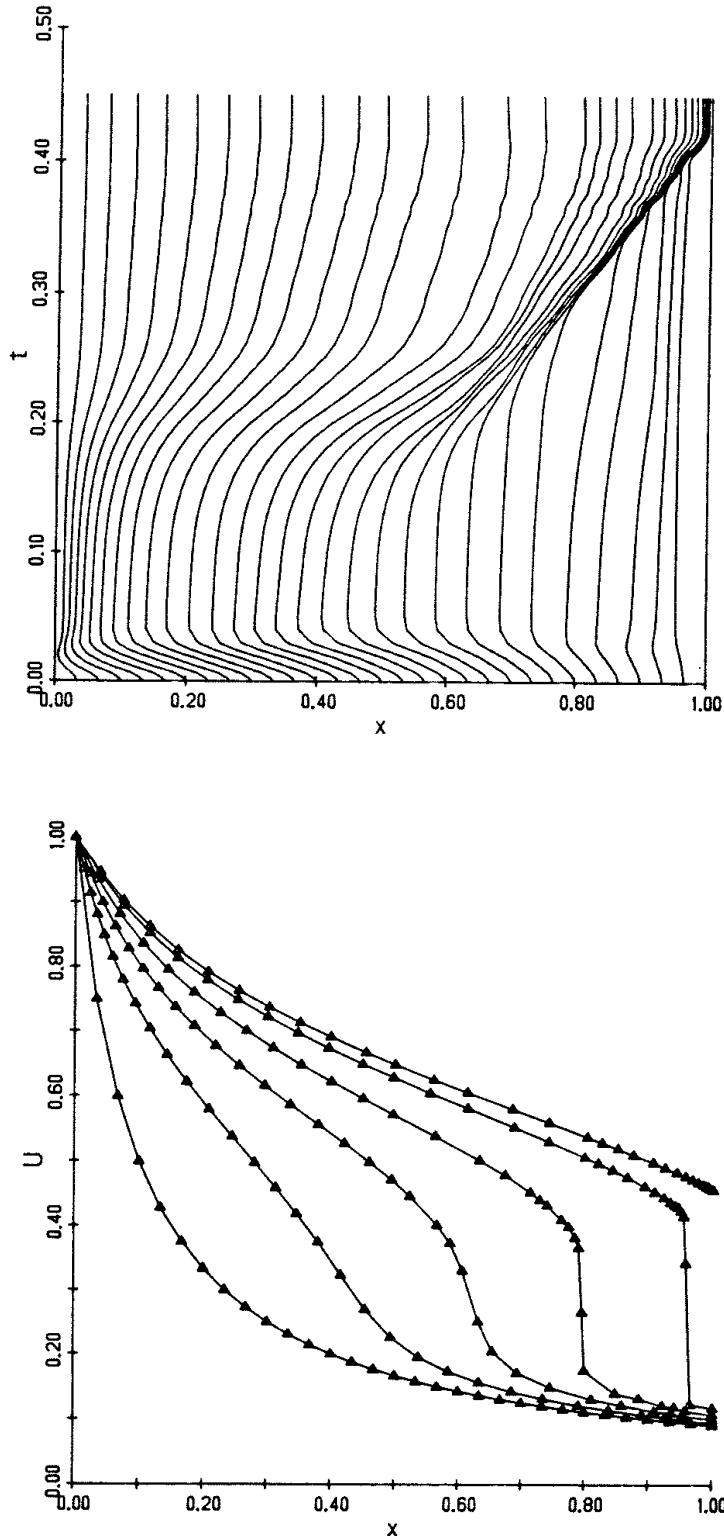


Figure 4.3.3: Method of lines solution of (4.3.12) (bottom) on a moving 30-element mesh (top) [2].

ware. The global parameter  $\bar{W}$  can be eliminated from (4.3.13a) by combining equations on two successive intervals to obtain

$$\dot{x}_{j-1} - 2\dot{x}_j + \dot{x}_{j+1} = -\lambda(W_{j+1} - W_j), \quad j = 1, 2, \dots, J-1.$$

The mesh trajectories and solution shown in Figure 4.3.3 used a mesh with  $J = 30$ . The mesh is concentrated in the vicinity of the sharp front and following it with approximately the correct velocity. Comparable accuracy with a uniform stationary mesh would require more than 1000 elements.

### 4.3.3 A Posteriori Error Estimation

*A priori* estimates of local discretization errors involve unknown derivatives of the exact solution. They are useful for ascertaining convergence rates, but rarely provide quantitative information. *A posteriori* error estimates use a computed solution to provide quantitative information. The two common techniques to obtain a posteriori error estimates involve comparing solutions on different meshes and comparing solutions of different orders. We will sketch the latter technique and describe the former in greater detail.

Suppose that we have two methods with local errors

$$u_j^{n+1} - U_j^{n+1} = \Delta t[c_1 \Delta t^p + c_2 \Delta x^p + O(\Delta t^{p+1}) + O(\Delta x^{p+1})] \quad (4.3.14a)$$

$$u_j^{n+1} - V_j^{n+1} = \Delta t[d_1 \Delta t^{p+1} + d_2 \Delta x^{p+1} + O(\Delta t^{p+2}) + O(\Delta x^{p+2})]. \quad (4.3.14b)$$

The spatial and temporal orders have been equated for simplicity. The technique that we are about to present works when these orders differ.

Adding and subtracting the higher order-solution yields

$$u_j^{n+1} - U_j^{n+1} = (u_j^{n+1} - V_j^{n+1}) + (V_j^{n+1} - U_j^{n+1}). \quad (4.3.15a)$$

Using (4.3.14) we see that the term on the left of (4.3.15a) is  $\Delta t(O(\Delta t^p) + O(\Delta x^p))$  and the first term on the right is  $\Delta t(O(\Delta t^{p+1}) + O(\Delta x^{p+1}))$ . Thus, the second term on the

right must be  $\Delta t(O(\Delta t^p) + O(\Delta x^p))$ . Hence, we can neglect the first term on the right to obtain the approximation

$$u_j^{n+1} - U_j^{n+1} \approx V_j^{n+1} - U_j^{n+1}. \quad (4.3.15b)$$

Thus, the difference between the lower- and higher-order solutions furnishes an estimate of the error of the lower-order solution. Unfortunately, the technique provides no error estimate of the higher-order solution, but it is common to propagate it forward in time rather than propagating the lower-order solution. This strategy is called *local extrapolation*. It is generally acceptable in regions where solutions are smooth, but can be dangerous near singularities.

Unless there are special properties of the difference scheme that can be exploited, the work involved in obtaining the error estimate is comparable to that of obtaining the solution. It's possible that the higher-order method can just provide a "correction" to the lower-order method. In this case, the work to obtain the lower-order solution need not be duplicated when obtaining the higher-order solution. This is called *order embedding* or *p-refinement*. There are often special *super convergence* points where the error converges at a faster rate than it does elsewhere. Knowledge of these points can reduce the computational cost of obtaining the error estimate [4].

The technique of using solutions computed on different meshes to obtain an *a posteriori* error estimate is called *Richardson's extrapolation* [16] or *h-refinement*. Suppose that a solution has been obtained using scheme (4.3.14a). Obtain a second solution using half the time and spatial step sizes, but starting from  $t_n$ . The error of the second solution, which we'll call  $\hat{U}_j^{n+1}$ , is obtained from (4.3.14a) as

$$u_j^{n+1} - \hat{U}_j^{n+1} = 2\frac{\Delta t}{2}[c_1(\frac{\Delta t}{2})^p + c_2(\frac{\Delta x}{2})^p + O(\Delta t^{p+1}) + O(\Delta x^{p+1})]$$

or

$$u_j^{n+1} - \hat{U}_j^{n+1} = \frac{\Delta t}{2^p}[c_1\Delta t^p + c_2\Delta x^p + O(\Delta t^p + 1) + O(\Delta x^p + 1)] \quad (4.3.16)$$

*Remark 1.* The indexing of this second (fine-mesh) solution should have been changed. Thus,  $\hat{U}_j^{n+1}$  corresponds to  $U_{2j}^{n+2}$ . We haven't changed the indexing to emphasize that

the two solutions are sampled at the same points. Additionally, the mesh can be changed by other than a factor of two, but more effort is involved.

Subtracting (4.3.16) from (4.3.14a) yields

$$\hat{U}_j^{n+1} - U_j^{n+1} = \left(1 - \frac{1}{2^p}\right)\Delta t[c_1\Delta t^p + c_2\Delta x^p] + \Delta t(O(\Delta t^{p+1}) + O(\Delta x^{p+1})).$$

Neglecting the higher-order terms yields

$$\Delta t[c_1\Delta t^p + c_2\Delta x^p] \approx \frac{\hat{U}_j^{n+1} - U_j^{n+1}}{1 - \frac{1}{2^p}}.$$

Thus, the error of the coarse-mesh solution (4.3.14a) may be estimated as

$$u_j^{n+1} - U_j^{n+1} \approx \frac{\hat{U}_j^{n+1} - U_j^{n+1}}{1 - \frac{1}{2^p}}. \quad (4.3.17)$$

Richardson's extrapolation can also be used to estimate the error of the fine-mesh solution  $\hat{U}_j^{n+1}$ , but only at the coarse mesh points. The work of obtaining the error estimate is approximately four times the work of obtaining the solution. This is usually considered excessive unless the fine-mesh solution is propagated forward in time. As with order embedding, this can be risky near singularities.

*Example 4.3.3.* Let us solve the kinematic wave equation

$$u_t + au_x = 0$$

by the forward time-backward space scheme (2.2.4)

$$U_j^{n+1} = (1 - \alpha)U_j^n + \alpha U_{j-1}^n.$$

As usual  $\alpha = a\Delta t/\Delta x$  is the Courant number. We choose  $a = 1$ ,  $\alpha = 1/2$ , and the periodic initial data

$$u(x, 0) = \sin x$$

Thus, the exact solution is

$$u(x, t) = \sin(x - t).$$

Since this scheme is first order accurate, we obtain error estimates by setting  $p = 1$  in (4.3.17). This gives

$$u_j^{n+1} - U_j^{n+1} \approx E_j^{n+1} = 2(\hat{U}_j^{n+1} - U_j^{n+1}).$$

Let us compare exact and estimated local errors by solving this problem for a single time step on  $0 < x < 2\pi$  using meshes with spacing  $\Delta x = 2\pi/J$ ,  $J = 4, 8, 16$ . Results for the exact errors and those estimated by Richardson's extrapolation, presented in Table 4.3.3, indicate no differences between the two. Indeed, the estimated errors are so accurate for this problem that when added to the computed solution with  $J = 4$  they produce a maximum error of  $0.2220 \cdot 10^{-15}$ . Results with less smooth solutions would not normally be this good, but, nevertheless, there are decided advantages to using Richardson's extrapolation.

$J$	$\ \mathbf{u}^1 - \mathbf{U}^1\ _\infty$	$\ \mathbf{E}^1\ _\infty$
4	0.2071	0.2071
8	0.0703	0.0703
16	0.0188	0.0188

Table 4.3.2: Exact and estimated local errors for Example 4.3.3.

### Problems

1. ([15], p. 91.) Let us consider the Du Fort-Frankel scheme (4.3.8) applied to the heat conduction equation (4.1.1a).
  - 1.1. Show that (4.3.8) satisfies the von Neumann necessary condition for stability for all  $r > 0$ .
  - 1.2. Calculate the leading terms of the local discretization error of (4.3.8) and examine the scheme's consistency. If the scheme is conditionally consistent, clearly state the conditions when it is and is not consistent.
2. ([15], p. 45.) Derive the following “summation by parts” formulas assuming that  $a_0 = a_J = b_0 = b_J = 0$ .
  - 2.1. 
$$\sum_{j=1}^{J-1} a_j(b_j - b_{j-1}) = - \sum_{j=1}^{J-1} b_j(a_{j+1} - a_j).$$
  - 2.2. 
$$\sum_{j=1}^{J-1} a_j(b_{j+1} - b_j) = - \sum_{j=1}^{J-1} b_{j+1}(a_{j+1} - a_j) - a_1 b_1.$$

2.3.

$$\sum_{j=1}^{J-1} a_j(b_{j+1} - 2b_j + b_{j-1}) = - \sum_{j=1}^{J-1} (b_{j+1} - b_j)(a_{j+1} - a_j) - a_1 b_1.$$

## 4.4 Variable-Coefficient and Nonlinear Problems

The methods presented in the preceding sections and chapters can be applied to variable-coefficient and nonlinear problems. Variable-coefficient problems pose virtually no computational difficulties. Likewise, explicit nonlinear problems lead to simple algebraic problems; however, implicit schemes will generally require an iterative solution technique. Analyses of consistency, convergence, and stability are more difficult for both variable-coefficient and nonlinear problems than they are for the constant-coefficient cases that we have been studying. If coefficients and/or solutions are smooth, one can often invoke a “local analysis,” which is a constant-coefficient analysis based on local values of the coefficients.

### 4.4.1 Linear Variable-Coefficient Problems

Let us begin with some examples.

*Example 4.4.1.* Consider the heat conduction problem

$$u_t = \sigma u_{xx} + au_x + bu + f, \quad (4.4.1)$$

where  $\sigma$ ,  $a$ ,  $b$ , and  $f$  are functions of  $x$  and  $t$ . The explicit forward time-centered space scheme for this problem would be

$$\frac{U_j^{n+1} - U_j^n}{\Delta t} = \sigma_j^n \frac{\delta^2 U_j^n}{\Delta x^2} + a_j^n \frac{\mu \delta U_j^n}{\Delta x} + b_j^n U_j^n + f_j^n,$$

where  $\sigma_j^n = \sigma(j\Delta x, n\Delta t)$ , etc. on a uniform mesh of spacing  $(\Delta x \times \Delta t)$ . Solving, as usual, for  $U_j^{n+1}$  using the definitions (Table 2.1.1) of the averaging  $\mu$  and central difference  $\delta$  operators, we find

$$U_j^{n+1} = c_{-1} U_{j-1}^n + c_0 U_j^n + c_1 U_{j+1}^n + \Delta t f_j^n, \quad (4.4.2a)$$

where

$$c_{-1} = (\sigma_j^n - \frac{\Delta x a_j^n}{2}) \frac{\Delta t}{\Delta x^2}, \quad (4.4.2b)$$

$$c_0 = 1 - (2\sigma_j^n - \Delta x^2 b_j^n) \frac{\Delta t}{\Delta x^2}, \quad (4.4.2c)$$

$$c_1 = (\sigma_j^n + \frac{\Delta x a_j^n}{2}) \frac{\Delta t}{\Delta x^2}. \quad (4.4.2d)$$

As noted, this scheme is used in exactly the same manner as described for the constant-coefficient problems studied earlier. Questions of consistency, convergence, and stability will be addressed shortly.

*Example 4.4.2.* Crank-Nicolson schemes for (4.4.1) offer some possibilities for variation. One possibility is to center the coefficients at  $(j, n + 1/2)$  and average the solution to get

$$\frac{U_j^{n+1} - U_j^n}{\Delta t} = [\sigma_j^{n+1/2} \frac{\delta^2}{\Delta x^2} + a_j^{n+1/2} \frac{\mu\delta}{\Delta x} + b_j^{n+1/2}] \left( \frac{U_j^{n+1} + U_j^n}{2} \right) + f_j^{n+1/2}. \quad (4.4.3a)$$

The symmetry of the discrete operator about  $(n + 1/2)\Delta t$  will give an  $O(\Delta t^2)$  local error in time. Centering about  $j\Delta x$ , as in the past, gives an  $O(\Delta x^2)$  local spatial error.

A second Crank-Nicolson scheme can be obtained by averaging the spatial operator on the right side of (4.4.1) to obtain

$$\begin{aligned} \frac{U_j^{n+1} - U_j^n}{\Delta t} &= \frac{1}{2} [(\sigma_j^{n+1} \frac{\delta^2}{\Delta x^2} + a_j^{n+1} \frac{\mu\delta}{\Delta x} + b_j^{n+1}) U_j^{n+1} + f_j^{n+1}] \\ &\quad + \frac{1}{2} [(\sigma_j^n \frac{\delta^2}{\Delta x^2} + a_j^n \frac{\mu\delta}{\Delta x} + b_j^n) U_j^n + f_j^n] \end{aligned} \quad (4.4.3b)$$

The form (4.4.3a) is analogous to midpoint quadrature while (4.4.3b) is analogous to the trapezoidal rule.

*Example 4.4.3.* The diffusion term in the heat conduction equation frequently appears in the *self-adjoint* form  $(\sigma u_x)_x$ . Problems in cylindrical coordinates, for example, would have the form  $((1/x)u_x)_x$ . Such expressions could clearly be expanded to  $\sigma u_x x + \sigma_x u_x$  and approximated as in Example 4.4.1; however, one would have to know  $\sigma_x$ . A better alternative is to approximate the self-adjoint diffusive term directly. For example, using central differences yields

$$\frac{\partial}{\partial x} (\sigma \frac{\partial u}{\partial x})|_j^n \approx \frac{\delta(\sigma_j^n \delta U_j^n)}{\Delta x^2} = \frac{\sigma_{j+1/2}^n U_{j+1}^n - (\sigma_{j+1/2}^n + \sigma_{j-1/2}^n) U_j^n + \sigma_{j-1/2}^n U_{j-1}^n}{\Delta x^2}. \quad (4.4.4)$$

The consistency, convergence, and stability of these variable-coefficient difference schemes must be established. In order to simplicity this discussion, let us suppose that (4.4.1) is solved by a general explicit finite difference scheme of the form

$$U_j^{n+1} = \sum_{|s| \leq S} c_s U_{j+s}^n + \Delta t f_j^n. \quad (4.4.5)$$

The coefficients  $c_s$ ,  $|s| \leq S$ , can depend on  $j$ ,  $n$ ,  $\Delta x$ , and  $\Delta t$ . A similar development is possible for implicit schemes.

Recall that (*cf.* Definitions 3.1.2, 3) consistency implies that the local error

$$u_j^{n+1} - U_j^{n+1} = -\Delta t \tau_j^n = u_j^{n+1} - \sum_{|s| \leq S} c_s u_{j+s}^n - \Delta t f_j^n \quad (4.4.6)$$

tend to zero faster than  $O(\Delta t)$ . Expanding (4.4.6) in a Taylor's series about  $(j, n)$  gives

$$-\Delta t \tau_j^n = u_j^n + \Delta t (u_t)_j^n + O(\Delta t^2) - \sum_{|s| \leq S} c_s [u_j^n + s \Delta x (u_x)_j^n + \frac{s^2 \Delta x^2}{2} (u_{xx})_j^n + O(\Delta x^3)] - \Delta t f_j^n.$$

Let us use the differential equation (4.4.1) to eliminate  $u_t$  in favor of spatial derivatives to obtain

$$\begin{aligned} -\Delta t \tau_j^n &= u_j^n + \Delta t [\sigma_j^n (u_{xx})_j^n + a_j^n (u_x)_j^n + b_j^n u_j^n + f_j^n] + O(\Delta t^2) \\ &\quad - \sum_{|s| \leq S} c_s [u_j^n + s \Delta x (u_x)_j^n + \frac{s^2 \Delta x^2}{2} (u_{xx})_j^n + O(\Delta x^3)] - \Delta t f_j^n. \end{aligned}$$

Regrouping terms

$$\begin{aligned} -\Delta t \tau_j^n &= [1 - \sum_{|s| \leq S} c_s + \Delta t b_j^n] u_j^n + [-\Delta x \sum_{|s| \leq S} s c_s + \Delta t a_j^n] (u_x)_j^n + \\ &\quad [-\Delta x^2 \sum_{|s| \leq S} \frac{s^2 c_s}{2} + \Delta t \sigma_j^n] (u_{xx})_j^n + O(\Delta t^2) + O(\Delta x^3). \end{aligned}$$

Enforcing that  $\tau_j^n \rightarrow 0$  as  $\Delta x, \Delta t \rightarrow 0$  yields the consistency conditions

$$\lim_{\Delta x, \Delta t \rightarrow 0} \frac{1}{\Delta t} [\sum_{|s| \leq S} c_s - 1] = b_j^n, \quad (4.4.7a)$$

$$\lim_{\Delta x, \Delta t \rightarrow 0} \frac{\Delta x}{\Delta t} \sum_{|s| \leq S} s c_s = a_j^n, \quad (4.4.7b)$$

$$\lim_{\Delta x, \Delta t \rightarrow 0} \frac{\Delta x^2}{\Delta t} \sum_{|s| \leq S} \frac{s^2 c_s}{2} = \sigma_j^n. \quad (4.4.7c)$$

Additional terms in the Taylor's series expansion may give rise to other consistency conditions if, *e.g.*, mesh ratios such as  $\Delta x^3/\Delta t$  did not approach zero. However, as we have seen, it's usually necessary to require  $\Delta x^2/\Delta t$  to be bounded for an explicit scheme. Thus, one would hardly compute with the smaller  $\Delta t = O(\Delta x^3)$ .

Stability is usually governed by the highest order terms in the difference equation. Let us demonstrate this by performing a von Neumann analysis of the explicit scheme (4.4.5) assuming that coefficients are constant ( $c_s$ ,  $|s| \leq S$ , are independent of  $j$  and  $n$ ),  $f_j^n = 0$ , and computation is performed with constant  $\Delta t/\Delta x^2$ . Under these restrictions, the von Neumann analysis produces the amplification factor

$$M_k = \sum_{|s| \leq S} c_s \omega_s^k, \quad \omega_s = e^{2\pi i s/J}. \quad (4.4.8)$$

Suppose the coefficients  $c_s$ ,  $|s| \leq S$ , are smooth enough to have series expansions in powers of  $\Delta t^{1/2}$  of the form

$$c_s = c_s^0 + \Delta t^{1/2} c_s^{1/2} + \Delta t c_s^1 + \dots \quad (4.4.9)$$

Substituting (4.4.9) into the consistency conditions (4.4.7) yields an alternate set of consistency conditions having the form

$$\sum_{|s| \leq S} c_s^0 = 1, \quad \sum_{|s| \leq S} c_s^{1/2} = 0, \quad \sum_{|s| \leq S} c_s^1 = b, \quad (4.4.10a)$$

$$\sum_{|s| \leq S} s c_s^0 = 0, \quad \sqrt{\frac{\Delta x^2}{\Delta t}} \sum_{|s| \leq S} s c_s^{1/2} = a, \quad (4.4.10b)$$

$$\frac{\Delta x^2}{\Delta t} \sum_{|s| \leq S} \frac{s^2 c_s^0}{2} = \sigma. \quad (4.4.10c)$$

The subscript  $j$  and superscript  $n$  on the coefficients  $a$ ,  $b$ , and  $\sigma$  have been omitted for this constant coefficient analysis.

*Remark 1.* We encountered the first consistency condition in (4.4.10) in conjunction with the Maximum Principle (Theorem 3.1.1).

Substituting the expansions (4.4.9) into (4.4.8) yields an expansion of the amplification factor as

$$M_k = M_k^0 + \Delta t^{1/2} M_k^{1/2} + \Delta t M_k^1 + \dots, \quad (4.4.11a)$$

where

$$M_k^l = \sum_{|s| \leq S} c_s^l \omega_s^k, \quad l = 0, 1, \dots. \quad (4.4.11b)$$

Using (4.4.10), we see that the leading term  $M_k^0$  is independent of  $a$  and  $b$ . Thus,  $M_k^0$  is the amplification factor of the differential equation

$$u_t = \sigma u_{xx}.$$

Suppose that the principal part of the operator is stable, *i.e.*, there exists a constant  $m_k^0$  such that

$$|M_k^0| \leq 1 + m_k^0 \Delta t.$$

For simplicity, let us assume that  $S$  is small relative to  $J$  and expand  $\omega_s^k$  in a series. Using (4.4.10) and (4.4.11b), this gives

$$\begin{aligned} M_k^0 &= 1 - \frac{\sigma \Delta t}{\Delta x^2} \left( \frac{2\pi k}{J} \right)^2 + O\left(\left(\frac{2\pi k S}{J}\right)^3\right), \\ M_k^{1/2} &= ia \sqrt{\frac{\Delta t}{\Delta x^2}} \left( \frac{2\pi k}{J} \right) + O\left(\left(\frac{2\pi k S}{J}\right)^3\right). \end{aligned}$$

An expansion of  $M_k^1$  is not needed. These equations imply that  $M_k^0$  is real and  $M_k^{1/2}$  is imaginary. Assuming this to be the case and using (4.4.11a) gives

$$|M_k| = (|M_k^0|^2 + \Delta t |M_k^{1/2}|^2 + \dots)^{1/2}.$$

Expanding the square root

$$|M_k| \leq 1 + O(\Delta t).$$

Thus, the difference scheme (4.4.5) satisfies the von Neumann condition and is stable according to Theorem 3.2.1. The proof of stability when  $M_k^0$  and  $M_k^1$  are complex follows similar lines [17].

The previous discussion quantify the effects of lower-order terms on stability when the coefficients are constant. Richtmyer and Morton [17], Chapter 5, show that a constant coefficient stability analysis with local values of the coefficients is sufficient for the stability of a variable coefficient problem when (i) (4.4.5) is consistent and (ii) the coefficients  $c_s$ ,  $|s| \leq S$ , are continuous and uniformly bounded on the domain of the differential equation as  $\Delta t, \Delta x \rightarrow 0$ . The local constant-coefficient analysis requires that stability be checked for all values of the coefficients in the domain. Consider, for example, solving a heat conduction problem with a variable diffusivity  $\sigma(x)$  for  $x \in (0, 1)$ . Using the explicit forward time-centered space scheme, we would have to show that

$$r = \frac{\sigma(x)\Delta t}{\Delta x^2} \leq \frac{1}{2}, \quad \forall x \in (0, 1).$$

#### 4.4.2 Convection-Diffusion Problems

The prior arguments imply that the lower-order terms in a parabolic differential equations do not appreciably affect stability; however, caution is needed when absolute stability is required. Consider the convection-diffusion equation

$$u_t = \sigma u_{xx} + au_x \quad (4.4.12)$$

which arises throughout fluid mechanics. The equation clearly has properties of both the kinematic wave equation (convection) and the heat equation (diffusion) and we would expect it to behave more like one or the other depending on the relative sizes of  $a$  and  $\sigma$ . Let's discretize this equation by the forward time-centered space difference approximation to obtain

$$U_j^{n+1} = U_j^n + r(U_{j-1}^n - 2U_j^n + U_{j+1}^n) + \frac{\alpha}{2}(U_{j+1}^n - U_{j-1}^n) \quad (4.4.13)$$

where  $\alpha$  is the Courant number.

It is not hard to show that solutions of initial value problems for (4.4.12) with periodic or compact data do not grow in time. Thus, it is reasonable to require

$$\|\mathbf{U}^n\| \leq \|\mathbf{U}^0\|.$$

Stability analyses involving only the higher-order terms ( $a = 0$ ) predict absolute stability when  $r \leq 1/2$ . We have just seen that this ensures stability when  $a \neq 0$ ; however, it does

not ensure absolute stability. Amplification factors may become larger than unity causing solutions to grow beyond their initial size. Maintaining no solution growth requires the additional condition

$$|P_e| \equiv \frac{|a|\Delta x}{2\sigma} \leq 1, \quad (4.4.14)$$

where  $P_e$  is the *cell Peclet* or *cell Reynolds number*. To verify (4.4.14) in the maximum norm, write (4.4.13) in the usual form

$$U_j^{n+1} = r(1 - P_e)U_{j-1}^n + (1 - 2r)U_j^n + r(1 + P_e)U_{j+1}^n$$

and invoke the Maximum Principle. All coefficients add to unity and are positive if  $r \leq 1/2$  and  $|P_e| \leq 1$ . The same condition may be verified in  $\mathcal{L}^2$  by using a von Neumann analysis. (The von Neumann analysis may also be inferred from the solution of Problem 3.2.3.)

If  $\sigma$  is small relative to  $|a|$ , maintaining  $|P_e| \leq 1$  will require a very fine mesh spacing. We could have anticipated difficulties when  $\sigma$  is small since the forward time-centered space scheme is not absolutely stable for the kinematic wave equation that results when  $\sigma = 0$ . With this view, it may be better to replace the centered-difference approximation of  $au_x$  by forward differences when  $a$  is positive.<sup>1</sup> Thus,

$$U_j^{n+1} = U_j^n + r(U_{j-1}^n - 2U_j^n + U_{j+1}^n) + \frac{a\Delta t}{2\Delta x}(U_{j+1}^n - U_j^n)$$

or

$$U_j^{n+1} = rU_{j-1}^n + [1 - 2r(1 + P_e)]U_j^n + r(1 + 2P_e)U_{j+1}^n \quad (4.4.15)$$

With  $P_e > 0$  for  $a > 0$ , use of the Maximum Principle states that there will be no growth if  $1 - 2r(1 + P_e) < 1$  or

$$2r(1 + P_e) = 2r + \alpha \leq 1. \quad (4.4.16)$$

This condition is typically much less restrictive than (4.4.14).

---

<sup>1</sup> $a$  is on the opposite side of the equation from our usual form of the kinematic wave equation.

The use of forward or backward differencing for the convective term of the convection-diffusion equation is called *upwind differencing*. When  $a$  can change sign, we may write (4.4.15) in the form

$$U_j^{n+1} = U_j^n + \frac{\beta}{2}(U_{j-1}^n - 2U_j^n + U_{j+1}^n) + \frac{\alpha}{2}(U_{j+1}^n - U_{j-1}^n) \quad (4.4.17a)$$

where

$$\beta = 2r + |\alpha|. \quad (4.4.17b)$$

(The von Neumann analysis of Problem 3.2.3 may again be used to study  $\mathcal{L}^2$  stability.) Notice that the “diffusion coefficient”  $\beta/2 = r + |\alpha|/2$  has increased relative to its value of  $r$  for the forward time-centered space scheme (4.4.13). This diffusion is often called artificial since it depends on mesh parameters rather than physical parameters.

*Example 4.4.4.* Let us apply the forward time-centered space (4.4.13) and upwind (4.4.17) schemes to a convection-diffusion problem with  $a = 1$ ,  $\sigma = 0.01$ , and the initial and boundary conditions

$$u(x, 0) = \phi(x) = \begin{cases} 0, & \text{if } 0 \leq x < 1/2 \\ 1, & \text{if } 1/2 \leq x \leq 1 \end{cases},$$

$$u(0, t) = 0, \quad u(1, t) = 1.$$

The initial square pulse propagates from  $x = 1/2$  to  $x = 0$  while diffusing to form a steady “boundary layer” at  $x = 0$  where the solution transitions from near unity to zero.

We choose  $J = 20$ , so that  $\Delta x = 0.05$ , and  $\Delta t = 0.025$ . Thus,  $\alpha = 0.5$ ,  $r = 0.1$ ,  $P_e = 2.5$ , and  $\beta = 0.7$ . With these parameters, the absolute stability condition (4.4.16) for the upwind scheme is satisfied, but condition (4.4.14) for the centered scheme is not.

The computed solutions for both the centered and upwind schemes are shown in Figure 4.4.1. The solution with centered differencing (4.4.13) of the convection term exhibits spurious oscillations. As noted, the scheme is stable but not absolutely stable. There is some growth of the initial data beyond unity. The upwind scheme is absolutely stable but only first-order accurate. In this example, this is seen through the excess diffusion at the moving front. Thus, with upwind differencing, the initial square pulse is

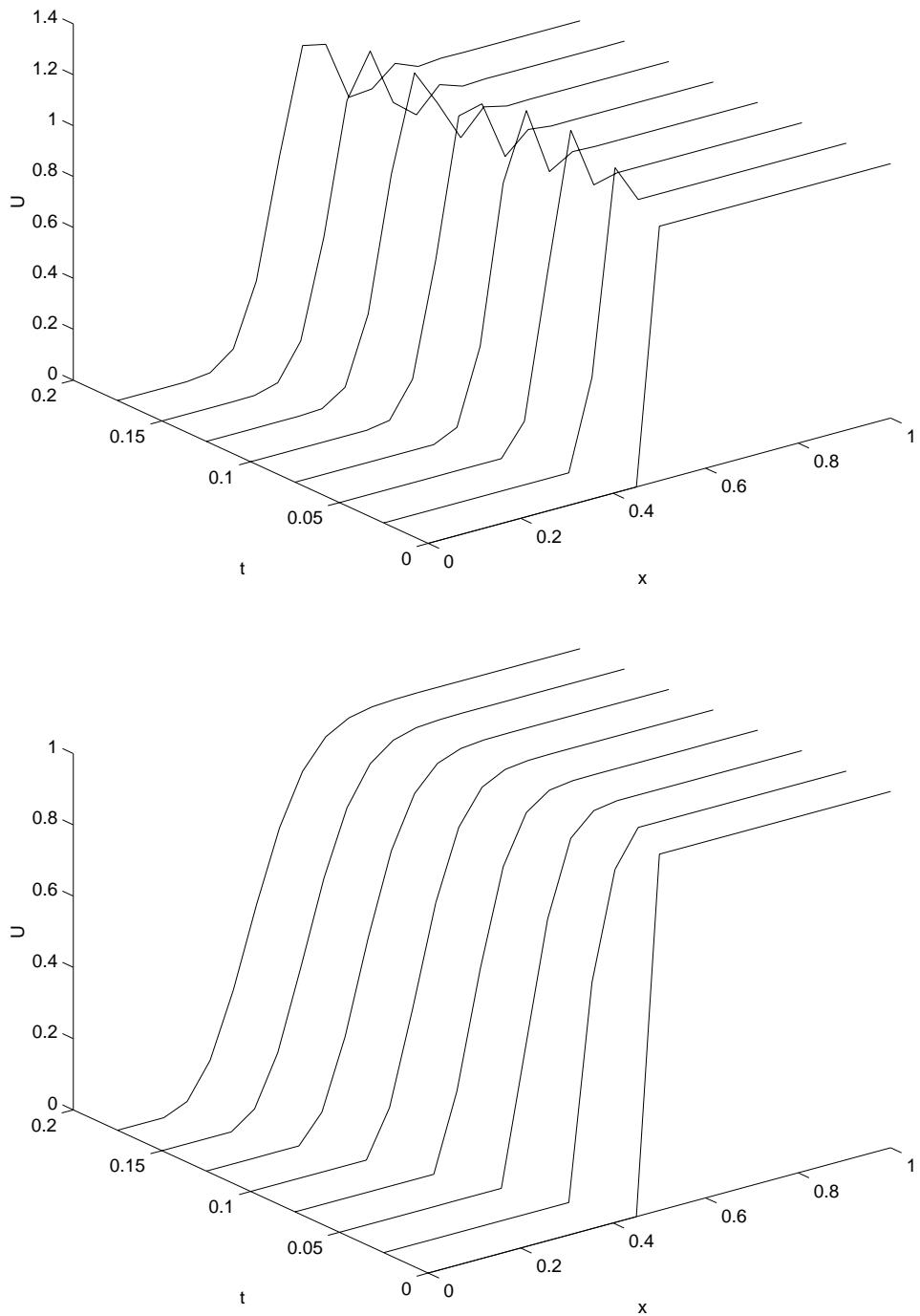


Figure 4.4.1: Solutions of Example 4.4.4 obtained by using centered differencing (4.4.13) (top) and upwind differencing (4.4.17) (bottom) of the convection term.

diffusing faster than it should. With the present state of the art, these are, unfortunately, the two choices:

- higher-order schemes introduce spurious oscillations and
- first-order upwind schemes have excess diffusion.

Some improvements using higher-order upwind schemes will be discussed in Chapter 6.

#### 4.4.3 Nonlinear Problems

Very few exact solutions of nonlinear problems exist and those that do are far removed from realistic applications. Numerical techniques such as finite difference methods often provide the only means of calculating approximations. Explicit schemes cause very few computational difficulties. Convergence and stability analyses, however, are far more difficult. Richtmyer and Morton [17] discuss a heuristic approach to stability that is useful when the coefficients and solution are smooth. Thus, in accord with the discussion of Section 4.1, they show that a constant-coefficient stability analysis performed with “frozen” coefficients yields essentially the correct results. We’ll subsequently illustrate this by example.

Implicit difference schemes lead to nonlinear algebraic systems that must typically be solved by iterative methods. Consider a nonlinear partial differential equation having the rather general form

$$u_t = f(x, t, u, u_x, u_{xx}). \quad (4.4.18)$$

The trapezoidal-rule form of the weighted average finite-difference approximation (4.1.7b) of this problem is

$$\begin{aligned} \frac{U_j^{n+1} - U_j^n}{\Delta t} &= \theta f(j\Delta x, (n+1)\Delta t, U_j^{n+1}, \frac{\mu\delta U_j^{n+1}}{\Delta x}, \frac{\delta^2 U_j^{n+1}}{\Delta x^2}) + \\ &\quad (1-\theta)f(j\Delta x, n\Delta t, U_j^n, \frac{\mu\delta U_j^n}{\Delta x}, \frac{\delta^2 U_j^n}{\Delta x^2}), \end{aligned} \quad (4.4.19a)$$

where  $\theta \in [0, 1]$ . The average and difference operators  $\mu$  and  $\delta$  are defined in Table 2.1.1. The midpoint-rule form of the weighted average scheme is

$$\frac{U_j^{n+1} - U_j^n}{\Delta t} = f(j\Delta x, (n+\theta)\Delta t, \theta U_j^{n+1} + (1-\theta)U_j^n, \frac{\mu\delta}{\Delta x}(\theta U_j^{n+1} + (1-\theta)U_j^n)),$$

$$\frac{\delta^2}{\Delta x^2}(\theta U_j^{n+1} + (1 - \theta)U_j^n). \quad (4.4.19b)$$

When  $\theta = 1/2$  both (4.4.19a) and (4.4.19b) are called Crank-Nicolson schemes. Although both are used, the trapezoidal rule form is more common for nonlinear problems.

*Example 4.4.5* ([17], p. 201). Consider the nonlinear diffusion equation

$$u_t = (u^m)_{xx} \quad (4.4.20a)$$

This differential equation arises in reaction problems associated with semiconductor device fabrication. Let us discretize it by the weighted average scheme (4.4.19a) to obtain

$$\begin{aligned} F_j(\mathbf{U}^{n+1}) &\equiv U_j^{n+1} - \theta \frac{\Delta t}{\Delta x^2} [(U_{j-1}^{n+1})^m - 2(U_j^{n+1})^m + (U_{j+1}^{n+1})^m] - \\ U_j^n - (1 - \theta) \frac{\Delta t}{\Delta x^2} [(U_{j-1}^n)^m - 2(U_j^n)^m + (U_{j+1}^n)^m] &= 0. \end{aligned} \quad (4.4.20b)$$

As usual, the vector  $\mathbf{U}^{n+1}$  represents the vector of unknowns at time level  $n + 1$ . If, for example, homogeneous Dirichlet boundary conditions

$$U_0^n = U_J^n = 0 \quad (4.4.20c)$$

were prescribed, then  $\mathbf{U}^n = [U_1^n, U_2^n, \dots, U_{J-1}^n]^T$  and we would have to solve the nonlinear system

$$\mathbf{F}(\mathbf{U}^{n+1}) = \begin{bmatrix} F_1(\mathbf{U}^{n+1}) \\ F_2(\mathbf{U}^{n+1}) \\ \vdots \\ F_{J-1}(\mathbf{U}^{n+1}) \end{bmatrix} = \mathbf{0}. \quad (4.4.20d)$$

With  $\theta = 0$ , we obtain the explicit forward time-centered space difference scheme

$$U_j^{n+1} = U_j^n + \frac{\Delta t}{\Delta x^2} [(U_{j-1}^n)^m - 2(U_j^n)^m + (U_{j+1}^n)^m]$$

which causes no computational difficulties.

We will need an iterative scheme to solve (4.4.19a) or (4.4.19b) whenever  $\theta \neq 0$ . Most iterative strategies are variants of Newton's method; however, functional (fixed-point) iteration is also used (*cf., e.g.*, Isaacson and Keller [14], Chapter 3). We'll discuss

Newton's method. Thus, suppose that  $\mathbf{V}^k$ ,  $k \geq 0$ , is a guess for the solution  $\mathbf{U}^{n+1}$  and expand the nonlinear system (4.4.20d) in a Taylor's series about  $\mathbf{V}^k$  to get

$$\mathbf{0} = \mathbf{F}(\mathbf{U}^{n+1}) = \mathbf{F}(\mathbf{V}^k) + \mathbf{F}_{\mathbf{U}^{n+1}}(\mathbf{V}^k)(\mathbf{U}^{n+1} - \mathbf{V}^k) + O(\|\mathbf{U}^{n+1} - \mathbf{V}^k\|^2).$$

Neglecting the quadratic terms and denoting the resulting approximation as  $\mathbf{V}^{k+1}$ , we obtain Newton's iteration

$$\mathbf{F}_{\mathbf{U}^{n+1}}(\mathbf{V}^k)(\mathbf{V}^{k+1} - \mathbf{V}^k) = -\mathbf{F}(\mathbf{V}^k), \quad k = 0, 1, \dots. \quad (4.4.21a)$$

The system (4.4.21a) defines a set of linear algebraic equations that must be solved at each iterative step for  $\mathbf{V}^{k+1}$ . The system matrix is the *Jacobian* which, for Dirichlet boundary conditions, is

$$\mathbf{F}_{\mathbf{U}^{n+1}}(\mathbf{V}^k) = \left[ \frac{\partial F_j(\mathbf{V}^k)}{\partial \mathbf{U}_i^{n+1}} \right] = \begin{bmatrix} \frac{\partial F_1(\mathbf{V}^k)}{\partial U_1^{n+1}} & \cdots & \frac{\partial F_1(\mathbf{V}^k)}{\partial U_{J-1}^{n+1}} \\ \vdots & \ddots & \vdots \\ \frac{\partial F_{J-1}(\mathbf{V}^k)}{\partial U_1^{n+1}} & \cdots & \frac{\partial F_{J-1}(\mathbf{V}^k)}{\partial U_{J-1}^{n+1}} \end{bmatrix}. \quad (4.4.21b)$$

The Jacobian will be tridiagonal for second-order partial differential equations with centered differences.

The iteration begins with an initial guess  $\mathbf{V}^0$ , which must be close to the true solution  $\mathbf{U}^{n+1}$  of (4.4.20d) for convergence. Fortunately, the solution  $\mathbf{U}^n$  at the previous time step frequently furnishes a good initial approximation. The iteration terminates either upon failure or satisfaction of a criterion such as

$$\|\mathbf{V}^{k+1} - \mathbf{V}^k\| < \epsilon \|\mathbf{V}^0\| \quad (4.4.21c)$$

for a prescribed tolerance  $\epsilon$ . The converged result is regarded as  $\mathbf{U}^{n+1}$ .

If  $p > 0$  is the smallest number for which

$$\lim_{k \rightarrow \infty} \frac{\|\mathbf{U}^{n+1} - \mathbf{V}^{k+1}\|}{\|\mathbf{U}^{n+1} - \mathbf{V}^k\|^p} = c$$

with  $c \neq 0$ , we say that the *convergence rate* of the iteration (4.4.21a) is  $p$ . Simple Taylor's series arguments show that the convergence rate of Newton's method is two when the Jacobian  $\mathbf{F}_{\mathbf{U}^{n+1}}(\mathbf{U}^{n+1})$  is not singular ([14], Chapter 3). This is called *quadratic convergence*. It implies that the error decreases in proportion to the square of the previous

error. When the Jacobian is singular, *e.g.*, at a bifurcation point, the convergence rate of Newton's method is typically linear ( $p = 1$ ).

The work associated with using Newton's method involves (*i*) calculating the Jacobian and (*ii*) solving the linear system (4.4.21a) at each iteration. The accuracy of the Jacobian does not affect the accuracy of the solution, just the convergence rate. Thus, it is not necessary to reevaluate the Jacobian after each iteration and we may choose to evaluate it only at the initial iteration and use

$$\mathbf{F}_{\mathbf{U}^{n+1}}(\mathbf{V}^0)(\mathbf{V}^{k+1} - \mathbf{V}^k) = -\mathbf{F}(\mathbf{V}^k), \quad k = 0, 1, \dots.$$

This scheme, often called the “chord method,” has a linear convergence rate. Although it has a slower convergence rate than Newton's method (4.4.21a), it may still use less computer time when Jacobian evaluations are complex. Along these lines, the partial derivatives within complex Jacobians may be approximated by finite differences, *e.g.*,

$$\frac{\partial F_j(\mathbf{V}^k)}{\partial \mathbf{U}_i^{n+1}} \approx \frac{F_j(\mathbf{V}^k) - F_j(V_1^k, \dots, V_{i-1}^k, V_i^{k-1}, V_{i+1}^k, \dots, V_{J-1}^k)}{V_i^k - V_i^{k-1}}.$$

This gives rise to the “secant method,” which converges at a rate between linear and quadratic ([14], Chapter 3).

*Example 4.4.6.* Consider the nonlinear diffusion equation (4.4.20a) with the trapezoidal-rule variant of the weighted average scheme (4.4.20b). Assuming that Dirichlet data is prescribed at  $x = 0$  and  $1$ , let's calculate the Jacobian of this system. Thus, differentiating  $F_j$  with respect to  $U_i^{n+1}$ .

$$\frac{\partial F_j}{\partial U_i^{n+1}} = \begin{cases} -m\theta \frac{\Delta t}{\Delta x^2} (U_i^{n+1})^{m-1}, & \text{if } i = j-1, j+1 \\ 1 + 2m\theta \frac{\Delta t}{\Delta x^2} (U_i^{n+1})^{m-1}, & \text{if } i = j \\ 0, & \text{otherwise} \end{cases}. \quad (4.4.22a)$$

In matrix form

$$\mathbf{F}_{\mathbf{U}^{n+1}}(\mathbf{V}^k) = \mathbf{I} + m\theta \frac{\Delta t}{\Delta x^2} \begin{bmatrix} 2(U_1^{n+1})^{m-1} & -(U_2^{n+1})^{m-1} & & & \\ -(U_1^{n+1})^{m-1} & 2(U_2^{n+1})^{m-1} & -(U_3^{n+1})^{m-1} & & \\ & & \ddots & & \\ & & & -(U_{J-2}^{n+1})^{m-1} & 2(U_{J-1}^{n+1})^{m-1} \end{bmatrix}_{(\mathbf{U}^{n+1} = \mathbf{V}^k)}. \quad (4.4.22b)$$

Thus, to solve this problem using Newton's method, we would use (4.4.21a) with  $\mathbf{F}(\mathbf{V}^k)$  given by (4.4.20b - 4.4.20d) and the Jacobian given by (4.4.22b). The Jacobian is tridiagonal; thus, one tridiagonal solution is required per iterative step.

Richtmyer and Morton [17], p. 201) solve an initial-boundary value problem for (4.4.20a) with  $m = 5$  when the exact solution is a “running wave”

$$u(x, t) = \psi(v(t - x + x_0)),$$

where  $v$  and  $x_0$  are prescribed constants. The function  $\psi$  is implicitly determined as the solution  $u$  of

$$\begin{aligned} \frac{5}{4}(u - u_0)^4 + \frac{20}{3}u_0(u - u_0)^3 + 15u_0^2(u + u_0)^2 + 20u_0^3(u - u_0) + \\ 5u_0^4 \ln(u - u_0) = v(vt - x + x_0), \end{aligned}$$

where  $u_0$  is another prescribed constant. Computationally, Richtmyer and Morton [17] solve an initial-boundary value problem with data that is consistent with the exact running wave solution. Thus, the initial and boundary conditions for (4.4.20b) are

$$U_j^0 = \psi(v(x_0 - x_j)), \quad j = 0, 1, \dots, J,$$

$$U_0^n = \psi(v(vt_n + x_0)), \quad U_J^n = \psi(v(vt_n - 1 + x_0)), \quad n > 0.$$

Richtmyer and Morton [17] solve a problem with  $\theta = 0.4$ ,  $v\Delta t/\Delta x = 0.75$ , and  $5u_0^4\Delta t/\Delta x^2 = 0.005$ . They only perform one Newton iteration per time step; thus, they take  $\mathbf{V}^1 = \mathbf{U}^{n+1}$ . (This not good practice. The convergence of Newton's method should be checked before proceeding to the next time step.) Their results for the numerical and exact solutions as functions of  $x$  ( $j$ ) are shown for several times  $t$  (time steps  $n$ ) in Figure 4.4.2.

The weighted average scheme (4.4.20b) is expected to have a stability restriction with  $\theta = 0.4$ . Thus, there is no logical reason for this choice relative to, *e.g.*, the Crank-Nicolson scheme with  $\theta = 0.5$  Richtmyer and Morton [17] choose  $\theta = 0.4$  to show that a constant-coefficient stability analysis suffices for this example since its solution is smooth. To do this, they write (4.4.20a)

$$u_t = (mu^{m-1}u_x)_x.$$

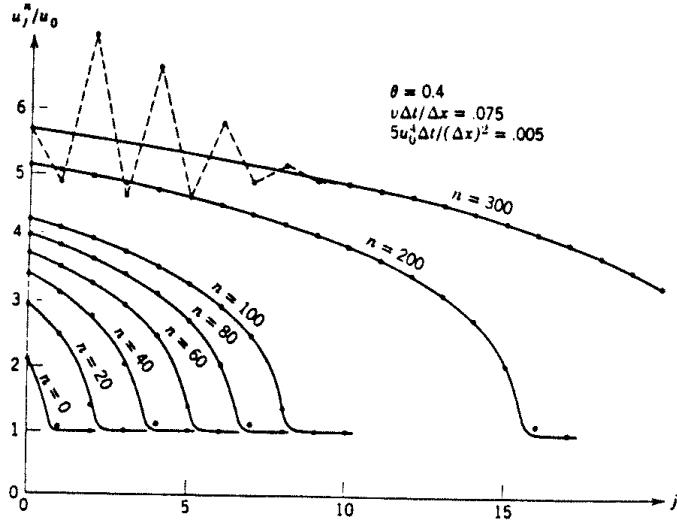


Figure 4.4.2: Richtmyer and Morton's [17] solution of Example 4.4.6. The solid curves are the exact solution and the dots show the numerical solution obtained by the weighted average scheme (4.4.20b) with  $m = 5$  and  $\theta = 0.4$ .

In this form, it has the appearance of a heat conduction equation

$$u_t = (\sigma u_x)_x$$

with the diffusion coefficient

$$\sigma = mu^{m-1}.$$

Based on the constant coefficient analysis of Section 4.1, we would expect this weighted average scheme to be absolutely stable when (4.1.20) is satisfied, *i.e.*, when

$$\frac{\sigma \Delta t}{\Delta x^2} \leq \frac{1}{2(1 - 2\theta)}.$$

With  $m = 5$ , we have  $\sigma = 5u^4$ ; thus, the stability condition is

$$\frac{5u^4 \Delta t}{\Delta x^2} \leq \frac{1}{2(1 - 2\theta)}.$$

Using the indicated parameter values

$$0.005 \left( \frac{u}{u_0} \right)^4 \leq \frac{1}{2(1 - 2(0.4))} = 2.5$$

or

$$\frac{u}{u_0} \leq 4.73.$$

The results of Figure 4.4.2 indicate that oscillations and a loss of absolute stability set in shortly after the solution at  $x = 0$  exceeds the above value.

### Problems

1. Use (4.4.2b - 4.4.2d) with (4.4.7) to show that the forward time-centered space scheme (4.4.2a) is consistent.
2. Consider the initial-boundary value problem for Burgers' equation

$$u_t + uu_x = \sigma u_{xx}, \quad 0 < x < 1, \quad t > 0,$$

$$u(x, 0) = \phi(x), \quad u(0, t) = g_L(t), \quad u(1, t) = g_R(t).$$

where  $\sigma$  is a positive constant. Burgers proposed this equation to model certain aspects of turbulent flows. It is also used as a proving ground for numerical methods because when  $\sigma$  is small the solution exhibits a shock like structure that is difficult to calculate.

- 2.1. Consider the two possibilities for discretizing the nonlinear term:

$$(uu_x)_j^n \approx U_j^n \frac{U_{j+1}^n - U_{j-1}^n}{2\Delta x}$$

and

$$(uu_x)_j^n = \frac{[(u^2)_x]_j^n}{2} \approx \frac{(U_{j+1}^n)^2 - (U_{j-1}^n)^2}{4\Delta x}.$$

Both formulas are  $O(\Delta x^2)$  approximations of  $uu_x$ . Rather than decide which one is best, let's use the linear combination

$$(uu_x) \approx \frac{Q(U_j^n)}{2\Delta x}$$

where

$$Q(U_j^n) = p \frac{(U_{j+1}^n)^2 - (U_{j-1}^n)^2}{2} + (1-p)U_j^n(U_{j+1}^n - U_{j-1}^n)$$

with  $p$  selected on  $[0, 1]$ .

Suppose that we discretize Burgers' equation using the Crank-Nicolson scheme

$$\frac{U_j^{n+1} - U_j^n}{\Delta t} + \frac{Q((U_j^{n+1} + U_j^n)/2)}{2\Delta x} = \sigma \frac{\delta^2((U_j^{n+1} + U_j^n)/2)}{\Delta x^2}$$

where, as usual,  $\delta^2$  denotes the second-centered difference.

Develop a procedure for solving the Crank-Nicolson system using the tridiagonal algorithm and Newton's iteration. Give formulas for evaluating any Jacobians.

- 2.2. Write a program to solve the Crank-Nicolson system developed in Part 1. Choose the initial and Dirichlet boundary data so that the exact solution is

$$u(x, t) = 1 - \tanh \frac{x-t}{2\sigma}.$$

Solve the problem with  $\sigma = 0.01$  for  $t \leq 1$  using  $p = 0, 2/3$ , and 1. Select  $\Delta x = 0.05, 0.02$ , and 0.01 and compute with  $\Delta t/\Delta x = 0.5$  and 1. Plot the solutions as functions of  $x$  at  $t = 0, 0.5$ , and 1. Tabulate or plot the error in the discrete  $L^2$  norm at  $t = 1$  as a function of  $\Delta x$  for each  $p$  and  $\Delta t/\Delta x$ . Discuss the results.

- 2.3. Show that the discrete  $\mathcal{L}^2$  norm

$$\|\mathbf{U}^n\|_2^2 = \sum_{j=0}^{J-1} (U_j^n)^2$$

satisfies

$$\|\mathbf{U}^{n+1}\|_2 \leq \|\mathbf{U}^n\|_2$$

when  $p = 2/3$  and the boundary data is trivial. Also show that

$$\|\mathbf{U}^{n+1}\|_2 = \|\mathbf{U}^n\|_2$$

is conserved in this case when  $\sigma = 0$ . (Hints: Multiply the Crank-Nicolson equation by  $(U_j^{n+1} + U_j^n)$ , sum over  $j$ , and use the summation by parts formulas of Problem 4.3.2. You may also consult Mitchell and Griffiths [15], Section 2.7 and Richtmyer and Morton [17], Section 6.3.)



# Bibliography

- [1] S. Adjerid, A. Aiffa, J.E. Flaherty, J.B. Hudson, and M.S. Shephard. Modeling and adaptive numerical techniques for oxidation of ceramic composites. *Ceramic Engineering and Science Proceedings*, 18:315–322, 1997.
- [2] S. Adjerid and J.E. Flaherty. A moving finite element method with error estimation and refinement for one-dimensional time dependent partial differential equations. *SIAM Journal on Numerical Analysis*, 23:778–795, 1986.
- [3] S. Adjerid and J.E. Flaherty. A moving-mesh finite element method with local refinement for parabolic partial differential equations. *Computer Methods in Applied Mechanics and Engineering*, 55:3–26, 1986.
- [4] S. Adjerid, J.E. Flaherty, P.K. Moore, and Y.J. Wang. High-order adaptive methods for parabolic systems. *Physica D*, 60:94–111, 1992.
- [5] U.M. Ascher and L.R. Petzold. *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*. SIAM, Philadelphia, 1998.
- [6] M. Berzins and R. Furzeland. A user’s manual for sprint - a versatile software package for solving systems of algebraic, ordinary and partial differential equations: Part 1 - algebraic and ordinary differential equations. Technical report, Thorton Research Centre, Shell Research Ltd., Amsterdam, 1985.
- [7] E.C. du Fort and S.P. Frankel. Stability conditions in the numerical treatment of parabolic differential equations. *Mathematical Tables and Other Aids to Computation*, 7:135–152, 1953.

- [8] G. Fairweather and I. Gladwell, editors. *Applied Numerical Mathematics*, volume 20, 1996. Special Issue on the Method of Lines for Time-Dependent Problems.
- [9] J.E. Flaherty and P.K. Moore. Integrated space-time adaptive hp-refinement methods for parabolic systems. *Applied Numerical Mathematics*, 16:317–341, 1995.
- [10] P.R. Garabedian. *Partial Differential Equations*. John Wiley and Sons, New York, 1964.
- [11] C.W. Gear. *Numerical Initial Value Problems in Ordinary Differential Equations*. Prentice Hall, Englewood Cliffs, 1971.
- [12] E. Hairer, S.P. Norsett, and G. Wanner. *Solving Ordinary Differential Equations I: Nonstiff Problems*. Springer-Verlag, Berlin, second edition, 1993.
- [13] E. Hairer and G. Wanner. *Solving Ordinary Differential Equations II: Stiff and Differential Algebraic Problems*. Springer-Verlag, Berlin, 1991.
- [14] E. Isaacson and H.B. Keller. *Analysis of Numerical Methods*. John Wiley and Sons, New York, 1966.
- [15] A.R. Mitchell and D.F. Griffiths. *The Finite Difference Method in Partial Differential Equations*. John Wiley and Sons, Chichester, 1980.
- [16] L.F. Richardson and J.A. Gaunt. The deferred approach to the limit. *Trans of the Royal Society of London*, 226A:299–361, 1927.
- [17] R.D. Richtmyer and K.W. Morton. *Difference Methods for Initial Value Problems*. John Wiley and Sons, New York, second edition, 1967.

# Chapter 5

## Multi-Dimensional Parabolic Problems

### 5.1 Alternating Direction Implicit (ADI) Methods

We would like to extend the one-dimensional explicit and implicit finite difference schemes that we have been studying to multi-dimensional parabolic problems. As a representative example, consider the three-dimensional heat conduction equation

$$\rho c u_t = \nabla \cdot (k \nabla u) = \frac{\partial}{\partial x} (k \frac{\partial u}{\partial x}) + \frac{\partial}{\partial y} (k \frac{\partial u}{\partial y}) + \frac{\partial}{\partial z} (k \frac{\partial u}{\partial z}), \quad (x, y, z) \in \Omega, \quad t > 0, \quad (5.1.1a)$$

subject to the initial and boundary conditions

$$u(x, y, z, 0) = \phi(x, y, z), \quad (x, y, z) \in \Omega \cup \partial\Omega, \quad (5.1.1b)$$

$$\alpha u + \beta u_{\mathbf{n}} = \gamma, \quad (x, y, z) \in \partial\Omega, \quad t > 0. \quad (5.1.1c)$$

As in one dimension,  $u$  is the temperature in a solid having density  $\rho$ , specific heat  $c$ , and thermal conductivity  $k$ . The temperature is to be determined for spatial coordinates  $(x, y, z)$  in the three-dimensional region  $\Omega$  and times  $t > 0$ . The boundary  $\partial\Omega$  of  $\Omega$  has unit outer normal vector  $\mathbf{n}$ .

Constructing a mesh for problems such as (5.1.1) on arbitrary regions is a major undertaking. Let's postpone it to Chapter 8 and first tackle problems on rectangular or hexahedral regions. In fact, let's begin with the two-dimensional, constant-coefficient,

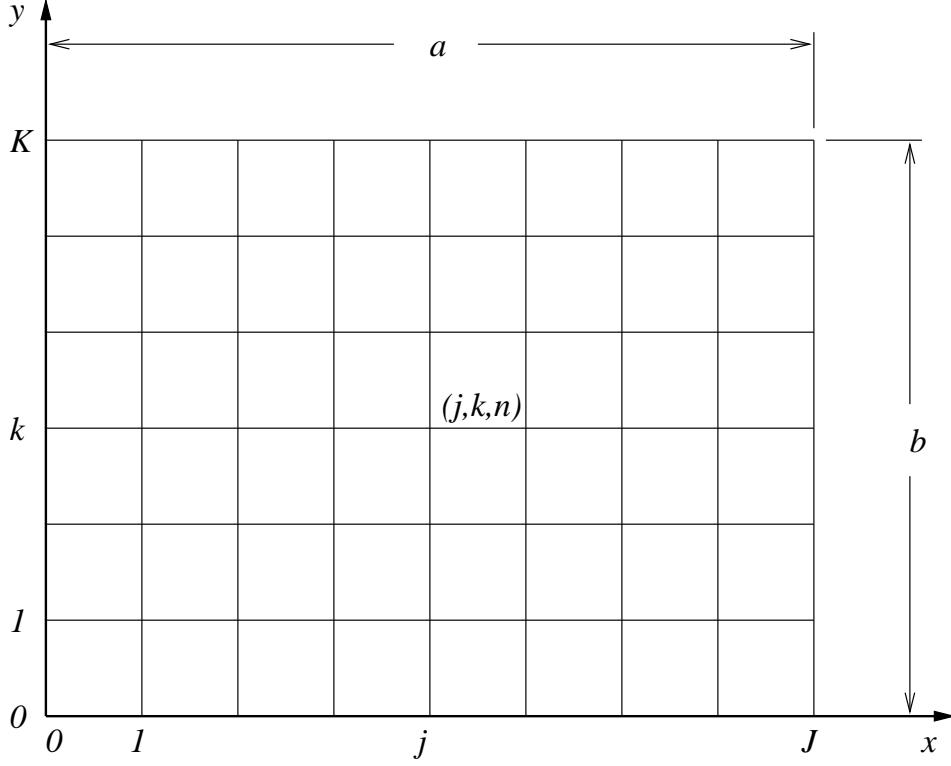


Figure 5.1.1: Two-dimensional rectangular domain  $\Omega$  and the uniform mesh used for finite difference approximations.

Dirichlet problem

$$u_t = \sigma(u_{xx} + u_{yy}), \quad (x, y) \in \Omega, \quad t > 0, \quad (5.1.2a)$$

$$u(x, y, 0) = \phi(x, y), \quad (x, y) \in \Omega \cup \partial\Omega, \quad (5.1.2b)$$

$$u(x, y, t) = \gamma(x, y, t), \quad (x, y) \in \partial\Omega, \quad t > 0, \quad (5.1.2c)$$

where  $\Omega$  is the rectangular region  $\{(x, y) \mid 0 < x < a, 0 < y < b\}$  shown in Figure 5.1.1.

Introduce a uniform rectangular grid of spacing  $\Delta x \times \Delta y$  on  $\Omega$  with  $\Delta x = a/J$  and  $\Delta y = b/K$  and partition time into planes parallel to the  $x, y$ -plane separated by a distance  $\Delta t$  (Figure 5.1.1). Following the one-dimensional notation, let  $U_{j,k}^n$  denote the finite difference approximation of  $u_{j,k}^n = u(j\Delta x, k\Delta y, n\Delta t)$ .

The extension of the explicit scheme (4.1.2) to two spatial dimensions is straightforward and is given by

$$\frac{U_{j,k}^{n+1} - U_{j,k}^n}{\Delta t} = \sigma \left[ \frac{\delta_x^2 U_{j,k}^n}{\Delta x^2} + \frac{\delta_y^2 U_{j,k}^n}{\Delta y^2} \right],$$

where the central difference operators  $\delta_x$  and  $\delta_y$  are defined in accordance with Table 2.1.1, but involve differences of the  $j$  and  $k$  indices, respectively, of  $U_{j,k}^n$ . Thus,

$$\delta_x^2 U_{j,k}^n = U_{j-1,k}^n - 2U_{j,k}^n + U_{j+1,k}^n, \quad \delta_y^2 U_{j,k}^n = U_{j,k-1}^n - 2U_{j,k}^n + U_{j,k+1}^n.$$

Solving for  $U_{j,k}^{n+1}$ ,

$$U_{j,k}^{n+1} = r_x(U_{j-1,k}^n + U_{j+1,k}^n) + r_y(U_{j,k-1}^n + U_{j,k+1}^n) + (1 - 2r_x - 2r_y)U_{j,k}^n, \quad (5.1.3a)$$

where

$$r_x = \frac{\sigma \Delta t}{\Delta x^2}, \quad r_y = \frac{\sigma \Delta t}{\Delta y^2}. \quad (5.1.3b)$$

The computational stencil for (5.1.3) is shown in Figure 5.1.2. The scheme is used exactly as in one dimension. Thus, beginning with the initial conditions

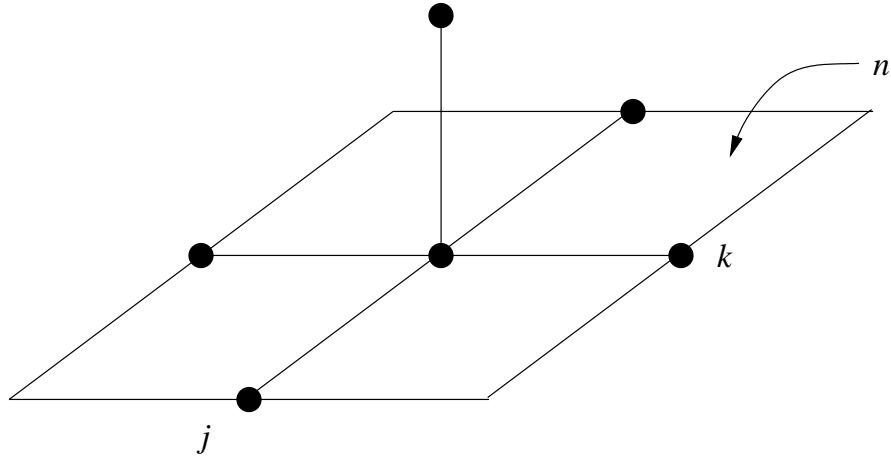


Figure 5.1.2: Computational stencil for the explicit scheme (5.1.3).

$$U_{j,k}^0 = \phi(j\Delta x, k\Delta y), \quad j = 0, 1, \dots, J, \quad k = 0, 1, \dots, K, \quad (5.1.3c)$$

and assuming that the solution  $U_{j,k}^n$ ,  $j = 0, 1, \dots, J$ ,  $k = 0, 1, \dots, K$ , has been calculated, (5.1.3a) is used to compute  $U_{j,k}^{n+1}$  at all interior points  $j = 1, 2, \dots, J-1$ ,  $k = 1, 2, \dots, K-1$ . The boundary conditions

$$U_{j,k}^{n+1} = \gamma(j\Delta x, k\Delta y, (n+1)\Delta t), \quad j = 0, J, \quad k = 0, K, \quad (5.1.3d)$$

furnish the solution on  $\partial\Omega$ .

Absolute stability of (5.1.3a) is assured in the maximum norm when

$$r_x + r_y \leq 1/2.$$

If  $\Delta x = \Delta y$  the stability requirement is  $\sigma \Delta t / \Delta x^2 \leq 1/4$ , which is more restrictive than in one dimension. Thus, there is an even greater motivation to study implicit methods in two dimensions.

When the Crank-Nicolson method is applied to (5.1.2), we find

$$\frac{U_{j,k}^{n+1} - U_{j,k}^n}{\Delta t} = \sigma \left[ \frac{\delta_x^2}{\Delta x^2} \frac{(U_{j,k}^{n+1} + U_{j,k}^n)}{2} + \frac{\delta_y^2}{\Delta y^2} \frac{(U_{j,k}^{n+1} + U_{j,k}^n)}{2} \right],$$

or

$$\left[ 1 - \frac{r_x \delta_x^2 + r_y \delta_y^2}{2} \right] U_{j,k}^{n+1} = \left[ 1 + \frac{r_x \delta_x^2 + r_y \delta_y^2}{2} \right] U_{j,k}^n. \quad (5.1.4a)$$

The computational stencil of (5.1.4a) is shown in Figure 5.1.3.

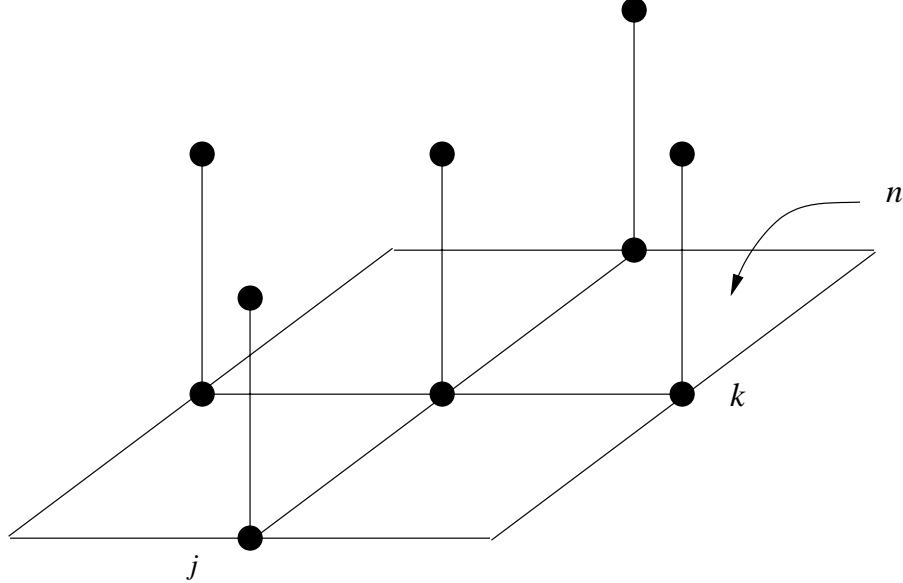


Figure 5.1.3: Computational stencil for the Crank-Nicolson scheme (5.1.4).

For simplicity, assume that trivial boundary data is prescribed, *i.e.*,  $\gamma = 0$  in (5.1.2c); order the equations (5.1.4a) and unknowns by rows; and write (5.1.4a) in the matrix form

$$(\mathbf{I} + \frac{1}{2}\mathbf{C})\mathbf{U}^{n+1} = (\mathbf{I} - \frac{1}{2}\mathbf{C})\mathbf{U}^n, \quad (5.1.4b)$$

where

$$\mathbf{U}^n = [U_{1,1}^n, \dots, U_{J-1,1}^n, U_{1,2}^n, \dots, U_{J-1,2}^n, \dots, U_{1,K-1}^n, \dots, U_{J-1,K-1}^n]^T, \quad (5.1.4c)$$

$$\mathbf{C} = \begin{bmatrix} \mathbf{D}_x & \mathbf{D}_y & & \\ \mathbf{D}_y & \mathbf{D}_x & \mathbf{D}_y & \\ & & \ddots & \\ & & & \mathbf{D}_y & \mathbf{D}_x \end{bmatrix}, \quad (5.1.4d)$$

$$\mathbf{D}_x = \begin{bmatrix} a & b & & \\ b & a & b & \\ & & \ddots & \\ & & b & a \end{bmatrix}, \quad \mathbf{D}_y = \begin{bmatrix} c & & & \\ & c & & \\ & & \ddots & \\ & & & c \end{bmatrix}, \quad (5.1.4e)$$

and

$$a = 2(r_x + r_y), \quad b = -r_x, \quad c = -r_y. \quad (5.1.4f)$$

The matrices  $\mathbf{D}_x$  and  $\mathbf{D}_y$  are  $(J-1) \times (J-1)$  tridiagonal and diagonal matrices, respectively, so  $\mathbf{C}$  is a  $(J-1)(K-1) \times (J-1)(K-1)$  block tridiagonal matrix. This system may be solved by an extension of the tridiagonal algorithm (Figure 4.1.4) to block systems ([3], Chapter 2); however, this method requires approximately  $(5/3)KJ^3$  multiplications per time step. This would normally be too expensive for practical computation. Iterative solution techniques can reduce the computational cost and we will reconsider these in Chapter 9. For the present, let us discuss a solution scheme called the *alternating direction implicit (ADI)* method. Variations of this method were introduced by Douglas [1] and Peaceman and Rachford [5].

The ADI method is a *predictor-corrector* scheme where part of the difference operator is implicit in the initial (prediction) step and another part is implicit in the final (correction) step. In the Peaceman-Rachford [5] variant of ADI, the predictor step consists of solving (5.1.2) for a time step  $\Delta t/2$  using the backward Euler method for the  $x$  derivative terms and the forward Euler method for the  $y$  derivative terms, *i.e.*,

$$U_{j,k}^{n+1/2} = U_{j,k}^n + \frac{r_x}{2} \delta_x^2 U_{j,k}^{n+1/2} + \frac{r_y}{2} \delta_y^2 U_{j,k}^n. \quad (5.1.5a)$$

The corrector step completes the solution process for a time step by using the forward Euler method for  $x$  derivative terms and the backward Euler method for  $y$  derivative terms; thus,

$$U_{j,k}^{n+1} = U_{j,k}^{n+1/2} + \frac{r_x}{2} \delta_x^2 U_{j,k}^{n+1/2} + \frac{r_y}{2} \delta_y^2 U_{j,k}^{n+1}. \quad (5.1.5b)$$

The computation stencils for both the predictor and corrector steps are shown in Figure 5.1.4. The predictor (5.1.5a) is implicit in the  $x$  direction and the corrector (5.1.5b) is implicit in the  $y$  direction.

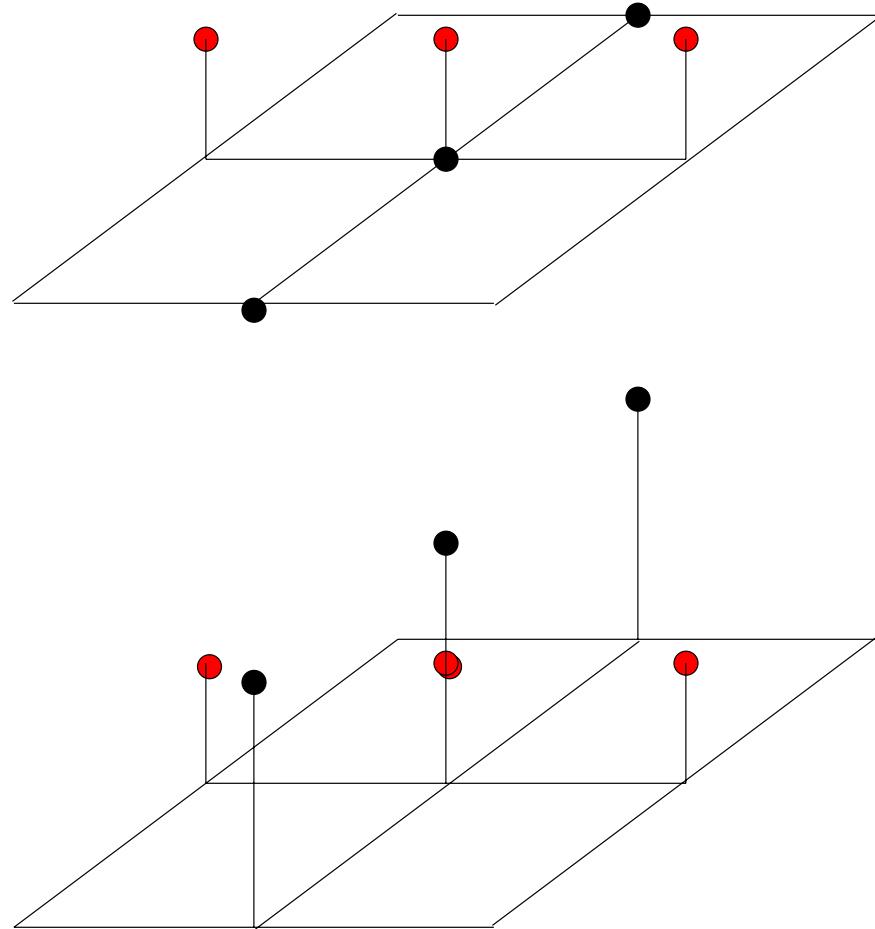


Figure 5.1.4: Computational stencil for the predictor (top) and corrector (bottom) steps of the ADI method (5.1.5a, 5.1.5b). Predicted solutions are shown in red and corrected solutions are black.

On a rectangular region, the predictor equations (5.1.5a) are solved by the tridiagonal algorithm with the unknowns ordered by rows. Thus, assuming that Dirichlet boundary

data is prescribed, we write (5.1.5a) at all interior points  $j = 1, 2, \dots, J - 1$ , in a given row  $k$  to obtain

$$(\mathbf{I} + \mathbf{C}_x)\mathbf{U}_k^{n+1/2} = \mathbf{g}_{y,k}^n, \quad (5.1.6a)$$

where

$$\mathbf{U}_k^{n+1/2} = \begin{bmatrix} U_{1,k}^{n+1/2} \\ U_{2,k}^{n+1/2} \\ \vdots \\ U_{J-1,k}^{n+1/2} \end{bmatrix}, \quad \mathbf{C}_x = \frac{r_x}{2} \begin{bmatrix} 2 & -1 & & \\ -1 & 2 & -1 & \\ & & \ddots & \\ & & -1 & 2 \end{bmatrix}, \quad (5.1.6b)$$

$$\mathbf{g}_{y,k}^n = \begin{bmatrix} U_{1,k}^n + r_y \delta_y^2 U_{1,k}^n / 2 \\ \vdots \\ U_{J-1,k}^n + r_y \delta_y^2 U_{J-1,k}^n / 2 \end{bmatrix}. \quad (5.1.6c)$$

Thus,  $\mathbf{U}_k^{n+1/2}$  is determined by solving (5.1.6) using the tridiagonal algorithm for all interior rows  $k = 1, 2, \dots, K - 1$ .

The corrector system (5.1.5b) is solved by ordering the unknowns by columns. In particular, writing (5.1.5b) for all interior points in column  $j$  gives

$$(\mathbf{I} + \mathbf{C}_y)\mathbf{U}_j^{n+1} = \mathbf{g}_{x,j}^{n+1/2}, \quad (5.1.7)$$

where  $\mathbf{U}_j^n$ ,  $\mathbf{C}_y$ , and  $\mathbf{g}_{x,j}^{n+1/2}$  follow from (5.1.6b, 5.1.6c) upon replacement of  $x$  by  $y$  and  $k$  by  $j$  and interchange of the spatial subscripts. Equation (5.1.7) is then solved by columns for  $j = 1, 2, \dots, J - 1$ , using the tridiagonal algorithm.

The horizontal predictor sweep requires the solution of  $K - 1$  tridiagonal systems of dimension  $J - 1$ . Each tridiagonal system requires approximately  $5J$  operations, where an operation is one multiplication or division plus one addition or subtraction (*cf.* Section 4.1). Thus, the predictor step requires approximately  $5JK$  operations. Similarly, in the corrector step, we have to solve  $J - 1$  tridiagonal systems of dimension  $K - 1$ , which also require approximately  $5JK$  operations. Therefore, the total operation count per time step is  $10JK$  operations, which would normally be far less than the  $(5/3)KJ^3$  operations needed for the block tridiagonal algorithm.

The local discretization error for multi-dimensional problems is defined exactly the same as for one-dimensional problems (Definition 3.1.1). The intermediate ADI solution introduces an added complication; thus, we have to either combine separate estimates of the local discretization errors of the predictor and corrector steps or eliminate  $U_{j,k}^{n+1/2}$  from (5.1.5b). The latter course is the simpler of the two for this application since  $U_{j,k}^{n+1/2}$  may be eliminated by adding and subtracting (5.1.5a) and (5.1.5b). The result is

$$\begin{aligned} U_{j,k}^{n+1} - U_{j,k}^n &= r_x \delta_x^2 U_{j,k}^{n+1/2} + \frac{r_y}{2} \delta_y^2 (U_{j,k}^{n+1} + U_{j,k}^n), \\ U_{j,k}^{n+1/2} &= \frac{1}{2}(U_{j,k}^{n+1} + U_{j,k}^n) - \frac{r_y}{4} \delta_y^2 (U_{j,k}^{n+1} - U_{j,k}^n). \end{aligned}$$

Substituting the second equation into the first

$$U_{j,k}^{n+1} - U_{j,k}^n = \frac{1}{2}(r_x \delta_x^2 + r_y \delta_y^2)(U_{j,k}^{n+1} + U_{j,k}^n) - \frac{r_x r_y}{4} \delta_x^2 \delta_y^2 (U_{j,k}^{n+1} - U_{j,k}^n).$$

Dividing by  $\Delta t$ , gathering all terms on the right side, replacing the numerical approximation by any smooth function, *e.g.*, the exact solution of the differential equation, and subtracting the result from the differential equation (5.1.2a) yields the local discretization error as

$$\begin{aligned} \Delta t \tau_{j,k}^n &= \Delta t(u_t - \sigma u_{xx} - \sigma u_{yy})|_{j,k}^n - (1 - \frac{r_x}{2} \delta_x^2 - \frac{r_y}{2} \delta_y^2) u_{j,k}^{n+1} + (1 + \frac{r_x}{2} \delta_x^2 + \frac{r_y}{2} \delta_y^2) u_{j,k}^n \\ &\quad - \frac{r_x r_y}{4} \delta_x^2 \delta_y^2 (u_{j,k}^{n+1} - u_{j,k}^n). \end{aligned}$$

*Remark 1.* The term  $\Delta t \tau_{j,k}^n$  is not the local error. Since this scheme is implicit, the expression for the local error is more complex.

The first three terms of the above expression are the product of  $\Delta t$  and the local discretization error of the Crank-Nicolson scheme (5.1.4a), *i.e.*,

$$\Delta t(\tau_{j,k}^n)_{CN} = \Delta t(u_t - \sigma u_{xx} - \sigma u_{yy})|_{j,k}^n - (1 - \frac{r_x}{2} \delta_x^2 - \frac{r_y}{2} \delta_y^2) u_{j,k}^{n+1} + (1 + \frac{r_x}{2} \delta_x^2 + \frac{r_y}{2} \delta_y^2) u_{j,k}^n.$$

A Taylor's series expansion would reveal that

$$(\tau_{j,k}^n)_{CN} = O(\Delta x^2) + O(\Delta y^2) + O(\Delta t^2).$$

Expanding the remaining term in a Taylor's series yields

$$\frac{r_x r_y}{4} \delta_x^2 \delta_y^2 (u_{j,k}^{n+1} - u_{j,k}^n) = \frac{r_x r_y}{4} \delta_x^2 \delta_y^2 \delta_t u_{j,k}^{n+1/2} = \frac{\sigma^2}{4} \Delta t^3 [(u_{txxyy})_{j,k}^{n+1/2} + \dots].$$

Thus, the local discretization error of the ADI method is

$$\tau_{j,k}^n = (\tau_{j,k}^n)_{CN} + O(\Delta t^2) = O(\Delta x^2) + O(\Delta y^2) + O(\Delta t^2),$$

which is the same order as that of the Crank-Nicolson method.

The stability of (5.1.5) can be analyzed by the von Neumann method. The two-dimensional form of the discrete Fourier series is

$$U_{j,k}^n = \sum_{p=0}^{J-1} \sum_{q=0}^{K-1} A_{p,q}^n e^{2\pi i(pj/J+qk/K)}. \quad (5.1.8a)$$

Substituting into (5.1.5) and proceeding as in one dimension, we find

$$A_{p,q}^n = (M_{p,q})^n A_{p,q}^0, \quad (5.1.8b)$$

where  $A_{p,q}^0$  is a Fourier component of the initial data and  $M_{p,q}$  is the amplification factor. Again, following the one-dimensional analysis, we verify that  $|M_{p,q}| \leq 1$  for all positive  $r_x$  and  $r_y$ ; hence, the Peaceman-Rachford version of the ADI method (5.1.5) is unconditionally stable.

## 5.2 Operator Splitting Methods

The ADI approach is often difficult to extend to problems on non-rectangular domains, to nonlinear problems, and to problems having mixed derivatives such as  $u_{xy}$ . The dimensional reduction developed for the ADI method can be viewed as an approximate factorization of the differential or discrete operator. Let us motivate the factorization by first examining the ordinary differential equation

$$\frac{dy}{dt} = (a + b)y$$

which, of course, has the solution

$$y(t) = e^{t(a+b)} y(0) = e^{ta} e^{tb} y(0).$$

The latter form suggests that the solution of the initial value problem may be obtained by first solving  $dy/dt = by$  to time  $t$  with  $y(0)$  prescribed as initial data, and then solving

$dy/dt = ay$  subject to the initial condition  $e^{tb}y(0)$ . This interpretation, however, does not extend to vector systems of the form

$$\frac{d\mathbf{y}}{dt} = (\mathbf{A} + \mathbf{B})\mathbf{y}$$

unless  $\mathbf{A}$  and  $\mathbf{B}$  commute. Thus, we may write the solution of the vector problem as

$$\mathbf{y}(t) = e^{t(\mathbf{A}+\mathbf{B})}\mathbf{y}(0),$$

where

$$e^{t\mathbf{C}} = \mathbf{I} + t\mathbf{C} + \frac{t^2}{2!}\mathbf{C}^2 + \dots$$

However,

$$\mathbf{y}(t) = e^{t(\mathbf{A}+\mathbf{B})}\mathbf{y}(0) \neq e^{t\mathbf{A}}e^{t\mathbf{B}}\mathbf{y}(0)$$

unless  $\mathbf{AB} = \mathbf{BA}$ . Nevertheless, let's push on and consider a linear partial differential equation

$$u_t = \mathcal{L}u = (\mathcal{L}_1 + \mathcal{L}_2)u, \quad (5.2.1)$$

where  $\mathcal{L}$  is a spatial differential operator that has been split into the sum of  $\mathcal{L}_1$  and  $\mathcal{L}_2$ . We'll think of  $\mathcal{L}_1$  as being associated with  $x$  derivatives and  $\mathcal{L}_2$  as being associated with  $y$  derivatives, but this is not necessary. Any splitting will do.

The solution of the linear partial differential equation can also be written as the exponential

$$u(x, y, t) = e^{t\mathcal{L}}u(x, y, 0)$$

when  $\mathcal{L}$  is independent of  $t$ . The interpretation of the exponential of the operator  $\mathcal{L}$  follows from a Taylor's series expansion of  $u$  in powers of  $t$ , *i.e.*,

$$u(x, y, t) = u(x, y, 0) + tu_t(x, y, 0) + \frac{t^2}{2!}u_{tt}(x, y, 0) + \dots = e^{t\frac{\partial}{\partial t}}u(x, y, 0)$$

or, using the partial differential equation,

$$u(x, y, t) = u(x, y, 0) + t\mathcal{L}u(x, y, 0) + \frac{t^2}{2!}\mathcal{L}^2u(x, y, 0) + \dots = e^{t\mathcal{L}}u(x, y, 0).$$

The above manipulations are similar to those used to obtain the Lax-Wendroff scheme of Section 3.3.

Unfortunately, once again,

$$u(x, y, t) = e^{t(\mathcal{L}_1 + \mathcal{L}_2)} u(x, y, 0) \neq e^{t\mathcal{L}_1} e^{t\mathcal{L}_2} u(x, y, 0),$$

unless the operators  $\mathcal{L}_1$  and  $\mathcal{L}_2$  commute. Let us verify this by using Taylor's series expansions of both sides of the above expression; thus,

$$e^{t(\mathcal{L}_1 + \mathcal{L}_2)} u(x, y, 0) = [\mathbf{I} + t(\mathcal{L}_1 + \mathcal{L}_2) + \frac{t^2}{2}(\mathcal{L}_1^2 + \mathcal{L}_1\mathcal{L}_2 + \mathcal{L}_2\mathcal{L}_1 + \mathcal{L}_2^2) + \dots] u(x, y, 0)$$

and

$$e^{t\mathcal{L}_1} e^{t\mathcal{L}_2} u(x, y, 0) = (\mathbf{I} + t\mathcal{L}_1 + \frac{t^2}{2}\mathcal{L}_1^2 + \dots)(\mathbf{I} + t\mathcal{L}_2 + \frac{t^2}{2}\mathcal{L}_2^2 + \dots) u(x, y, 0)$$

or

$$e^{t\mathcal{L}_1} e^{t\mathcal{L}_2} u(x, y, t) = [(\mathbf{I} + t(\mathcal{L}_1 + \mathcal{L}_2) + \frac{t^2}{2}(\mathcal{L}_1^2 + 2\mathcal{L}_1\mathcal{L}_2 + \mathcal{L}_2^2) + \dots)] u(x, y, 0).$$

Hence,

$$[e^{t(\mathcal{L}_1 + \mathcal{L}_2)} - e^{t\mathcal{L}_1} e^{t\mathcal{L}_2}] u(x, y, 0) = [\frac{t^2}{2}(\mathcal{L}_2\mathcal{L}_1 - \mathcal{L}_1\mathcal{L}_2) + O(t^3)] u(x, y, 0).$$

The difference between the two expressions is  $O(t^2)$  unless  $\mathcal{L}_1\mathcal{L}_2 u = \mathcal{L}_2\mathcal{L}_1 u$ . The factorization “almost” works when  $t$  is small; hence, we can replace  $t$  by a small time increment  $\Delta t$  to obtain

$$u(x, y, \Delta t) = e^{\Delta t(\mathcal{L}_1 + \mathcal{L}_2)} u(x, y, 0) = [e^{\Delta t\mathcal{L}_1} e^{\Delta t\mathcal{L}_2} + \frac{\Delta t^2}{2}(\mathcal{L}_2\mathcal{L}_1 - \mathcal{L}_1\mathcal{L}_2) + O(\Delta t^3)] u(x, y, 0).$$

To obtain a numerical method, we (i) discretize the spatial operators  $\mathcal{L}_1$  and  $\mathcal{L}_2$ , (ii) neglect the local error terms, and (iii) use the resulting method from time step-to-time step. Thus,

$$\mathbf{U}^{n+1} = e^{\Delta t\mathbf{L}_{1,\Delta}} e^{\Delta t\mathbf{L}_{2,\Delta}} \mathbf{U}^n, \quad (5.2.2)$$

where  $\mathbf{L}_{1,\Delta}$  and  $\mathbf{L}_{2,\Delta}$  are discrete approximations of  $\mathcal{L}_1$  and  $\mathcal{L}_2$ . This technique, often called the *method of fractional steps* or *operator splitting*, has several advantages:

1. If the operators  $\mathbf{L}_{1,\Delta}$  and  $\mathbf{L}_{2,\Delta}$  satisfy the von Neumann conditions

$$\|e^{\Delta t\mathbf{L}_{k,\Delta}}\| \leq 1 + c_k \Delta t, \quad k = 1, 2,$$

then the combined scheme is stable, since, using (5.2.2)

$$\|\mathbf{U}^{n+1}\| \leq \|e^{\Delta t \mathbf{L}_{1,\Delta}}\| \|e^{\Delta t \mathbf{L}_{2,\Delta}}\| \|\mathbf{U}^n\| \leq (1 + c\Delta t) \|\mathbf{U}^n\|.$$

Similarly, if the individual operators are absolutely stable, the combined scheme will be absolutely stable.

2. With operator splitting, the local error is  $O(\Delta t^2)$  unless the operators  $\mathcal{L}_1$  and  $\mathcal{L}_2$  commute, in which case it is  $O(\Delta t^3)$ .

Let us examine some possibilities

*Example 5.2.1.* In order to solve (5.2.1) by operator splitting, we solve

$$u_t = \mathcal{L}_2 u$$

for a time step and then repeat the time step solving

$$u_t = \mathcal{L}_1 u.$$

If we discretize the partial differential equations with Crank-Nicolson approximations, we have

$$(\mathbf{I} - \frac{\Delta t}{2} \mathbf{L}_{2,\Delta}) \hat{U}_{j,k}^{n+1} = (\mathbf{I} + \frac{\Delta t}{2} \mathbf{L}_{2,\Delta}) U_{j,k}^n, \quad (5.2.3a)$$

$$(\mathbf{I} - \frac{\Delta t}{2} \mathbf{L}_{1,\Delta}) U_{j,k}^{n+1} = (\mathbf{I} + \frac{\Delta t}{2} \mathbf{L}_{1,\Delta}) \hat{U}_{j,k}^{n+1}. \quad (5.2.3b)$$

We have used a  $\hat{\cdot}$  to denote the “predicted solution” of (5.2.3a).

*Remark 1.* If the operators  $\mathbf{L}_{1,\Delta}$  and  $\mathbf{L}_{2,\Delta}$  commute, then we may easily verify that (5.2.3) is equivalent to the Peaceman-Rachford form of ADI (5.1.5) for the heat conduction equation. Unlike the Peaceman-Rachford implementation, however, a portion of the operator is neglected at each step.

Let's apply (5.2.3) to the variable-coefficient heat conduction equation

$$u_t = (\sigma u_x)_x + (\sigma u_y)_y$$

where  $\sigma = \sigma(x, y)$ . Suppose that we select

$$\mathcal{L}_1 u = (\sigma u_x)_x, \quad \mathcal{L}_2 u = (\sigma u_y)_y,$$

We discretize each operator using (4.4.4) and introduce the shorthand notation

$$\mathbf{L}_{1,\Delta} U_{j,k}^n = \frac{\hat{\delta}_x^2 U_{j,k}^n}{\Delta x^2} \equiv \frac{\delta_x(\sigma_{j,k} \delta_x U_{j,k}^n)}{\Delta x^2}$$

where

$$\delta_x(\sigma_{j,k} \delta_x U_{j,k}^n) = \sigma_{j+1/2,k} (U_{j+1,k}^n - U_{j,k}^n) - \sigma_{j-1/2,k} (U_{j,k}^n - U_{j-1,k}^n).$$

A similar formula may be written for  $\mathbf{L}_{2,\Delta}$ . Upon using (5.2.3)

$$(\mathbf{I} - \frac{r_y^0}{2} \hat{\delta}_y^2) \hat{U}_{j,k}^{n+1} = (\mathbf{I} + \frac{r_y^0}{2} \hat{\delta}_y^2) U_{j,k}^n, \quad (5.2.4a)$$

$$(\mathbf{I} - \frac{r_x^0}{2} \hat{\delta}_x^2) U_{j,k}^{n+1} = (\mathbf{I} + \frac{r_x^0}{2} \hat{\delta}_x^2) \hat{U}_{j,k}^{n+1} \quad (5.2.4b)$$

where

$$r_x^0 = \Delta t / \Delta x^2, \quad r_y^0 = \Delta t / \Delta y^2. \quad (5.2.4c)$$

The combined method (5.2.4a, 5.2.4b) is solved in alternating directions, like the ADI method (5.1.5). The way that we have split the operator, (5.2.4a) would be solved by columns and (5.2.4b) would be solved by rows. Proceeding in the opposite manner is acceptable.

*Example 5.2.2.* Consider the Taylor's series expansions about time level  $n + 1/2$

$$u_{j,k}^{n+1} = [\mathbf{I} + \frac{\Delta t}{2} \mathcal{L} + \frac{\Delta t^2}{4 \cdot 2!} \mathcal{L}^2 + O(\Delta t^3)] u_{j,k}^{n+1/2},$$

$$u_{j,k}^n = [\mathbf{I} - \frac{\Delta t}{2} \mathcal{L} + \frac{\Delta t^2}{4 \cdot 2!} \mathcal{L}^2 + O(\Delta t^3)] u_{j,k}^{n+1/2}.$$

Adding and subtracting

$$u_{j,k}^{n+1} - u_{j,k}^n = [\Delta t \mathcal{L} + O(\Delta t^3)] u_{j,k}^{n+1/2},$$

$$u_{j,k}^{n+1} + u_{j,k}^n = 2[\mathbf{I} + \frac{\Delta t^2}{4 \cdot 2!} \mathcal{L}^2 + O(\Delta t^3)] u_{j,k}^{n+1/2}.$$

Eliminating  $u_{j,k}^{n+1/2}$

$$u_{j,k}^{n+1} - u_{j,k}^n = \frac{\Delta t}{2} \mathcal{L}(u_{j,k}^{n+1} + u_{j,k}^n) + O(\Delta t^3).$$

Splitting the operator into its component parts

$$[\mathbf{I} - \frac{\Delta t}{2}\mathcal{L}_1 - \frac{\Delta t}{2}\mathcal{L}_2]u_{j,k}^{n+1} = [\mathbf{I} + \frac{\Delta t}{2}\mathcal{L}_1 + \frac{\Delta t}{2}\mathcal{L}_2]u_{j,k}^n + O(\Delta t^3).$$

This is just the trapezoidal rule integration of (5.2.1) for a time step. Indeed, were we to discretize the spatial operators  $\mathcal{L}_1$  and  $\mathcal{L}_2$  using centered differences, we would obtain the same Crank-Nicolson scheme (5.1.4a) that we rejected. Now, instead, let's factor each operator as

$$\mathbf{I} \pm \frac{\Delta t}{2}\mathcal{L}_1 \pm \frac{\Delta t}{2}\mathcal{L}_2 = (\mathbf{I} \pm \frac{\Delta t}{2}\mathcal{L}_1)(\mathbf{I} \pm \frac{\Delta t}{2}\mathcal{L}_2) - \frac{\Delta t^2}{4}\mathcal{L}_1\mathcal{L}_2.$$

Thus, we have

$$\begin{aligned} (\mathbf{I} - \frac{\Delta t}{2}\mathcal{L}_1)(\mathbf{I} - \frac{\Delta t}{2}\mathcal{L}_2)u_{j,k}^{n+1} &= (\mathbf{I} + \frac{\Delta t}{2}\mathcal{L}_1)(\mathbf{I} + \frac{\Delta t}{2}\mathcal{L}_2)u_{j,k}^n + \\ &\quad \frac{\Delta t^2}{4}\mathcal{L}_1\mathcal{L}_2(u_{j,k}^{n+1} - u_{j,k}^n) + O(\Delta t^3). \end{aligned}$$

We have already shown that the next-to-last term on the right is  $O(\Delta t^3)$ ; thus, it may be combined with the discretization error to obtain

$$(\mathbf{I} - \frac{\Delta t}{2}\mathcal{L}_1)(\mathbf{I} - \frac{\Delta t}{2}\mathcal{L}_2)u_{j,k}^{n+1} = (\mathbf{I} + \frac{\Delta t}{2}\mathcal{L}_1)(\mathbf{I} + \frac{\Delta t}{2}\mathcal{L}_2)u_{j,k}^n + O(\Delta t^3).$$

If we neglect the local discretization error and discretize the spatial operators, we obtain

$$(\mathbf{I} - \frac{\Delta t}{2}\mathbf{L}_{1,\Delta})(\mathbf{I} - \frac{\Delta t}{2}\mathbf{L}_{2,\Delta})U_{j,k}^{n+1} = (\mathbf{I} + \frac{\Delta t}{2}\mathbf{L}_{1,\Delta})(\mathbf{I} + \frac{\Delta t}{2}\mathbf{L}_{2,\Delta})U_{j,k}^n \quad (5.2.5)$$

Peaceman and Rachford [5] solved (5.2.3) as

$$(\mathbf{I} - \frac{\Delta t}{2}\mathbf{L}_{1,\Delta})U_{j,k}^{n+1/2} = (\mathbf{I} + \frac{\Delta t}{2}\mathbf{L}_{2,\Delta})U_{j,k}^n, \quad (5.2.6a)$$

$$(\mathbf{I} - \frac{\Delta t}{2}\mathbf{L}_{2,\Delta})U_{j,k}^{n+1} = (\mathbf{I} + \frac{\Delta t}{2}\mathbf{L}_{1,\Delta})U_{j,k}^{n+1/2}. \quad (5.2.6b)$$

When applied to the heat conduction equation with centered spatial differences, this scheme is also identical to the ADI scheme (5.1.5).

Let us verify that (5.2.5) and (5.2.6) are equivalent. Thus, operate on (5.2.6b) with  $\mathbf{I} - \Delta t\mathbf{L}_{1,\Delta}/2$  to obtain

$$(\mathbf{I} - \frac{\Delta t}{2}\mathbf{L}_{1,\Delta})(\mathbf{I} - \frac{\Delta t}{2}\mathbf{L}_{2,\Delta})U_{j,k}^{n+1} = (\mathbf{I} - \frac{\Delta t}{2}\mathbf{L}_{1,\Delta})(\mathbf{I} + \frac{\Delta t}{2}\mathbf{L}_{1,\Delta})U_{j,k}^{n+1/2}.$$

The operators on the right may be interchanged to obtain

$$(\mathbf{I} - \frac{\Delta t}{2}\mathbf{L}_{1,\Delta})(\mathbf{I} - \frac{\Delta t}{2}\mathbf{L}_{2,\Delta})U_{j,k}^{n+1} = (\mathbf{I} + \frac{\Delta t}{2}\mathbf{L}_{1,\Delta})(\mathbf{I} - \frac{\Delta t}{2}\mathbf{L}_{1,\Delta})U_{j,k}^{n+1/2}.$$

Using (5.2.6a) yields (5.2.5).

*Example 5.2.3.* D'Yakonov (*cf.* [4], Section 2.12) introduced the following scheme for solving (5.2.5)

$$(\mathbf{I} - \frac{\Delta t}{2}\mathbf{L}_{1,\Delta})\hat{U}_{j,k}^{n+1} = (\mathbf{I} + \frac{\Delta t}{2}\mathbf{L}_{1,\Delta})(\mathbf{I} + \frac{\Delta t}{2}\mathbf{L}_{2,\Delta})U_{j,k}^n \quad (5.2.7a)$$

$$(\mathbf{I} - \frac{\Delta t}{2}\mathbf{L}_{2,\Delta})U_{j,k}^{n+1} = \hat{U}_{j,k}^{n+1}. \quad (5.2.7b)$$

This scheme has the same order of accuracy and characteristics as the Peaceman-Rachford ADI scheme (5.2.6).

*Example 5.2.4.* Douglas and Rachford [2] developed an alternative scheme for (5.2.1) using backward-difference approximations. Thus, consider integrating (5.2.1) for a time step by the backward Euler method to obtain

$$(\mathbf{I} - \Delta t\mathcal{L}_1 - \Delta t\mathcal{L}_2)u_{j,k}^{n+1} = u_{j,k}^n + O(\Delta t)^2.$$

Let us rewrite this as

$$(\mathbf{I} - \Delta t\mathcal{L}_1 - \Delta t\mathcal{L}_2 + \Delta t^2\mathcal{L}_1\mathcal{L}_2)u_{j,k}^{n+1} = (\mathbf{I} + \Delta t^2\mathcal{L}_1\mathcal{L}_2)u_{j,k}^n +$$

$$\Delta t^2\mathcal{L}_1\mathcal{L}_2(u_{j,k}^{n+1} - u_{j,k}^n) + O(\Delta t)^3.$$

As in Example 5.2.2, we may show that the next-to-last term on the right is  $O(\Delta t^3)$  and, hence, may be neglected. Also neglecting the temporal discretization error term, discretizing the operators  $\mathcal{L}_1$  and  $\mathcal{L}_2$ , and factoring the left side gives

$$(\mathbf{I} - \Delta t\mathbf{L}_{1,\Delta})(\mathbf{I} - \Delta t\mathbf{L}_{2,\Delta})U_{j,k}^{n+1} = (\mathbf{I} + \Delta t^2\mathbf{L}_{1,\Delta}\mathbf{L}_{2,\Delta})U_{j,k}^n.$$

Douglas and Rachford [2] factored this as

$$(\mathbf{I} - \Delta t\mathbf{L}_{1,\Delta})\hat{U}_{j,k}^{n+1} = (\mathbf{I} + \Delta t\mathbf{L}_{2,\Delta})U_{j,k}^n \quad (5.2.8a)$$

$$(\mathbf{I} - \Delta t \mathbf{L}_{2,\Delta}) U_{j,k}^{n+1} = \hat{U}_{j,k}^{n+1} - \Delta t \mathbf{L}_{2,\Delta} U_{j,k}^n \quad (5.2.8b)$$

Assuming that the discrete spatial operators are second-order accurate, the local discretization error is  $O(\Delta t) + O(\Delta x^2) + O(\Delta y^2)$ . This is lower order than the  $O(\Delta t^2)$  local discretization error that would be obtained from the ADI factorization (5.2.6); however, backward differencing gives greater stability than (5.2.6) which may be useful for nonlinear problems.

*Example 5.2.5.* The previous examples suggest a simplicity that is not always present. Boundary conditions must be treated very carefully since the intermediate solutions or  $\mathbf{U}^{n+1/2}$  or  $\hat{\mathbf{U}}^{n+1}$  need not be consistent approximations of  $u(x, (n + 1/2)\Delta t)$  or  $u(x, (n + 1)\Delta t)$ . Yanenko [8] presents a good example of the complications that can arise at boundaries. Strikwerda [7], Section 7.3, suggests using a combination of  $\mathbf{U}^n$  and  $\mathbf{U}^{n+1}$  to get the intermediate boundary condition. Let us illustrate this for the Dirichlet problem (5.1.2) using the Peaceman-Rachford ADI scheme (5.2.6). During the horizontal sweep (5.2.6a), we need boundary conditions for  $\mathbf{U}^{n+1/2}$  at  $x = 0$  and  $1$ . Adding (5.2.6a) and (5.2.6b) gives

$$U_{j,k}^{n+1/2} = \frac{1}{2}(\mathbf{I} + \frac{\Delta t}{2} \mathbf{L}_{2,\Delta}) U_{j,k}^n + \frac{1}{2}(\mathbf{I} - \frac{\Delta t}{2} \mathbf{L}_{2,\Delta}) U_{j,k}^{n+1}.$$

Using the boundary condition (5.1.2c)

$$U_{j,k}^{n+1/2} = \frac{1}{2}(\mathbf{I} + \frac{\Delta t}{2} \mathbf{L}_{2,\Delta}) \gamma_{j,k}^n + \frac{1}{2}(\mathbf{I} - \frac{\Delta t}{2} \mathbf{L}_{2,\Delta}) \gamma_{j,k}^{n+1} \quad (5.2.9)$$

which can be used as a boundary condition for  $\mathbf{U}^{n+1/2}$ . The obvious boundary condition

$$U_{j,k}^{n+1/2} = \frac{\gamma_{j,k}^n + \gamma_{j,k}^{n+1}}{2}$$

is only first-order accurate as apparent from (5.2.9).

Boundary conditions for the Douglas-Rachford scheme can be obtained from the corrector equation (5.2.8b) and the boundary condition (5.1.2c) as

$$\hat{U}_{j,k}^{n+1} = (\mathbf{I} - \Delta t \mathbf{L}_{2,\Delta}) \gamma_{j,k}^{n+1} + \Delta t \mathbf{L}_{2,\Delta} \gamma_{j,k}^n. \quad (5.2.10)$$

*Example 5.2.6.* Strang [6] developed a factorization technique that has a faster rate of convergence than the splitting (5.2.2) when the operators  $\mathcal{L}_1$  and  $\mathcal{L}_2$  do not commute.

Strang computes

$$U_{j,k}^{n+1} = e^{(\Delta t/2)\mathcal{L}_2} e^{\Delta t\mathcal{L}_1} e^{(\Delta t/2)\mathcal{L}_2} U_{j,k}^n. \quad (5.2.11)$$

This scheme appears to require an extra solution per step; however, if results are output every  $n$  time steps then

$$U_{j,k}^n = (e^{(\Delta t/2)\mathcal{L}_2} e^{\Delta t\mathcal{L}_1} e^{(\Delta t/2)\mathcal{L}_2})(e^{(\Delta t/2)\mathcal{L}_2} e^{\Delta t\mathcal{L}_1} e^{(\Delta t/2)\mathcal{L}_2}) \dots (e^{(\Delta t/2)\mathcal{L}_2} e^{\Delta t\mathcal{L}_1} e^{(\Delta t/2)\mathcal{L}_2}) U_{j,k}^0$$

or

$$U_{j,k}^n = e^{(\Delta t/2)\mathcal{L}_2} e^{\Delta t\mathcal{L}_1} e^{\Delta t\mathcal{L}_2} e^{\Delta t\mathcal{L}_1} \dots e^{(\Delta t/2)\mathcal{L}_2} U_{j,k}^0.$$

Hence, the factorization (5.2.11) is the same as the simpler splitting (5.2.2) except for the first and last time steps.

Let us estimate the local error; thus, assuming that  $U_{j,k}^n = u_{j,k}^n$ ,

$$\begin{aligned} U_{j,k}^{n+1} &= (\mathbf{I} + \frac{\Delta t}{2}\mathcal{L}_2 + \frac{\Delta t^2}{8}\mathcal{L}_2^2 + \dots)(\mathbf{I} + \Delta t\mathcal{L}_1 + \frac{\Delta t^2}{2}\mathcal{L}_1^2 + \dots) \\ &\quad (\mathbf{I} + \frac{\Delta t}{2}\mathcal{L}_2 + \frac{\Delta t^2}{8}\mathcal{L}_2^2 + \dots)u_{j,k}^n \end{aligned}$$

or

$$U_{j,k}^{n+1} = [\mathbf{I} + \Delta t(\mathcal{L}_1 + \mathcal{L}_2) + \frac{\Delta t^2}{2}(\mathcal{L}_1^2 + \mathcal{L}_1\mathcal{L}_2 + \mathcal{L}_2\mathcal{L}_1 + \mathcal{L}_2^2) + O(\Delta t^3)]u_{j,k}^n.$$

Thus,

$$u_{j,k}^{n+1} - U_{j,k}^{n+1} = [e^{\Delta t(\mathcal{L}_1 + \mathcal{L}_2)} - e^{(\Delta t/2)\mathcal{L}_2} e^{\Delta t\mathcal{L}_1} e^{(\Delta t/2)\mathcal{L}_1}]u_{j,k}^n = O(\Delta t^3).$$

### Problems

1. Although operator splitting has primarily been used for dimensional splitting, it may be used in other ways. Consider the nonlinear reaction-diffusion problem

$$u_t = \sigma u_{xx} + u(1-u), \quad 0 < x < 1, \quad t > 0,$$

$$u(0, t) = u(1, t) = 0, \quad t > 0,$$

$$u(x, 0) = \phi(x), \quad 0 \leq x \leq 1.$$

Develop a procedure for solving this problem that involves splitting the diffusion ( $\sigma u_{xx}$ ) and reaction ( $u(1-u)$ ) operators. Discuss its stability and local discretization errors.



# Bibliography

- [1] J. Douglas. On the numerical integration of  $\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = \frac{\partial u}{\partial t}$  by implicit methods. *Joural of SIAM*, 3:42–65, 1955.
- [2] J. Douglas and H.H. Rachford. On the numerical solution of heat conduction problems in two and three space variables. *Trasactions of the American Mathematics Society*, 82:421–439, 1956.
- [3] E. Isaacson and H.B. Keller. *Analysis of Numerical Methods*. John Wiley and Sons, New York, 1966.
- [4] A.R. Mitchell and D.F. Griffiths. *The Finite Difference Method in Partial Differential Equations*. John Wiley and Sons, Chichester, 1980.
- [5] D.W. Peaceman and Jr. H.H. Rachford. The numerical solution of parabolic and elliptic equations. *Journal of SIAM*, 3:28–41, 1955.
- [6] G. Strang. On the construction and comparison of difference schemes. *SIAM Journal on Numerical Analysis*, 5:506–517, 1968.
- [7] J.C. Strikwerda. *Finite Difference Schemes and Partial Differential Equations*. Chapman and Hall, Pacific Grove, 1989.
- [8] N. Yanenko. *The Method of Fractional Steps*. Springer-Verlag, Heidelberg, 1971.

# Chapter 6

## Hyperbolic Problems

### 6.1 Vector Systems and Characteristics

Let's resume the study of finite difference techniques for hyperbolic conservation laws by examining the characteristics of one-dimensional vector systems having the form

$$\mathbf{u}_t + \mathbf{f}(\mathbf{u})_x = \mathbf{b}(x, t, \mathbf{u}). \quad (6.1.1a)$$

Here,  $\mathbf{u}(x, t)$ ,  $\mathbf{f}(\mathbf{u})$ , and  $\mathbf{b}(x, t, \mathbf{u})$  are  $m$ -dimensional vectors. As in Chapter 1, it is sometimes convenient to write (6.1.1a) in the convective form

$$\mathbf{u}_t + \mathbf{A}\mathbf{u}_x = \mathbf{b}, \quad (6.1.1b)$$

where the Jacobian

$$\mathbf{A}(\mathbf{u}) = \mathbf{f}_{\mathbf{u}}(\mathbf{u}) \quad (6.1.1c)$$

is an  $m \times m$  matrix.

**Definition 6.1.1.** *If  $\mathbf{A}$  has  $m$  real and distinct eigenvalues  $\lambda_1 < \lambda_2 < \dots < \lambda_m$  and, hence,  $m$  linearly independent eigenvectors  $\mathbf{p}^{(1)}, \mathbf{p}^{(2)}, \dots, \mathbf{p}^{(m)}$ , then (6.1.1a) is said to be hyperbolic.*

Physical problems where dissipative effects can be neglected often lead to hyperbolic systems. Areas where these arise include acoustics, dynamic elasticity, electromagnetics, and gas dynamics.

Let

$$\mathbf{P} = [\mathbf{p}^{(1)}, \mathbf{p}^{(2)}, \dots, \mathbf{p}^{(m)}] \quad (6.1.2a)$$

and write the eigenvalue-eigenvector relation in matrix form as

$$\mathbf{A}\mathbf{P} = \mathbf{P}\Lambda, \quad (6.1.2b)$$

where

$$\Lambda = \begin{bmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_m \end{bmatrix} \quad (6.1.2c)$$

Multiplication of (6.1.1b) by  $\mathbf{P}^{-1}$  and the use of (6.1.2b) gives

$$\mathbf{P}^{-1}\mathbf{u}_t + \mathbf{P}^{-1}\mathbf{A}\mathbf{u}_x = \mathbf{P}^{-1}\mathbf{u}_t + \Lambda\mathbf{P}^{-1}\mathbf{u}_x = \mathbf{P}^{-1}\mathbf{b}.$$

Let

$$\mathbf{w} = \mathbf{P}^{-1}\mathbf{u} \quad (6.1.3)$$

so that

$$\mathbf{w}_t + \Lambda\mathbf{w}_x = \mathbf{P}^{-1}\mathbf{u}_t + (\mathbf{P}^{-1})_t\mathbf{u} + \Lambda[\mathbf{P}^{-1}\mathbf{u}_x + (\mathbf{P}^{-1})_x\mathbf{u}].$$

Using (6.1.3)

$$\mathbf{w}_t + \Lambda\mathbf{w}_x = \mathbf{Q}\mathbf{w} + \mathbf{g}, \quad (6.1.4a)$$

where

$$\mathbf{Q} = [(\mathbf{P}^{-1})_t + \Lambda(\mathbf{P}^{-1})_x]\mathbf{P}, \quad \mathbf{g} = \mathbf{P}^{-1}\mathbf{b}. \quad (6.1.4b)$$

In component form, (6.1.4a) is

$$(w_i)_t + \lambda_i(w_i)_x = \sum_{j=1}^m q_{i,j}w_j + g_i, \quad i = 1, 2, \dots, m. \quad (6.1.4c)$$

Thus, the transformation (6.1.3) has uncoupled the differentiated terms of the original system (6.1.1b).

As in Chapter 1, consider the directional derivative of each component  $w_i$ ,  $i = 1, 2, \dots, m$ , of  $\mathbf{w}$ ,

$$\frac{dw_i}{dt} = (w_i)_t + (w_i)_x \frac{dx}{dt}, \quad i = 1, 2, \dots, m,$$

in the directions

$$\frac{dx}{dt} = \lambda_i, \quad i = 1, 2, \dots, m, \tag{6.1.5}$$

and use (6.1.4c) to obtain

$$\frac{dw_i}{dt} = \sum_{j=1}^m q_{i,j} w_j + g_i, \quad i = 1, 2, \dots, m. \tag{6.1.6}$$

As a reminder, the curves (6.1.5) are the *characteristics* of the system (6.1.1a, 6.1.1b).

The partial differential system (6.1.1b) may be solved by integrating the  $2m$  ordinary differential equations (6.1.5, 6.1.6). This system is uncoupled through its differentiated terms but coupled through  $\mathbf{Q}$  and  $\mathbf{g}$ . This method of solution is, quite naturally, called the method of characteristics. While we could develop effective numerical methods based on the method of characteristics, they are generally not efficient when  $m > 2$ .

In Chapter 1, we studied homogeneous ( $\mathbf{b} \equiv \mathbf{0}$ ) scalar problems ( $m = 1$ ) and found that the solution  $u$  did not change along the characteristic. Examining (6.1.6), we see that  $w_i$  will change along the characteristic unless the right side of (6.1.6) vanishes. This generally requires both  $\mathbf{b} = \mathbf{0}$  and  $\mathbf{P}$  to be constant. Similarly, we'll have to modify the definition of domain of dependence that was introduced in Chapter 2 to accommodate the inhomogeneous term in (6.1.1b).

**Definition 6.1.2.** *The set of all points that determine the solution at a point  $P(x_0, t_0)$  is called the domain of dependence of  $P$ .*

Consider the arbitrary point  $P(x_0, t_0)$  and the characteristics passing through it as shown in Figure 6.1.1. The solution  $u(x_0, t_0)$  depends on the initial data on the interval  $[A, B]$  and on the values of  $\mathbf{b}$  in the region  $APB$ , bounded by  $[A, B]$  and the characteristic curves  $\dot{x} = \lambda_1$  and  $\dot{x} = \lambda_m$ . Thus, the region  $APB$  is the domain of dependence of  $P$ .

*Example 6.1.1.* Consider the initial value problem for the forced wave equation

$$u_{tt} = a^2 u_{xx} + q(x), \quad -\infty < x < \infty, \quad t > 0,$$

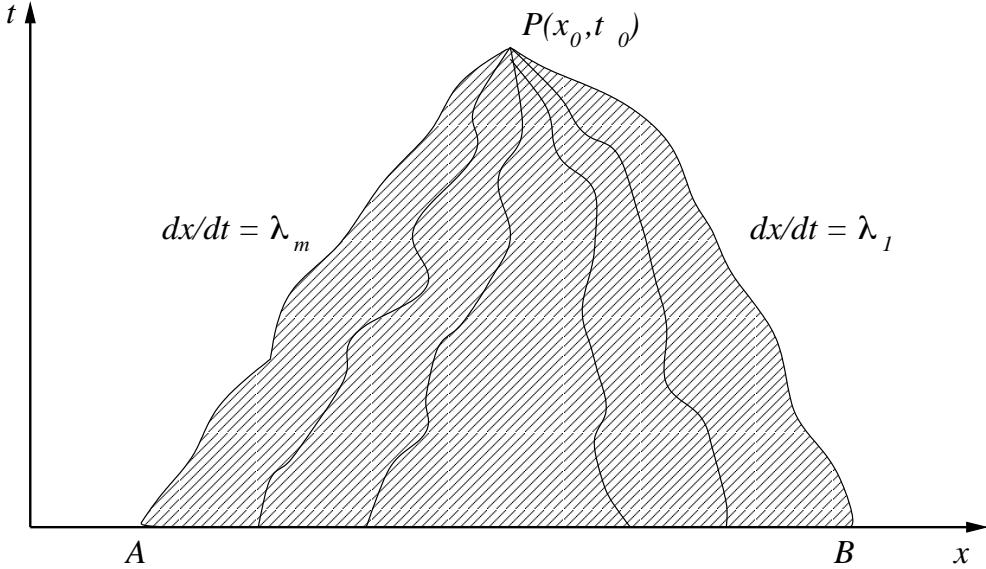


Figure 6.1.1: Domain of dependence of a point  $P(x_0, t_0)$ . The solution at  $P$  depends on the initial data on the line  $[A, B]$  and the values of  $\mathbf{b}$  within the region  $APB$  bounded by the characteristic curves  $dx/dt = \lambda_1, \lambda_m$ .

$$u(x, 0) = u^0(x), \quad u_t(x, 0) = \dot{u}^0(x), \quad -\infty < x < \infty.$$

If  $u(x, t)$  corresponds to the transverse displacement of a taut string, then the constant  $a^2 = T/\rho$ , where  $T$  is the tension and  $\rho$  is the linear density of the string, and  $q(x)$  is a lateral load applied to the string in the direction of  $u$ . The lateral load could, for example, represent the weight of the string.

We'll transform the wave equation to a first-order system by letting

$$u_1 = u_t, \quad u_2 = au_x.$$

Physically,  $u_1(x, t)$  is the velocity and  $u_2(x, t)$  is the stress at point  $x$  and time  $t$  in the string.

The transformation gives

$$(u_1)_t = a(u_2)_x + q$$

and the compatibility condition

$$(u_2)_t = au_{xt} = au_{tx} = a(u_1)_x.$$

Thus, the first-order system has the form of (6.1.1b) with

$$\mathbf{u} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}, \quad \mathbf{A} = \begin{bmatrix} 0 & -a \\ -a & 0 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} q \\ 0 \end{bmatrix}.$$

The initial conditions become

$$u_1(x, 0) = \dot{u}^0(x), \quad u_2(x, 0) = au_x^0(x), \quad -\infty < x < \infty.$$

The eigenvalues of  $\mathbf{A}$  are  $\lambda = \pm a$ , the characteristics are

$$\dot{x} = \pm a,$$

and the eigenvectors are

$$\mathbf{P} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}.$$

Since  $\mathbf{P}^{-1} = \mathbf{P}$ , we may use (6.1.3) to determine the canonical variables as

$$w_1 = \frac{u_1 + u_2}{\sqrt{2}}, \quad w_2 = \frac{u_1 - u_2}{\sqrt{2}}.$$

From (6.1.4), the canonical form of the problem is

$$(w_1)_t - a(w_1)_x = \frac{q}{\sqrt{2}}, \quad (w_2)_t + a(w_2)_x = \frac{q}{\sqrt{2}}.$$

The characteristics integrate to

$$x = x_0 - at, \quad x = x_0 + at,$$

and along the characteristics, we have

$$\frac{dw_k}{dt} = \frac{q}{\sqrt{2}}, \quad k = 1, 2.$$

Integrating, we find

$$w_1(x, t) = w_1^0(x_0) + \frac{1}{\sqrt{2}} \int_0^t q(x_0 - a\tau) d\tau$$

or

$$w_1(x, t) = w_1^0(x_0) - \frac{1}{a\sqrt{2}} \int_{x_0}^{x_0 - at} q(\xi) d\xi.$$

It's usual to eliminate  $x_0$  by using the characteristic equation to obtain

$$w_1(x, t) = w_1^0(x + at) - \frac{1}{a\sqrt{2}} \int_{x+at}^x q(\xi) d\xi.$$

Likewise

$$w_2(x, t) = w_2^0(x - at) + \frac{1}{a\sqrt{2}} \int_{x-at}^x q(\xi) d\xi.$$

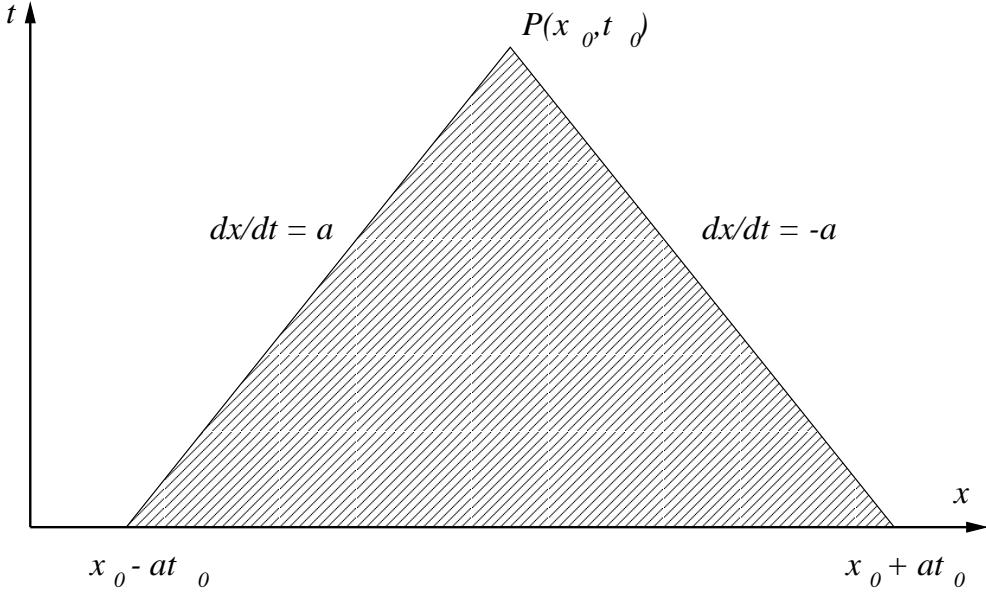


Figure 6.1.2: The domain of dependence of a point  $P(x_0, t_0)$  for Example 6.1.1 is the triangle connecting the points  $P$ ,  $(x_0 - at_0, 0)$ , and  $(x_0 + at_0, 0)$ .

The domain of dependence of a point  $P(x_0, t_0)$  is shown in Figure 6.1.2. Using the bounding characteristics, it is the triangle connecting the points  $(x_0, t_0)$ ,  $(x_0 - at_0, 0)$ , and  $(x_0 + at_0, 0)$ . (Actually, with  $q$  being a function of  $x$  only, the domain of dependence only involves values of  $q(x)$  on the subinterval  $(x_0 - at_0, 0)$  to  $(x_0 + at_0, 0)$ .)

Transforming back to the physical variables

$$u_1(x, t) = \frac{1}{\sqrt{2}}(w_1 + w_2) = \frac{1}{\sqrt{2}}[w_1^0(x + at) + w_2^0(x - at)] + \frac{1}{2a} \int_{x-at}^{x+at} q(\xi) d\xi,$$

$$u_2(x, t) = \frac{1}{\sqrt{2}}(w_1 - w_2) = \frac{1}{\sqrt{2}}[w_1^0(x + at) - w_2^0(x - at)] - \frac{1}{2a} [\int_{x+at}^x q(\xi) d\xi + \int_{x-at}^x q(\xi) d\xi].$$

Suppose, for simplicity, that  $\dot{u}^0(x) = 0$ , then

$$u_1(x, 0) = 0 = \frac{1}{\sqrt{2}}[w_1^0(x) + w_2^0(x)],$$

$$u_2(x, 0) = au_x^0(x) = \frac{1}{\sqrt{2}}[w_1^0(x) - w_2^0(x)].$$

Thus,

$$w_1^0(x) = -w_2^0(x) = \frac{au_x^0(x)}{\sqrt{2}},$$

and

$$u_1(x, t) = \frac{a}{2}[u_x^0(x + at) - u_x^0(x - at)] + \frac{1}{2a} \int_{x-at}^{x+at} q(\xi) d\xi,$$

$$u_2(x, t) = \frac{a}{2}[u_x^0(x + at) + u_x^0(x - at)] - \frac{1}{2a} \left[ \int_{x+at}^x q(\xi) d\xi + \int_{x-at}^x q(\xi) d\xi \right].$$

Since  $u_2 = au_x$ , we can integrate to find the solution in the original variables. In order to simplify the manipulations, let's do this for the homogeneous system ( $q(x) = 0$ ). In this case, we have

$$u_2(x, t) = \frac{a}{2}[u_x^0(x + at) + u_x^0(x - at)];$$

hence,

$$u(x, t) = \frac{1}{2}[u^0(x + at) + u^0(x - at)].$$

The solution for an initial value problem when

$$u^0(x) = \begin{cases} x + 1, & \text{if } -1 \leq x \leq 0 \\ 1 - x, & \text{if } 0 \leq x \leq 1 \\ 0, & \text{otherwise} \end{cases}$$

is shown in Figure 6.1.3. The initial data splits into two waves having half the initial amplitude and traveling in the positive and negative  $x$  directions with speeds  $a$  and  $-a$ , respectively.

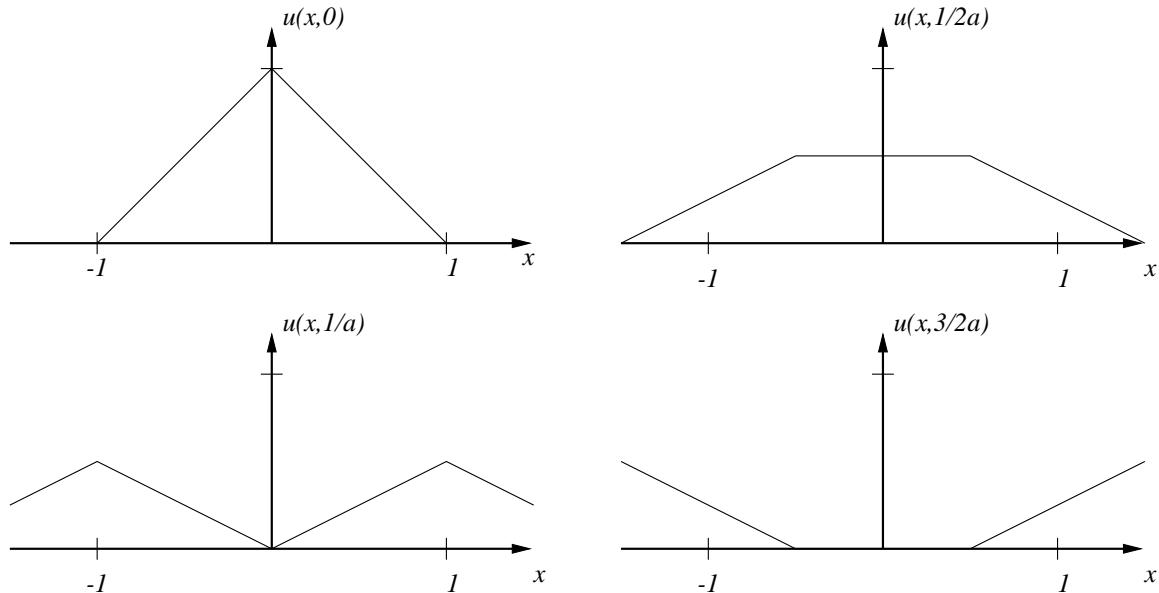


Figure 6.1.3: Solution of Example 6.1.1 at  $t = 0$  (upper left),  $1/2a$  (upper right),  $1/a$  (lower left), and  $3/2a$  (lower right).

## 6.2 The Courant, Friedrichs, Lewy Theorem

All finite difference schemes that we have developed and studied for scalar problems extend to vector systems of the form (6.1.1a) or (6.1.1b). Thus, for example, the Lax-Friedrichs scheme for (6.1.1b) is

$$\frac{\mathbf{U}_j^{n+1} - (\mathbf{U}_{j+1}^n + \mathbf{U}_{j-1}^n)/2}{\Delta t} + \mathbf{A}_j^n \frac{\mathbf{U}_{j+1}^n - \mathbf{U}_{j-1}^n}{2\Delta x} = \mathbf{b}_j^n,$$

where  $\mathbf{A}_j^n \equiv \mathbf{A}(j\Delta x, n\Delta t \mathbf{U}_j^n)$ , etc. The Courant, Friedrichs, Lewy Theorem also applies to vector systems of the form (6.1.1b).

**Theorem 6.2.1.** (*Courant, Friedrichs, Lewy*). *A necessary condition for the convergence of a finite difference scheme is that its domain of dependence contain the domain of dependence of the partial differential system (6.1.1b).*

*Proof.* We'll prove the theorem for constant-coefficient problems. Proofs in more general situations are similar. The matrix  $\mathbf{Q}$  of (6.1.4b) is zero for constant-coefficient problems and the canonical form of (6.1.1b) becomes

$$\mathbf{w}_t + \mathbf{\Lambda} \mathbf{w}_x = \mathbf{g}. \quad (6.2.1)$$

The eigenvalues  $\lambda_1 < \lambda_2 < \dots < \lambda_m$  are constant and the characteristics are straight lines. A typical set of characteristics passing through an arbitrary point  $P(x_0, t_0)$  is shown in Figure 6.2.1. Again, for simplicity, suppose that a four-point one-level explicit finite difference scheme

$$\mathbf{W}_j^{n+1} = \mathbf{C}_{-1} \mathbf{W}_{j-1}^n + \mathbf{C}_0 \mathbf{W}_j^n + \mathbf{C}_1 \mathbf{W}_{j+1}^n + \Delta t \mathbf{g}_j^n. \quad (6.2.2)$$

is used to obtain the numerical solution of (6.2.1). The stencil of this difference scheme and its domain of dependence are shown in Figure 6.2.2. The domain of dependence of the finite difference scheme is the region  $ABP$ , which is bounded by the  $x$  axis and the lines of slope  $\pm\Delta t/\Delta x$ . The domain of dependence of the partial differential equation is shown as the region  $CDP$  in Figures 6.2.1 and 6.2.2. We suppose that the domain of dependence of the partial differential equation does not contain that of the difference

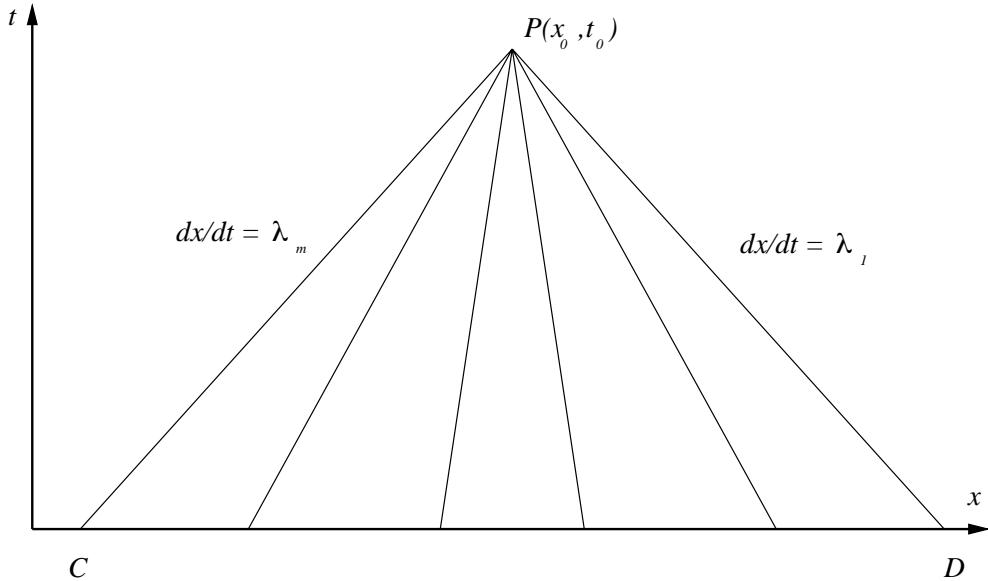


Figure 6.2.1: Characteristics and domain of dependence for a constant coefficient problem of the form (6.1.1b).

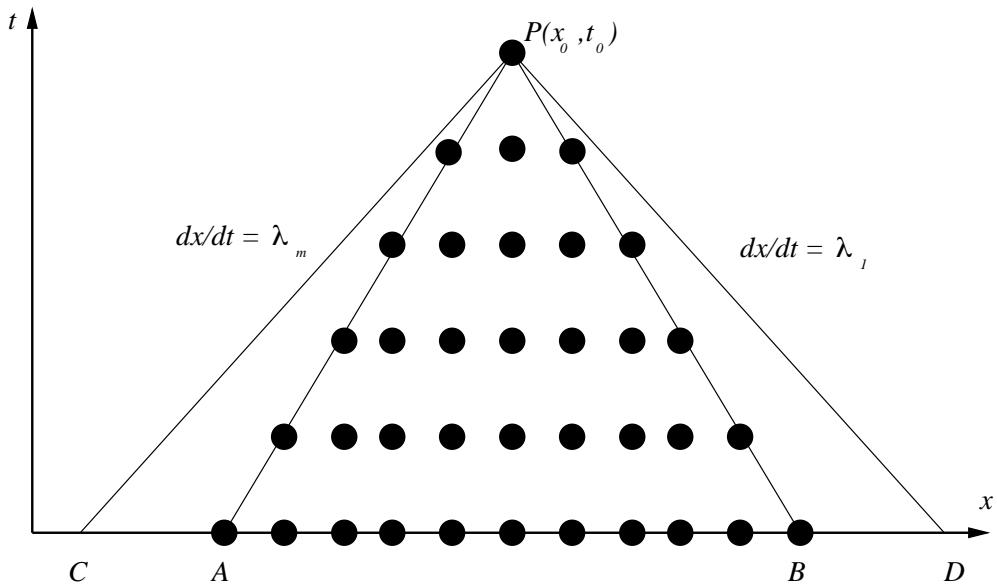


Figure 6.2.2: Sample domains of dependence of constant coefficient partial differential equation of the form (6.1.1b) and a difference scheme of the form of (6.2.2).

scheme, again, as shown in Figure 6.2.2. Consider a problem where the initial data vanishes on  $AB$  and the source  $\mathbf{g}$  vanishes on  $ABP$ . The finite difference solution at  $P$  is, therefore, trivial. However, if the initial data is nonzero on  $CA$  or  $BD$  or the source  $\mathbf{g}$  is nonzero on  $CAP$  or  $BDP$ , the solution of the partial differential system need not

be zero at  $P$ . Moreover, the solution of the difference scheme cannot converge to that of the partial differential equation as long as the mesh is refined with  $\Delta t/\Delta x$  fixed, since the solution of the difference equation will always be zero at  $P$ . We, therefore, have a generalization of the Courant, Friedrichs, Lewy Theorem considered in Chapter 2.  $\square$

*Remark 1.* The Courant, Friedrichs, Lewy Theorem restricts all explicit finite difference schemes to satisfy

$$\max_{1 \leq i \leq m} |\lambda_i| \frac{\Delta t}{\Delta x} \leq 1. \quad (6.2.3)$$

*Remark 2.* The Courant, Friedrichs, Lewy Theorem is usually stated as a stability restriction rather than a convergence restriction. Our proof would have to be supplemented by the Lax equivalence theorem in order to demonstrate that it also imposes a stability restriction.

The von Neumann method can be used to study the stability of homogeneous ( $\mathbf{b} = \mathbf{0}$ ) constant coefficient systems with periodic initial data. Thus, consider

$$\mathbf{u}_t + \mathbf{A}\mathbf{u}_x = \mathbf{0}, \quad -\infty < x < \infty, \quad t > 0, \quad (6.2.4a)$$

$$\mathbf{u}(x, 0) = \phi(x), \quad -\infty < x < \infty, \quad (6.2.4b)$$

where  $\mathbf{A}$  is constant and  $\phi(x)$  is periodic in  $x$ . As an example, consider a one-level finite difference scheme having the form

$$\sum_{|s| \leq S} \mathbf{D}_s \mathbf{U}_{j+s}^{n+1} = \sum_{|s| \leq S} \mathbf{C}_s \mathbf{U}_{j+s}^n, \quad (6.2.5)$$

where  $\mathbf{C}_s$  and  $\mathbf{D}_s$  are  $m \times m$  matrices and  $S$  is an integer.

As in the scalar case, assume a solution in the form of a discrete Fourier series

$$\mathbf{U}_j^n = \sum_{k=0}^{J-1} \mathbf{A}_k^n e^{2\pi i j k / J}. \quad (6.2.6)$$

Substituting (6.2.6) into (6.2.5)

$$\sum_{|s| \leq S} \sum_{k=0}^{J-1} [\mathbf{D}_s \mathbf{A}_k^{n+1} e^{2\pi i (j+s)k / J} - \mathbf{C}_s \mathbf{A}_k^n e^{2\pi i (j+s)k / J}] = \mathbf{0}.$$

Rearranging the sums

$$\sum_{k=0}^{J-1} \sum_{|s| \leq S} [\mathbf{D}_s \mathbf{A}_k^{n+1} e^{2\pi iks/J} - \mathbf{C}_s e^{2\pi iks/J} \mathbf{A}_k^n] e^{2\pi ijk/J} = \mathbf{0}.$$

Using the orthogonality condition (3.2.2), we find

$$\sum_{|s| \leq S} \mathbf{D}_s e^{2\pi iks/J} \mathbf{A}_k^{n+1} = \sum_{|s| \leq S} \mathbf{C}_s e^{2\pi iks/J} \mathbf{A}_k^n. \quad (6.2.7)$$

Solving for  $\mathbf{A}_k^n$

$$\mathbf{A}_k^n = (\mathbf{M}_k)^n \mathbf{A}_k^0, \quad (6.2.8a)$$

where

$$\mathbf{M}_k = \left( \sum_{|s| \leq S} \mathbf{D}_s e^{2\pi iks/J} \right)^{-1} \left( \sum_{|s| \leq S} \mathbf{C}_s e^{2\pi iks/J} \right). \quad (6.2.8b)$$

The amplification factor  $\mathbf{M}_k$  is an  $m \times m$  matrix. In order to demonstrate the stability of the difference scheme (6.2.5), we must find a constant  $C$  such that

$$\|(\mathbf{M}_k)^n\| \leq C, \quad \Delta t \rightarrow 0, \quad n\Delta t \leq T. \quad (6.2.9)$$

The von Neumann necessary condition for stability implies that there exists a constant  $c$  such that

$$\rho(\mathbf{M}_k) \equiv \max_{1 \leq i \leq m} |\lambda_i(\mathbf{M}_k)| \leq 1 + c\Delta t, \quad \Delta t \rightarrow 0. \quad (6.2.10a)$$

For absolute stability, we require

$$\rho(\mathbf{M}_k) \leq 1. \quad (6.2.10b)$$

Conditions (6.2.9) and (6.2.10) may be quite difficult to verify in practice. We may try to bound the eigenvalues, but such bounds must be accurate in order to draw conclusions about stability.

### 6.3 Dissipation and Dispersion

As a prelude to understanding those difference schemes that will be effective for solving nonlinear problems with discontinuous solutions, let us return to the linear scalar initial value problem

$$u_t + au_x = 0, \quad t > 0, \quad u(x, 0) = e^{2\pi i kx}, \quad -\infty < x < \infty, \quad (6.3.1)$$

Many of the results developed in this section are qualitatively the same for nonlinear problems and vector systems, but, naturally, more difficult to analyze.

Consider an explicit finite difference approximation of (6.3.1) having the form

$$U_j^{n+1} = U_j^n - \frac{\alpha}{2}(U_{j+1}^n - U_{j-1}^n) + \frac{\beta}{2}(U_{j+1}^n - 2U_j^n + U_{j-1}^n), \quad (6.3.2)$$

where

$$\alpha = a \frac{\Delta t}{\Delta x} \quad (6.3.3)$$

is the Courant number. Several popular difference schemes have this form for different values of  $\beta$  and some are listed in Table 6.3. The centered scheme is not absolutely stable for any choice of  $\alpha$ , whereas the other schemes are absolutely stable for some range of  $\alpha$  (Chapter 3) and we seek to understand the reasons why this is so. For completeness, we'll also include the leap frog scheme

$$U_j^{n+1} = U_j^{n-1} - \alpha(U_{j+1}^n - U_{j-1}^n) \quad (6.3.4)$$

in our study.

Method	$\beta$	$\sigma$
Centered	0	0
Lax-Wendroff	$\alpha^2$	$a^2 \Delta t / 2$
Upwind	$ \alpha $	$ a  \Delta x / 2$
Lax-Friedrichs	1	$\Delta x^2 / 2 \Delta t$

Table 6.3.1: Values of the parameters  $\beta$  and  $\sigma$  for several difference schemes having the form of (6.3.2).

The scheme (6.3.2) may also be regarded as an approximation of the convection-diffusion problem

$$u_t + au_x = \sigma u_{xx}, \quad t > 0, \quad u(x, 0) = e^{2\pi ikx}, \quad -\infty < x < \infty, \quad (6.3.5a)$$

with forward time and centered space differencing. The exact solution of this problem is

$$u(x, t) = e^{-4\pi^2\sigma k^2 t} e^{2\pi ik(x-at)}. \quad (6.3.5b)$$

We have already seen that:

- The term  $\sigma u_{xx}$  in (6.3.5a) is dissipative or diffusive. It decreases the  $\mathcal{L}^2$  energy of the system (Section 1.2). When  $\sigma = 0$  the  $\mathcal{L}^2$  energy of the system is conserved.
- The dissipation is frequency dependent with the higher frequency terms having the greatest decrease in amplitude.
- When  $4\pi^2\sigma k^2 t \ll 1$ , the solution of (6.3.5a) is a good approximation of the solution of (6.3.1).

We have also seen that instabilities in finite difference schemes result in a rapid growth in the amplitudes of the high-frequency modes of the discrete solution. This suggests the addition of a small diffusion term (*e.g.*, a discrete approximation of  $\sigma u_{xx}$ ) to a finite difference scheme to kill the growth of the high-frequency oscillations without greatly affecting accuracy of the solution.

The methods of Table 6.3 are meant to approximate (6.3.1); however, with the exception of the centered scheme, they can also be regarded as discrete approximations of (6.3.5a) with various diffusivities. For example:

- The Lax-Wendroff scheme is an approximation of (6.3.5a) with forward time differences, centered space differences and  $\sigma = \beta\Delta x^2/2\Delta t = a^2\Delta t/2$ . Since the diffusivity  $\sigma = O(\Delta t)$ , the Lax-Wendroff scheme is consistent with (6.3.1) as  $\Delta t \rightarrow 0$ . The diffusivity is *artificial* since it is proportional to a numerical parameter ( $\Delta t$ ).
- The upwind scheme is an approximation of (6.3.5a) using forward time and centered space differences with  $\sigma = \beta\Delta x^2/2\Delta t = |a|\Delta x/2$ .

- Similarly, the Lax-Friedrichs scheme approximates (6.3.5a) with the same differencing and  $\sigma = \beta\Delta x^2/2\Delta t = \Delta x^2/2\Delta t$ .

The solution of (6.3.2) for the special initial conditions (6.3.1) is (*cf.* Problem 3.2.3)

$$U_j^n = (M_k)^n e^{2\pi i k j / J}, \quad (6.3.6a)$$

where

$$M_k = 1 - 2\beta \sin^2 \theta - i\alpha \sin 2\theta, \quad (6.3.6b)$$

and

$$\theta = k\pi/J. \quad (6.3.6c)$$

The magnitude of the amplification factor is

$$|M_k|^2 = 1 + (\alpha^2 - \beta^2) \sin^2 2\theta - 4\beta(1 - \beta) \sin^2 \theta. \quad (6.3.6d)$$

The magnitude of the amplification factor when  $\alpha = 0.75$  is shown as a function of  $\theta$  in Figure 6.3.1 for each of the schemes given in Table 6.3. The centered scheme amplifies the initial data for all values of  $\theta$  while the upwind, Lax-Friedrichs, and Lax-Wendroff schemes dissipate the initial amplitude for virtually all frequencies.

**Definition 6.3.1.** *A finite difference scheme is dissipative of order  $2r$  if there exists a constant  $c > 0$ , independent of  $\Delta t$  and  $\Delta x$ , such that*

$$|M_k(\theta)| \leq 1 - c|\theta|^{2r}, \quad 0 \leq \theta \leq \pi/2. \quad (6.3.7a)$$

There are other equivalent definitions. First, observe that it suffices to find a constant  $\hat{c} > 0$  satisfying

$$|M_k(\theta)|^2 \leq 1 - \hat{c}|\theta|^{2r}, \quad 0 \leq \theta \leq \pi/2. \quad (6.3.7b)$$

Strikwerda [21], Section 5.1, observes that many common schemes have an amplification factor satisfying

$$|M_k(\theta)|^2 \leq 1 - \bar{c} \sin^{2r} \theta, \quad 0 \leq \theta \leq \pi/2, \quad (6.3.7c)$$

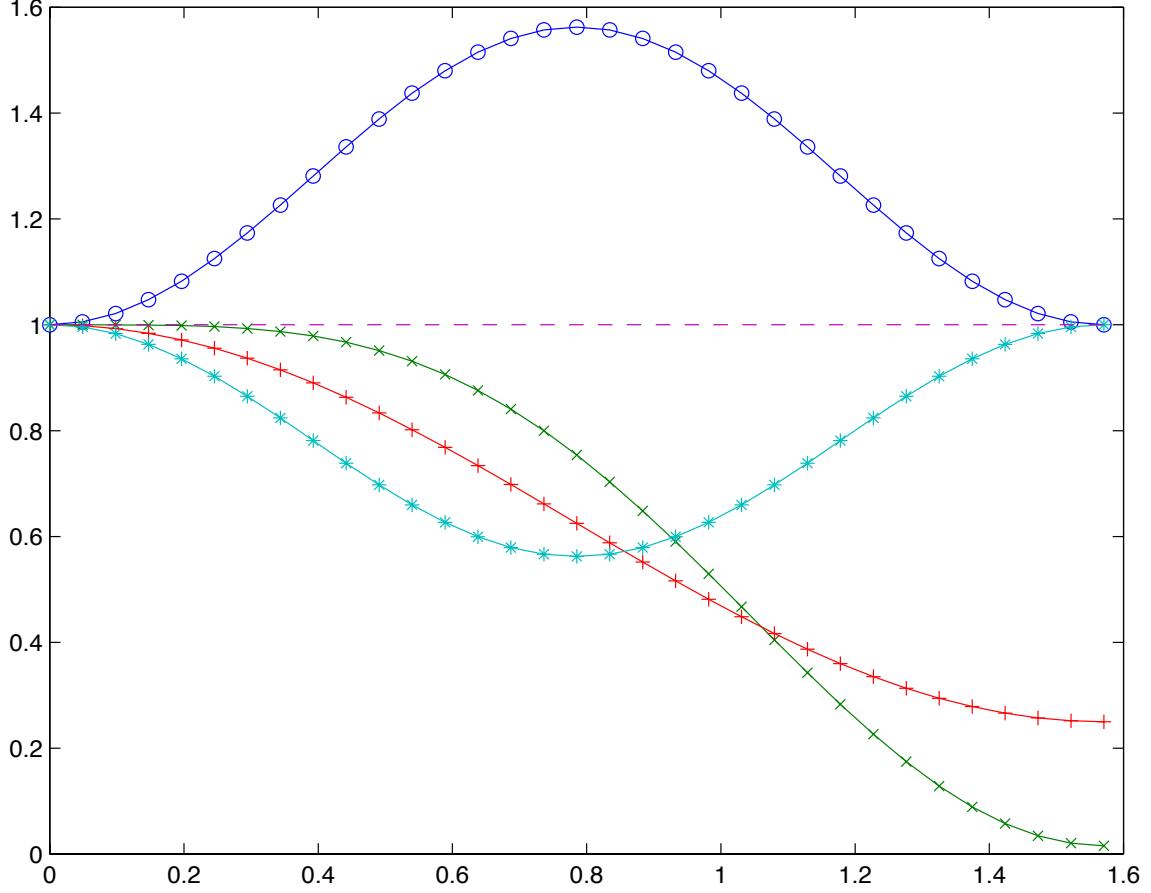


Figure 6.3.1: Magnitudes of the amplification factors when  $\alpha = 0.75$  as functions of  $\theta = k\pi/J$  for the centered (o), Lax-Wendroff (x), upwind (+), Lax-Friedrichs (\*), and leap frog (--) schemes.

with  $\bar{c} > 0$ . Since  $\theta \geq \sin \theta$ ,  $0 \leq \theta \leq \pi/2$ , (6.3.7a) and (6.3.7c) are equivalent.

*Example 6.3.1.* Using Definition 6.3.1 and (6.3.6d), let's examine the order of dissipation of the Lax-Wendroff, upwind, Lax-Friedrichs, and centered schemes. Beginning with the Lax-Wendroff scheme, we set  $\beta = \alpha^2$  in (6.3.6d) to obtain

$$|M_k|^2 = 1 + \alpha^2(1 - \alpha^2) \sin^2 2\theta - 4\alpha^2(1 - \alpha^2) \sin^2 \theta.$$

Using the double angle formula

$$|M_k|^2 = 1 - 4\alpha^2(1 - \alpha^2) \sin^4 \theta.$$

Thus, the Lax-Wendroff scheme is dissipative of order four.

For upwind differencing, we set  $\beta = |\alpha|$  in (6.3.6d) and find

$$|M_k|^2 = 1 - 4|\alpha|(1 - |\alpha|)\sin^2 \theta.$$

Thus, upwind differencing is dissipative of order two. The Lax-Wendroff scheme with fourth-order dissipation gives a more accurate representation of the lower-frequency modes than upwind differencing and greater dissipation of the high-frequency modes.

For the Lax-Friedrichs scheme, we set  $\beta = 1$  in (6.3.6d) and obtain

$$|M_k|^2 = 1 - (1 - \alpha^2)\sin^2 2\theta.$$

For small  $\theta$ , we can find a constant  $c$  ( $\approx 4(1 - \alpha^2)$ ) so that (6.3.7a) is satisfied. However, this bound will not hold when  $\theta \approx \pi/2$ . Thus, the Lax-Friedrichs scheme is not strictly dissipative. It will reduce the magnitude of the amplification factor for most frequencies, but not those of the highest-frequency modes (Figure 6.3.1).

Finally, the centered space scheme has  $|M_k(\theta)| > 1$ ,  $0 \leq \theta \leq \pi/2$ , and is not dissipative.

Using the von Neumann method on the leap frog scheme (6.3.4), we find the two amplification factors as (*cf.* Problem 1 at the end of this Section)

$$M_k^\pm = i\alpha \sin 2\theta \pm \sqrt{1 - \alpha^2 \sin^2 2\theta}. \quad (6.3.8a)$$

There are two amplification factors since the leap frog method is a two-level scheme and, as described in Section 4.3, the discrete solution involves a linear combination of the two. Assuming that  $|\alpha| \leq 1$ , we see that both factors have unit magnitude, *i.e.*,

$$|M_k^\pm|^2 = 1. \quad (6.3.8b)$$

Thus, the leap frog scheme neither dissipates nor amplifies the magnitudes of the initial data for any frequency (Figure 6.3.1).

With the prescribed initial data, the exact solution of (6.3.1) has unit magnitude. Thus, the leap frog scheme has no amplitude error. The question, of course, is if the leap frog scheme has no amplitude error, what error does it have? In order to answer this query, write the amplification factor (either (6.3.6b) or (6.3.8a)) in the polar form

$$M_k = |M_k|e^{-i\phi_k} \quad (6.3.9a)$$

where

$$\tan \phi_k = \frac{\alpha \sin 2\theta}{1 - 2\beta \sin^2 \theta} \quad (6.3.9b)$$

for (6.3.6b) and

$$\tan \phi_k^\pm = \pm \frac{\alpha \sin 2\theta}{\sqrt{1 - \alpha^2 \sin^2 2\theta}} \quad (6.3.9c)$$

for (6.3.8a). Using (6.3.6a), the solution of the difference equation (6.3.2) may be written in the form

$$U_j^n = |M_k|^n e^{-in\phi_k} e^{2\pi i k j / J} = |M_k|^n e^{2\pi i k (x_j - \gamma_k t_n)} \quad (6.3.10a)$$

where

$$x_j = j\Delta x = \frac{j}{J}, \quad t_n = n\Delta t, \quad \gamma_k = \frac{\phi_k}{2\pi k \Delta t}. \quad (6.3.10b)$$

We will also regard this as being one component of the solution of the leap frog scheme (6.3.4) when (6.3.9c) is used for  $\phi$ .

The *phase speed* of a wave of the form

$$e^{i(\kappa x - \omega t)}$$

is the quantity  $\omega/\kappa$ . It is the speed at which waves of frequency  $\omega$  propagate. If the phase speed depends on the wave number  $\kappa$  then the wave is said to be *dispersive*, otherwise it is non-dispersive. The exact solution of the model problem (6.3.1) is

$$u(x, t) = e^{2\pi i k (x - at)}.$$

Hence,  $\omega = 2\pi ka$  and  $\kappa = 2\pi k$ . The phase speed is  $a$  and the wave is non-dispersive. However, the phase speed  $\gamma_k$  of either finite difference scheme (6.3.2) or (6.3.4) depends on  $k$  and, hence, these waves are dispersive. Normalized phase speeds ( $\gamma_k/a$ ) are shown as functions of  $\theta$  for the schemes of Table 6.3 and the leap frog scheme (6.3.4). Jumps in the phase speed correspond to branch cuts in the arctangent function. While phase speeds appear reasonable when  $\theta$  is small, all have large errors when  $\theta$  is near  $\pi/2$ . With  $\theta = \pi/2$ , for example, both (6.3.9b) and (6.3.9c) give  $\tan \phi_k = 0$  and, consequently,

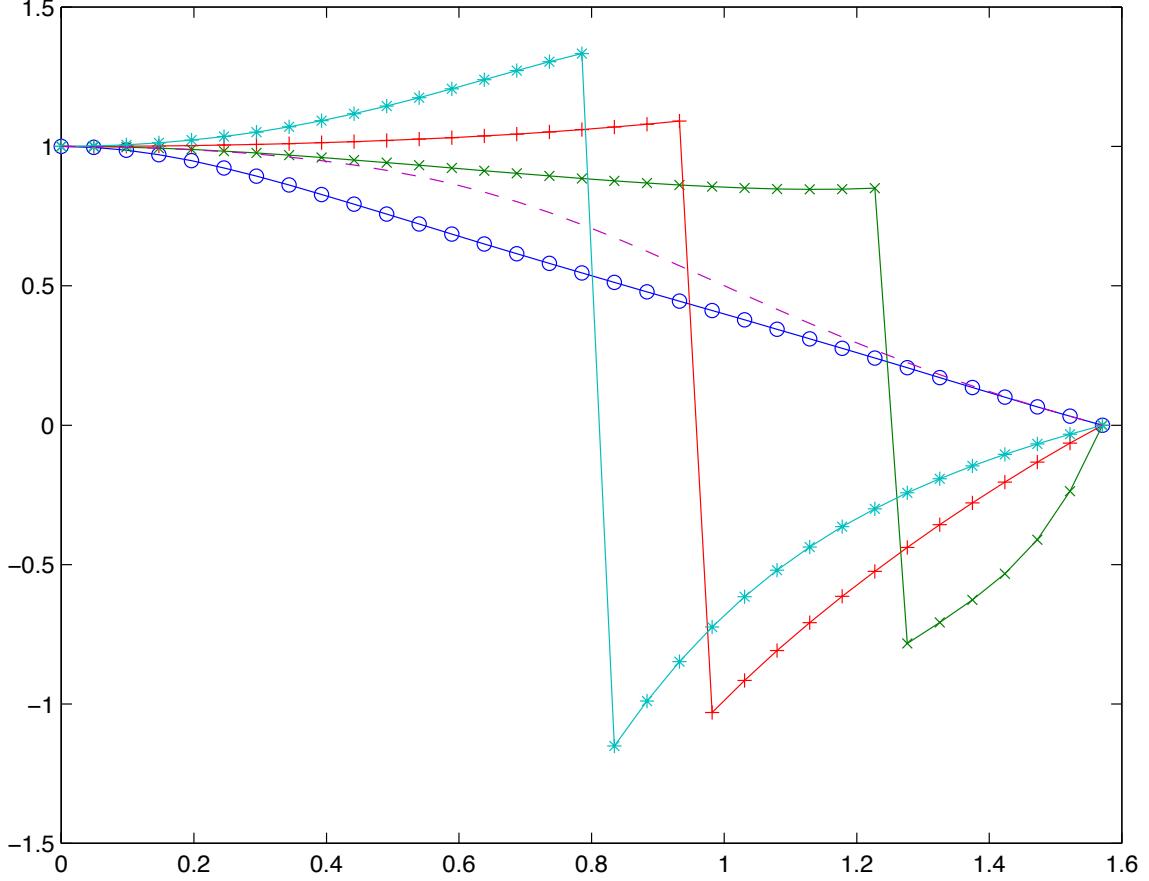


Figure 6.3.2: Normalized phase speeds  $\gamma_k/a$  when  $\alpha = 0.75$  as functions of  $\theta = k\pi/J$  for the centered (o), Lax-Wendroff (x), upwind (+), Lax-Friedrichs (\*), and leap frog (--) schemes.

$\gamma_k = 0$  (Figure 6.3.9b). Thus, instead of having speed  $a$ , waves of both schemes (6.3.2) and (6.3.4) are practically stationary when  $\theta \approx \pi/2$ .

In order to get additional qualitative information about the degradation of the phase speed with increasing  $\theta$ , let's expand (6.3.9b) in a series about  $\theta = 0$ . Thus,

$$\tan \phi_k \approx 2\theta\alpha(1 - 2\theta^2/3)(1 + 2\beta\theta^2) \approx 2\theta\alpha[1 + 2\theta^2(\beta - 1/3)].$$

For small  $z$ ,  $\tan^{-1} z \approx z - z^3/3$ ; thus,

$$\phi_k \approx 2\theta\alpha[1 - \frac{2\theta^2}{3}(1 + 2\alpha^2 - 3\beta)].$$

Using (6.3.10b)

$$\gamma_k = \frac{\phi_k}{2\pi k\Delta t} \approx \frac{2(k\pi/J)(a\Delta t/\Delta x)}{2\pi k\Delta t}[1 - \frac{2\theta^2}{3}(1 + 2\alpha^2 - 3\beta)]$$

Method	$\beta$	Approximate $\gamma_k$
Centered	0	$a[1 - 2\theta^2(1 + 2\alpha^2)/3]$
Lax-Wendroff	$\alpha^2$	$a[1 - 2\theta^2(1 - \alpha^2)/3]$
Upwind	$ \alpha $	$a[1 - 2\theta^2(1 - \alpha)(1 - 2\alpha)/3]$
Lax-Friedrichs	1	$a[1 + 4\theta^2(1 - \alpha^2)/3]$
Leap frog		$\pm a[1 - 2\theta^2(1 - \alpha^2)/3]$

Table 6.3.2: Approximate phase speeds for the difference schemes (6.3.2) and (6.3.4) when  $\theta \ll 1$ .

or

$$\gamma_k \approx a[1 - \frac{2\theta^2}{3}(1 + 2\alpha^2 - 3\beta)]. \quad (6.3.11a)$$

The results are summarized in Table 6.3 for the four schemes of Table 6.3. A similar analysis of the leap frog scheme gives (*cf.* Problem 1 at the end of this Section)

$$\gamma_k \approx \pm a[1 - \frac{2\theta^2}{3}(1 - \alpha^2)]. \quad (6.3.11b)$$

Clearly, the positive sign corresponds to the correct solution and the negative sign to a parasitic solution introduced by the multi-level scheme.

Comparing the data in Figure 6.3.2 and Table 6.3, all schemes considered have approximately the correct phase speed for low-frequency waves ( $\theta \ll 1$ ), but have large errors when  $\theta \approx \pi/2$ . Assuming that  $0 < \alpha \leq 1$ , the data of Table 6.3 indicates that low-frequency waves travel slower than the correct speed for the centered scheme, the Lax-Wendroff scheme, the leap frog scheme, and the upwind scheme when  $\alpha \leq 1/2$ . Under the same assumption, low-frequency waves travel faster than the correct speed for the Lax-Friedrichs scheme and the upwind scheme when  $\alpha > 1/2$ . These conclusions are verified for  $\alpha = 0.75$  in Figure 6.3.2.

Since the high-frequency waves have large phase-speed errors, it seems appropriate to dissipate them. The upwind and Lax-Wendroff schemes dissipate the high-frequency waves the most (Figure 6.3.1); thus, they would be good schemes to use when high-frequency information is present. The leap frog scheme, on the other hand, does not dissipate waves of any frequency and would not be a good one to use in connection with high-frequency data. While the Lax-Friedrichs scheme is dissipative for most frequencies,

it is less so for the highest-frequency modes. Finally, the centered scheme is not dissipative and should not be used.

### Problems

1. Use the Von Neumann method to show that the amplification factors of the Leap-frog difference scheme (6.3.4) for the initial value problem (6.3.1) are given by (6.3.8a). Show that the amplitudes and phase speeds of the amplification factors are given by (6.3.8b) and (6.3.9c), respectively. Obtain approximations of the phase speeds for small and large values of  $\theta$ . Compare the approximate and true phase speeds.

## 6.4 Nonlinear Problems

The time step restriction implied by the Courant, Friedrichs, Lewy Theorem is not restrictive for many hyperbolic problems; thus, explicit finite difference schemes will often be acceptable. Implicit schemes are useful when (*i*) a problem involves widely disparate characteristic speeds, (*ii*) a steady-state (time independent) solution is sought, or (*iii*) reactions are involved. The treatment of implicit schemes for hyperbolic systems is fundamentally the same as for parabolic systems, so we'll concentrate on the simpler explicit schemes for solving a system of one-dimensional conservation laws having the form (6.1.1a).

We have just learned that the Lax-Wendroff scheme has good dissipative properties, particularly regarding high-frequency modes. It is, however, difficult to use because it generally requires knowledge of the derivatives of the Jacobian  $\mathbf{A}$  (*cf.* (6.1.1c and Section 3.3)). An analogy with Runge-Kutta methods for ordinary differential equations would suggest that derivatives of  $\mathbf{A}$  could be avoided at the expense of additional evaluations of the flux  $\mathbf{f}$ . This typically leads to predictor-corrector methods and the first one that we will study is due to Richtmyer ([17], Section 12.7). Using this method, a predicted solution of (6.1.1a) is calculated by using the Lax-Friedrichs scheme for one-half time step, *i.e.*,

$$\mathbf{U}_{j+1/2}^{n+1/2} = \mu_x \mathbf{U}_{j+1/2}^n - \frac{\Delta t}{2\Delta x} \delta_x \mathbf{f}_{j+1/2}^n = \frac{\mathbf{U}_{j+1}^n + \mathbf{U}_j^n}{2} - \frac{\Delta t}{2\Delta x} (\mathbf{f}_{j+1}^n - \mathbf{f}_j^n). \quad (6.4.1a)$$

This solution is corrected using the leap frog scheme

$$\mathbf{U}_j^{n+1} = \mathbf{U}_j^n - \frac{\Delta t}{\Delta x} \delta_x \mathbf{f}_j^{n+1/2} = \mathbf{U}_j^n - \frac{\Delta t}{\Delta x} (\mathbf{f}_{j+1/2}^{n+1/2} - \mathbf{f}_{j-1/2}^{n+1/2}). \quad (6.4.1b)$$

The computational stencils of the predictor and corrector steps of this scheme, which we'll call the *Richtmyer two-step method*, are shown in Figure 6.4.1. The local discretization error can be shown to be  $O(\Delta t^2) + O(\Delta x^2)$ . The  $O(\Delta t^2)$  error is due to the centering of the leap frog corrector about  $t_{n+1/2}$ .

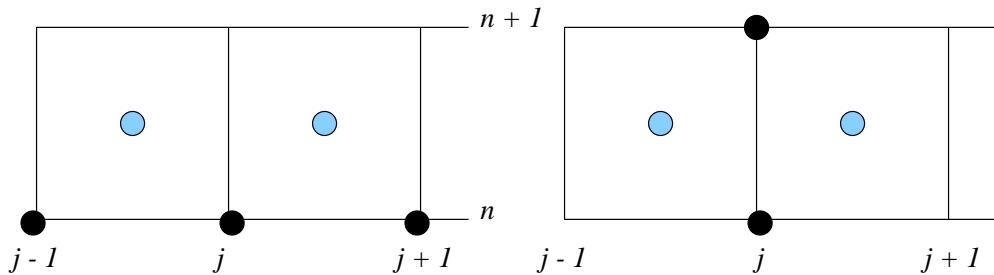


Figure 6.4.1: Computational stencils of the predictor (left) and corrector (right) stages of the Richtmyer two-step method (6.4.1). Predicted solutions are shown in light blue.

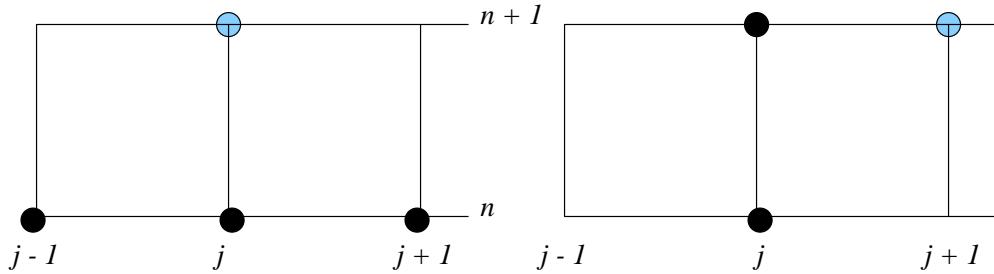


Figure 6.4.2: Computational stencils of the predictor (left) and corrector (right) stages of MacCormack's method (6.4.2). Predicted solutions are shown in light blue.

The Richtmyer two-step and Lax-Wendroff schemes are identical when  $\mathbf{f}(\mathbf{u}) = \mathbf{A}\mathbf{u}$  and  $\mathbf{A}$  is a constant. However, the Richtmyer two-step scheme is much simpler to implement than the Lax-Wendroff scheme for nonlinear systems. The equivalence of the two methods implies that the Richtmyer two-step scheme has similar properties to the Lax-Wendroff scheme. Indeed, both are stable for linear problems when the Courant, Friedrichs, Lewy condition  $\max_{1 \leq i \leq m} |\lambda_i| \Delta t / \Delta x \leq 1$ , is satisfied. Here  $\lambda_i$ ,  $1 \leq i \leq m$ , is an eigenvalue of  $\mathbf{A}$ .

MacCormack's scheme (*cf.* [16], Section 4.5) uses a forward time-backward space predictor and a backward time-forward space corrector for one-half time step, *i.e.*,

$$\hat{\mathbf{U}}_j^{n+1} = \mathbf{U}_j^n - \frac{\Delta t}{\Delta x} \nabla_x \mathbf{f}_j^n = \mathbf{U}_j^n - \frac{\Delta t}{\Delta x} (\mathbf{f}_j^n - \mathbf{f}_{j-1}^n), \quad (6.4.2a)$$

$$\mathbf{U}_j^{n+1} = \frac{\mathbf{U}_j^n + \hat{\mathbf{U}}_j^{n+1}}{2} - \frac{\Delta t}{2\Delta x} \Delta_x \hat{\mathbf{f}}_j^{n+1} = \frac{\mathbf{U}_j^n + \hat{\mathbf{U}}_j^{n+1}}{2} - \frac{\Delta t}{2\Delta x} (\hat{\mathbf{f}}_{j+1}^{n+1} - \hat{\mathbf{f}}_j^{n+1}) \quad (6.4.2b)$$

where  $\hat{\mathbf{f}}_j^{n+1} \equiv \mathbf{f}(\hat{\mathbf{U}}_j^{n+1})$ .

The computational stencils of the predictor and corrector stages of MacCormack's scheme are shown in Figure 6.4.2. MacCormack's scheme can be shown to possess the properties noted for the Richtmyer two-step scheme, including identity with the Lax-Wendroff scheme for linear constant-coefficient problems. Finally, MacCormack's scheme can also be used in reverse order as

$$\hat{\mathbf{U}}_j^{n+1} = \mathbf{U}_j^n - \frac{\Delta t}{\Delta x} \Delta_x \mathbf{f}_j^n = \mathbf{U}_j^n - \frac{\Delta t}{\Delta x} (\mathbf{f}_{j+1}^n - \mathbf{f}_j^n), \quad (6.4.3a)$$

$$\mathbf{U}_j^{n+1} = \frac{\mathbf{U}_j^n + \hat{\mathbf{U}}_j^{n+1}}{2} - \frac{\Delta t}{2\Delta x} \nabla_x \hat{\mathbf{f}}_j^{n+1} = \frac{\mathbf{U}_j^n + \hat{\mathbf{U}}_j^{n+1}}{2} - \frac{\Delta t}{2\Delta x} (\hat{\mathbf{f}}_j^{n+1} - \hat{\mathbf{f}}_{j-1}^{n+1}). \quad (6.4.3b)$$

### 6.4.1 Artificial Viscosity

We have seen that nonlinear conservation laws can give rise to discontinuous solutions such as shock waves and contact surfaces. The principal computational approaches to treat problems with discontinuities involve (*i*) explicit tracking and (*ii*) implicit “capturing.” Tracking is the more accurate of the two, but it is difficult to implement for problems in two and three dimensions. We'll concentrate on shock-capturing schemes that isolate discontinuities without knowing their locations. Let us begin with a simple scalar problem.

*Example 6.4.1.* Consider the Riemann problem for the inviscid Burgers' equation (*cf.* Problem 1.3.1)

$$u_t + \frac{(u^2)_x}{2} = 0, \quad -\infty < x < \infty, \quad t > 0, \quad (6.4.4a)$$

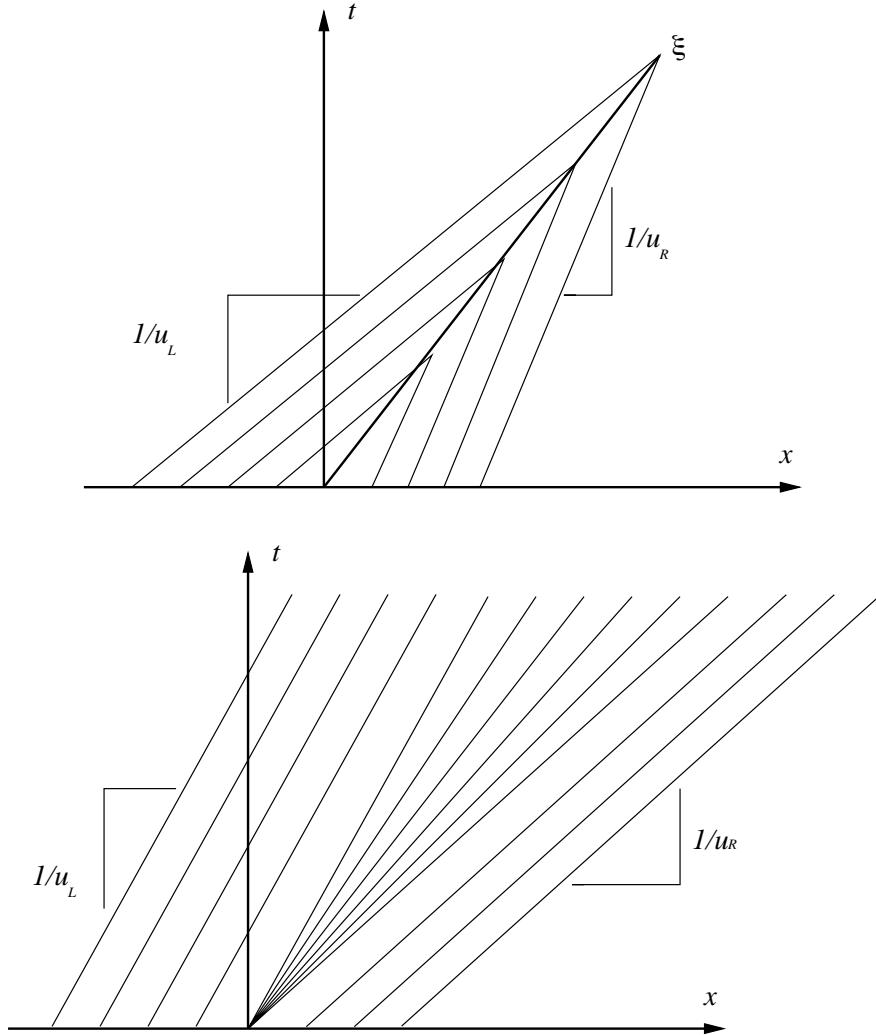


Figure 6.4.3: Shock (top) and expansion (bottom) wave characteristics of the Riemann problem (6.4.4).

$$u(x, 0) = \begin{cases} u_L, & \text{if } x < 0 \\ u_R, & \text{if } x \geq 0 \end{cases}. \quad (6.4.4b)$$

Recall that a Riemann problem is a Cauchy (initial value) problem with piecewise constant initial data. If  $u_L > u_R$  the solution of (6.4.4) features the shock wave

$$u(x, t) = \begin{cases} u_L, & \text{if } x/t < \dot{\xi} \\ u_R, & \text{if } x/t \geq \dot{\xi} \end{cases}, \quad \dot{\xi} = \frac{u_L + u_R}{2}. \quad (6.4.5a)$$

If  $u_L < u_R$  the solution of (6.4.4) has the expansion

$$u(x, t) = \begin{cases} u_L, & \text{if } x/t < u_L \\ x/t, & \text{if } u_L \leq x/t < u_R \\ u_R, & \text{if } x/t \geq u_R \end{cases}. \quad (6.4.5b)$$

The characteristics of the shock and expansion solutions are shown in Figure 6.4.3. The characteristics are straight lines with slopes of  $1/u_L$  for  $x/t < u_L$  and  $1/u_R$  for  $x/t \geq u_R$ . No characteristics are generated within the expansion wave. The entropy condition that shocks can only exist when characteristics intersect can be used to infer that the two characteristic families are not separated by a shock. Viscosity arguments can be used to construct the expansion fan [14].

Consider solving a shock problem for (6.4.4) with  $u_L = 1$  and  $u_R = 0$  by the Lax-Friedrichs scheme

$$U_j^{n+1} = \frac{U_{j+1}^n + U_{j-1}^n}{2} - \frac{\bar{\alpha}}{4}[(U_{j+1}^n)^2 - (U_{j-1}^n)^2]$$

and the Richtmyer two-step (6.4.1) scheme

$$U_{j+1/2}^{n+1/2} = \frac{U_{j+1}^n + U_j^n}{2} - \frac{\bar{\alpha}}{4}[(U_{j+1}^n)^2 - (U_j^n)^2],$$

$$U_j^{n+1} = U_j^n - \frac{\bar{\alpha}}{2}[(U_{j+1/2}^{n+1/2})^2 - (U_{j-1/2}^{n+1/2})^2],$$

where

$$\bar{\alpha} = \frac{\Delta t}{\Delta x}. \quad (6.4.6)$$

Selecting  $\Delta x = 0.1$  and  $\bar{\alpha} = 0.75$ , we show solutions obtained by the two methods for five time steps in Figure 6.4.4. The two numerical solutions are compared with the exact solution at  $t = 3/40$  in Figure 6.4.5. The Lax-Friedrichs solution is a monotone decreasing function of  $x$  through the shock while the Richtmyer two step solution has an overshoot.

This situation is the same as the one that we observed for convection-diffusion systems (Section 4.4.2):

- first-order finite difference methods preserve monotonicity of the solution near a discontinuity but have excess dissipation and
- higher-order methods introduce oscillation near discontinuities.

Let us formally explore these notions, at least for scalar problems.

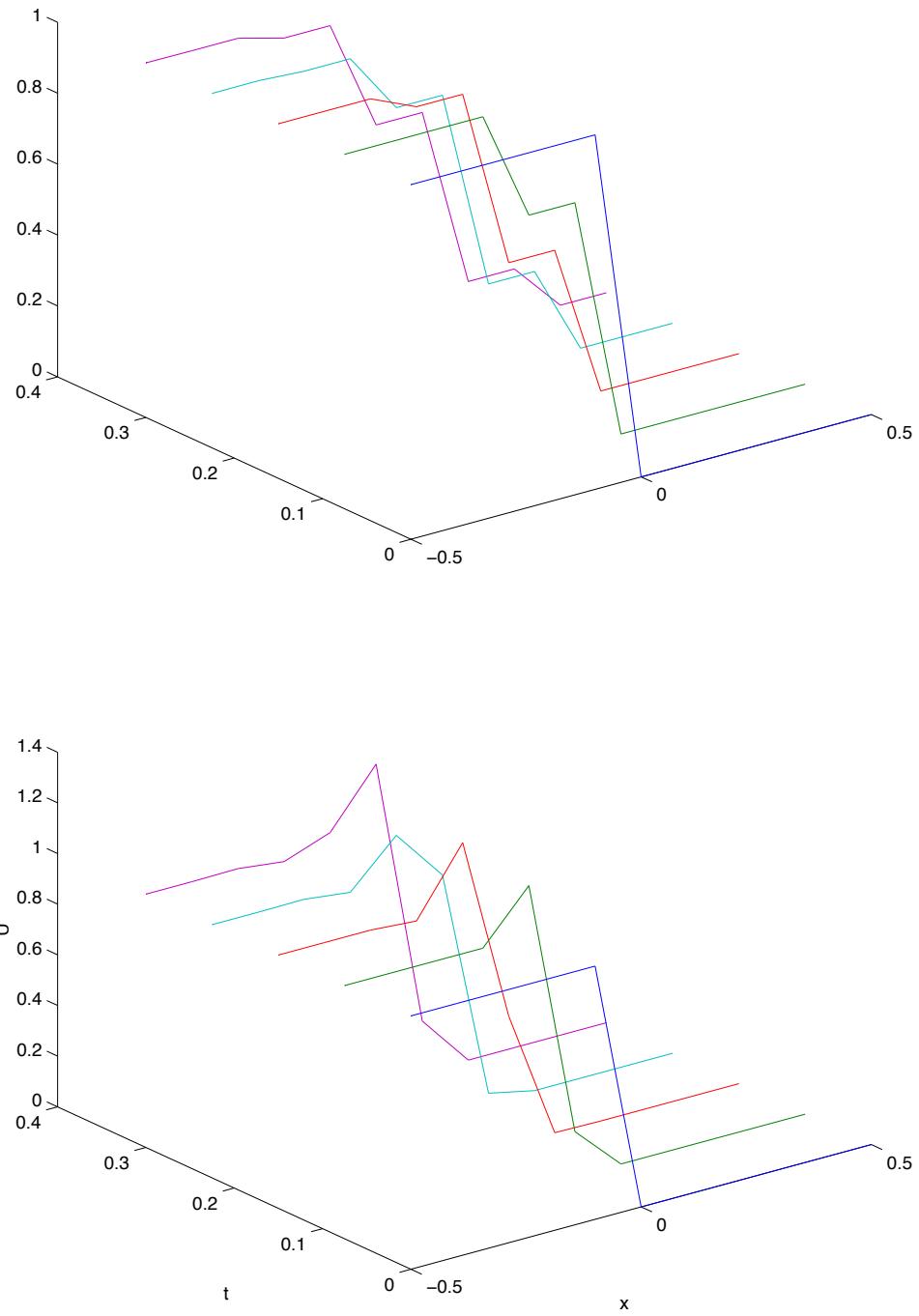


Figure 6.4.4: Shock wave solution of the Riemann problem (6.4.4) using the Lax-Friedrichs (top) and Richtmyer two-step (bottom) schemes for five time steps.

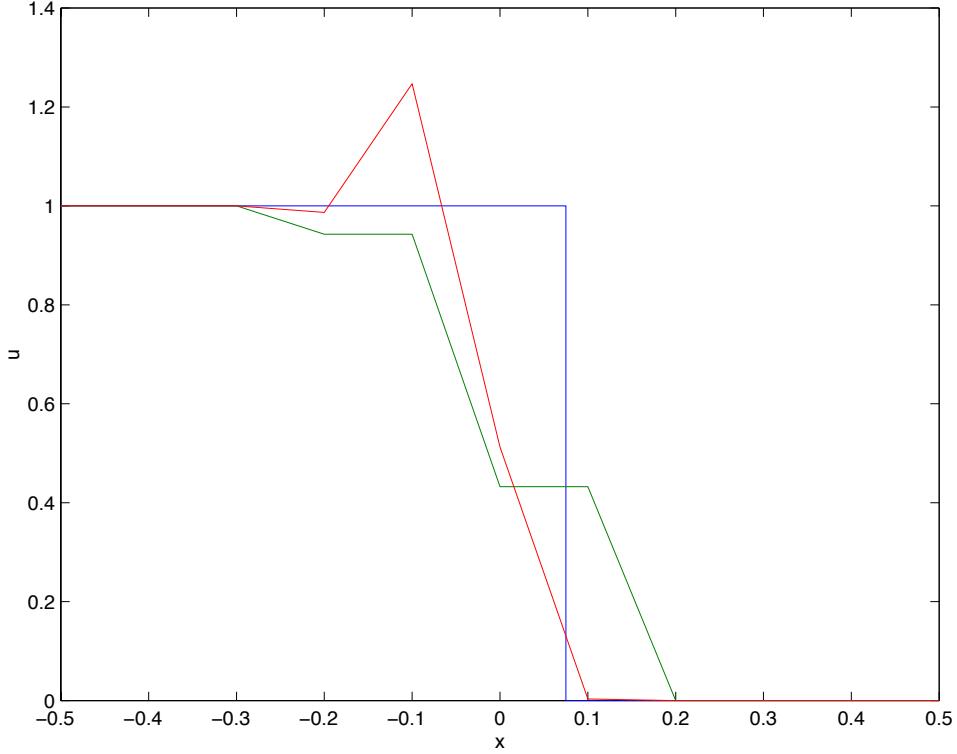


Figure 6.4.5: Comparison of the exact, Lax-Friedrichs, and Richtmyer two-step solutions of (6.4.4) at  $t = 2\Delta t$ .

**Definition 6.4.1.** *A finite-difference approximation of (6.1.1a) is in conservation form if it can be written as*

$$\frac{\mathbf{U}_j^{n+1} - \mathbf{U}_j^n}{\Delta t} + \frac{\mathbf{F}_{j+1/2}^n - \mathbf{F}_{j-1/2}^n}{\Delta x} = 0 \quad (6.4.7a)$$

where

$$\mathbf{F}_{j+1/2}^n = \mathbf{F}(\mathbf{U}_{j-S+1}^n, \mathbf{U}_{j-S+2}^n, \dots, \mathbf{U}_{j+S}^n) \quad (6.4.7b)$$

and

$$\mathbf{F}_{j-1/2}^n = \mathbf{F}(\mathbf{U}_{j-S}^n, \mathbf{U}_{j-S+1}^n, \dots, \mathbf{U}_{j+S-1}^n) \quad (6.4.7c)$$

for some integer  $S > 0$ . The function  $\mathbf{F}$  is called the numerical flux and it is consistent with the physical flux  $\mathbf{f}$  if

$$\mathbf{F}(\mathbf{v}, \mathbf{v}, \dots, \mathbf{v}) = \mathbf{f}(\mathbf{v}) \quad (6.4.7d)$$

for any smooth function  $v$ .

*Remark 1.* Consistency, in this situation, implies that the numerical and exact fluxes agree for uniform data. You might check whether or not this is equivalent to our usual notion of consistency.

**Definition 6.4.2.** A scalar finite difference scheme having the form

$$U_j^{n+1} = G(U_{j-S}^n, U_{j-S+1}^n, \dots, U_{j+S}^n) \quad (6.4.8a)$$

is monotone if it is a monotone non-decreasing function of its arguments, i.e., if

$$\frac{\partial G}{\partial U_{j+k}^n} \geq 0, \quad |k| \leq S. \quad (6.4.8b)$$

For conservative schemes (6.4.7),

$$G(U_{j-S}^n, U_{j-S+1}^n, \dots, U_{j+S}^n) = U_j^n - \bar{\alpha}(F_{j+1/2}^n - F_{j-1/2}^n). \quad (6.4.8c)$$

*Remark 2.* This definition applies to vector systems of conservation laws; however, we present it for scalar equations for simplicity.

*Example 6.4.2.* Consider the Lax-Friedrichs scheme for a scalar conservation law having the form (6.1.1a)

$$U_j^{n+1} = G(U_{j-1}^n, U_j^n, U_{j+1}^n) = \frac{U_{j+1}^n + U_{j-1}^n}{2} - \frac{\bar{\alpha}}{2}[f(U_{j+1}^n) - f(U_{j-1}^n)].$$

Differentiating

$$\frac{\partial G}{\partial U_{j-1}^n} = \frac{1 + \alpha(U_{j-1}^n)}{2}, \quad \frac{\partial G}{\partial U_j^n} = 0, \quad \frac{\partial G}{\partial U_{j+1}^n} = \frac{1 - \alpha(U_{j+1}^n)}{2},$$

where

$$\alpha(v) = \frac{a(v)\Delta t}{\Delta x}, \quad a(v) = f'(v).$$

Thus,

$$\frac{\partial G}{\partial U_k^n} \geq 0, \quad k = j, j \pm 1,$$

and the scheme is monotone when  $|\alpha| \leq 1$ .

Define

$$F_{j+1/2}^n = F(U_j^n, U_{j+1}^n) = -\frac{1}{2\bar{\alpha}}(U_{j+1}^n - U_j^n) + \frac{f_{j+1}^n + f_j^n}{2}$$

and

$$F_{j-1/2}^n = F(U_{j-1}^n, U_j^n) = -\frac{1}{2\bar{\alpha}}(U_j^n - U_{j-1}^n) + \frac{f_j^n + f_{j-1}^n}{2}$$

Thus, the Lax-Friedrichs scheme can be written in conservation form (6.4.7). In a similar fashion, we can show that the Richtmyer two-step scheme (6.4.1) can be written in conservation form but it is not monotone (*cf.* Problem 1 at the end of this Section).

Monotone difference schemes produce smooth transitions near discontinuities; however, as indicated by the following theorem, their existence is limited.

**Theorem 6.4.1.** *A monotone finite difference scheme in the conservation form (6.1.1a) must be first-order accurate.*

*Proof.* *cf.* Sod [19] or Harten, *et al.* [9]. □

The results of Section 6.3 (Figure 6.3.1) indicate that the Lax-Wendroff (hence, the Richtmyer two-step) scheme has less dissipation for moderately high frequencies than either the Lax-Friedrichs or upwind schemes. Since discontinuous solutions have all frequencies present, perhaps the oscillations could be reduced by adding even more viscosity.

Consider any second-order difference scheme and add a diffusive term to its right-hand side having the form

$$\frac{\Delta x}{2} \frac{\delta_x(\mathbf{Q}_j \delta_x \mathbf{U}_j^n)}{\Delta x^2} = \frac{1}{2\Delta x} [\mathbf{Q}_{j+1/2}(\mathbf{U}_{j+1}^n - \mathbf{U}_j^n) - \mathbf{Q}_{j-1/2}(\mathbf{U}_j^n - \mathbf{U}_{j-1}^n)], \quad (6.4.9a)$$

where  $\mathbf{Q}_{j+1/2} = \mathbf{Q}_{j+1/2}(\mathbf{U}_j^n, \mathbf{U}_{j+1}^n)$  is an  $m \times m$  matrix that vanishes when  $\mathbf{U}_{j+1}^n - \mathbf{U}_j^n = 0$ . Artificial viscosity terms of this form were originally studied by Lax and Wendroff [15] and are discussed in Richtmyer and Morton [17], Section 12.14, and Sod [19], Section 4.2. For scalar problems, Lax and Wendroff [15] chose

$$Q_{j+1/2}(U_j^n, U_{j+1}^n) = \frac{\epsilon}{2} |a_{j+1}^n - a_j^n|, \quad (6.4.9b)$$

where  $\epsilon$  is a parameter of order unity. Thus, for example, the Lax-Wendroff scheme with artificial viscosity would become (*cf.* Section 3.3)

$$U_j^{n+1} = U_j^n - \alpha_j^n \mu_x \delta_x U_j^n + (\alpha_j^n)^2 \delta_x^2 U_j^n + \frac{\Delta t}{2\Delta x} \delta_x(Q_j \delta_x \mathbf{U}_j^n), \quad (6.4.9c)$$

where  $\alpha_j^n$  is the Courant number.

Extending the artificial viscosity model to vector systems of conservation laws is more difficult. Lax and Wendroff [15] define the matrix  $\mathbf{Q}_{j+1/2}$  in (6.4.9a) to have eigenvalues  $\epsilon_i |\lambda_{i,j+1} - \lambda_{i,j}|$ ,  $i = 1, 2, \dots, m$ , where  $\lambda_{i,j}$ ,  $i = 1, 2, \dots, m$ , are the eigenvalues of the Jacobian  $\mathbf{A}(\mathbf{U}_j^n) = \mathbf{f}_u(\mathbf{U}_j^n)$  and  $\epsilon_i$ ,  $i = 1, 2, \dots, m$ , are approximately one. This eigenvalue condition is not sufficient to uniquely determine  $\mathbf{Q}_{j\pm 1/2}$  and Lax and Wendroff [15] required  $\mathbf{Q}_{j\pm 1/2}$  to commute with  $\mathbf{A}$ . They did this by prescribing  $\mathbf{Q}_{j\pm 1/2}$  as  $m-1$  *th* degree polynomials in  $\mathbf{A}$ .

Let us illustrate the performance of the artificial viscosity technique using a simple example.

*Example 6.4.3.* Consider an initial value problem for the inviscid Burgers' equation (6.4.4a) with the data

$$u(x, 0) = \phi(x) = \begin{cases} 1, & \text{if } x < 0 \\ 1-x, & \text{if } 0 \leq x < 1 \\ 0, & \text{if } 1 \leq x \end{cases} .$$

Recall (Section 1.3) that the exact solution is

$$u(x, t) = \begin{cases} 1, & \text{if } x < t \\ \frac{x-1}{t-1}, & \text{if } t \leq x < 1 \\ 0, & \text{if } 1 \leq x \end{cases} , \quad t < 1,$$

$$u(x, t) = \begin{cases} 1, & \text{if } x < (t+1)/2 \\ 0, & \text{if } x \geq (t+1)/2 \end{cases} , \quad t \geq 1.$$

The initial ramp steepens into a shock wave at  $t = 1$ . For  $t > 1$ , the shock travels in the positive  $x$  direction with speed  $1/2$ .

We solved this problem using upwind differencing, the Lax-Friedrichs method, and the Richtmyer two-step method with and without the artificial viscosity model (6.4.9). Sod [19] solved the same problem using other methods as well as these. In each case, we used  $\Delta x = 0.02$ ,  $\Delta t = 0.01$ , and the artificial viscosity parameter  $\epsilon = 0.9$ . Solutions are shown in Figures 6.4.6 - 6.4.9 as functions of  $x$  for  $t = 0.5, 1.0$ , and  $2.0$ .

The solution shown in Figure 6.4.7 using the Lax-Friedrichs scheme has the greatest dissipation. The compression wave and shock have been spread over several computational cells. The upwind scheme, shown in Figure 6.4.6, has much less apparent dissipation. As expected, the Richtmyer two-step scheme has oscillations trailing the shock;

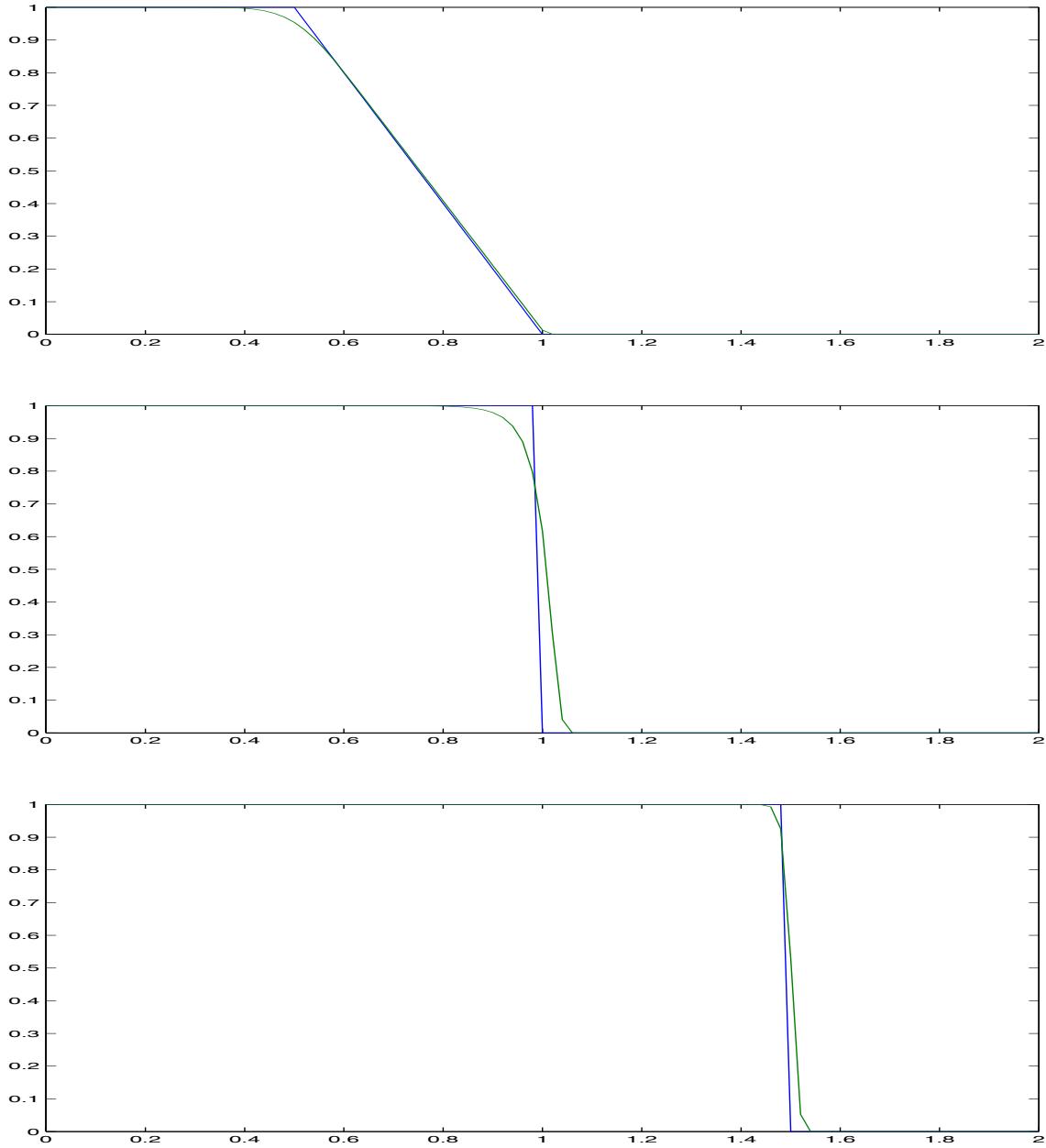


Figure 6.4.6: Comparison of exact and upwind-difference solutions of Example 6.4.3 at  $t = 0.5, 1.0, 2.0$  (top to bottom).

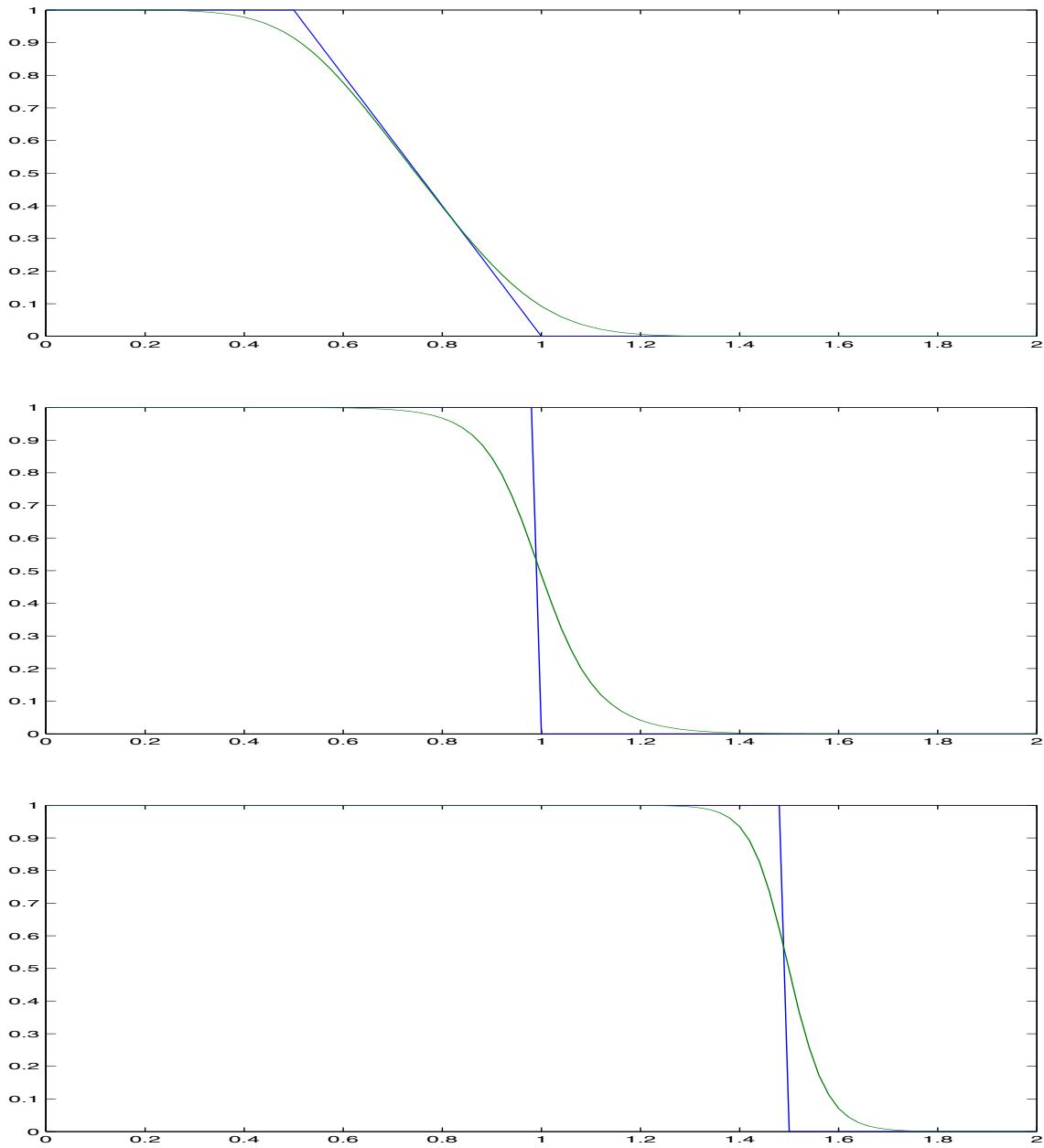


Figure 6.4.7: Comparison of exact and Lax-Friedrichs solutions of Example 6.4.3 at  $t = 0.5, 1.0, 2.0$  (top to bottom).

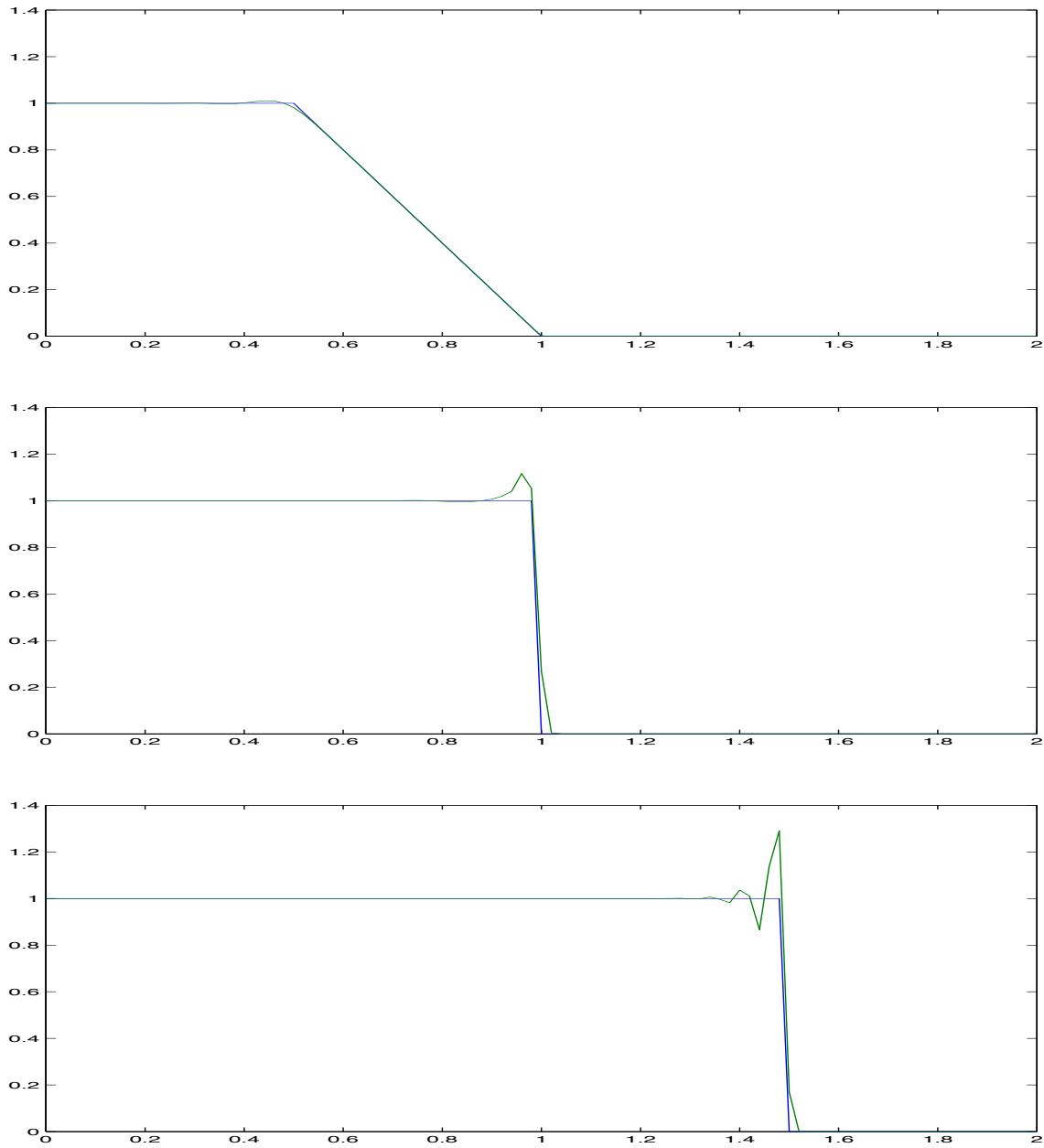


Figure 6.4.8: Comparison of exact and Richtmyer two-step solutions of Example 6.4.3 at  $t = 0.5, 1.0, 2.0$  (top to bottom).

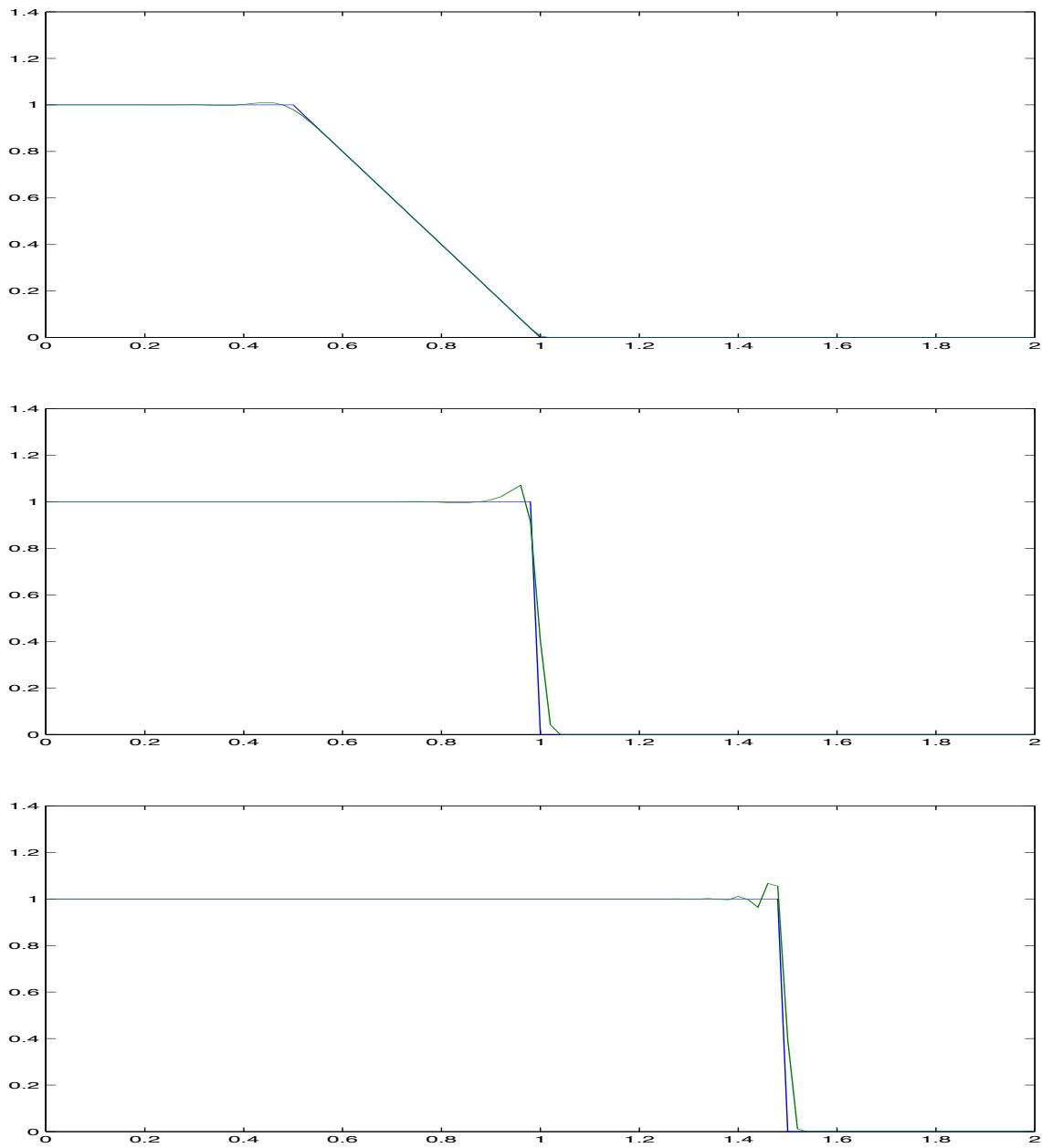


Figure 6.4.9: Comparison of exact and Richtmyer two-step with added viscosity solutions of Example 6.4.3 at  $t = 0.5, 1.0, 2.0$  (top to bottom).

however, accuracy when the solution is continuous is quite good. The oscillations of the Richtmyer two-step scheme are reduced when artificial viscosity is added (Figure 6.4.9). For shock waves, the excess dissipation does not spread as time increases. This is not the case with the linear waves studied in Section 6.3. The intersecting characteristics tend to confine the excess dissipation to the vicinity of the shock. This does not happen for linear problems where discontinuities propagate along characteristics.

### Problems

1. Show that the Richtmyer two-step scheme (6.4.1) can be written in conservation form but is not monotone. What is the numerical flux function?
2. Consider the initial value problem

$$\mathbf{u}_t + \mathbf{f}(\mathbf{u})_x = \mathbf{g}(\mathbf{u}), \quad \mathbf{u}(x, 0) = \boldsymbol{\phi},$$

where  $\mathbf{u}$ ,  $\mathbf{f}$ ,  $\mathbf{g}$ , and  $\boldsymbol{\phi}$  are  $m$ -vectors that are periodic in  $x$  with period 1.

- 2.1. Write a computer program to solve the above initial value problem on  $0 < x < 1$ ,  $0 < t \leq T$  using the Richtmyer two-step procedure (6.4.1).
- 2.2. Let  $m = 2$ ,  $u_1 = u$ ,  $u_2 = v$ , and consider the linear wave equation

$$u_t - v_x = 0, \quad v_t - u_x = 0,$$

with the initial conditions on  $0 \leq x < 1$  given as

$$u(x, 0) = 0, \quad v(x, 0) = \begin{cases} 1, & \text{if } 0 \leq x < 1/2 \\ -1, & \text{if } 1/2 \leq x < 1 \end{cases}.$$

Solve the problem on  $0 < t \leq T = 1$  for all combinations of  $J, N = 10, 20, 30$  such that  $\bar{\alpha} = \Delta t / \Delta x \leq 1$ . Calculate the exact solution and plot the exact and numerical values of  $u_1$ ,  $u_2$  as functions of  $x$  for  $t = 0, 0.25, 0.5, 0.75, 1$ . Calculate the errors in the maximum norm at  $t = 1$  for each component of the solution and each mesh. Comment on the accuracy and convergence rate.

- 2.3. Let  $m = 1$ ,  $u = u_1$ , and consider the inviscid Burgers' equation (6.4.4a) with the initial conditions on  $0 \leq x < 1$  given as

$$u(x, 0) = \begin{cases} 1, & \text{if } 0 \leq x < 0.25 \\ 0, & \text{if } 0.25 \leq x < 0.75 \\ 1, & \text{if } 0.75 \leq x < 1 \end{cases}.$$

Solve this problem on  $0 < t \leq T = 2$  for the same combinations of  $J$  and  $N$  given in Part 2. Calculate the exact solution and compare the exact and numerical solutions at by plotting them at  $t = 0, 0.5, 1, 1.5, 2$ . Calculate errors in the maximum norm at  $t = 2$ . Comment on the accuracy and estimate the convergence rate.

## 6.5 Godunov Schemes

The shock capturing finite difference schemes for nonlinear hyperbolic problems have not been very satisfying. To reiterate, the first-order methods can preserve solution monotonicity near discontinuities by introducing excess dissipation and the higher-order methods oscillate near discontinuities. In this section, we will study some modern approaches that try to alleviate these difficulties. The problem is not solved and the subject matter is still an active area of research. Thus, our results will be somewhat tentative.

Once again, let us focus on a one-dimensional system of conservation laws having the form (6.1.1). When discontinuities are present, solutions are expected to satisfy the integrated form of (6.1.1a)

$$\frac{d}{dt} \int_a^b \mathbf{u} dx = -\mathbf{f}(\mathbf{u})|_a^b \quad (6.5.1)$$

on any interval  $a < x < b$  (*cf.* Section 1.3). Discontinuities move according to the Rankine-Hugoniot conditions

$$\dot{\xi}(\mathbf{u}_R - \mathbf{u}_L) = \mathbf{f}(\mathbf{u}_R) - \mathbf{f}(\mathbf{u}_L), \quad (6.5.2)$$

where  $\mathbf{u}_R$  and  $\mathbf{u}_L$  are solutions values immediately to the right and left, respectively, of the discontinuity. The Rankine-Hugoniot conditions do not necessarily determine a unique shock speed for vector systems (6.1.1). A way of introducing discontinuities directly into the solution is to construct a “weak form” of the problem using Galerkin’s method. Weak solutions automatically satisfy the partial differential system (6.1.1a) in regions where  $\mathbf{u}$  is smooth and the Rankine-Hugoniot conditions (6.5.2) across jumps. Galerkin’s method is at the core of the finite element method. While we won’t discuss this approach here, we will note that problems still exist. Weak solutions are not uniquely determined from

the initial data. The expansion wave of Example 6.4.1 illustrates this. One way of eliminating the non-uniqueness is to view solutions of (6.1.1) as limiting solutions of a viscous problem, *e.g.*,

$$\mathbf{u}_t + \mathbf{f}(\mathbf{u})_x = \epsilon \mathbf{u}_{xx}, \quad (6.5.3)$$

as the viscosity  $\epsilon \rightarrow 0$ . The proper form of the viscous term may not be known and different limiting solutions may result from different viscous models.

An alternate and preferable means of eliminating the non-uniqueness of solutions is through the use of an entropy function. Again, we will not develop this topic further but note that additional material may be found in, *e.g.*, Harten and Lax [10] and Jeffrey and Taniuti [13].

Having studied Riemann problems in Section 6.4, let us attempt to use their solutions to construct numerical schemes. Thus, let  $\mathbf{v}(x/t, \mathbf{u}_L, \mathbf{u}_R)$  be the solution of a Riemann problem for (6.1.1a) with the piecewise-constant initial data

$$\mathbf{u}(x, 0) = \begin{cases} \mathbf{u}_L, & \text{if } x < 0 \\ \mathbf{u}_R, & \text{if } x \geq 0 \end{cases}. \quad (6.5.4)$$

*Example 6.5.1.* The solution of the Riemann problem for the inviscid Burgers' equation (6.4.4a) is

$$v(x/t, u_L, u_R) = \begin{cases} u_L, & \text{if } x/t < (u_L + u_R)/2 \\ u_R, & \text{if } x/t \geq (u_L + u_R)/2 \end{cases}, \quad \text{if } u_L > u_R,$$

and

$$v(x/t, u_L, u_R) = \begin{cases} u_L, & \text{if } x/t < u_L \\ x/t, & \text{if } u_L \leq x/t < u_R \\ u_R, & \text{if } x/t \geq u_R \end{cases}, \quad \text{if } u_L \leq u_R.$$

Godunov [7] was the first to use solutions of Riemann problems to construct finite-difference procedures. Glimm [6] introduced a random component into the solution of Riemann problems like (6.1.1a, 6.5.4) as an analytical tool to study existence and uniqueness of solutions of initial value problems for (6.1.1). Chorin [2] showed that Glimm's study could be used to create a numerical technique called the "random choice method." There are several variants of Chorin's approach and Sod [19], Section 3.12, discusses some of them. The difference between the Glimm-Chorin random choice procedure and

Godunov's finite-difference scheme and its successors is that the former renders shocks as exact discontinuities. Shock speeds will generally only be correct on average, but the discontinuities will not have any dissipation associated with them. Godunov schemes are shock capturing schemes that have dissipation associated with them and, thus, spread discontinuities over a few computational cells.

Shocks do not have to be perfectly sharp for most applications. Some dissipation is generally acceptable. Additionally, exact solution of the associated Riemann problem may neither be possible nor practical. Let us, thus, consider the construction of finite-difference schemes that are based on the approximate solution of Riemann problems and we'll begin with a difference approximation of (6.1.1) in conservative form (6.4.7). Lax and Wendroff [15] and Sod [19], Section 4.2, show that finite difference schemes in conservation form converge as  $\Delta x$  and  $\Delta t$  approach zero to functions  $\mathbf{w}(x, t)$  that are weak solutions of (6.1.1a).

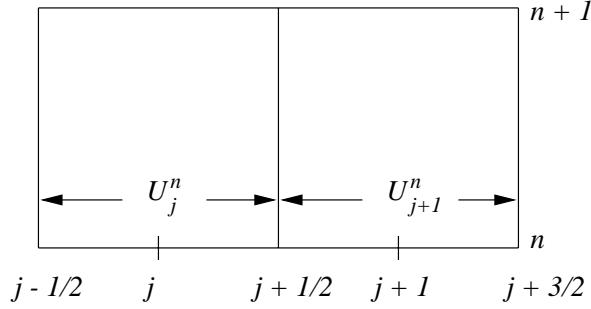


Figure 6.5.1: Grid structure for Godunov's method.

Let us introduce a uniform grid of spacing  $\Delta x \times \Delta t$  and regard the solution at time level  $n$  as being a piecewise constant function of  $x$  given by

$$\mathbf{U}^n(x) = \mathbf{U}_j^n, \quad (j - 1/2)\Delta x < x \leq (j + 1/2)\Delta x, \quad -\infty < j < \infty, \quad (6.5.5)$$

(Figure 6.5.1). The solution of a Riemann problem “breaking” at  $(x_{j+1/2}, t_n)$  with this piecewise constant data is  $\mathbf{v}((x - x_{j+1/2})/(t - t_n), \mathbf{U}_j^n, \mathbf{U}_{j+1}^n)$ ,  $x \in (x_j, x_{j+1}]$ ,  $t \in (t_n, t_{n+1}]$ , provided that solutions of Riemann problems on neighboring subintervals do not interact. This will be the case if time steps are selected to satisfy the Courant, Friedrichs, Lewy

condition

$$\max_j \max_{1 \leq i \leq m} |\lambda_i(\mathbf{A}_j^n)| \frac{\Delta t}{\Delta x} \leq \frac{1}{2}, \quad (6.5.6a)$$

where  $\lambda_i$ ,  $i = 1, 2, \dots, m$ , are the eigenvalues of  $\mathbf{A}_j^n$ . Assuming (6.5.6a) is satisfied, the solution of all of the local Riemann problems can be written in the compact form

$$\mathbf{V}(x, t) = \mathbf{v}((x - x_{j+1/2})/(t - t_n), \mathbf{U}_j^n, \mathbf{U}_{j+1}^n), \quad x \in (x_j, x_{j+1}], \quad t \in (t_n, t_{n+1}]. \quad (6.5.6b)$$

With Godunov's [7] method, the solution of an initial value problem at time  $t_{n+1}$  is obtained by projecting the solution of the Riemann problems onto piecewise constant functions, *i.e.*,

$$\mathbf{U}_j^{n+1} = \frac{1}{\Delta x} \int_{x_{j-1/2}}^{x_{j+1/2}} \mathbf{V}(x, t_{n+1}) dx. \quad (6.5.7)$$

Since  $\mathbf{V}(x, t)$  is an exact solution of the the partial differential system (6.1.1a),

$$\mathbf{0} = \int_{x_{j-1/2}}^{x_{j+1/2}} \int_{t_n}^{t_{n+1}} [\mathbf{V}_t + \mathbf{f}_x(\mathbf{V})] dx dt = \int_{x_{j-1/2}}^{x_{j+1/2}} \mathbf{V}(x, t)|_{t_n}^{t_{n+1}} dx + \int_{t_n}^{t_{n+1}} \mathbf{f}(\mathbf{V})|_{x_{j-1/2}}^{x_{j+1/2}} dt. \quad (6.5.8)$$

However,  $\mathbf{V}(x_{j-1/2}, t) = \mathbf{v}(0, \mathbf{U}_{j-1}^n, \mathbf{U}_j^n)$  and  $\mathbf{V}(x_{j+1/2}, t) = \mathbf{v}(0, \mathbf{U}_j^n, \mathbf{U}_{j+1}^n)$  are independent of  $t$ ; thus,

$$\int_{t_n}^{t_{n+1}} \mathbf{f}(\mathbf{V})|_{x_{j-1/2}}^{x_{j+1/2}} dt = [\mathbf{f}(\mathbf{v}(0, \mathbf{U}_j^n, \mathbf{U}_{j+1}^n)) - \mathbf{f}(\mathbf{v}(0, \mathbf{U}_{j-1}^n, \mathbf{U}_j^n))] \Delta t.$$

Using this and (6.5.7) in (6.5.8) yields

$$\mathbf{U}_j^{n+1} = \mathbf{U}_j^n - \bar{\alpha} [\mathbf{f}(\mathbf{v}(0, \mathbf{U}_j^n, \mathbf{U}_{j+1}^n)) - \mathbf{f}(\mathbf{v}(0, \mathbf{U}_{j-1}^n, \mathbf{U}_j^n))] \quad (6.5.9a)$$

where  $\bar{\alpha} = \Delta t / \Delta x$ . This is Godunov's [7] method. It clearly has the conservation form (6.4.7) with the numerical flux

$$\mathbf{F}_G(\mathbf{U}_j^n, \mathbf{U}_{j+1}^n) = \mathbf{f}(\mathbf{v}(0, \mathbf{U}_j^n, \mathbf{U}_{j+1}^n)). \quad (6.5.9b)$$

The subscript  $G$  is used merely to identify the numerical flux associated with Godunov's method.

*Example 6.5.2.* Using the solution of the Riemann problem for the inviscid Burgers' equation of Example 6.5.1, we have

$$v(0, u_L, u_R) = \begin{cases} u_L, & \text{if } u_L, u_R > 0 \\ u_R, & \text{if } u_L, u_R < 0 \\ 0, & \text{if } u_L < 0, u_R > 0 \\ u_L, & \text{if } u_L > 0, u_R < 0, (u_L + u_R)/2 > 0 \\ u_R, & \text{if } u_L > 0, u_R < 0, (u_L + u_R)/2 < 0 \end{cases}.$$

The characteristics for these solutions are illustrated in Figure 6.5.2. The numerical flux is

$$F_G(u_L, u_R) = \begin{cases} u_L^2/2, & \text{if } u_L, u_R > 0 \\ u_R^2/2, & \text{if } u_L, u_R < 0 \\ 0, & \text{if } u_L < 0, u_R > 0 \\ u_L^2/2, & \text{if } u_L > 0, u_R < 0, (u_L + u_R)/2 > 0 \\ u_R^2/2, & \text{if } u_L > 0, u_R < 0, (u_L + u_R)/2 < 0 \end{cases}.$$

This flux can be written more compactly by letting

$$u^+ = \max(u, 0), \quad u^- = \min(u, 0). \quad (6.5.10a)$$

Then

$$F_G(u_L, u_R) = \max[(u_L^+)^2/2, (u_R^-)^2/2]. \quad (6.5.10b)$$

Several schemes deriving from Godunov's [7] early work are now available. We'll describe techniques developed by Roe [18], van Leer [22], and Cockburn and Shu [3].

### 6.5.1 Roe's Method

Roe [18] developed a scheme having the conservation law form (6.4.7) with the numerical flux

$$\mathbf{F}_R(\mathbf{U}_j^n, \mathbf{U}_{j+1}^n) = \mathbf{f}(\mathbf{w}(0, \mathbf{U}_j^n, \mathbf{U}_{j+1}^n)), \quad (6.5.11a)$$

where  $\mathbf{w}(x/t, \mathbf{u}_L, \mathbf{u}_R)$  is the exact solution of the linearized Riemann problem

$$\mathbf{w}_t + \hat{\mathbf{A}}(\mathbf{u}_L, \mathbf{u}_R)\mathbf{w}_x = 0. \quad (6.5.11b)$$

The  $m \times m$  matrix  $\hat{\mathbf{A}}$  is an approximation of the Jacobian  $\mathbf{f}_u$  satisfying

$$\mathbf{f}(\mathbf{u}_R) - \mathbf{f}(\mathbf{u}_L) = \hat{\mathbf{A}}(\mathbf{u}_L, \mathbf{u}_R)(\mathbf{u}_R - \mathbf{u}_L). \quad (6.5.11c)$$

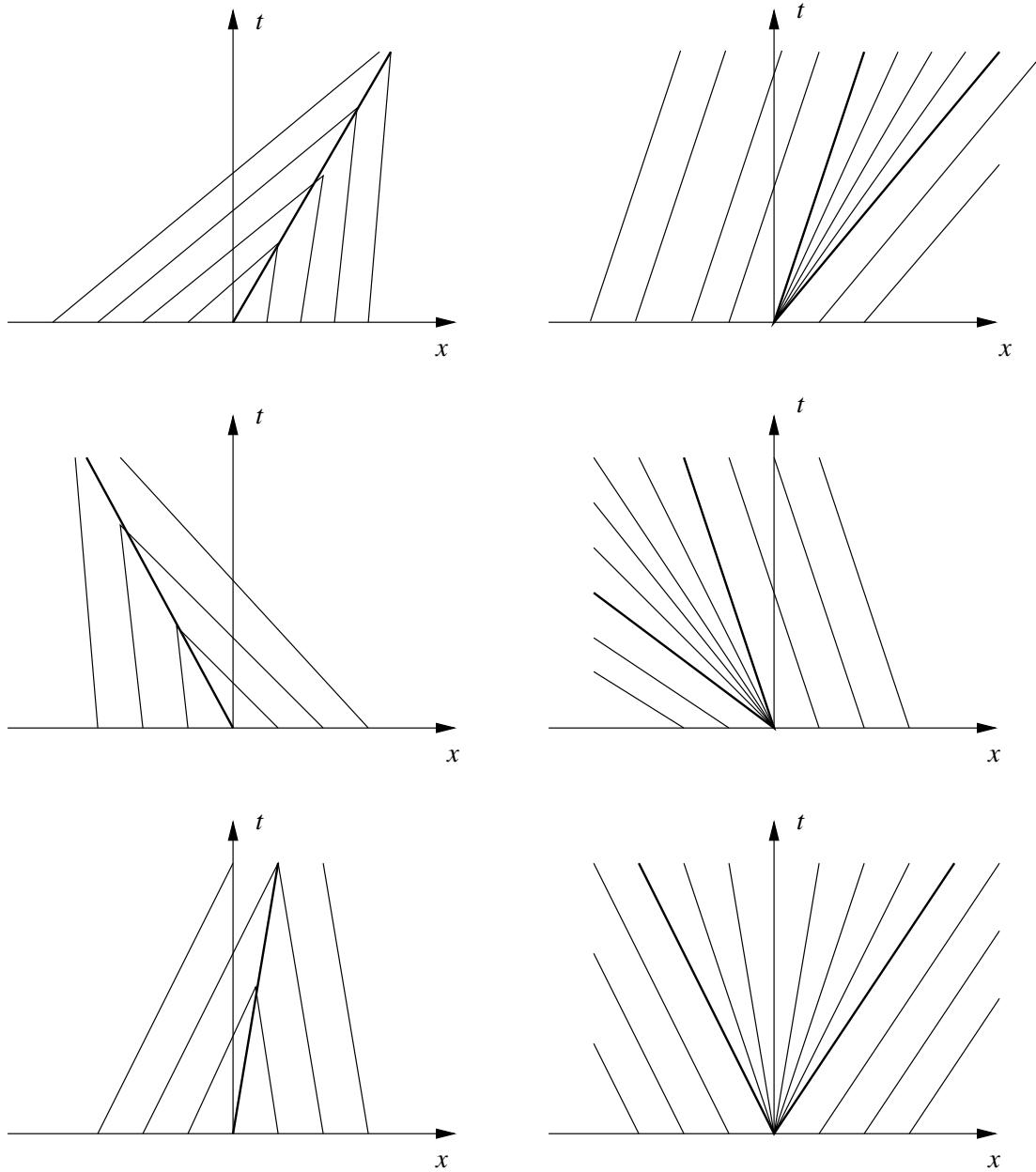


Figure 6.5.2: Characteristics of Riemann problems for Burgers' equation when  $u_L, u_R > 0$  (top);  $u_L, u_R < 0$  (center);  $u_L > 0$ ,  $u_R < 0$ ,  $(u_L + u_R)/2 > 0$  (bottom left); and  $u_L < 0, u_R > 0$  (bottom right).

*Example 6.5.3.* The flux for the inviscid Burgers' equation is  $f(u) = u^2/2$  and (6.5.11c) becomes

$$(u_R^2 - u_L^2)/2 = \hat{A}(u_L, u_R)(u_R - u_L).$$

Thus, for this scalar problem

$$\hat{A}(u_L, u_R) = (u_R + u_L)/2 = \dot{\xi},$$

where  $\dot{\xi}$  is the shock speed. Thus, Roe's numerical flux for the inviscid Burgers' equation is

$$F_R(u_L, u_R) = \begin{cases} u_L^2/2, & \text{if } \dot{\xi} \geq 0 \\ u_R^2/2, & \text{if } \dot{\xi} < 0 \end{cases}.$$

The approximate Riemann solver is exact when  $u_L > u_R$  and a shock is produced; however, shocks are also produced when  $u_L < u_R$  and an expansion wave should occur. Thus, Roe's scheme violates the entropy condition. Harten and Hyman [8] introduced a modification that eliminates the entropy violation. We'll depart from their development and present an approach due to Harten, Lax, and van Leer [11]. Both approaches appear in Sod [19], Section 4.4.

Let  $\nu_{min}$ ,  $\nu_{max}$ , be the minimum and maximum signal speeds associated with a Riemann problem breaking at  $(x_{j+1/2}, t_n)$  (Figure 6.5.3). The solution for

$$\frac{x - x_{j+1/2}}{t - t_n} > \nu_{max}$$

is  $\mathbf{u}_R$ . Similarly, the solution for

$$\frac{x - x_{j+1/2}}{t - t_n} < \nu_{min}$$

is  $\mathbf{u}_L$  (Figure 6.5.3). Several solution states may exist between these extremes. In order to simplify the resulting difference method, Harten, Lax, and van Leer [11] considered approximations with one and two states between the maximum and minimum signals. We'll illustrate their method when one state is present. In this case, their approximate Riemann solution is

$$\mathbf{w}(x/t, \mathbf{u}_L, \mathbf{u}_R) = \begin{cases} \mathbf{u}_L, & \text{if } x/t < \nu_{min} \\ \mathbf{u}_*, & \text{if } \nu_{min} \leq x/t < \nu_{max} \\ \mathbf{u}_R, & \text{if } \nu_{max} \leq x/t \end{cases}. \quad (6.5.12)$$

In order to determine the intermediate state  $\mathbf{u}_*$ , they integrate the conservation law (6.1.1a) over the cell  $(x_j, x_{j+1}) \times (t_n, t_{n+1})$  (Figure 6.5.3) to obtain

$$0 = \int_{x_j}^{x_{j+1}} \int_{t_n}^{t_{n+1}} [\mathbf{u}_t + \mathbf{f}_x(\mathbf{u})] dx dt = \int_{x_j}^{x_{j+1}} \mathbf{u}(x, t)|_{t_n}^{t_{n+1}} dx + \int_{t_n}^{t_{n+1}} \mathbf{f}(\mathbf{u})|_{x_j}^{x_{j+1}} dt.$$

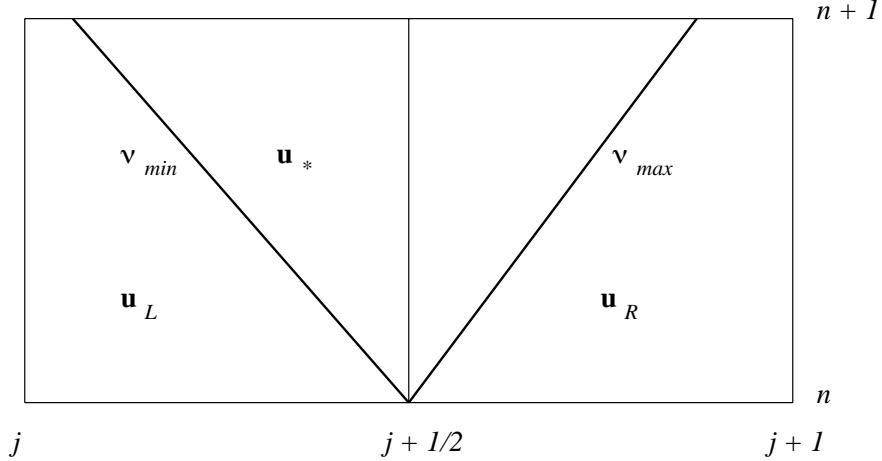


Figure 6.5.3: Computational cell for the Riemann problem for Roe's method. The minimum  $\nu_{min}$  and maximum  $\nu_{max}$  signal speeds are assumed to be separated by one constant state  $\mathbf{u}_*$ .

Since the solutions at  $x = x_j$  and  $x_{j+1}$  are  $\mathbf{u}_L$  and  $\mathbf{u}_R$ , respectively, we have

$$\int_{x_j}^{x_{j+1}} \mathbf{u}(x, t)|_{t_n}^{t_{n+1}} dx + \Delta t[\mathbf{f}(\mathbf{u}_R) - \mathbf{f}(\mathbf{u}_L)] = 0$$

From (6.5.12), we have

$$\mathbf{u}(x, t_n) = \begin{cases} \mathbf{u}_L, & \text{if } x_j \leq x < x_{j+1/2} \\ \mathbf{u}_R, & \text{if } x_{j+1/2} \leq x < x_{j+1} \end{cases}$$

and

$$\mathbf{u}(x, t_{n+1}) = \begin{cases} \mathbf{u}_L, & \text{if } x_{j-1} \leq x < x_{j+1/2} + \nu_{min}\Delta t \\ \mathbf{u}_*, & \text{if } x_{j+1/2} + \nu_{min}\Delta t \leq x < x_{j+1/2} + \nu_{max}\Delta t \\ \mathbf{u}_R, & \text{if } x_{j+1/2} + \nu_{max}\Delta t \leq x < x_{j+1} \end{cases}.$$

(The speed  $\nu_{min}$  has been assumed positive and not as shown in Figure 6.5.3.) Thus, the remaining integral may be evaluated to obtain

$$\begin{aligned} & \left( \frac{\Delta x}{2} + \nu_{min}\Delta t \right) \mathbf{u}_L + (\nu_{max} - \nu_{min})\Delta t \mathbf{u}_* + \left( \frac{\Delta x}{2} - \nu_{max}\Delta t \right) \mathbf{u}_R - \frac{\Delta x}{2}(\mathbf{u}_L + \mathbf{u}_R) \\ & + \Delta t[\mathbf{f}(\mathbf{u}_R) - \mathbf{f}(\mathbf{u}_L)] = 0 \end{aligned}$$

or

$$\mathbf{u}_* = \frac{\nu_{max}\mathbf{u}_R - \nu_{min}\mathbf{u}_L}{\nu_{max} - \nu_{min}} - \frac{\mathbf{f}(\mathbf{u}_R) - \mathbf{f}(\mathbf{u}_L)}{\nu_{max} - \nu_{min}}. \quad (6.5.13)$$

Harten, Lax, and van Leer [11] show that the modified Roe's scheme (6.4.7, 6.5.11a, 6.5.12, 6.5.13) no longer violates the entropy condition.

Harten and Hyman [8] described a procedure for calculating the signal speeds  $\nu_{min}$  and  $\nu_{max}$ , which we describe for scalar problems. They begin by selecting  $\tilde{\nu}_{min}$  and  $\tilde{\nu}_{max}$ , respectively, as approximate speeds of the left and right ends of an expansion fan centered at  $x = x_{j+1/2}$ . These may be taken as

$$\tilde{\nu}_{min} = \min_{\theta \in [0,1]} \hat{A}(u_L, (1-\theta)u_L + \theta u_R), \quad (6.5.14a)$$

$$\tilde{\nu}_{max} = \max_{\theta \in [0,1]} \hat{A}((1-\theta)u_L + \theta u_R, u_R). \quad (6.5.14b)$$

Their search over values of  $u \in [u_L, u_R]$  seeks to identify minimum and maximum signal speeds that are not at the extreme values of  $u_L$  and  $u_R$ .

They then select their signal speeds as

$$\nu_{min} = \hat{A}(u_L, u_R) - (\hat{A}(u_L, u_R) - \tilde{\nu}_{min})^+ \quad (6.5.14c)$$

and

$$\nu_{max} = \hat{A}(u_L, u_R) + (\tilde{\nu}_{max} - \hat{A}(u_L, u_R))^+. \quad (6.5.14d)$$

Using (6.5.11c) in (6.5.13), they compute  $u_*$  as

$$u_* = \frac{\hat{A}(u_L, u_R) - \nu_{min}}{\nu_{max} - \nu_{min}} u_L + \frac{\nu_{max} - \hat{A}(u_L, u_R)}{\nu_{max} - \nu_{min}} u_R \quad (6.5.14e)$$

The approximate solution of the Riemann problem is obtained from (6.5.12) translated to  $(x_{j+1/2}, t_n)$ . Equations (6.5.14) may also be applied to vector systems that have been put into canonical form by the transformation (6.1.3) ([19], Section 4.4).

*Example 6.5.4.* For the inviscid Burgers' equation, we have  $\hat{A}(u_L, u_R) = (u_L + u_R)/2 = \dot{\xi}$  (Example 6.5.3). Using (6.5.14a)

$$\tilde{\nu}_{min} = \min_{\theta \in [0,1]} \frac{u_L + (1-\theta)u_L + \theta u_R}{2} = \min_{\theta \in [0,1]} \frac{(2-\theta)u_L + \theta u_R}{2}.$$

Since the argument of the minimization is a linear function of  $\theta$  for Burgers' equation, extreme values occur at  $\theta = 0, 1$ .

$$\tilde{\nu}_{min} = \min(u_L, \frac{u_L + u_R}{2}) = \min(u_L, \dot{\xi}).$$

Similarly, using (6.5.14b)

$$\tilde{\nu}_{max} = \max_{\theta \in [0,1]} \frac{(1-\theta)u_L + (1+\theta)u_R}{2} = \max(\dot{\xi}, u_R).$$

In an expansion region,  $u_L < \dot{\xi} < u_R$  and  $\tilde{\nu}_{min} = u_L$  and  $\tilde{\nu}_{max} = u_R$ . Using these in (6.5.14c, 6.5.14d) yields  $\nu_{min} = u_L$  and  $\nu_{max} = u_R$ , which are the exact speeds of the endpoints of the expansion fan. Using these in (6.5.14e) gives  $u_* = \dot{\xi}$  and, from (6.5.12),

$$w(x/t, u_L, u_R) = \begin{cases} u_L, & \text{if } x/t < u_L \\ \dot{\xi}, & \text{if } u_L < x/t \leq u_R \\ u_R, & \text{if } u_R \leq x/t \end{cases}.$$

For a shock,  $u_L > \dot{\xi} > u_R$ , so  $\tilde{\nu}_{min} = \dot{\xi}$  and  $\tilde{\nu}_{max} = \dot{\xi}$ . Substitution into (6.5.14c, 6.5.14d) yields  $\nu_{min} = \nu_{max} = \dot{\xi}$ . The intermediate solution disappears from (6.5.12) and the solution of the Riemann problem is

$$w(x/t, u_L, u_R) = \begin{cases} u_L, & \text{if } x/t < \dot{\xi} \\ u_R, & \text{if } \dot{\xi} \leq x/t \end{cases}.$$

### 6.5.2 Van Leer's Method

Van Leer's [22] technique is simplest to present for the linear scalar problem

$$u_t + au_x = 0. \quad (6.5.15)$$

Suppose that we have an approximate solution  $W^n(x)$  at time  $t_n$ , with  $W^0(x)$  corresponding to the prescribed initial data. Using the mesh structure of Figure 6.5.1, construct a piecewise-constant approximation  $U^n(x)$  of  $W^n(x)$  as

$$U^n(x) = U_j^n = \frac{1}{\Delta x} \int_{x_{j-1/2}}^{x_{j+1/2}} W^n(x) dx, \quad x \in (x_{j-1/2}, x_{j+1/2}). \quad (6.5.16a)$$

The solution  $W^{n+1}(x)$  at  $t_{n+1}$  is obtained as the exact solution of (6.5.15); thus,

$$W^{n+1}(x) = U^n(x - a\Delta t). \quad (6.5.16b)$$

The process is repeated for the next time step. The steps in the procedure are illustrated in Figure 6.5.4. If the Courant, Friedrichs, Lewy condition  $|a|\Delta t/\Delta x < 1$  is satisfied, then the translation of  $U^n(x)$  by the amount  $a\Delta t$  does not exceed  $\Delta x$ .

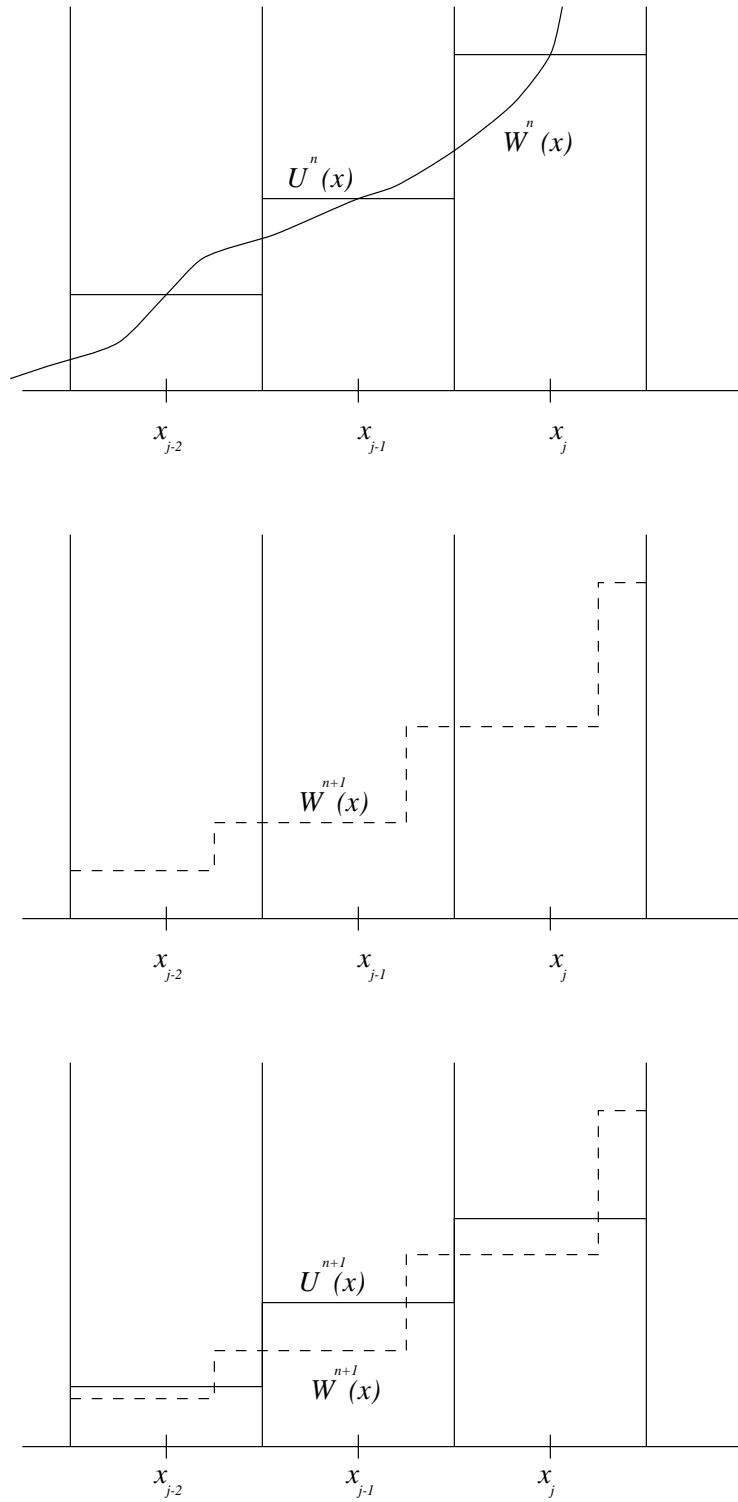


Figure 6.5.4: Projection of  $W^n(x)$  onto a piecewise constant function  $U^n(x)$  (top). Convective transport of the piecewise constant functions  $U^n(x)$  (center) to obtain  $W^{n+1}(x)$ . Projection of  $W^{n+1}(x)$  onto the piecewise constant function  $U^{n+1}(x)$  (bottom).

If  $a > 0$  then

$$W^{n+1}(x) = \begin{cases} U_{j-1}^n, & \text{if } x_{j-1/2} < x \leq x_{j-1/2} + a\Delta t \\ U_j^n, & \text{if } x_{j-1/2} + a\Delta t < x \leq x_{j+1/2} \end{cases}.$$

Substituting this into (6.5.16a) and performing the integration yields

$$U_j^{n+1} = a\bar{\alpha}U_{j-1}^n + (1 - a\bar{\alpha})U_j^n = U_j^n - a\bar{\alpha}(U_j^n - U_{j-1}^n), \quad (6.5.17)$$

which is recognized as the first-order upwind difference scheme.

Van Leer [22] extended this approach to higher orders of accuracy. For example, he obtained a second-order method by projecting  $W^n(x)$  onto a space of piecewise linear polynomials. When this is done, one must figure a way of determining the slope of the solution on each subinterval. Van Leer [22] describes several possibilities and some of these also appear in Sod [19], Section 4.3. One possibility uses centered difference approximations from solution values on neighboring subintervals. The description of a second approach using moments follows.

### 6.5.3 Higher-Order Methods

Higher order methods can achieve excellent accuracy and performance in regions where the solution is smooth. We'll describe a method of constructing high-order methods that uses moments of the solution; however, instead of following the traditional approach [22], we'll describe an alternative treatment, called the *discontinuous Galerkin* method, as developed by Cockburn and Shu [3].

We'll use a method of lines formulation and let  $\mathbf{W}(x, t)$  be a piecewise linear approximation of  $\mathbf{u}(x, t)$  whose restriction to  $(x_{j-1/2}, x_{j+1/2})$  is (Figure 6.5.5)

$$\mathbf{W}(x, t) = \mathbf{U}_j(t) + D\mathbf{U}_j(t)(x - x_j) \quad (6.5.18)$$

where  $\mathbf{U}_j(t)$  and  $D\mathbf{U}_j(t)$  are approximations of  $u(x_j, t)$  and  $u_x(x_j, t)$ , respectively. Let us use the integral form of the conservation law (6.5.1) with (6.5.18) to obtain

$$\frac{d}{dt} \int_{x_{j-1/2}}^{x_{j+1/2}} \mathbf{W}(x, t) dx = -\mathbf{f}(\mathbf{W})|_{x_{j-1/2}}^{x_{j+1/2}}.$$

Using the representation (6.5.18) and replacing  $\mathbf{f}$  by a numerical flux yields

$$\Delta x \dot{\mathbf{U}}_j(t) = -\mathbf{F}_{j+1/2} + \mathbf{F}_{j-1/2} \quad (6.5.19a)$$

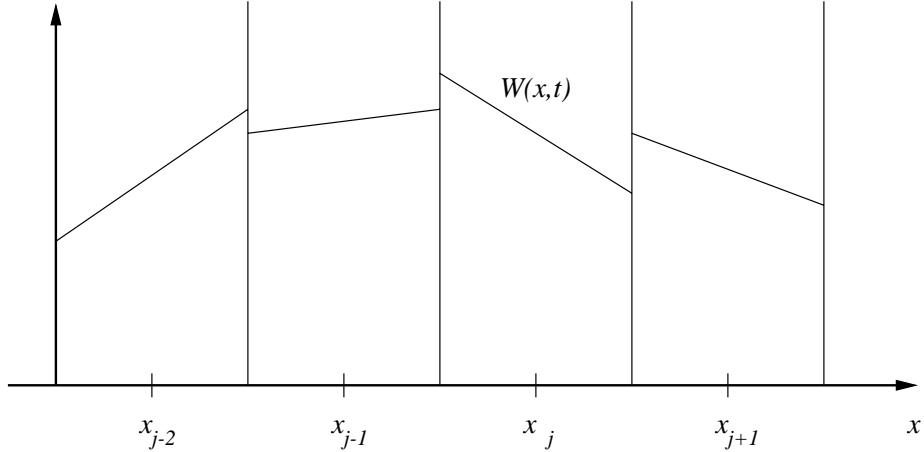


Figure 6.5.5: Piecewise linear function  $W^n(x)$  used for the method of Cockburn and Shu [3].

where

$$\mathbf{F}_{j+1/2} = \mathbf{F}_{j+1/2}(\mathbf{U}_j(t) + \frac{\Delta x}{2} D\mathbf{U}_j(t), \mathbf{U}_{j+1}(t) - \frac{\Delta x}{2} D\mathbf{U}_{j+1}(t)). \quad (6.5.19b)$$

The gradient  $D\mathbf{U}_j$  is determined by taking a moment of (6.1.1a); thus, we multiply (6.1.1a) by  $x - x_j$ , integrate over the subinterval  $(x_{j-1/2}, x_{j+1/2})$ , and replace the exact solution by the approximation (6.5.18) to obtain

$$\frac{d}{dt} \int_{x_{j-1/2}}^{x_{j+1/2}} (x - x_j) \mathbf{W}(x, t) dx + \int_{x_{j-1/2}}^{x_{j+1/2}} (x - x_j) \mathbf{f}_x(\mathbf{W}) dx = 0$$

Integrating the flux term by parts

$$\frac{d}{dt} \int_{x_{j-1/2}}^{x_{j+1/2}} (x - x_j) \mathbf{W}(x, t) dx + (x - x_j) \mathbf{f}(\mathbf{W})|_{x_{j-1/2}}^{x_{j+1/2}} = \int_{x_{j-1/2}}^{x_{j+1/2}} \mathbf{f}(\mathbf{W}) dx.$$

Using (6.5.18)

$$\frac{\Delta x^3}{12} D\dot{\mathbf{U}}_j + \frac{\Delta x}{2} [\mathbf{F}_{j+1/2} + \mathbf{F}_{j-1/2}] = \int_{x_{j-1/2}}^{x_{j+1/2}} \mathbf{f}(\mathbf{W}) dx. \quad (6.5.19c)$$

Equations (6.5.19b) and (6.5.19c) are two vector ( $2m$  scalar) equations for the unknowns  $\mathbf{U}_j(t)$  and  $D\mathbf{U}_j(t)$ . To complete the specification of the scheme (6.5.18, 6.5.19), we must (i) choose a numerical flux, (ii) evaluate the integral on the right of (6.5.19c), (iii) choose a time integration strategy, and (iv) define initial conditions.

The flux can be evaluated by a Riemann or other numerical flux. Cockburn and Shu [3] describe several choices. Perhaps the simplest is the Lax-Friedrichs flux which, in this context, is

$$\mathbf{F}_{j+1/2}(\mathbf{W}_L, \mathbf{W}_R) = \frac{1}{2}[\mathbf{f}(\mathbf{W}_L) + \mathbf{f}(\mathbf{W}_R) - \max(|\lambda_{min}|, |\lambda_{max}|)(\mathbf{W}_R - \mathbf{W}_L)] \quad (6.5.20a)$$

where, using (6.5.18),

$$\mathbf{W}_L = \mathbf{U}_j(t) + \frac{\Delta x}{2}D\mathbf{U}_j(t), \quad \mathbf{W}_R = \mathbf{U}_{j+1}(t) - \frac{\Delta x}{2}D\mathbf{U}_{j+1}(t) \quad (6.5.20b)$$

are the solutions to the immediate left and right, respectively, of  $x_{j+1/2}$  and  $\lambda_{min}$  and  $\lambda_{max}$  are, respectively, the minimum and maximum eigenvalues of  $\mathbf{f}_u(\mathbf{u})$ ,  $\mathbf{W}_L \leq \mathbf{u} \leq \mathbf{W}_R$ .

The integral on the right of (6.5.19c) is usually evaluated numerically. With the midpoint rule, we have

$$\int_{x_{j-1/2}}^{x_{j+1/2}} \mathbf{f}(\mathbf{W})dx = \Delta x \mathbf{f}(\mathbf{U}_j) + O(\Delta x^3). \quad (6.5.20c)$$

Cockburn and Shu [3] recommend a total variation diminishing (TVD) Runge-Kutta scheme. Biswas *et al.* [1] found that classical Runge-Kutta formulas gave similar results. With forward Euler integration and (6.5.20c), the scheme (6.5.19) becomes

$$\mathbf{U}_j^{n+1} = \mathbf{U}_j^n - \frac{\Delta t}{\Delta x}[\mathbf{F}_{j+1/2}^n - \mathbf{F}_{j-1/2}^n] \quad (6.5.21a)$$

$$D\mathbf{U}_j^{n+1} = D\mathbf{U}_j^n + \frac{6\Delta t}{\Delta x^2}[\mathbf{F}_{j+1/2}^n + \mathbf{F}_{j-1/2}^n - 2\mathbf{f}(\mathbf{U}_j^n)]. \quad (6.5.21b)$$

While the forward Euler method may be used for time integration, a second-order method would be more compatible with the higher spatial accuracy.

Finally, initial conditions may also be obtained by taking moments

$$\int_{x_{j-1/2}}^{x_{j+1/2}} [\mathbf{W}(x, 0) - \boldsymbol{\phi}(x)]dx = 0, \quad \int_{x_{j-1/2}}^{x_{j+1/2}} (x - x_j)[\mathbf{W}(x, 0) - \boldsymbol{\phi}(x)]dx = 0. \quad (6.5.22)$$

The scheme may be recognized by some as a Galerkin method, which is the principal technique used with finite element methods. It uses a discontinuous solution representation; hence, the name discontinuous Galerkin method. It has several advantages:

- it readily extends to higher orders of accuracy by using higher-degree polynomials and taking higher-order moments of the conservation law (6.1.1a) [3, 1];
- it may be used in two and three dimensions in much the same way as in one dimension [1, 4];
- it extends to unstructured meshes of triangular or tetrahedral elements [4]; and
- it has a compact structure that is suitable for parallel computation.

Unfortunately, since it is second order, it will have oscillations near discontinuities. These can be reduced by *limiting* the computed solution. With limiting, the gradient  $D\mathbf{U}_j(t)$  variable is reevaluated to eliminate overshoots in the solution. In particular  $D\mathbf{U}_j$  is modified so that:

1. the linear function (6.5.18) does not take on values outside of the adjacent grid averages (Figure 6.5.6, upper left);
2. local extrema are set to zero (Figure 6.5.6, upper right); and
3. the gradient is replaced by zero if its sign is not consistent with its neighbors (Figure 6.5.6, lower center).

A formula for accomplishing these goals can be summarized concisely using the minimum modulus function as

$$D\mathbf{U}_{j,mod} = \text{minmod}(D\mathbf{U}_j, \nabla\mathbf{U}_j, \Delta\mathbf{U}_j) \quad (6.5.23a)$$

where

$$\text{minmod}(a, b, c) = \begin{cases} \text{sgn}(a) \min(|a|, |b|, |c|), & \text{if } \text{sgn}(a) = \text{sgn}(b) = \text{sgn}(c) \\ 0, & \text{otherwise} \end{cases} \quad (6.5.23b)$$

and  $\nabla$  and  $\Delta$  are the backward and forward difference operators, respectively.

*Example 6.5.5.* Biswas *et al.* [1] solve the inviscid Burgers' equation (6.4.4a) with the initial data

$$u(x, 0) = \frac{1 + \sin x}{2}.$$

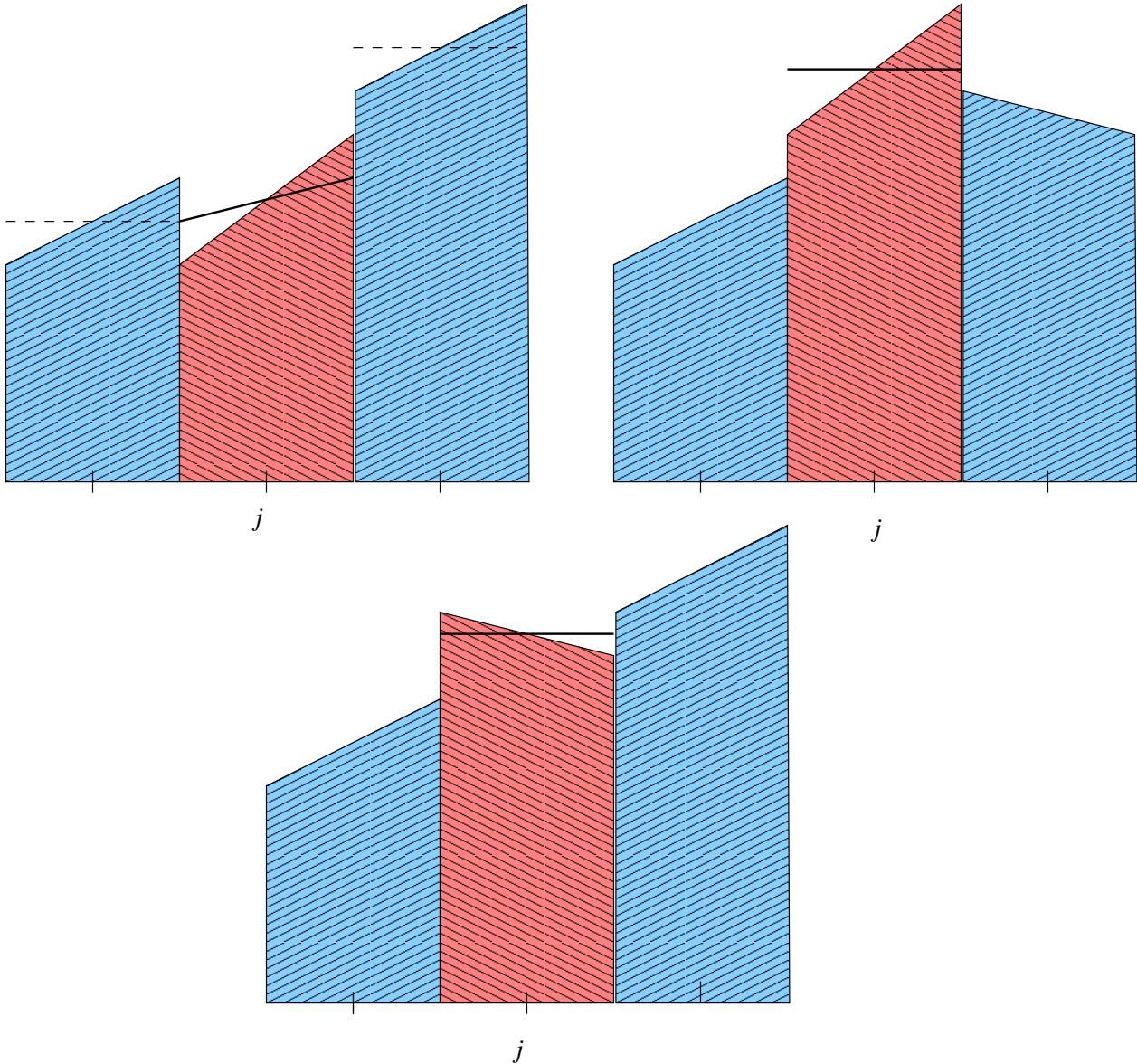


Figure 6.5.6: Solution limiting: reduce slopes to be within neighboring averages (upper left); set local extrema to zero (upper right); and set slopes to zero if they disagree with neighboring trends.

They use an upwind numerical flux

$$F_{j+1/2}(W_L, W_R) = \begin{cases} f(W_L), & \text{if } f'(U_j) \geq 0 \\ f(W_R), & \text{if } f'(U_j) < 0 \end{cases}$$

and solve problems with  $\Delta x = 1/32$  using first-, second- and third-order methods. The first-order method is obtained from (6.5.21a) by setting  $DU_j = 0$ ,  $j = 0, 1, \dots, J$ . This is just upwind differencing. The second-order method is (6.5.18, 6.5.21–6.5.23) with the upwind numerical flux. The third-order method is derived in the same manner as the

second-order method using a piecewise quadratic function to represent the solution. Time integration was done using classical Runge-Kutta methods of orders 1-3, respectively, for the first-, second-, and third-order methods. The solutions are shown in Figure 6.5.7. The piecewise polynomial functions used to represent the solution are plotted at eleven points on each subinterval.

The first-order solution (upper left of Figure 6.5.7) is characteristically diffusive. The second-order solution (upper right of Figure 6.5.7) has greatly reduced the diffusion while not introducing any spurious oscillations. The minimum modulus limiter (6.5.23) has a tendency to flatten solutions near extrema. This can be seen in the third-order solution shown at the lower left of Figure 6.5.7. There is a loss of (local) monotonicity near the shocks. (Average solution values are monotone and this is all that the limiter (6.5.23) was designed to produce.) Biswas *et al.* [1] developed an adaptive limiter that reduces “flattening” and does a better job of preserving local monotonicity near discontinuities. The solution with this limiter is shown in the lower portion of Figure 6.5.7.

Evaluating numerical fluxes and using limiting for vector systems is more complicated than indicated by the previous scalar example. Cockburn and Shu [3] reported problems when applying limiting component-wise. At the price of additional computation, they applied limiting to the characteristic fields obtained by diagonalizing the Jacobian  $\mathbf{f}_u$ . Biswas *et al.* [1] proceeded in a similar manner. “Flux-vector splitting” may provide a compromise between the two extremes. As an example, consider the solution and flux vectors for the one-dimensional Euler equations of compressible flow

$$\mathbf{u} = \begin{bmatrix} \rho \\ \rho u \\ e \end{bmatrix}, \quad \mathbf{f}(\mathbf{u}) = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ u(e + p) \end{bmatrix} \quad (6.5.24a)$$

where  $\rho$ ,  $u$ ,  $e$ , and

$$p = (\gamma - 1)(e - \rho u^2)/2 \quad (6.5.24b)$$

are, respectively, the fluid’s density, velocity, internal energy per unit volume, and pressure.

For this and related differential systems, the flux vector is a homogeneous function

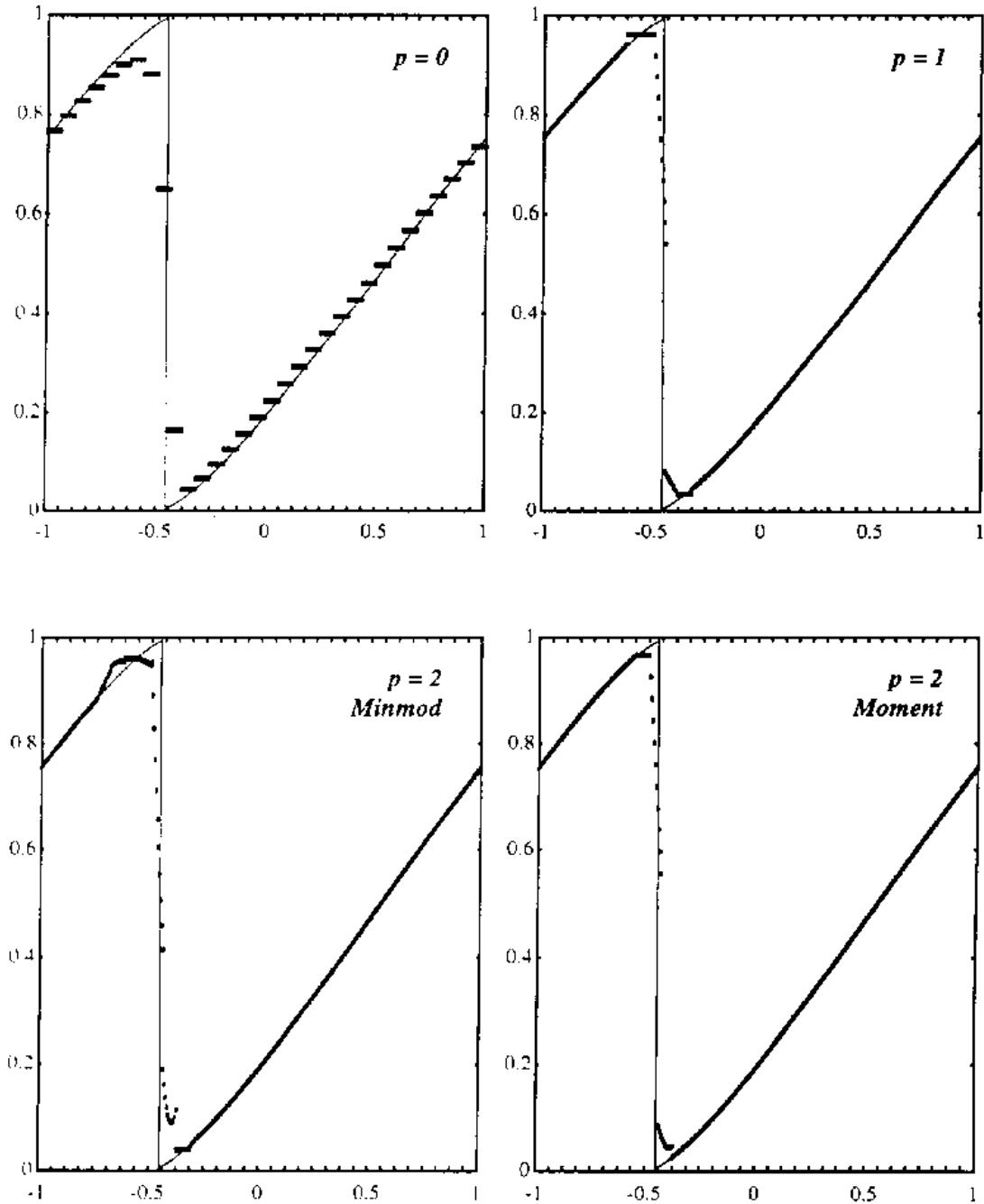


Figure 6.5.7: Exact (line) and numerical solutions of Example 5 for methods with piecewise polynomial approximations of degrees  $p = 0, 1$ , and  $2$ . Solutions with a minmod limiter and an adaptive limiter of Biswas et al. [1] are shown for  $p = 2$ .

that may be expressed as

$$\mathbf{f}(\mathbf{u}) = \mathbf{A}\mathbf{u} = \mathbf{f}_u(\mathbf{u})\mathbf{u}. \quad (6.5.25a)$$

Since the system is hyperbolic, the Jacobian  $\mathbf{A}$  may be diagonalized as described in Section 6.1 to yield

$$\mathbf{f}(\mathbf{u}) = \mathbf{P}^{-1} \mathbf{\Lambda} \mathbf{P} \mathbf{u} \quad (6.5.25b)$$

where the diagonal matrix  $\mathbf{\Lambda}$  contains the eigenvalues of  $\mathbf{A}$

$$\mathbf{\Lambda} = \begin{bmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_m \end{bmatrix} = \begin{bmatrix} u - c & & & \\ & u & & \\ & & u + c & \\ & & & \end{bmatrix}. \quad (6.5.25c)$$

The variable  $c = \sqrt{\partial p / \partial \rho}$  is the speed of sound in the fluid. The matrix  $\mathbf{\Lambda}$  can be decomposed into components

$$\mathbf{\Lambda} = \mathbf{\Lambda}^+ + \mathbf{\Lambda}^- \quad (6.5.26a)$$

where  $\mathbf{\Lambda}^+$  and  $\mathbf{\Lambda}^-$  are, respectively, composed of the non-negative and non-positive components of  $\mathbf{\Lambda}$

$$\lambda_i^\pm = \frac{\lambda_i \pm |\lambda_i|}{2}, \quad i = 1, 2, \dots, m. \quad (6.5.26b)$$

Writing the flux vector in similar fashion using (6.5.25)

$$\mathbf{f}(\mathbf{u}) = \mathbf{P}^{-1}(\mathbf{\Lambda}^+ + \mathbf{\Lambda}^-)\mathbf{P}\mathbf{u} = \mathbf{f}(\mathbf{u})^+ + \mathbf{f}(\mathbf{u})^-. \quad (6.5.27)$$

Split fluxes for the Euler equations were presented by Steger and Warming [20]. Van Leer [23] found an improvement that provided improvements near sonic and stagnation points of the flow. The split fluxes are evaluated by upwind techniques. Letting  $\mathbf{W}_L$  and  $\mathbf{W}_R$  be solution vectors on the left and right of an interface (*e.g.*, at  $x_{j+1/2}$ ), then  $\mathbf{f}^+$  is evaluated using  $\mathbf{W}_L$  and  $\mathbf{f}^-$  is evaluated using  $\mathbf{W}_R$ .

## 6.6 Nonlinear Instability and Boundary Conditions

Linear stability analyses (*e.g.*, using von Neumann's method) are necessary for stability, but, possibly, not sufficient because of nonlinearity or boundary conditions. In order to illustrate this effect, consider solving the inviscid Burgers' equation

$$u_t + uu_x = 0$$

by the leap frog scheme

$$U_j^{n+1} = U_j^{n-1} - \bar{\alpha} U_j^n (U_{j+1}^n - U_{j-1}^n), \quad (6.6.1)$$

where  $\bar{\alpha} = \Delta t / \Delta x$ . In order to analyze the sensitivity of (6.6.1) to small perturbations, let

$$U_j^n = \bar{U}_j^n + \delta U_j^n \quad (6.6.2)$$

where  $\bar{U}_j^n$  is a solution of (6.6.1) and  $\delta U_j^n$  is a perturbation. Substituting (6.6.2) into (6.6.1)

$$\bar{U}_j^{n+1} + \delta U_{j+1}^n = U_j^{n-1} + \delta U_j^{n-1} - \bar{\alpha} (\bar{U}_j^n + \delta U_j^n) [(\bar{U}_{j+1}^n + \delta U_{j+1}^n) - (\bar{U}_{j-1}^n + \delta U_{j-1}^n)].$$

Expanding the above expression while using the fact that  $\bar{U}_j^n$  satisfies (6.6.1) and neglecting squares and products of perturbations, we find

$$\delta U_j^{n+1} = \delta U_j^{n-1} - \bar{\alpha} \bar{U}_j^n (\delta U_{j+1}^n - \delta U_{j-1}^n) - \bar{\alpha} \delta U_j^n (\bar{U}_{j+1}^n - \bar{U}_{j-1}^n). \quad (6.6.3a)$$

The difference equation (6.6.3a) is linear, but has variable coefficients. In order to use a von Neumann stability analysis we would also have to assume that  $\bar{U}_j^n = \bar{U}$ , a constant. Were this done, (6.6.3a) would become

$$\delta U_j^{n+1} = \delta U_j^{n-1} - \bar{\alpha} \bar{U} (\delta U_{j+1}^n - \delta U_{j-1}^n). \quad (6.6.3b)$$

Using the von Neumann method (Problem 6.3.1), we would conclude that the leap frog scheme is absolutely stable when the Courant, Friedrichs, Lewy condition  $|\bar{U}| \bar{\alpha} \leq 1$  is satisfied. We would like to use this condition as a local stability analysis (*cf.* Section 4.4) to infer that the leap frog scheme is absolutely stable for this nonlinear problem when

$$\bar{\alpha} \max_j |U_j^n| \leq 1. \quad (6.6.3c)$$

A local stability analysis works for many finite difference schemes and problems; however, if disturbances are sufficiently strong, they may cause nonlinear instabilities. Gary [5] introduced an example where the solution of (6.6.1) has the form

$$U_j^n = \bar{U} + c_n \cos \pi j / 2 + s_n \sin \pi j / 2 + v_n \cos \pi j. \quad (6.6.4)$$

The trigonometric terms may be regarded as high-frequency perturbations to a constant solution  $\bar{U}$ . By substituting (6.6.4) into (6.6.1) we may show that  $c_n$ ,  $s_n$ , and  $v_n$  satisfy the difference equations

$$c_{n+1} - c_{n-1} = 2\bar{\alpha}s_n(v_n - \bar{U}), \quad (6.6.5a)$$

$$s_{n+1} - s_{n-1} = 2\bar{\alpha}c_n(v_n - \bar{U}), \quad (6.6.5b)$$

$$v_{n+1} = v_{n-1}. \quad (6.6.5c)$$

The solution of (6.6.5c) only takes on two values

$$v_n = \begin{cases} A, & \text{if } n \text{ is even} \\ B, & \text{if } n \text{ is odd} \end{cases}. \quad (6.6.5d)$$

We may use (6.6.5d) to eliminate  $v_n$  and  $s_n$  from (6.6.5a, 6.6.5b) to get

$$c_{n+2} - 2\mu c_n + c_{n-2} = 0, \quad (6.6.6a)$$

where

$$\mu = 1 + 2\bar{\alpha}^2(B - \bar{U})(A + \bar{U}). \quad (6.6.6b)$$

The constant-coefficient difference equation (6.6.6a) may be solved to yield

$$c_n = C_+ r_+^n + C_- r_-^n \quad (6.6.7a)$$

where

$$r_{\pm} = -\mu \pm i\sqrt{1 - \mu^2}. \quad (6.6.7b)$$

The constants  $C_+$  and  $C_-$  are determined from the initial and starting conditions.

The coefficients  $c_n$ ,  $n \geq 0$ , will not grow as  $n$  increases as long as  $|r_{\pm}| \leq 1$ . This in turn requires that  $|\mu| \leq 1$ . Using (6.6.4–6.6.7), we may show that the local linear stability condition (6.6.3c) is satisfied when the following two inequalities are satisfied:

$$\bar{\alpha}[|\bar{U}| + \max(|A|, |B|)] \leq 1 \quad (6.6.8a)$$

and

$$-1 < \bar{\alpha}^2(B - \bar{U})(A + \bar{U}). \quad (6.6.8b)$$

Under these conditions:

- If  $|A| < \bar{U}$ ,  $|B| \leq \bar{U}$ , and the linear stability conditions (6.6.8a, 6.6.8b) are satisfied then  $|\mu| < 1$ .
- If  $|A|$  and  $|B|$  are not bounded by  $\bar{U}$ , then it is possible to satisfy (6.6.8a, 6.6.8b) with  $|\mu| > 1$ . For example, choose  $A = 0$ ,  $B = \bar{U} + \epsilon$ , with  $\bar{U}$  and  $\epsilon$  positive. Then, (6.6.8b) is satisfied since

$$\bar{\alpha}^2 \epsilon \bar{U} > 0.$$

Choosing  $\bar{\alpha}$  such that

$$\bar{\alpha}(2\bar{U} + \epsilon) < 1$$

ensures that (6.6.8a) is satisfied. However, using (6.6.6b),

$$\mu = 1 + 2\bar{\alpha}^2 \epsilon \bar{U} > 1.$$

Thus, solutions will grow in magnitude. This example illustrates that high-frequency modes of a solution may interact with other modes to cause a nonlinear instability. This provides another reason to choose a scheme that dissipates high-frequency modes (Section 6.3).

### 6.6.1 Boundary Conditions

We need extra “artificial” boundary conditions with centered finite difference schemes like the Lax-Friedrichs and Lax-Wendroff methods. Let us illustrate this using a simple linear problem.

*Example 6.6.1.* Consider the initial-boundary value problem for the kinematic wave equation

$$u_t - u_x = 0, \quad 0 < x < 1, \quad t > 0, \tag{6.6.9a}$$

$$u(x, 0) = \phi(x), \quad u(1, t) = \psi(t), \tag{6.6.9b}$$

which has the exact solution

$$u(x, t) = \begin{cases} \phi(x + t), & \text{if } x + t \leq 1 \\ \psi(x + t - 1), & \text{if } x + t > 1 \end{cases}.$$

Establishing a uniform grid on  $[0, 1] \times t > 0$  and using a centered spatial scheme with the four-point stencil shown in Figure 6.6.1, we see that additional information is needed to compute the solution along the line  $x = 0$ .

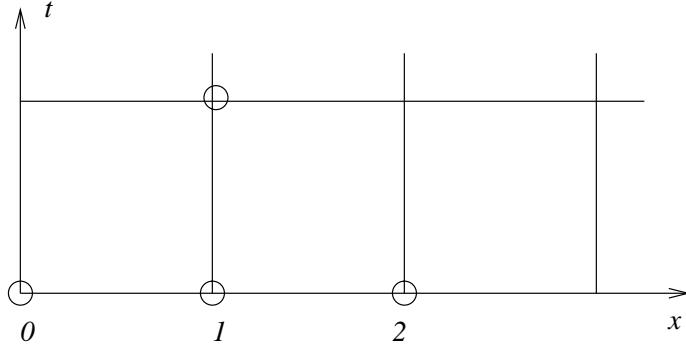


Figure 6.6.1: Uniform grid and four-point finite difference scheme illustrating the need for extra information at  $x = 0$ .

Consider, for example, a problem with  $\phi(x) = x$ . The exact solution for  $x + t < 1$  is  $u(x, t) = x + t$ . Let's solve this problem using the leap frog scheme

$$U_j^{n+1} = U_j^{n-1} + \frac{\Delta t}{\Delta x} (U_{j+1}^n - U_{j-1}^n) \quad (6.6.10a)$$

and the Lax-Friedrichs scheme

$$U_j^{n+1} = \frac{U_{j+1}^n + U_{j-1}^n}{2} + \frac{\Delta t}{2\Delta x} (U_{j+1}^n - U_{j-1}^n) \quad (6.6.10b)$$

with  $\Delta t = \Delta x = 0.1$  and simple constant extrapolation

$$U_0^{n+1} = U_1^{n+1}$$

as a boundary condition at  $x = 0$ . We further use the exact solution to start the multi-level leap frog scheme. The results, for a few time steps are shown in Table 6.6.1.

Both the leap frog and Lax-Friedrichs schemes produce the exact solution of this initial value problem since the Courant number is unity. Thus, the only errors arise due to the boundary approximation. From the table, we see that the Lax-Friedrichs scheme has confined the rather inaccurate treatment of the solution at  $x = 0$  to the one cell adjacent to the boundary. With the leap frog scheme, however, the poor boundary approximation is polluting the accuracy of the solution in the interior of the domain. The choice of proper

n	Leap Frog				Lax-Friedrichs			
	$j$				$j$			
	0	1	2	3	0	1	2	3
0	0.0	0.1	0.2	0.3	0.0	0.1	0.2	0.3
1	0.1	0.2	0.3	0.4	0.2	0.2	0.3	0.4
2	0.3	0.3	0.4	0.5	0.3	0.3	0.4	0.5
3	0.3	0.3	0.5	0.6	0.4	0.4	0.5	0.6
4	0.5	0.5	0.7	0.7	0.5	0.5	0.6	0.7
5	0.5	0.5	0.7	0.7	0.6	0.6	0.7	0.8

Table 6.6.1: Solution of Example 6.6.1 using the leap frog and Lax-Friedrichs methods with constant extrapolation at the boundary  $x = 0$ .

artificial boundary conditions for hyperbolic problems has been thoroughly studied. A good account of the theory and practice appears in Strikwerda [21], Chapter 11. The two principal methods of analyzing stability involve Laplace transforms and matrix methods. The former approach is often called the Gustafsson, Kreiss, Sundström, and Osher theory ([21], Section 11.3). The latter is the technique that we studied in Section 3.3. We'll be much more brief and note that, as a rule of thumb, one should try to use dissipative difference schemes and dissipative boundary conditions. Thus, with the present example, we could consider using the Lax-Friedrichs scheme in the interior and the forward time-forward space scheme

$$U_0^{n+1} = U_0^n + \frac{\Delta t}{\Delta x} (U_1^n - U_0^n)$$

at the boundary. With unit Courant number, this combination gives the exact solution.

Prescribing boundary conditions for  $m \times m$  linear vector systems (6.1.1b) requires the canonical form

$$\begin{bmatrix} \mathbf{w}^- \\ \mathbf{w}^+ \end{bmatrix}_t + \begin{bmatrix} \mathbf{\Lambda}^- & \mathbf{0} \\ \mathbf{0} & \mathbf{\Lambda}^+ \end{bmatrix} \begin{bmatrix} \mathbf{w}^- \\ \mathbf{w}^+ \end{bmatrix}_x = \mathbf{0}, \quad (6.6.11a)$$

where

$$\mathbf{AP} = \mathbf{P}\mathbf{\Lambda}, \quad \mathbf{\Lambda} = \begin{bmatrix} \mathbf{\Lambda}^- & \mathbf{0} \\ \mathbf{0} & \mathbf{\Lambda}^+ \end{bmatrix}, \quad (6.6.11b)$$

$$\mathbf{\Lambda}^- = \begin{bmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_p \end{bmatrix}, \quad \mathbf{\Lambda}^+ = \begin{bmatrix} \lambda_{p+1} & & & \\ & \lambda_{p+2} & & \\ & & \ddots & \\ & & & \lambda_m \end{bmatrix}, \quad (6.6.11c)$$

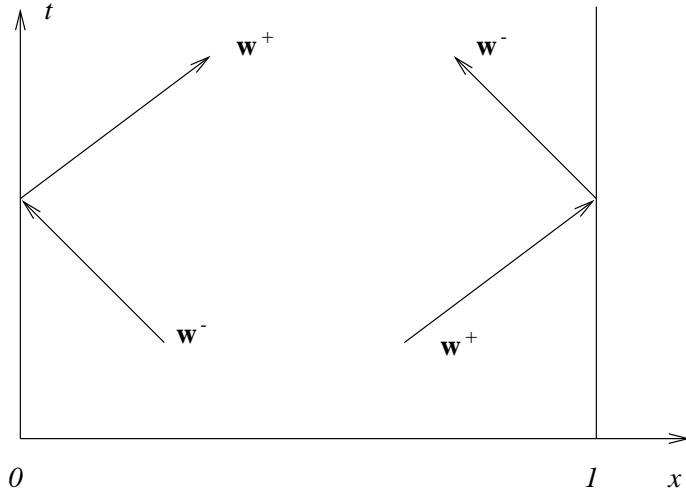


Figure 6.6.2: Boundary condition prescription for an initial-boundary value problem for (6.1.1b).

with  $\lambda_1 < \lambda_2 < \dots < \lambda_p < 0$  and  $\lambda_{p+1} < \lambda_{p+2} < \dots < \lambda_m > 0$ . Boundary conditions are needed along those characteristics that enter the region. Thus, for an initial boundary value problem on  $0 < x < 1$ , boundary conditions should be prescribed for  $\mathbf{w}^+$  at  $x = 0$  and for  $\mathbf{w}^-$  at  $x = 1$  (Figure 6.6.2). Artificial boundary conditions would, therefore, be needed for  $\mathbf{w}^-$  at  $x = 0$  and for  $\mathbf{w}^+$  at  $x = 1$ .

*Example 6.6.2.* Initial-boundary value problems on infinite domains must be approximated on finite domains. Artificial boundaries, placed far from the region of interest, can cause spurious wave reflections back into the domain. The goal is to construct boundary conditions on the artificial boundaries to eliminate or reduce these spurious reflections. Let us partition  $\mathbf{P}$  as

$$\mathbf{A}\mathbf{P}^- = \mathbf{P}^-\Lambda^-, \quad \mathbf{A}\mathbf{P}^+ = \mathbf{P}^+\Lambda^+. \quad (6.6.12a)$$

For simplicity, assume that  $\mathbf{P}$  is orthogonal in the sense that

$$(\mathbf{P}^-)^T \mathbf{P}^+ = 0. \quad (6.6.12b)$$

Assuming that  $\mathbf{P}$  is constant, the canonical transformation (6.1.3) may be differentiated to obtain

$$\dot{\mathbf{u}} = \mathbf{P}\dot{\mathbf{w}} = \mathbf{P}^-\dot{\mathbf{w}}^- + \mathbf{P}^+\dot{\mathbf{w}}^+$$

where  $(\cdot) = d(\cdot)/dt$ .

Let the artificial boundary be  $x = 0$  of Figure 6.6.2. No waves will enter the region if  $\dot{\mathbf{w}}^+ = 0$ ; thus,

$$\dot{\mathbf{u}} = \mathbf{P}^- \dot{\mathbf{w}}^-$$

or, multiplying by  $(\mathbf{P}^+)^T$  and using the orthogonality condition (6.6.12b),

$$(\mathbf{P}^+)^T \dot{\mathbf{u}} = 0. \quad (6.6.13a)$$

There are many possibilities for evaluating the directional derivative

$$\dot{\mathbf{u}} = \mathbf{u}_t + \dot{x}\mathbf{u}_x. \quad (6.6.13b)$$

Hedstrom [12] evaluates (6.6.13b) along the line  $x = 0$ . Another possibility is to assume that the initial data is trivial for  $x \leq 0$ ; thus, in particular  $\mathbf{u}_x = 0$ . In either case, the nonreflecting boundary conditions (6.6.13) become

$$(\mathbf{P}^+)^T \mathbf{u}_t = 0. \quad (6.6.14)$$

## Problems

1. For the solution (6.6.4) of the leap frog problem (6.6.1) verify that relations (6.6.5a–6.6.5c), (6.6.6a, 6.6.6b), (6.6.7a, 6.6.7b), and (6.6.8a, 6.6.8b) are satisfied.
2. ([21], Section 11.2). Consider the initial-boundary value problem (6.6.9) with the initial and boundary conditions prescribed so that its exact solution is

$$u(x, t) = \sin 2\pi(x + t).$$

- 2.1. Solve this problem using the leap frog scheme (6.6.10a) with the following artificial boundary conditions applied at  $x = 0$ :

$U_0^{n+1} = U_1^{n+1}$	Constant extrapolation
$U_0^{n+1} = U_1^n$	Constant diagonal extrapolation
$U_0^{n+1} = U_0^n + \lambda(U_1^n - U_0^n)$	Upwind differencing
$U_0^{n+1} = U_0^{n-1} + 2\lambda(U_1^n - U_0^n)$	Pseudo leap frog
$U_0^{n+1} = 2U_1^{n+1} - U_2^{n+1}$	Linear extrapolation
$U_0^{n+1} = 2U_1^n - U_2^{n-1}$	Linear diagonal extrapolation

Use  $\Delta x = 0.05$  and  $\lambda = 0.95$ . Display the solution at  $t = 1.9$ . Comment on the stability of each boundary condition based on the computational results. Stop any calculation prematurely if the computed solution exceeds two in magnitude. Display the solution at the last acceptable time in this case.

2.2. Repeat the computation for the Lax-Wendroff scheme.



# Bibliography

- [1] R. Biswas, K.D. Devine, and J.E. Flaherty. Parallel adaptive finite element methods for conservation laws. *Applied Numerical Mathematics*, 14:255–284, 1993.
- [2] A.J. Chorin. Random choice solution of hyperbolic systems. *Journal of Computational Physics*, 25:517–533, 1976.
- [3] B. Cockburn and C.-W. Shu. TVB Runge-Kutta local projection discontinuous finite element method for conservation laws II: General framework. *Mathematics of Computation*, 52:411–435, 1989.
- [4] J.E. Flaherty, R.M. Loy, C. Özturan, M.S. Shephard, B.K. Szymanski, J.D. Teresco, and L.H. Ziantz. Parallel structures and dynamic load balancing for adaptive finite element computation. *Applied Numerical Mathematics*, 26:241–265, 1998.
- [5] J. Gary. Lecture notes on the numerical solution of partial differential equations. Technical report, University of Colorado, Boulder, 1975.
- [6] J. Glimm. Solutions in the large for nonlinear hyperbolic systems of equations. *Communications on Pure and Applied Mathematics*, 18:697–715, 1965.
- [7] S.K. Godunov. A finite difference method for the numerical computation of discontinuous solutions of the equations of fluid dynamics. *Mat. Sbornik.*, 47:271–306, 1959.
- [8] A. Harten and J.M. Hyman. Self-adjusting grid methods for one-dimensional hyperbolic conservation laws. Technical Report LASL Report LA-9105, Los Alamos Scientific Laboratory, Los Alamos, 1983.

- [9] A. Harten, J.M. Hyman, and P.D. Lax. On finite-difference approximations and entropy conditions for shocks. *Communications on Pure and Applied Mathematics*, 29:297–322, 1976.
- [10] A. Harten and P.D. Lax. A random choice finite difference scheme for hyperbolic conservation laws. *SIAM Journal on Numerical Analysis*, 18:289–315, 1981.
- [11] A. Harten, P.D. Lax, and B. van Leer. On upstream difference and Godunov-type schemes for hyperbolic conservation laws. *SIAM Review*, 25:35–61, 1983.
- [12] G. Hedstrom. Nonreflecting boundary conditions for nonlinear hyperbolic systems. *Journal of Computational Physics*, 30:222–237, 1979.
- [13] A. Jeffrey and T. Taniuti. *Non-linear Wave Propagation*. Academic Press, New York, 1964.
- [14] P.D. Lax. *Hyperbolic Systems of Conservation Laws and the Mathematical Theory of Shock Waves*. Regional Conference Series in Applied Mathematics, No. 11. SIAM, Philadelphia, 1973.
- [15] P.D. Lax and B. Wendroff. Systems of conservation laws. *Communications on Pure and Applied Mathematics*, 13:217–237, 1960.
- [16] A.R. Mitchell and D.F. Griffiths. *The Finite Difference Method in Partial Differential Equations*. John Wiley and Sons, Chichester, 1980.
- [17] R.D. Richtmyer and K.W. Morton. *Difference Methods for Initial Value Problems*. John Wiley and Sons, New York, second edition, 1967.
- [18] P.L. Roe. Approximate Riemann solvers, parameter vectors, and difference schemes. *Journal of Computational Physics*, 43:357–372, 1981.
- [19] G.A. Sod. *Numerical Methods in Fluid Dynamic*. Cambridge University Press, Cambridge, 1985.

- [20] J.L Steger and R.F. Warming. Flux vector splitting of the inviscid gasdynamic equations with applications to finite difference methods. *Journal of Computational Physics*, 40:263–293, 1981.
- [21] J.C. Strikwerda. *Finite Difference Schemes and Partial Differential Equations*. Chapman and Hall, Pacific Grove, 1989.
- [22] B. van Leer. Towards the ultimate conservation difference scheme IV. A new approach of numerical convection. *Journal of Computational Physics*, 23:276–299, 1977.
- [23] B. van Leer. Flux-vector splitting gor the Euler equations. *Lecture Notes in Physics*, 170:507–512, 1982.

# Chapter 7

## Multidimensional Hyperbolic Problems

### 7.1 Split and Unsplit Difference Methods

Our study of multidimensional parabolic problems in Chapter 5 has laid most of the groundwork for our present task of creating difference approximations for multidimensional hyperbolic problems. By now we know that

- definitions of consistency, convergence, and stability for one-dimensional problems carry over to multiple dimensions;
- stability analyses by von Neumann's method are equivalent to those in one dimension but algebraically more complex; and
- implicit schemes are difficult to implement without splitting the operator or alternating directions.

Geometrical complexities will, again, be postponed until Chapter 8, so we'll consider two-dimensional conservation laws of the form

$$\mathbf{u}_t + \mathbf{f}(\mathbf{u})_x + \mathbf{g}(\mathbf{u})_y = 0 \quad (7.1.1a)$$

where  $\mathbf{u}$ ,  $\mathbf{f}$ , and  $\mathbf{g}$  are  $m$ -vectors. The convective form of this system is

$$\mathbf{u}_t + \mathbf{A}\mathbf{u}_x + \mathbf{B}\mathbf{u}_y = 0 \quad (7.1.1b)$$

where

$$\mathbf{A}(\mathbf{u}) = \mathbf{f}_u(\mathbf{u}), \quad \mathbf{B}(\mathbf{u}) = \mathbf{g}_u(\mathbf{u}). \quad (7.1.1c)$$

Multidimensional finite difference schemes can be simple extensions of one-dimensional methods. Thus, for example, a Lax-Friedrichs approximation of (7.1.1a) would be

$$\begin{aligned} \mathbf{U}_{jk}^{n+1} = & \frac{1}{4}(\mathbf{U}_{j+1,k}^n + \mathbf{U}_{j-1,k}^n + \mathbf{U}_{j,k+1}^n + \mathbf{U}_{j,k-1}^n) \\ & - \frac{\Delta t}{2\Delta x}(\mathbf{f}_{j+1,k}^n - \mathbf{f}_{j-1,k}^n) - \frac{\Delta t}{2\Delta y}(\mathbf{g}_{j,k+1}^n - \mathbf{g}_{j,k-1}^n). \end{aligned} \quad (7.1.2)$$

As in Chapter 5, a uniform mesh of spacing  $\Delta x \times \Delta y$  has been introduced with  $U_{jk}^n$  being the finite difference approximation of  $u(j\Delta x, k\Delta y, n\Delta t)$ .

Extending the Richtmyer two-step method (6.4.1) to two dimensions is a bit more difficult. The predictor step is the Lax-Friedrichs method

$$\begin{aligned} \mathbf{U}_{lm}^{n+1} = & \frac{1}{4}(\mathbf{U}_{l+1,m}^n + \mathbf{U}_{l-1,m}^n + \mathbf{U}_{l,m+1}^n + \mathbf{U}_{l,m-1}^n) - \frac{\Delta t}{2\Delta x}(\mathbf{f}_{l+1,m}^n - \mathbf{f}_{l-1,m}^n) \\ & - \frac{\Delta t}{2\Delta y}(\mathbf{g}_{l,m+1}^n - \mathbf{g}_{l,m-1}^n), \quad (l, m) = (j \pm 1, k), (j, k \pm 1), \end{aligned} \quad (7.1.3a)$$

applied to the four points surrounding  $(j, k)$  (Figure 7.1.1). The corrector step is the leap frog method

$$\mathbf{U}_{jk}^{n+2} = \mathbf{U}_{jk}^n - \frac{\Delta t}{\Delta x}(\mathbf{f}_{j+1,k}^{n+1} - \mathbf{f}_{j-1,k}^{n+1}) - \frac{\Delta t}{\Delta y}(\mathbf{g}_{j,k+1}^{n+1} - \mathbf{g}_{j,k-1}^{n+1}). \quad (7.1.3b)$$

The computational stencil of (7.1.3) is shown in Figure 7.1.1. The mesh spacing  $\Delta t$ ,  $\Delta x$ , and  $\Delta y$  has been doubled relative to the one-dimensional scheme (6.4.1). This was done to simplify the writing of the scheme. As shown on the bottom of Figure 7.1.1, the Richtmyer two-step scheme can be regarded as a nine-point difference formula on a staggered grid. This is possible because the physical flux vector and the divergence operator are rotationally invariant. Halving the mesh spacing to get a scheme that is more in line with our usual notation is much simpler with this staggered grid interpretation.

The Courant, Friedrichs, Lewy Theorem is still available to restrict the domain of dependence of a difference scheme to contain that of the partial differential equation. For example, the solution of the model initial value problem

$$u_t + au_x + bu_y = 0, \quad -\infty < x, y < \infty, \quad t > 0, \quad (7.1.4a)$$

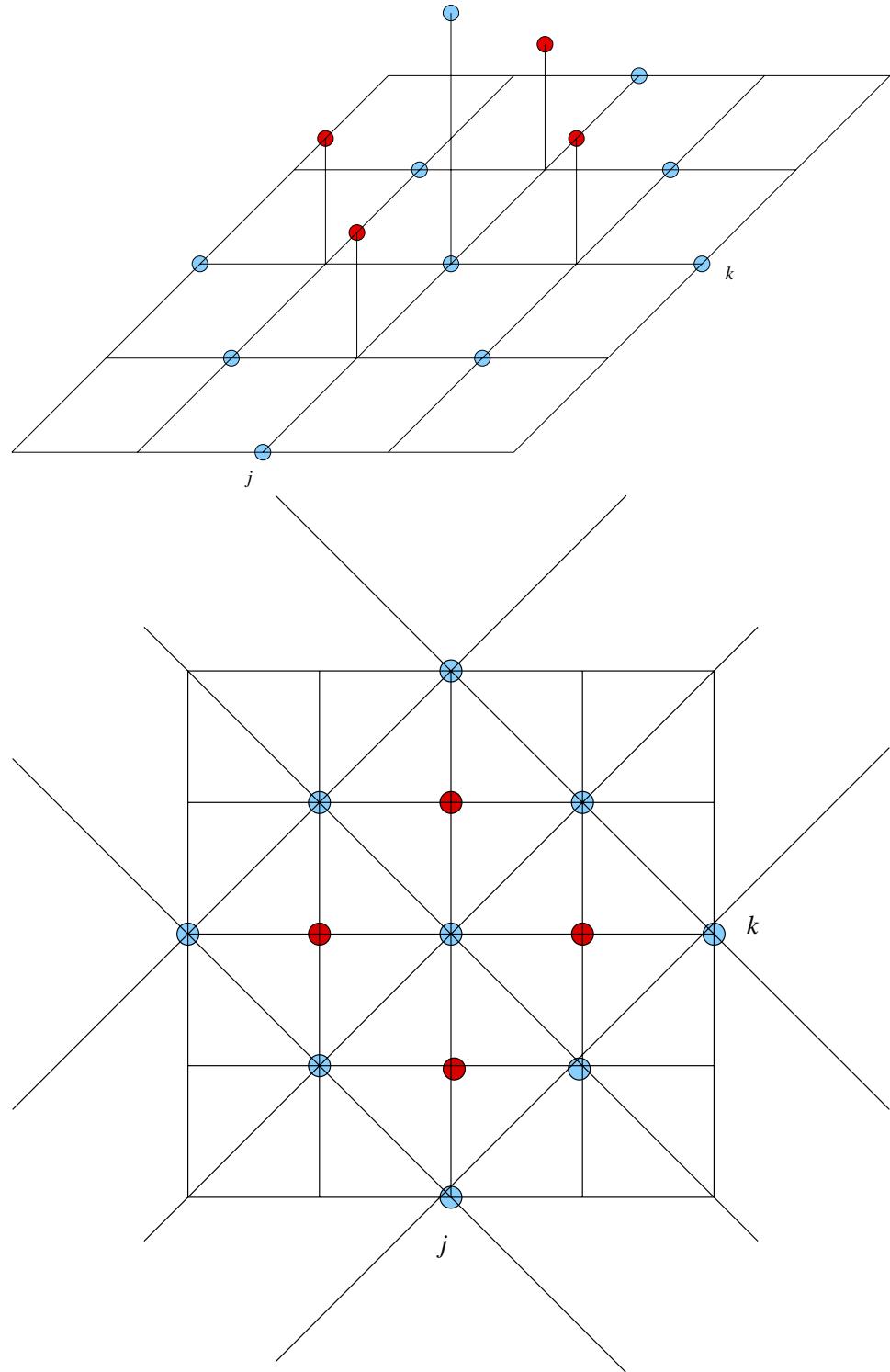


Figure 7.1.1: Computational stencil of the Richtmyer two-step method (7.1.3). Predicted solutions are shown with filled circles and corrected solutions are shown with blue circles and corrected solutions are shown in red. The Richtmyer two-step scheme can be regarded as a nine-point formula on a staggered grid (bottom).

$$u(x, y, 0) = \phi(x, y), \quad -\infty < x < \infty, \quad (7.1.4b)$$

is

$$u(x, y, t) = \phi(x - at, y - bt). \quad (7.1.4c)$$

Thus, the domain of dependence of a point  $(x_0, y_0, t_0)$  is the single point  $(x_0 - at_0, y_0 - bt_0)$ .

Let us consider a five- or nine-point explicit difference scheme of the form

$$U_{jk}^{n+1} = \sum_{l=j-1}^{j+1} \sum_{m=k-1}^{k+1} C_{lm} U_{lm}^n$$

The domain of dependence of the mesh point  $(j, k, n + 1)$  for the nine-point scheme is the rectangle

$$\mathcal{D} = \{(x, y) | x_{j-1} \leq x \leq x_{j+1}, y_{k-1} \leq y \leq y_{k+1}\}. \quad (7.1.5a)$$

A five-point scheme would have  $C_{j\pm 1, k\pm 1} = 0$ . Its domain of dependence would be less clear; however, for simplicity, we'll define it as

$$\mathcal{D} = \{(x, y) | \frac{|x - x_j|}{\Delta x} + \frac{|y - y_k|}{\Delta y} \leq 1\}. \quad (7.1.5b)$$

The domain of dependence of the point  $(j, k, n + 1)$  for the partial differential equation is the single point  $(x_j - a\Delta t, y_k - b\Delta t)$ . The various domains of dependence are shown in Figure 7.1.2. Thus, the Courant, Friedrichs, Lewy condition is

$$\max\left(\frac{|a|\Delta t}{\Delta x}, \frac{|b|\Delta t}{\Delta y}\right) \leq 1. \quad (7.1.6a)$$

for the nine-point scheme and

$$\frac{|a|\Delta t}{\Delta x} + \frac{|b|\Delta t}{\Delta y} \leq 1 \quad (7.1.6b)$$

for the five-point scheme.

The Lax-Friedrichs scheme applied to (7.1.4a) yields

$$U_{jk}^{n+1} = \frac{1}{4}(1 - 2\alpha)U_{j+1,k}^n + \frac{1}{4}(1 + 2\alpha)U_{j-1,k}^n + \frac{1}{4}(1 - 2\beta)U_{j,k+1}^n + \frac{1}{4}(1 + 2\beta)U_{j,k-1}^n$$

where

$$\alpha = \frac{a\Delta t}{\Delta x}, \quad \beta = \frac{b\Delta t}{\Delta y}.$$

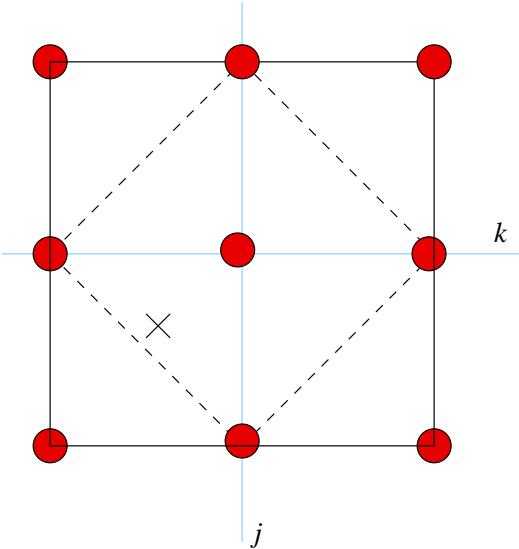


Figure 7.1.2: Domain of dependence of nine-point (solid line) and five-point (dashed line) difference schemes and of the partial differential equation ( $\times$ ).

Using the Maximum Principle, we find a sufficient condition for stability in the maximum norm as

$$\max(|\alpha|, |\beta|) \leq \frac{1}{2},$$

which is more restrictive than in one dimension. Using the von Neumann method (*cf.* Problem 1 at the end of this section), we may show that the Lax-Friedrichs scheme is stable in  $\mathcal{L}^2$  when

$$\alpha^2 + \beta^2 \leq \frac{1}{2}.$$

*Example 7.1.1.* With the more stringent stability restrictions, methods that use operator splitting become attractive. Let us consider splitting MacCormack's predictor-corrector scheme (6.4.2) for the conservation laws (7.1.2). In the  $x$ -direction, we neglect  $\mathbf{g}$ , predict using forward time-backward space differencing, and correct using backward time-forward space differencing for one half time step to obtain

$$\hat{\mathbf{U}}_{jk}^{n+1} = \mathbf{U}_{jk}^n - \frac{\Delta t}{\Delta x} (\mathbf{f}_{jk}^n - \mathbf{f}_{j-1,k}^n), \quad (7.1.7)$$

$$\bar{\mathbf{U}}_{jk}^{n+1} = \frac{\mathbf{U}_{jk}^n + \hat{\mathbf{U}}_{jk}^{n+1}}{2} - \frac{\Delta t}{2\Delta x} (\hat{\mathbf{f}}_{j+1,k}^{n+1} - \hat{\mathbf{f}}_{jk}^{n+1}) \quad (7.1.8)$$

where  $\hat{\mathbf{f}}_{jk}^{n+1} \equiv \mathbf{f}(\hat{\mathbf{U}}_{jk}^{n+1})$ . The solution obtained at the conclusion of sweeping the mesh by rows is denoted as  $\bar{\mathbf{U}}_{jk}^{n+1}$ . The sweep in the  $y$ -direction follows the same pattern with  $\mathbf{f}$  neglected and  $\mathbf{g}$  included and the solution  $\bar{\mathbf{U}}_{jk}^{n+1}$  used as initial data. Thus,

$$\tilde{\mathbf{U}}_{jk}^{n+1} = \bar{\mathbf{U}}_{jk}^{n+1} - \frac{\Delta t}{\Delta y} (\bar{\mathbf{g}}_{jk}^{n+1} - \bar{\mathbf{g}}_{j,k-1}^{n+1}), \quad (7.1.9)$$

$$\mathbf{U}_{jk}^{n+1} = \frac{\bar{\mathbf{U}}_{jk}^{n+1} + \tilde{\mathbf{U}}_{jk}^{n+1}}{2} - \frac{\Delta t}{2\Delta y} (\tilde{\mathbf{g}}_{j,k+1}^{n+1} - \tilde{\mathbf{g}}_{jk}^{n+1}). \quad (7.1.10)$$

This split scheme merely needs to satisfy the one-dimensional stability conditions when the boundary conditions between the  $x$  and  $y$  sweeps are implemented with care (*cf.* Section 5.2).

## Problems

- Analyze the stability of the Lax-Friedrichs scheme (7.1.2) for the kinematic wave equation (7.1.4) using von Neumann's method.

# Chapter 8

## Elliptic Problems

### 8.1 Difference Schemes for Poisson's Equation

Models of steady (time independent) problems in science and engineering give rise to elliptic partial differential systems. Examples arise in heat conduction, incompressible flow, electrostatic potentials, and static elasticity. Elliptic problems are boundary value problems rather than propagation problems. Poisson's equation is canonical and a typical two-dimensional problem involves determining  $u(x, y)$  satisfying

$$\mathcal{L}u = -\Delta u \equiv -u_{xx} - u_{yy} = f(x, y), \quad (x, y) \in \Omega, \quad (8.1.1a)$$

Subject to the boundary condition

$$\alpha u + \beta u_{\mathbf{n}} = \gamma, \quad (x, y) \in \partial\Omega. \quad (8.1.1b)$$

Here,  $\Omega \subset \Re^2$  with boundary  $\partial\Omega$  and  $\mathbf{n}$  is the unit outward normal to  $\partial\Omega$ .

Let us begin with the finite-difference solution of the Dirichlet problem ( $\alpha = 1, \beta = 0$ ) for (8.1.1) on the rectangular region

$$\Omega = \{(x, y) | 0 < x < a, 0 < y < b\}.$$

As usual, we introduce a uniform grid on  $\Omega$  having spacing  $\Delta x = a/J$  and  $\Delta y = b/K$  (Figure 8.1.1) and let  $U_{j,k}$  denote the finite difference approximation of  $u(j\Delta x, k\Delta y)$ . Replacing the derivatives in (8.1.1a) by centered differences yields the discrete system

$$-\frac{U_{j+1,k} - 2U_{j,k} + U_{j-1,k}}{\Delta x^2} - \frac{U_{j,k+1} - 2U_{j,k} + U_{j,k-1}}{\Delta y^2} = f_{j,k}$$

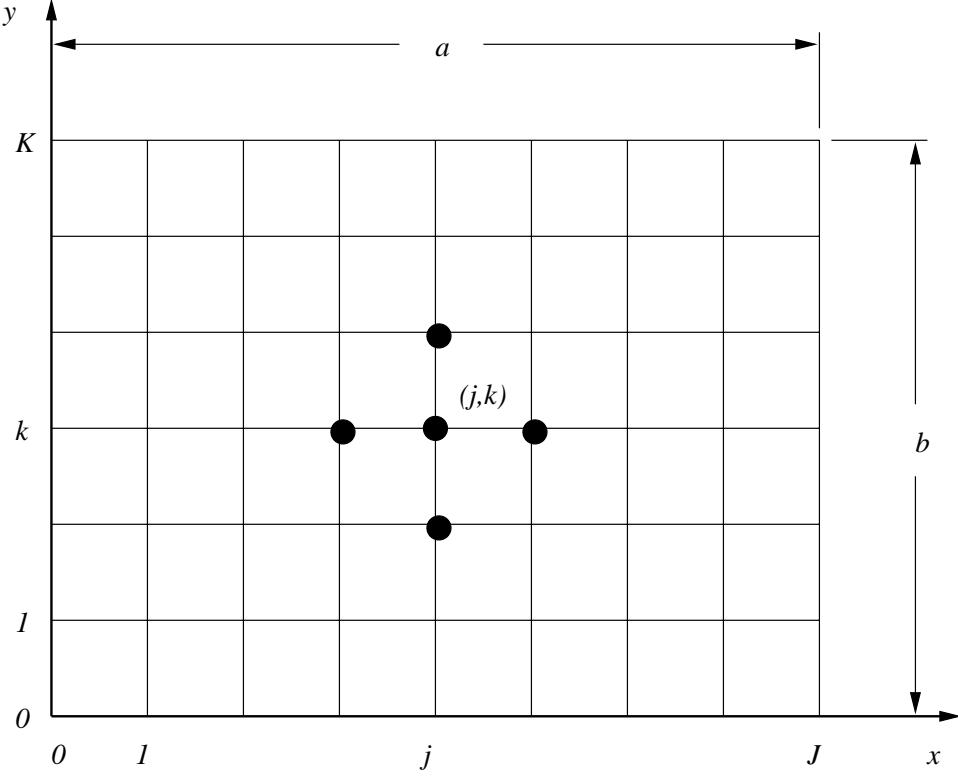


Figure 8.1.1: Grid structure for the solution of Poisson's equation on a rectangle.

or

$$U_{j,k} - \theta_x(U_{j+1,k} + U_{j-1,k}) - \theta_y(U_{j,k+1} + U_{j,k-1}) = \theta_{xy}f_{j,k}, \quad (8.1.2a)$$

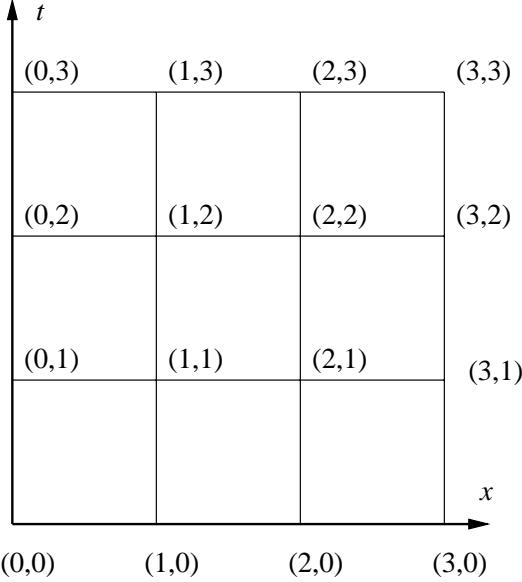
where

$$\theta_x = \frac{\Delta y^2}{2(\Delta x^2 + \Delta y^2)}, \quad \theta_y = \frac{\Delta x^2}{2(\Delta x^2 + \Delta y^2)}, \quad \theta_{xy} = \frac{\Delta x^2 \Delta y^2}{2(\Delta x^2 + \Delta y^2)}. \quad (8.1.2b)$$

The computational stencil for (8.1.2) is a five-point cross shown in Figure 8.1.1. The usual Taylor's series expansion reveals that (8.1.2) has an  $O(\Delta x^2) + O(\Delta y^2)$  local discretization error since centered difference formulas are used on a uniform mesh.

We obtain a discrete problem by writing the difference formula (8.1.2a) at each interior mesh point ( $j \in [1, J - 1]$ ,  $k \in [1, K - 1]$ ) and using the Dirichlet boundary conditions to obtain a linear algebraic system. Let us first illustrate this system for a  $3 \times 3$  mesh as shown in Figure 8.1.2. Writing (8.1.2a) at the mesh points  $(1, 1)$ ,  $(2, 1)$ ,  $(1, 2)$ , and  $(2, 2)$  gives

$$U_{1,1} - \theta_x U_{2,1} - \theta_y U_{1,2} = \theta_{xy} f_{1,1} + \theta_x \gamma_{0,1} + \theta_y \gamma_{1,0},$$

Figure 8.1.2: Mesh nomenclature for a  $3 \times 3$  grid.

$$-\theta_x U_{1,1} + U_{2,1} - \theta_y U_{2,2} = \theta_{xy} f_{2,1} + \theta_x \gamma_{3,1} + \theta_y \gamma_{2,0},$$

$$-\theta_y U_{1,1} + U_{1,2} - \theta_x U_{2,2} = \theta_{xy} f_{1,2} + \theta_x \gamma_{0,2} + \theta_y \gamma_{1,3},$$

$$-\theta_y U_{2,1} - \theta_x U_{1,2} + U_{2,2} = \theta_{xy} f_{2,2} + \theta_x \gamma_{3,2} + \theta_y \gamma_{2,3}.$$

The known boundary data is written on the right side of the equations to give us a  $4 \times 4$  linear system of equations for the four unknowns  $U_{j,k}$ ,  $j, k = 1, 2$ , at interior mesh points.

Following the nomenclature used in Chapter 5 for parabolic problems, let

$$\mathbf{U}_k = [U_{1,k}, U_{2,k}, \dots, U_{J-1,k}]^T \quad (8.1.3a)$$

denote the vector of unknowns in row  $k$  of the mesh and let

$$\mathbf{U} = [\mathbf{U}_1^T, \mathbf{U}_2^T, \dots, \mathbf{U}_{K-1}^T]^T \quad (8.1.3b)$$

denote the vector of all unknowns ordered by rows. Writing the difference equation (8.1.2a) at all interior mesh points gives the linear system

$$\mathbf{AU} = \mathbf{b}, \quad (8.1.4a)$$

or

$$\begin{bmatrix} \mathbf{C}_1 & \mathbf{D}_1 & & \\ \mathbf{D}_2 & \mathbf{C}_2 & \mathbf{D}_2 & \\ & \ddots & & \\ & & \mathbf{D}_{K-1} & \mathbf{C}_{K-1} \end{bmatrix} \begin{bmatrix} \mathbf{U}_1 \\ \mathbf{U}_2 \\ \vdots \\ \mathbf{U}_{K-1} \end{bmatrix} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \vdots \\ \mathbf{b}_{K-1} \end{bmatrix}, \quad (8.1.4b)$$

where

$$\mathbf{C}_k = \begin{bmatrix} 1 & -\theta_x & & \\ -\theta_x & 1 & -\theta_x & \\ & \ddots & \ddots & \\ & & -\theta_x & 1 \end{bmatrix}, \quad \mathbf{D}_k = \begin{bmatrix} -\theta_y & & & \\ & -\theta_y & & \\ & & \ddots & \\ & & & -\theta_y \end{bmatrix}, \quad (8.1.4c)$$

and

$$\mathbf{b}_k = \theta_{xy} \begin{bmatrix} f_{1,k} \\ f_{2,k} \\ \vdots \\ f_{J-1,k} \end{bmatrix} + \theta_x \begin{bmatrix} \gamma_{0,k} \\ 0 \\ \vdots \\ 0 \\ \gamma_{J,k} \end{bmatrix} + \delta_{1,k} \theta_y \begin{bmatrix} \gamma_{1,0} \\ \gamma_{2,0} \\ \vdots \\ \gamma_{J-1,0} \end{bmatrix} + \delta_{K-1,k} \theta_y \begin{bmatrix} \gamma_{1,K} \\ \gamma_{2,K} \\ \vdots \\ \gamma_{J-1,K} \end{bmatrix}. \quad (8.1.4d)$$

The matrices  $\mathbf{C}_k$  and  $\mathbf{D}_k$  are  $(J-1) \times (J-1)$  tridiagonal and diagonal matrices, respectively; thus,  $\mathbf{A}$  is  $(J-1)(K-1) \times (J-1)(K-1)$  block tridiagonal matrix. The Kronecker delta  $\delta_{j,k}$  is unity when  $j = k$  and zero otherwise.

Problems involving Neumann or Robin boundary conditions are treated as described in Section 4.2 for parabolic problems. Problems on nonrectangular domains may be treated by finite element or finite volume (Section 8.2) methods.

We conclude this section by noting that linear elliptic partial differential equations satisfy a maximum principle which states that the maximum (minimum) value of  $u$  in a closed region  $\Omega$  occurs on the boundary  $\partial\Omega$ .

*Example 8.1.1.* Let us illustrate the maximum principle for Laplace's equation

$$\Delta u = u_{xx} + u_{yy} = 0.$$

Functions  $u$  satisfying Laplace's equation also satisfy the “mean value theorem”

$$u(\mathbf{P}) = \frac{1}{2\pi} \int_0^{2\pi} u(r \cos \theta, r \sin \theta) d\theta \quad (8.1.5)$$

around any circle of radius  $r$  centered at the point  $\mathbf{P}$ . Since  $u(\mathbf{P})$  at the center is the average of values on any circle surrounding  $\mathbf{P}$ ,  $u$  must assume its maximum and minimum values on the circle. Since this holds for any circle, we have proven the maximum principle for harmonic functions (*i.e.*, functions that satisfy Laplace's equation).

We would like a discrete model of an elliptic problem to have as many properties of the continuous problem as possible. The maximum principle, in particular, is a good

property to have because it can be used to establish the uniqueness of solutions. Consider the discrete approximation (8.1.2) of Laplace's equation ( $f(x, y) = 0$ ) and, for simplicity, let  $\Delta x = \Delta y$ . In this case, use of (8.1.2b) implies that  $\theta_x = \theta_y = 1/4$  and (8.1.2a) becomes

$$U_{j,k} = \frac{1}{4}(U_{j+1,k} + U_{j-1,k} + U_{j,k+1} + U_{j,k-1}).$$

Thus, the centered difference approximation (8.1.2) satisfies a discrete version of the maximum principle.

Let us pursue this matter in a more abstract setting by considering a difference approximation of the form

$$\mathbf{L}_\Delta U_{j,k} = c_0 U_{j,k} - \sum_{|\mathbf{s}| \leq S, |\mathbf{s}| \neq 0} c_{\mathbf{s}} U_{\mathbf{Q}_{\mathbf{s}}} = f_{j,k} \quad (8.1.6)$$

to a linear elliptic problem

$$\mathcal{L}u = f(x, y).$$

The summation in (8.1.6) is taken over a finite set of points surrounding  $(x_j, y_k)$ . We regard  $\mathbf{s} = [s_1, s_2]^T$  as a vector having integral coefficients indicating the points in the stencil of the difference scheme with the interpretation  $\mathbf{Q}_{\mathbf{s}} = (j + s_1, k + s_2)$ .

We'll begin with some properties that (8.1.6) might possess.

**Definition 8.1.1.** *The finite difference scheme (8.1.6) is non-negative if*

$$c_0 > 0, \quad c_{\mathbf{s}} \geq 0, \quad \forall |\mathbf{s}| \leq S, \quad \forall (x_j, y_k) \in \Omega. \quad (8.1.7)$$

**Definition 8.1.2.** *The finite difference scheme (8.1.6) is diagonally dominant if*

$$c_0 \geq \sum_{|\mathbf{s}| \leq S, |\mathbf{s}| \neq 0} |c_{\mathbf{s}}|, \quad \forall (x_j, y_k) \in \Omega. \quad (8.1.8)$$

*with strict inequality holding for at least one mesh point of  $\Omega$ .*

*Example 8.1.2.* Consider the solution of Laplace's equation using the centered scheme (8.1.2) with  $f(x, y) = 0$ , i.e.,

$$U_{j,k} - \theta_x(U_{j+1,k} + U_{j-1,k}) - \theta_y(U_{j,k+1} + U_{j,k-1}) = 0.$$

In comparison with (8.1.6), we have

$$c_0 = 1, \quad c_{1,0} = c_{-1,0} = \theta_x, \quad c_{0,1} = c_{0,-1} = \theta_y.$$

From (8.1.2b) we see that the coefficients  $\theta_x$  and  $\theta_y$  are positive; thus, this difference scheme is non-negative. Also, using (8.1.2b)

$$\sum_{|\mathbf{s}| \leq S, |\mathbf{s}| \neq 0} |c_{\mathbf{s}}| = 2\theta_x + 2\theta_y = 1.$$

The question of diagonal dominance is, as yet, undecided. Both  $c_0$  and the sum of the off-diagonal entries in the difference scheme are unity. This will be true at all mesh points that are not adjacent to boundaries. Thus, the scheme will be diagonal dominant or not depending on the prescribed boundary conditions. Homogeneous Dirichlet data, for example, will drop at least one of the four terms in the difference scheme at mesh points adjacent to the boundary and result in diagonal dominance.

The extension of the maximum principle to elliptic operators having the form of (8.1.6) follows.

**Definition 8.1.3.** *Let  $\mathbf{V}$  be any function defined on the mesh that satisfies*

$$\mathbf{L}_\Delta V_{j,k} \leq 0, \quad \forall (x_j, y_k) \in \Omega. \quad (8.1.9)$$

*If the value  $V_{j,k}$  at mesh points  $(x_j, y_k) \in \Omega$  nowhere exceeds its value at mesh points  $(x_j, y_k) \in \partial\Omega$  then  $\mathbf{L}_\Delta$  is said to satisfy a discrete maximum principle.*

*Remark 1.* At an extreme point of a function  $u(x, y)$  we have  $u_x = u_y = 0$ . If the extreme point is a maximum then we also have  $u_{xx} < 0$  and  $u_{yy} < 0$ . We would, therefore, expect an elliptic operator, such as  $-u_{xx} - u_{yy}$ , to be positive at a maximum point of  $u$ . For this reason, it suffices to consider discrete functions  $\mathbf{V}$  satisfying  $\mathbf{L}_\Delta V_{j,k} \leq 0$ .

The importance of a non-negative diagonally dominant difference scheme in relation to the maximum principle is expressed in Theorem 8.1.1.

**Theorem 8.1.1.** *A non-negative diagonally dominant finite difference scheme of the form (8.1.6) satisfies a discrete maximum principle.*

*Proof.* Without loss of generality, assume that  $\mathbf{V} > 0$ , since a large positive constant can be added to  $\mathbf{V}$  without affecting the location of extreme points. From (8.1.6) and (8.1.9) we have

$$c_0 V_{j,k} \leq \sum_{|\mathbf{s}| \leq S, |\mathbf{s}| \neq 0} c_{\mathbf{s}} V_{\mathbf{Q}_{\mathbf{s}}} \leq \max_{|\mathbf{s}| \leq S, |\mathbf{s}| \neq 0} V_{\mathbf{Q}_{\mathbf{s}}} \left( \sum_{|\mathbf{s}| \leq S, |\mathbf{s}| \neq 0} |c_{\mathbf{s}}| \right).$$

Using (8.1.7) and (8.1.8)

$$c_0 V_{j,k} \leq c_0 \max_{|\mathbf{s}| \leq S, |\mathbf{s}| \neq 0} V_{\mathbf{Q}_{\mathbf{s}}}.$$

Therefore,

$$V_{j,k} \leq \max_{|\mathbf{s}| \leq S, |\mathbf{s}| \neq 0} V_{\mathbf{Q}_{\mathbf{s}}}.$$

□

As with continuous problems, the discrete maximum principle may be used to prove uniqueness of finite difference solutions. Additional results involving maximum principles are given in Mitchell and Griffiths [6], Section 3.8, and Strikwerda [7], Section 12.3.

## 8.2 Finite Volume Methods

Let us consider the solution of a two-dimensional problem on a non-rectangular domain  $\Omega$  as shown in Figure 8.2.1. As with a rectangular domain, let us cover  $\Omega$  with a uniform rectangular grid of spacing  $\Delta x \times \Delta y$ . Interior mesh points, such as  $A$  of Figure 8.2.1, can be treated by standard finite difference methods. Points near the boundary, such as  $P$ , require special treatment. Approximating  $\partial\Omega$  by a piecewise rectangular function, so that standard difference formulas may be used, is not very accurate. A more satisfactory procedure is to use “unequal arm” finite difference formulas. Recall, that such approximations were introduced in Section 2.1; thus, for example, using (2.1.11) we obtain

$$(u_x)_P = \frac{1}{2} \left[ \frac{u_E - u_P}{\Delta x_E} + \frac{u_P - u_W}{\Delta x_W} \right] - \frac{1}{4} (\Delta x_E - \Delta x_W) (u_{xx})_P + O(\Delta x^2) \quad (8.2.1a)$$

and

$$(u_{xx})_P = \frac{2}{\Delta x_E + \Delta x_W} \left[ \frac{u_E - u_P}{\Delta x_E} - \frac{u_P - u_W}{\Delta x_W} \right] - \frac{1}{6} (\Delta x_E - \Delta x_W) (u_{xxx})_P + O(\Delta x^2). \quad (8.2.1b)$$

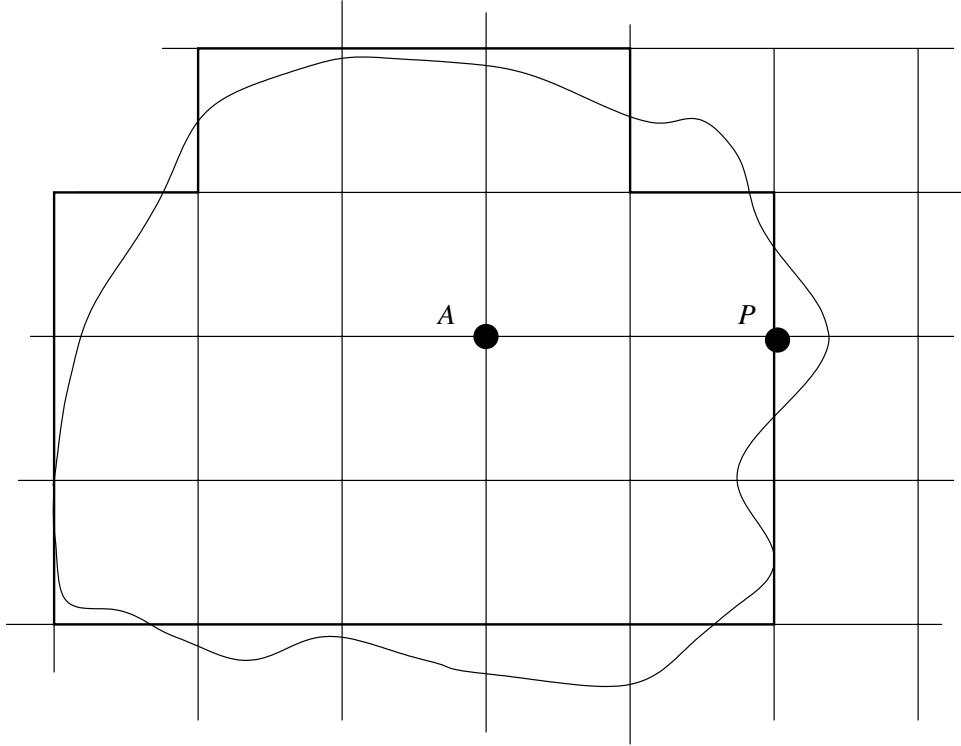


Figure 8.2.1: Two-dimensional non-rectangular domain  $\Omega$  with a superimposed uniform rectangular grid. Standard difference approximations can be used at interior points such as  $A$ . Points, such as  $P$ , near the boundary require special treatment.

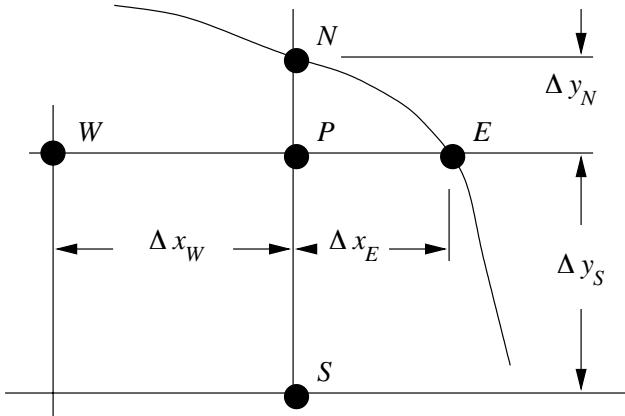


Figure 8.2.2: Notation used for unequal-arm finite difference approximations near a boundary point  $P$ .

As shown in Figure 8.2.2, we are using a single-subscript notation. Thus, if  $P$  has the indices  $(j, k)$ , we let  $u_P = u_{j,k}$ ,  $u_E = u_{j+1,k}$ , etc. Similarly,  $\Delta x_E = x_{j+1} - x_j$ , etc., and  $\Delta x = \max(\Delta x_E, \Delta x_W)$ . Approximations obtained by neglecting the  $u_{xx}$  term in (8.2.1a) and the  $u_{xxx}$  term in (8.2.1b) are only first-order accurate. The order of accuracy can be

improved by retaining more terms in Taylor's series expansions of the solution about  $P$ . As noted in Chapter 2, this will enlarge the computational stencil by bringing in points to the left of  $W$ . A second-order approximation for  $u_x$  is easily obtained by substituting the expression (8.2.1b) for  $u_{xx}$  into (8.2.1a)

$$(u_x)_P = \frac{1}{\Delta x_E + \Delta x_W} \left[ \frac{\Delta x_W}{\Delta x_E} (u_E - u_P) + \frac{\Delta x_E}{\Delta x_W} (u_P - u_W) \right] + O(\Delta x^2). \quad (8.2.1c)$$

Dirichlet boundary conditions using unequal-arm differences such as (8.2.1) are only slightly more complicated to implement than those on a uniform mesh. For example, the discrete form of (8.1.2a) with prescribed Dirichlet data  $u = \gamma(x, y)$ ,  $(x, y) \in \partial\Omega$ , would be

$$-\frac{2}{\Delta x_E + \Delta x_W} \left[ \frac{\gamma_E - U_P}{\Delta x_E} - \frac{U_P - U_W}{\Delta x_W} \right] - \frac{2}{\Delta y_N + \Delta y_S} \left[ \frac{\gamma_N - U_P}{\Delta y_N} - \frac{U_P - U_S}{\Delta x_S} \right] = f_P$$

at a point such as  $P$  of Figure 8.2.2. Here,  $\gamma_E$  and  $\gamma_N$  denote the values of  $\gamma$  at the boundary points  $E$  and  $N$ , respectively.

Implementing Neumann boundary data, however, is substantially more complex and, in fact, does not have a satisfactory resolution using an approach like we have described. The “finite volume” technique offers a better way of treating Neumann boundary data and, in general, of solving problems on nonuniform meshes. Actually, finite element techniques are the most effective way of both satisfying Neumann data and implementing solution strategies on nonuniform meshes. Finite element techniques are not studied in these notes, but we will describe the finite volume relative to the model problem

$$-\nabla \cdot \sigma \nabla u = -(\sigma u_x)_x - (\sigma u_y)_y = f(x, y), \quad (x, y) \in \Omega, \quad (8.2.2)$$

where  $\sigma = \sigma(x, y)$ .

Let us integrate (8.2.2) on some region  $R \subset \Omega$  and use the divergence theorem to obtain

$$0 = \iint_R [\nabla \cdot \sigma \nabla u + f] dx dy = \int_{\partial R} \sigma u_{\mathbf{n}} ds + \iint_R f dx dy \quad (8.2.3)$$

where  $\mathbf{n}$  is a unit outward normal to  $\partial R$  and  $s$  is a coordinate along  $\partial R$ .

Approximate solution techniques are obtained by choosing  $R$  appropriately and using finite differences to approximate derivatives. Let us first apply the technique to a

nonuniform mesh of rectangular cells. Consider a cluster of four cells as shown in Figure 8.2.3 and let  $R$  be the rectangular region obtained by bisecting each arm and connecting the normals through the bisection points.

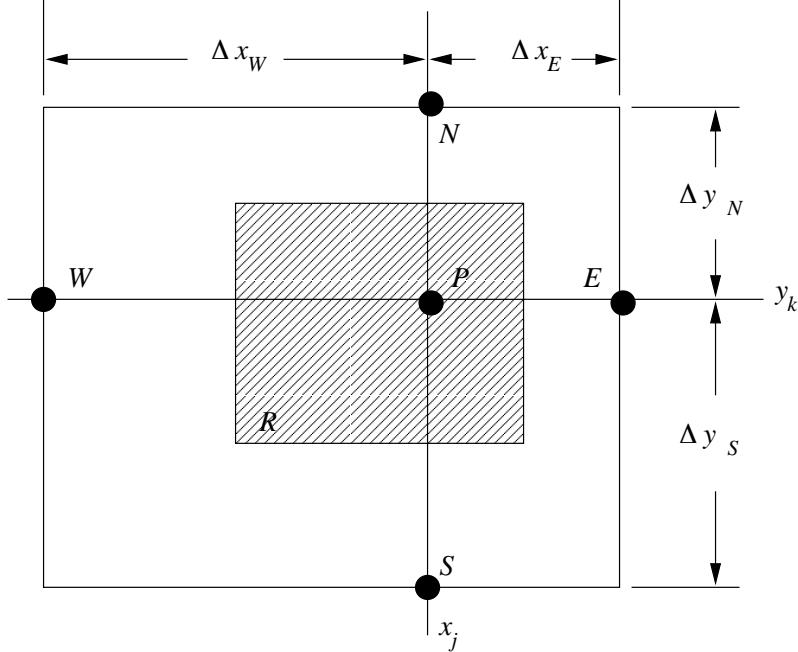


Figure 8.2.3: Group of four rectangular cells and the region  $R$  used for the finite volume technique.

Line integrals in (8.2.3) are approximated using the midpoint rule and normal derivatives are approximated using centered differences; thus, for example, the portion of line integral in (8.2.3) on the right edge of  $R$  would be

$$\int_{y_k-1/2\Delta y_S}^{y_k+1/2\Delta y_N} \sigma u_x dy \approx \sigma(x_j + \Delta x_E/2, y_k) \left( \frac{U_E - U_P}{\Delta x_E} \right) \left( \frac{\Delta y_N + \Delta y_S}{2} \right). \quad (8.2.4a)$$

The area integral in (8.2.3) is approximated by the rectangular rule using data at the point  $P$  to get

$$\iint_R f(x, y) dx dy \approx f(x_j, y_k) \left( \frac{\Delta x_E + \Delta x_W}{2} \right) \left( \frac{\Delta y_N + \Delta y_S}{2} \right). \quad (8.2.4b)$$

Using (8.2.4) in (8.2.3), we obtain the complete scheme as

$$0 = f_P \left( \frac{\Delta x_E + \Delta x_W}{2} \right) \left( \frac{\Delta y_N + \Delta y_S}{2} \right)$$

$$\begin{aligned}
& + \sigma_{j+1/2,k} \left( \frac{U_E - U_P}{\Delta x_E} \right) \left( \frac{\Delta y_N + \Delta y_S}{2} \right) + \sigma_{j,k+1/2} \left( \frac{U_N - U_P}{\Delta y_N} \right) \left( \frac{\Delta x_E + \Delta x_W}{2} \right) \\
& + \sigma_{j-1/2,k} \left( \frac{U_W - U_P}{\Delta x_W} \right) \left( \frac{\Delta y_N + \Delta y_S}{2} \right) + \sigma_{j,k-1/2} \left( \frac{U_S - U_P}{\Delta y_S} \right) \left( \frac{\Delta x_E + \Delta x_W}{2} \right).
\end{aligned} \tag{8.2.5}$$

This version of the finite volume scheme is only first-order accurate unless the mesh spacing is uniform. With uniform spacing and  $\sigma = 1$ , (8.2.5) is identical to centered differences (8.1.2). Even though it is first order, the unequal-arm difference scheme may use fewer computational cells on a nonuniform mesh and produce comparable accuracy to a second-order scheme on a denser uniform mesh. Such relative tradeoffs have to be appraised.

The finite volume scheme can be applied on non-rectangular regions. This is useful near curved boundaries and for unstructured-mesh computation. Let us illustrate the approximation of boundary data first. Two possibilities are shown in Figure 8.2.4. In the case shown on the top, the point  $P$  is not on  $\partial\Omega$  and the region  $R$  is selected as a triangularly-shaped region. The line integral in (8.2.3) is approximated as

$$\begin{aligned}
\int_{\partial R} \sigma u_{\mathbf{n}} ds & \approx \sigma_{NE}(U_{\mathbf{n}})_{NE} \Delta s_{NE} + \sigma_{j-1/2,k} \left( \frac{U_W - U_P}{\Delta x_W} \right) (\Delta y_N + \Delta y_S / 2) \\
& + \sigma_{j,k-1/2} \left( \frac{U_S - U_P}{\Delta y_S} \right) (\Delta x_E + \Delta x_W / 2),
\end{aligned} \tag{8.2.6a}$$

where  $\Delta s_{NE}$  is the length of the outer boundary of  $R$  and  $(U_{\mathbf{n}})_{NE}$  is the approximation of  $u_{\mathbf{n}}$  at the midpoint of the outer boundary of  $R$  (cf. Figure 4). The value of  $(U_{\mathbf{n}})_{NE}$  is obtained from the prescribed Neumann boundary data, *e.g.*,

$$(U_{\mathbf{n}})_{NE} = \gamma_{NE}. \tag{8.2.6b}$$

The above choice of mesh is reasonable but it doesn't provide for solutions to be computed on the boundary. Another alternative, shown at the bottom of Figure 8.2.4, is to design the mesh so that point  $P$  is on the boundary. Then, taking the region  $R$  as shown, the line integral in (8.2.3) is approximated as

$$\int_{\partial R} \sigma u_{\mathbf{n}} ds \approx \sigma_{PSE}(U_{\mathbf{n}})_{PSE} \Delta s_{PSE} + \sigma_{PNW}(U_{\mathbf{n}})_{PNW} \Delta s_{PNW}$$

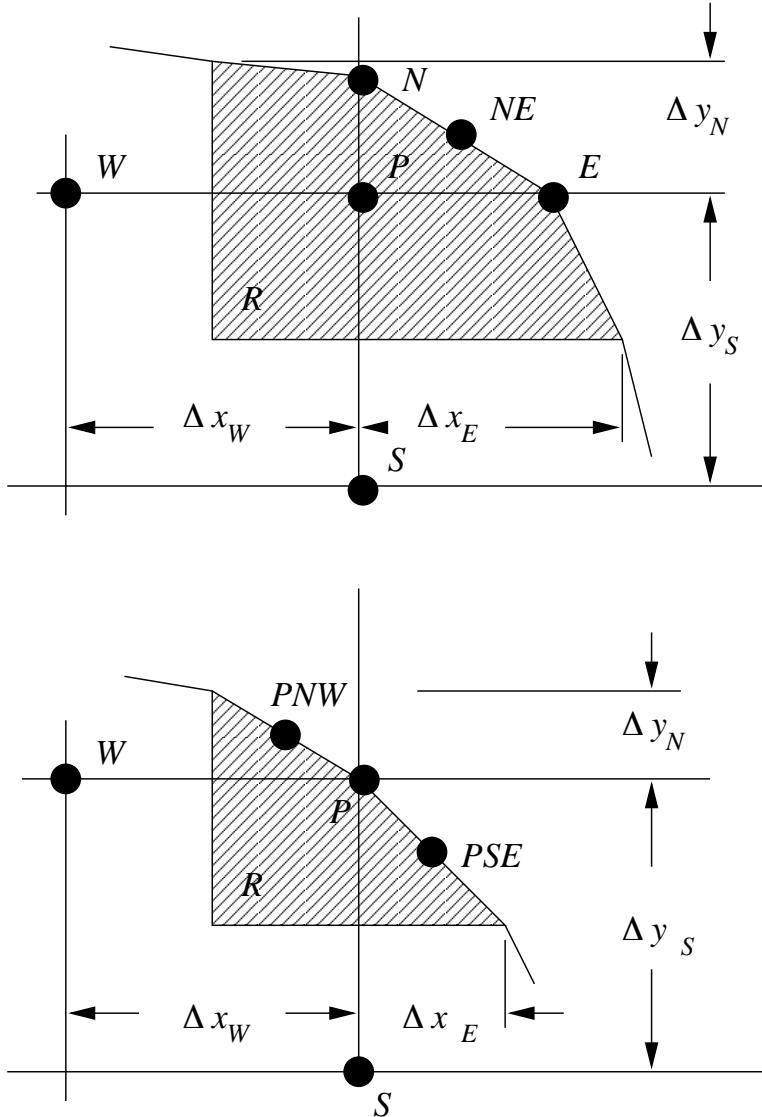


Figure 8.2.4: Boundary approximation by the finite volume method when the point  $P$  is off (top) and on (bottom) the boundary.

$$\sigma_{j-1/2,k} \left( \frac{U_W - U_P}{\Delta x_W} \right) (\Delta y_N + \Delta y_S/2) + \sigma_{j,k-1/2} \left( \frac{U_S - U_P}{\Delta y_S} \right) (\Delta x_E/2 + \Delta x_W). \quad (8.2.7)$$

The subscript  $PSE$  identify the point midway between points  $P$  and  $E$  (Figure 8.2.4). Similarly, point  $PNW$  is midway between points  $P$  and  $NW$ . The lengths of the arcs containing  $PSE$  and  $PNW$  are  $\Delta s_{PSE}$  and  $\Delta s_{PNW}$ , respectively.

Completely “unstructured” meshes of triangular or quadrilateral elements in two dimensions and tetrahedra and hexahedra in three dimensions are used for finite element computation and have become popular with finite volume methods as well. A portion

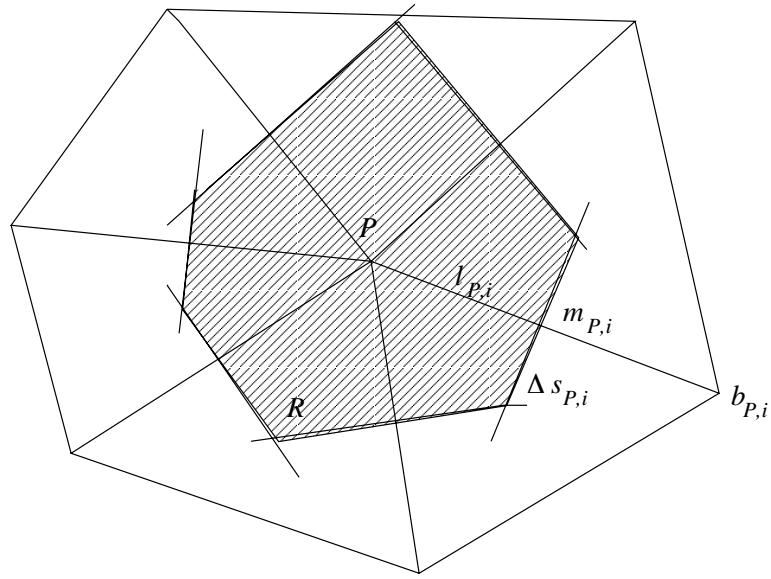


Figure 8.2.5: Portion of a mesh of triangular elements and the region  $R$  surrounding a point  $P$ .

of a mesh surrounding a point  $P$  is shown in Figure 8.2.5. The region  $R$  is taken as the polygon formed from the perpendicular bisectors of the element edges intersecting point  $P$ . The approximation of (8.2.3) on such a mesh is

$$f_P A_R + \sum_{i=1}^{N_P} \sigma_{m_{P,i}} \left( \frac{U_{b_{P,i}} - U_P}{l_{P,i}} \right) \Delta s_{P,i} = 0 \quad (8.2.8)$$

where  $A_R$  is the area of region  $R$ ;  $b_{P,i}$ ,  $i = 1, 2, \dots, N_P$ , are the neighbors of  $P$ ;  $m_{P,i}$  identifies the point midway between  $P$  and its neighbor  $b_{P,i}$ ;  $l_{P,i}$  is the length of the edge from  $P$  to  $b_{P,i}$ ; and  $\Delta s_{P,i}$  is the length of the segment of  $\partial R$  perpendicular to  $l_{P,i}$ .

Clearly the geometry has become more complex than with uniform meshes. It would now be necessary to store the coordinates of all of the vertices of the triangular elements and their connectivity. We'll say a bit more about data structures in the next section. It has been difficult to extend finite volume methods to higher orders of accuracy; however, the discontinuous Galerkin method described in Section 6.5 provides a systematic means of doing this. The finite element method also provides a means of constructing arbitrarily high-order approximations.

### Problems

1. A finite volume method may be constructed on an unstructured grid of, *e.g.*, trian-

gular elements by placing the unknowns (at points such as  $P$ ) at cell centroids rather than vertices. Develop an approximation for (8.2.3) using such a mesh structure.

2. Consider a cluster of four rectangular computational cells contained within

$$D_{j,k} = \{(x, y) \mid x_{j-1} \leq x \leq x_{j+1}, y_{k-1} \leq y \leq y_{k+1}\}.$$

Assume that  $\sigma(x, y) = 1$  above the line  $C$  of  $D_{j,k}$  connecting  $(x_{j-1}, y_{k-1})$ ,  $(x_j, y_k)$ , and  $(x_{j+1}, y_{k+1})$  and that  $\sigma(x, y) = 2$  below this line. Calculate a discrete approximation to the partial differential equation

$$-(\sigma u_x)_x - (\sigma u_y)_y + qu = f(x, y)$$

at  $(x_j, y_k)$  using the finite volume method. Do not assume that  $(x_j, y_k)$  is at the center of  $D_{j,k}$ .

### 8.3 Mesh Generation

Discretizing two-dimensional domains into meshes having nonuniform cell spacings can either be simple or difficult depending on geometric or solution complexities. Discretizing three-dimensional domains is currently not simple. Nonuniform meshes may be appropriate for some problems with simple geometry due to rapid solution variation in some regions, *e.g.*, in boundary layers. Some computational specialists argue that constructing the proper mesh for a problem is the major impediment to its solution. Several attempts to automate the mesh generation process are underway. Adaptive solution-based mesh refinement procedures automatically concentrate meshes in such regions and seek to automate the task of modifying (refining/coarsening) an existing mesh [1, 3]. While we will not attempt a thorough treatment of all approaches, we will discuss the essential ideas of mesh generation by (i) mapping techniques where a complex domain is transformed into a simpler one so that a mesh may be easily generated and (ii) direct techniques where a mesh is generated on the original domain.

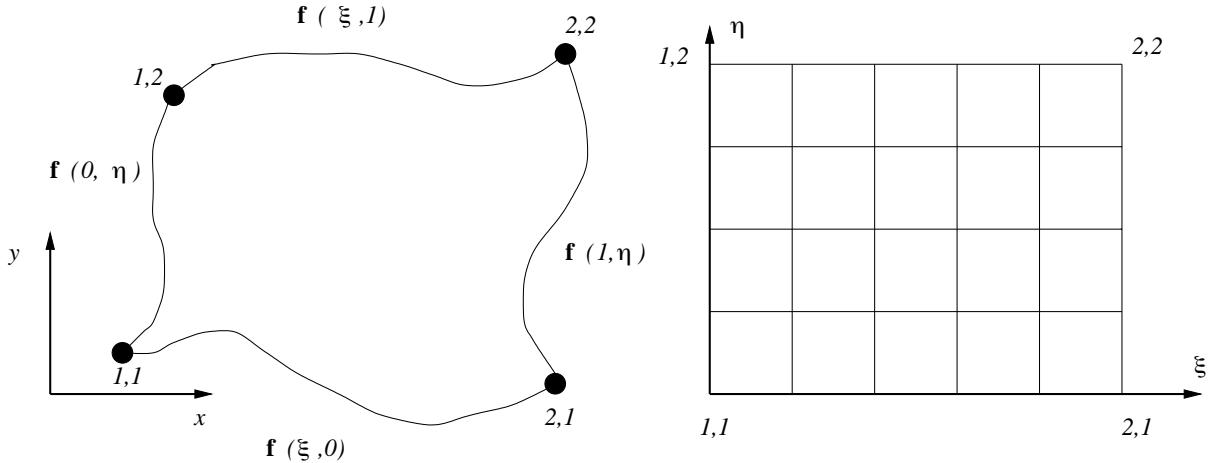


Figure 8.3.1: Mapping of a simply connected region (left) onto a rectangular computational domain (right).

### 8.3.1 Mesh Generation by Coordinate Mapping

Scientists and engineers have used coordinate mappings for some time to transform complicated domains into simpler ones. The mappings can employ either analytical or piecewise polynomial functions. The procedure usually starts with relations

$$x = f_1(\xi, \eta), \quad y = f_2(\xi, \eta)$$

that define the domain in physical  $(x, y)$  space to its image in the simpler  $(\xi, \eta)$  space. A simply connected region and its computational counterpart appear in Figure 8.3.1. It's convenient to introduce the vectors

$$\mathbf{x}^T = [x, y], \quad \mathbf{f}(\xi, \eta)^T = [f_1(\xi, \eta), f_2(\xi, \eta)] \quad (8.3.1a)$$

and write the coordinate transformation as

$$\mathbf{x} = \mathbf{f}(\xi, \eta) \quad (8.3.1b)$$

In Figure 8.3.1, we have shown a region with four segments  $\mathbf{f}(\xi, 0)$ ,  $\mathbf{f}(\xi, 1)$ ,  $\mathbf{f}(0, \eta)$ , and  $\mathbf{f}(1, \eta)$  that are related to the computational lines  $\xi = 0$ ,  $\xi = 1$ ,  $\eta = 0$ , and  $\eta = 1$ , respectively. (The four curved segments may involve different functions, but we have written them all as  $\mathbf{f}$  for simplicity.)

Let us also consider the projections

$$\mathbf{P}_\xi(\mathbf{f}) = \bar{\psi}_1(\xi)\mathbf{f}(0, \eta) + \bar{\psi}_2(\xi)\mathbf{f}(1, \eta), \quad (8.3.2a)$$

$$\mathbf{P}_\eta(\mathbf{f}) = \bar{\psi}_1(\eta)\mathbf{f}(\xi, 0) + \bar{\psi}_2(\eta)\mathbf{f}(\xi, 1), \quad (8.3.2b)$$

where

$$\bar{\psi}_1(\xi) = 1 - \xi, \quad \bar{\psi}_2(\xi) = \xi. \quad (8.3.2c)$$

With  $\xi$  ranging on  $[0, 1]$ , the mapping  $\mathbf{x} = \mathbf{P}_\xi(\mathbf{f})$  transforms the left and right edges of the domain correctly, but ignores the top and bottom while the mapping  $\mathbf{x} = \mathbf{P}_\eta(\mathbf{f})$  transforms the top and bottom boundaries correctly but not the sides (Figure 8.3.2). Coordinate lines of constant  $\xi$  and  $\eta$  are mapped as either curves or straight lines on the physical domain as indicated in Figure 8.3.2.

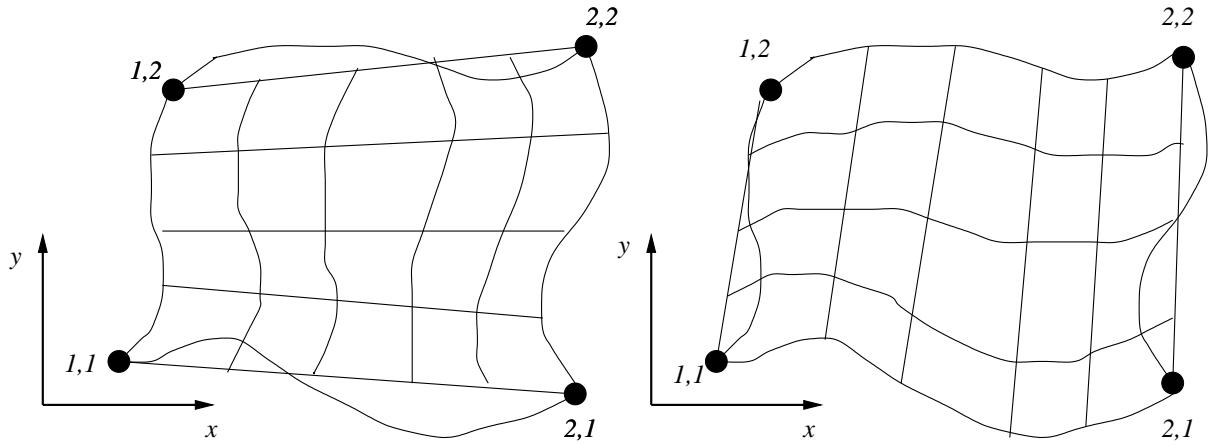


Figure 8.3.2: The transformations  $\mathbf{x} = \mathbf{P}_\xi(\mathbf{f})$  (left) and  $\mathbf{x} = \mathbf{P}_\eta(\mathbf{f})$  (right) as applied to the simply-connected domain shown in Figure 8.3.1.

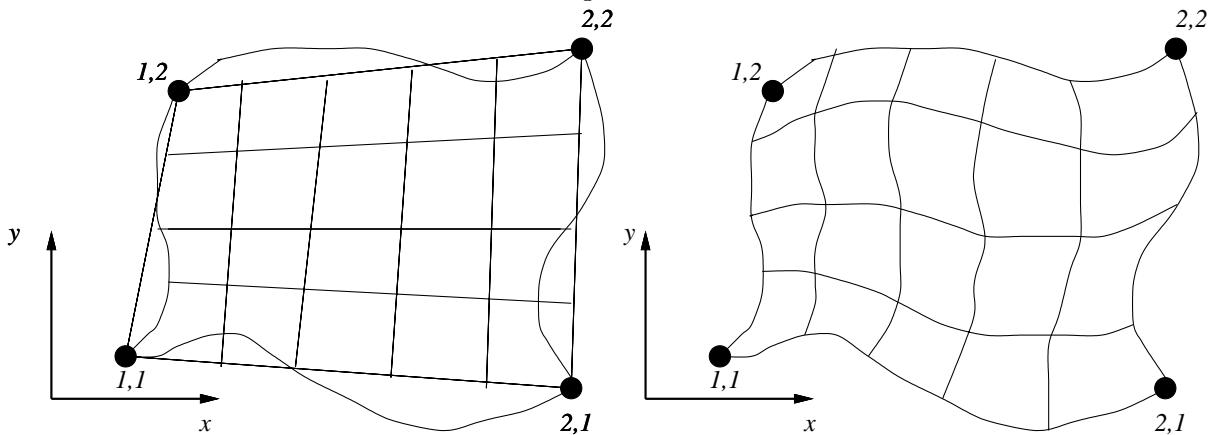


Figure 8.3.3: Illustrations of the transformations  $\mathbf{x} = \mathbf{P}_\xi \mathbf{P}_\eta(\mathbf{f})$  (left) and  $\mathbf{x} = \mathbf{P}_\xi^c plus \mathbf{P}_\eta(\mathbf{f})$  (right) as applied to the simply-connected domain shown in Figure 8.3.1.

With a goal of constructing an effective mapping, let us also examine the tensor

product and Boolean sums of the projections (8.3.2); thus,

$$\mathbf{P}_\xi \mathbf{P}_\eta(\mathbf{f}) = \sum_{i=1}^2 \sum_{j=1}^2 \bar{\psi}_i(\xi) \bar{\psi}_j(\eta) \mathbf{f}(i, j) \quad (8.3.3a)$$

$$\mathbf{P}_\xi \oplus \mathbf{P}_\eta = \mathbf{P}_\xi(\mathbf{f}) + \mathbf{P}_\eta(\mathbf{f}) - \mathbf{P}_\xi \mathbf{P}_\eta(\mathbf{f}). \quad (8.3.3b)$$

An application of these transformations to a simply-connected domain is shown in Figure 8.3.3. The tensor product (8.3.3a) is a bilinear function of  $x$  and  $y$ . As shown, this can be used to cancel the errors in (8.3.2) to reveal the Boolean sum (8.3.3b) as the desired mapping of the simply connected domain onto the computational plane. Lines of constant  $\xi$  and  $\eta$  in the computational  $(\xi, \eta)$ -plane become curves in the physical  $(x, y)$ -plane (figure 8.3.3).

Although these transformations are quite simple, they have been used to generate grids on complex two- and three-dimensional regions. Two examples involving the flow about an airfoil are shown in Figure 8.3.4. With the transformation shown at the top of the figure, the entire surface of the airfoil is mapped to  $\eta = 0$  (2-3). A cut is made from the trailing edge of the airfoil and the curve so defined is mapped to the left ( $\xi = 0$ , 2-1) and right ( $\xi = 0$ , 3-4) edges of the computational domain. The entire far field is mapped to the top ( $\eta = 1$ , 1-4) of the computational domain. Lines of constant  $\xi$  are rays from the airfoil surface to the far field boundary in the physical plane. Lines of constant  $\eta$  are closed curves encircling the airfoil. Meshes constructed in this manner are called “O-grids.” On the bottom of Figure 8.3.4, the surface of the airfoil is mapped to a portion (2-3) of the  $\xi$  axis. The cut from the trailing edge is mapped to the rest (1-2 and 3-4) of the axis. The (right) outflow boundary is mapped to the left (1-5) and right (4-6) edges of the computational domain, and the top, left, and bottom far field boundaries are mapped to the top ( $\eta = 1$ , 5-6) of the computational domain. Lines of constant  $\xi$  become curves beginning and ending at the outflow boundary and surrounding the airfoil. Lines of constant  $\eta$  are rays from the airfoil surface or the cut to the outer boundary. This mesh is called a “C-grid.”

With the mapping complete, the differential equation is transformed to the computational  $(\xi, \eta)$ -plane where it is solved on a uniform grid. Let us illustrate the procedure for

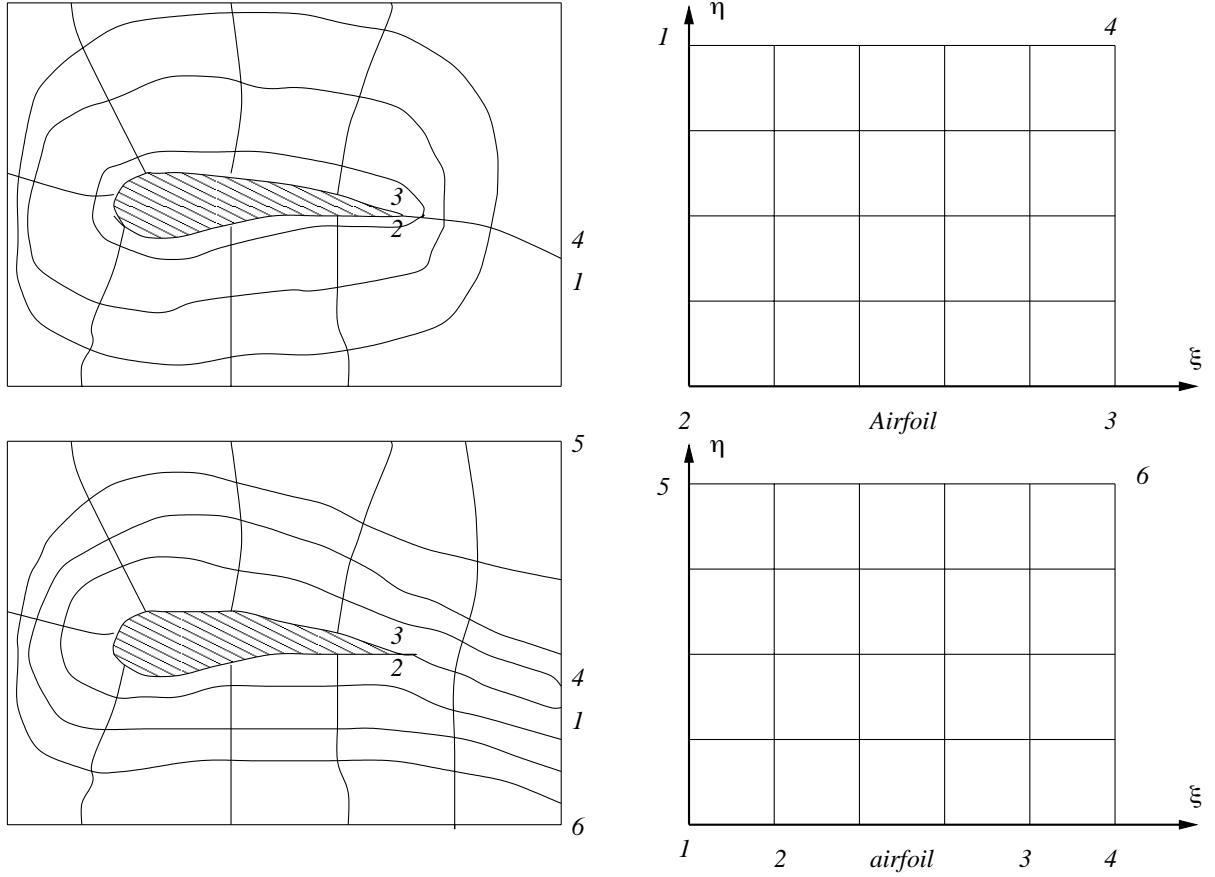


Figure 8.3.4: “O-grid” (top) and “C-grid” (bottom) mappings of the flow about an airfoil.

the model self-adjoint differential equation (8.2.2). Returning to the scalar form of the coordinate transformations, we write (8.3.3b) in the generic form  $x = x(\xi, \eta)$ ,  $y = y(\xi, \eta)$ . Then

$$( )_x = ( )_\xi \xi_x + ( )_\eta \eta_x, \quad ( )_y = ( )_\xi \xi_y + ( )_\eta \eta_y. \quad (8.3.4)$$

Using (8.3.4) in (8.2.2)

$$\begin{aligned} -\xi_x [\hat{\sigma}(u_\xi \xi_x + u_\eta \eta_x)]_\xi - \eta_x [\hat{\sigma}(u_\xi \xi_x + u_\eta \eta_x)]_\eta \\ -\xi_y [\hat{\sigma}(u_\xi \xi_y + u_\eta \eta_y)]_\xi - \eta_y [\hat{\sigma}(u_\xi \xi_y + u_\eta \eta_y)]_\eta = \hat{f} \end{aligned} \quad (8.3.5)$$

where  $\hat{\sigma} = \sigma(x(\xi, \eta), y(\xi, \eta))$  and  $\hat{f} = f(x(\xi, \eta), y(\xi, \eta))$ . Although (8.3.5) may be re-written in several forms, this one is as good to discretize as any. The *metrics* of the transformation  $\xi_x$ ,  $\eta_x$ ,  $\xi_y$ , and  $\eta_y$  are obtained by using (8.3.4) with  $x$  and  $y$  inserted in

the parentheses; thus,

$$1 = x_\xi \xi_x + x_\eta \eta_x, \quad 0 = y_\xi \xi_x + y_\eta \eta_x,$$

$$0 = x_\xi \xi_y + x_\eta \eta_y, \quad 1 = y_\xi \xi_y + y_\eta \eta_y.$$

or

$$\mathbf{J} \begin{bmatrix} \xi_x & \xi_y \\ \eta_x & \eta_y \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \mathbf{J} = \begin{bmatrix} x_\xi & x_\eta \\ y_\xi & y_\eta \end{bmatrix}. \quad (8.3.6)$$

The elements of the Jacobian are obtained by differentiating the transformation (8.3.3b).

### 8.3.2 Unstructured Mesh Generation

There are several approaches to unstructured mesh generation. Early attempts used manual techniques where point-coordinates were explicitly defined. Semi-automatic generation techniques required generating a coarse mesh which could be uniformly refined by dividing each element edge into  $K$  segments and connecting segments on opposite sides of an element to create  $K^2$  (triangular) elements. More automatic procedures use advancing fronts [5], point insertion, and recursive bisection. We'll discuss the latter procedure because it was pioneered and developed at Rensselaer.

With recursive bisection [2], a two-dimensional region  $\Omega$  is embedded in a square “universe” that is recursively quartered to create a set of disjoint squares called *quadrants*. Quadrants are related through a hierarchical *quadtree* structure. The original square universe is regarded as the root of the tree and smaller quadrants created by subdivision are regarded as offspring of larger ones. Quadrants intersecting  $\partial\Omega$  are recursively quartered until a prescribed spatial resolution of  $\Omega$  is obtained. At this stage, quadrants that are leaf nodes of the tree and intersect  $\Omega \cup \partial\Omega$  are further divided into small sets of triangular or quadrilateral elements. Severe mesh gradation is avoided by imposing a maximal one-level difference between quadrants sharing a common edge. This implies a maximal two-level difference between quadrants sharing a common vertex.

A simple example involving a domain consisting of a rectangle and a region within a curved arc, as shown in Figure 8.3.5, will illustrate the process. In the upper left portion of the figure, the square universe containing the problem domain is quartered creating the

one-level tree structure shown at the upper right. The quadrant containing the curved arc is quartered and the resulting quadrant that intersects the circular arc is quartered again to create the three-level tree shown in the lower right portion of the figure. A triangular mesh generated for this tree structure is also shown. Quadrants and a mixed triangular- and quadrilateral-element mesh for a more complex example are shown in Figure 8.3.6.

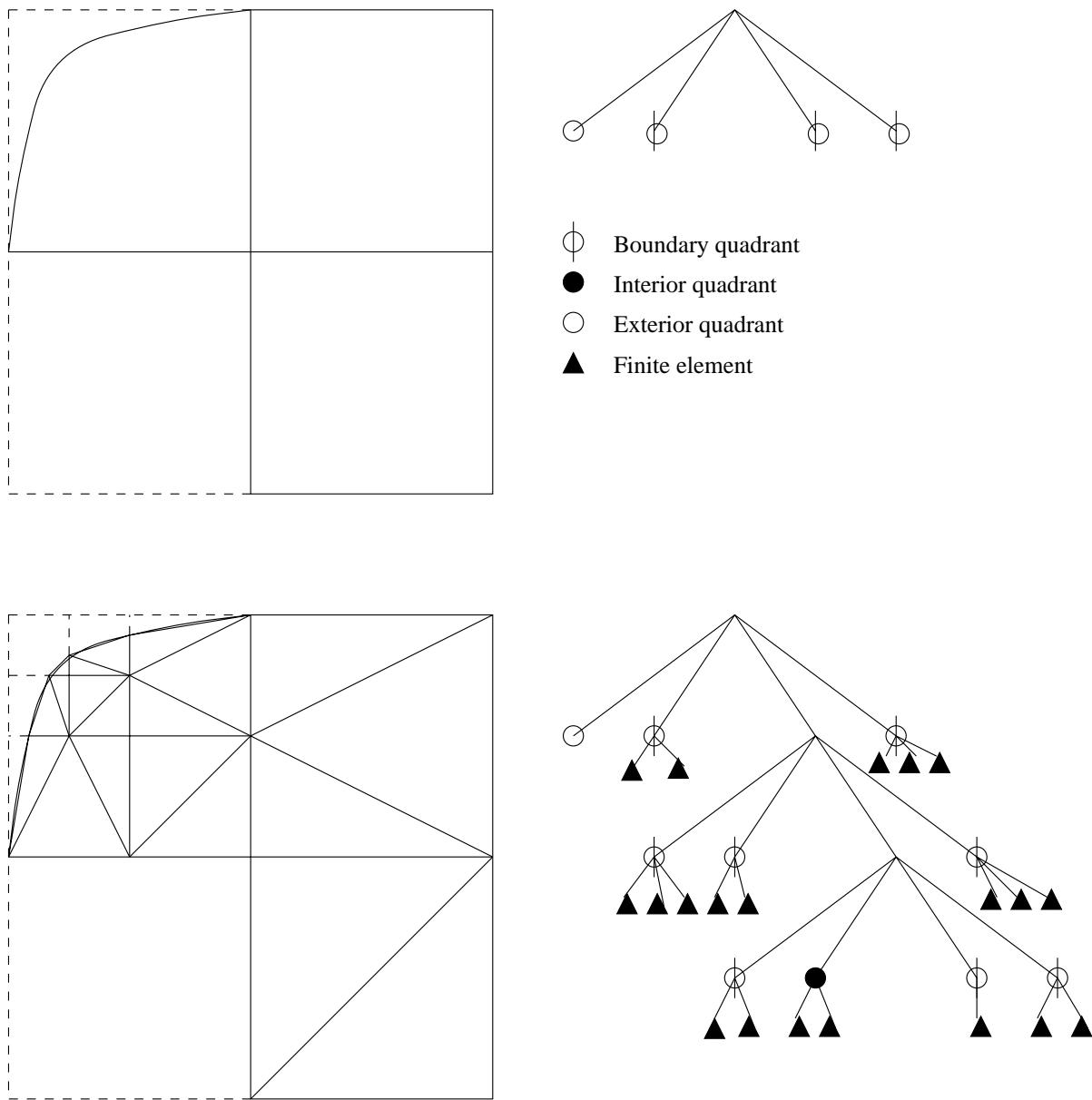


Figure 8.3.5: Finite quadtree mesh generation for a domain consisting of a rectangle and a quarter circle. One-level and three-level tree structures and their associated meshes of triangular elements are shown at the top and bottom of the figure, respectively.

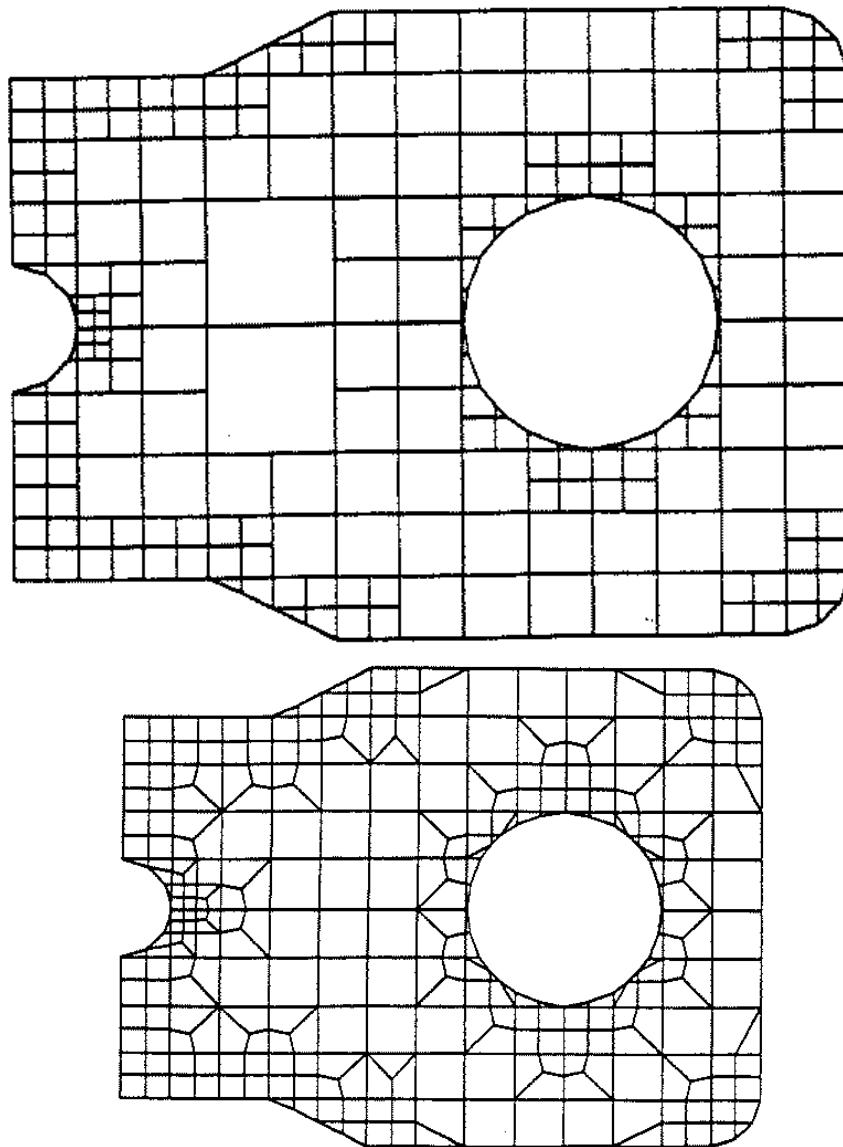


Figure 8.3.6: Quadtree structure and mixed triangular- and quadrilateral-element mesh generated from it.

The quadtree procedure may be extended to three dimensions where an octree decomposition is invoked. Elements produced by the quadtree and octree techniques may have poor geometric shapes near boundaries. A final “smoothing” of the mesh improves element shapes and further reduces mesh gradation near  $\partial\Omega$ . Element vertices on  $\partial\Omega$  are moved along the boundary to provide a better approximation to it. Pairs of boundary vertices that are too close to each other may be *collapsed* to a single vertex. Interior vertices are smoothed by a *Laplacian* operation that places each vertex at the “centroid”

of its neighboring vertices.

Arbitrarily complex two- and three-dimensional domains may be discretized by the quadtree and octree processes to produce unstructured grids. Further solution-based mesh refinement may be done by subdividing appropriate terminal quadrants or octants and generating a new mesh locally. This unites mesh generation and adaptive mesh refinement by a common tree data structure. The underlying tree structure is also suitable for load balancing on a parallel computer [4].

Unstructured mesh computation requires a data structure to store the geometric information. Perhaps, the minimal storage requirements include the vertex coordinates and the element connectivity. A sample mesh containing seven triangular elements with eight vertices appears in Figure 8.3.7. Element indices are shown in parentheses. The element-vertex and the vertex-coordinate information for this mesh are shown in Table 8.3.1. In establishing these tables, we assumed that the elemental data is traversed in a particular order, *i.e.*, we stored element vertices in a counterclockwise order (Table 8.3.1, left) beginning with an arbitrary initial vertex. We also need to know the edges of elements that are on  $\partial\Omega$  in order to apply boundary conditions. This can be done by creating another table and adopting a convention. Thus, suppose that the edge between the first and second vertices of an element is labeled as Edge 1, that between the second and third vertices is Edge 2, and that between the third and first vertices is Edge 3. A boundary data table could then indicate those element edges that coincide with  $\partial\Omega$ . Such a table is shown on the right of Table 8.3.1 for the mesh of Figure 8.3.7. The first row of the table identifies Edge 1 of Element 1 as being on a boundary of the domain. Similarly, the second row identifies Edge 3 of Element 1 as a boundary edge, *etc.*

### Problems

1. Consider the domain

$$\{(x, y) \mid 0 \leq x \leq L, 4h(x/L)[1 - (x/L)] \leq y \leq H\}$$

and develop a mapping of the form (8.3.1). Construct the mappings  $\mathbf{P}_\xi$ ,  $\mathbf{P}_\eta$ ,  $\mathbf{P}_\xi \mathbf{P}_\eta$ , and  $\mathbf{P}_\xi \oplus \mathbf{P}_\eta$  for the region and show several lines of constant  $\xi$  and constant  $\eta$ .

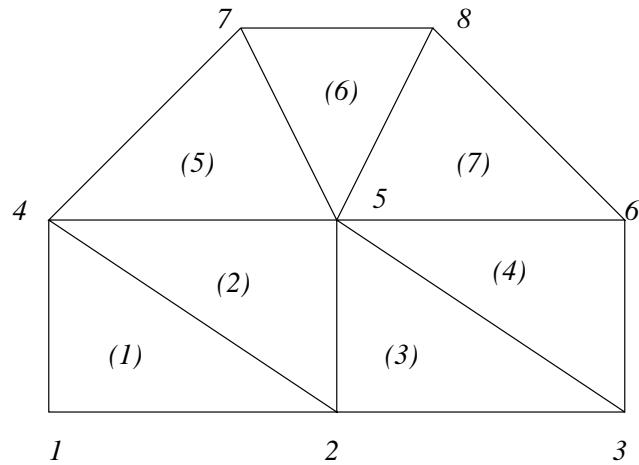


Figure 8.3.7: Sample mesh involving seven triangular elements having eight vertices.

Element	Vertices
1	1 2 4
2	2 5 4
3	2 3 5
4	5 3 6
5	4 5 7
6	7 5 8
7	5 6 8

Vertex	Coordinates	
1	0.00	0.00
2	1.50	0.00
3	3.00	0.00
4	0.00	1.00
5	1.50	1.00
6	3.00	1.00
7	1.00	2.00
8	2.00	2.00

Element	Edge
1	1
1	3
3	1
4	2
5	2
6	3
7	2

Table 8.3.1: Element-vertex data (left), vertex-coordinate data (center), and boundary data (right) for the mesh of Figure 7.



# Bibliography

- [1] I. Babuška, J.E. Flaherty, W.D. Henshaw, J.E. Hopcroft, J.E. Oliger, and T. Tezduyar, editors. *Modeling, Mesh Generation, and Adaptive Numerical Methods for Partial Differential Equations*, volume 75 of *The IMA Volumes in Mathematics and its Applications*, New York, 1995. Springer-Verlag.
- [2] P.L. Baehmann, S.L. Wittchen, M.S. Shephard, K.R. Grice, and M.A. Yerry. Robust geometrically based automatic two-dimensional mesh generation. *International Journal of Numerical Methods in Engineering*, 24:1043–1078, 1995.
- [3] M.W. Bern, J.E. Flaherty, and M. Luskin, editors. *Grid Generation and Adaptive Algorithms*, volume 113 of *The IMA Volumes in Mathematics and its Applications*, New York, 1999. Springer-Verlag.
- [4] J.E. Flaherty, R.M. Loy, C. Özturan, M.S. Shephard, B.K. Szymanski, J.D. Teresco, and L.H. Ziantz. Parallel structures and dynamic load balancing for adaptive finite element computation. *Applied Numerical Mathematics*, 26:241–265, 1998.
- [5] R. Löhner. Finite element methods in CFD: Grid generation, adaptivity and parallelization. In H. Deconinck and T. Barth, editors, *Unstructured Grid Methods for Advection Dominated Flows*, Neuilly sur Seine, 1992. AGARD Report AGARD-R-787.
- [6] A.R. Mitchell and D.F. Griffiths. *The Finite Difference Method in Partial Differential Equations*. John Wiley and Sons, Chichester, 1980.
- [7] J.C. Strikwerda. *Finite Difference Schemes and Partial Differential Equations*. Chapman and Hall, Pacific Grove, 1989.

# Chapter 9

## Solution Techniques for Elliptic Problems

### 9.1 Direct Solution Methods

In Sections 8.1, 2, we saw that the discretization of an elliptic partial differential equation led to the solution of a large, sparse, linear algebraic system. In this chapter, we address the solution of such systems. A slight change in notation will simplify the presentation. Thus, our goal is to solve linear algebraic systems of the form

$$\mathbf{A}\mathbf{x} = \mathbf{b}, \quad (9.1.1)$$

where  $\mathbf{A}$  is a large, sparse, and typically positive-definite  $N \times N$  matrix. In this section, we study *direct* techniques where the solution of (9.1.1) is found after a finite number of algebraic operations. In subsequent sections, we'll consider *iterative* techniques where the solution of (9.1.1) will only be found as the number of steps becomes infinite.

Direct solution utilize Gaussian elimination and its many variants. Pivoting is unnecessary with positive-definite systems and we shall assume that this is the case here. Gaussian elimination is regarded as a factorization of  $\mathbf{A}$  into the product

$$\mathbf{A} = \mathbf{L}\mathbf{U} = \begin{bmatrix} 1 & & & \\ l_{21} & 1 & & \\ l_{31} & l_{32} & & \\ \vdots & \vdots & \ddots & \\ l_{N1} & l_{N2} & \cdots & 1 \end{bmatrix} \begin{bmatrix} u_{11} & u_{12} & \cdots & u_{1N} \\ u_{22} & \cdots & u_{2N} \\ \ddots & & \vdots \\ & & u_{NN} \end{bmatrix}. \quad (9.1.2)$$

The matrix  $\mathbf{L}$  is lower triangular and  $\mathbf{U}$  is upper triangular. Zero elements above the

diagonal of  $\mathbf{L}$  and below the diagonal of  $\mathbf{U}$  are not shown. Expanding (9.1.2)

$$u_{ij} = a_{ij} - \sum_{k=1}^{i-1} l_{ik} u_{kj}, \quad j = i, i+1, \dots, N, \quad (9.1.3a)$$

$$l_{ji} = \frac{1}{u_{ii}} (a_{ji} - \sum_{k=1}^{i-1} l_{jk} u_{ki}), \quad j = i+1, i+2, \dots, N, \quad i = 1, 2, \dots, N. \quad (9.1.3b)$$

The summations involved in (9.1.3) are understood to be zero if the lower limit exceeds the upper one.

Once  $\mathbf{L}$  and  $\mathbf{U}$  have been determined, (9.1.1) may be solved by forward and backward substitution; thus, we have

$$\mathbf{Ax} = \mathbf{LUx} = \mathbf{b}.$$

Let

$$\mathbf{Ux} = \mathbf{y} \quad (9.1.4a)$$

then

$$\mathbf{Ly} = \mathbf{b}. \quad (9.1.4b)$$

Expressed in scalar form, the forward substitution (9.1.4b) and the backward substitution (9.1.4a) are

$$y_i = b_i - \sum_{k=1}^{i-1} l_{ik} y_k, \quad i = 1, 2, \dots, N, \quad (9.1.5a)$$

$$x_i = \frac{1}{u_{ii}} (y_i - \sum_{k=i+1}^N u_{ik} x_k), \quad i = N, N-1, \dots, 1. \quad (9.1.5b)$$

The procedure fails if  $u_{ii} = 0$ ,  $i = 1, 2, \dots, N$ . Pivoting may be necessary if this should occur.

The above procedures ignore sparsity in  $\mathbf{A}$  which, as we'll show, is not practical. As a first step towards this end, let us consider the banded structure of  $\mathbf{A}$ .

**Definition 9.1.1.** A matrix  $\mathbf{A}$  is called a band matrix of bandwidth  $p + q + 1$  if  $a_{ij} = 0$  for  $j > i + p$  and  $i > j + q$ .

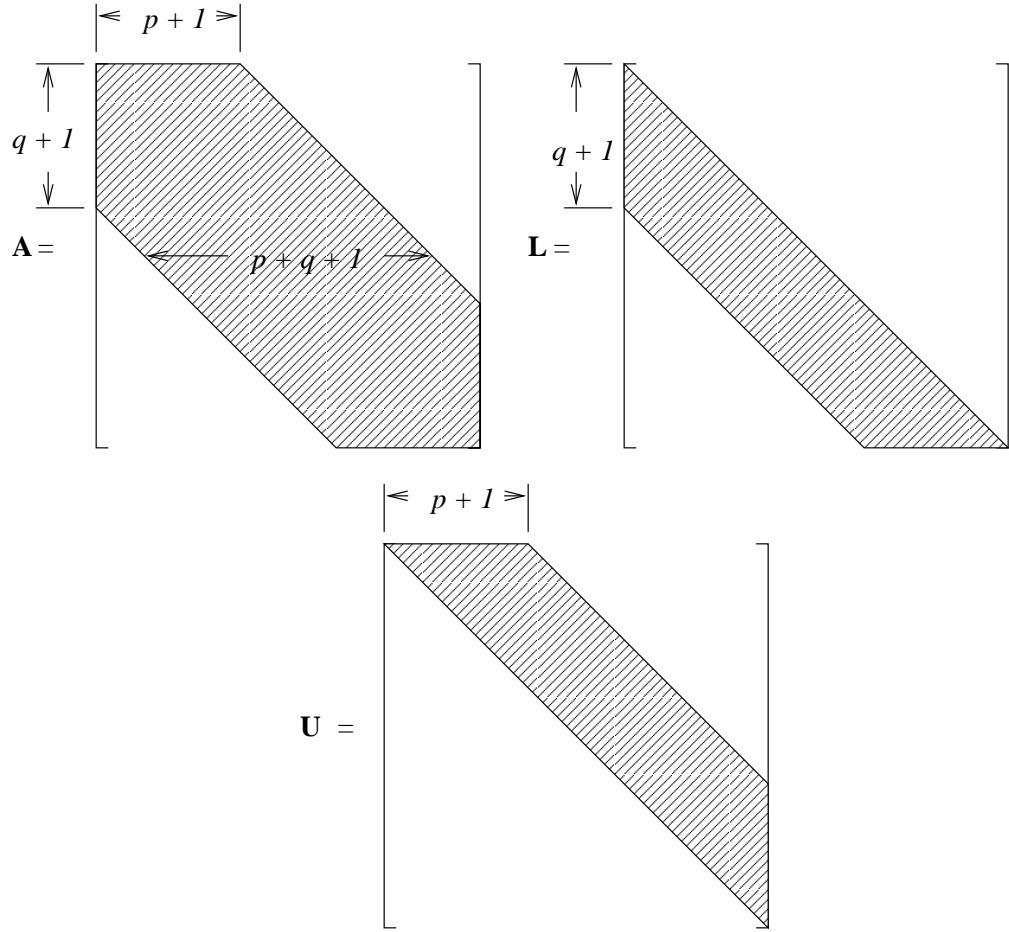


Figure 9.1.1: Structure of a band matrix  $\mathbf{A}$  of bandwidth  $p + q + 1$  (left) and of its lower and upper triangular factors  $\mathbf{L}$  (right) and  $\mathbf{U}$  (bottom), respectively. Elements not in the shaded regions are zero.

When using Gaussian elimination with a band matrix, it is easily shown that the factors  $\mathbf{L}$  and  $\mathbf{U}$ , respectively, have the structures of the lower and upper portions of  $\mathbf{A}$  (Figure 9.1.1); thus,  $l_{ij} = 0$  for  $i > j + q$  and  $j > i$  and  $u_{ij} = 0$  for  $i > j$  and  $j > i + p$ .

For a band matrix with  $q = p$ , the factorization (9.1.3) and forward and backward substitution (9.1.5) phases of Gaussian elimination become

$$u_{ij} = a_{ij} - \sum_{k=\max(1,j-p)}^{i-1} l_{ik} u_{kj}, \quad j = i, i+1, \dots, i+p, \quad (9.1.6a)$$

$$l_{ji} = \frac{1}{u_{ii}} (a_{ji} - \sum_{k=\max(1,j-p)}^{i-1} l_{jk} u_{ki}), \quad j = i+1, i+2, \dots, i+p, \quad i = 1, 2, \dots, N. \quad (9.1.6b)$$

$$y_i = b_i - \sum_{k=\max(1,i-p)}^{i-1} l_{ik} y_k, \quad i = 1, 2, \dots, N, \quad (9.1.7a)$$

$$x_i = \frac{1}{u_{ii}} (y_i - \sum_{k=i+1}^{\min(N,i+p)} u_{ik} y_k), \quad i = N, N-1, \dots, 1. \quad (9.1.7b)$$

The algorithm (9.1.6,9.1.7) ignores embedded zeros within the band. Accounting for these is typically not necessary with a direct solution method since, as will be discussed, they become nonzero during the elimination process.

The reduction in time and space complexity is significant when  $p \ll N$ . Approximate operation counts for the factorization and forward and backward substitution phases of the full and banded procedures are given in Table 9.1.1. In order to provide some meaning to these estimates, consider the solution of a Dirichlet problem for Laplace's equation on a square. With  $\Delta x = \Delta y$  the resulting algebraic system (8.1.4) has the form shown in Figure 9.1.2. Using (8.1.2), reveals that the diagonal elements of  $\mathbf{A}$  are unity and all off-diagonal terms are  $-1/4$ . Each block of  $\mathbf{A}$  is  $(J-1) \times (J-1)$  and there are  $J-1$  blocks. Thus,  $N = (J-1)^2$  and  $p = J-1$ . Using this data with the estimates shown in Table 9.1.1 gives the approximate operation counts reported on the right of Table 9.1.1. Even modest values of  $J$  indicate the impracticality of ignoring the sparsity present in  $\mathbf{A}$ .

	Full	Banded		Full	Banded
Factor	$N^3/3$	$Np(p+1)$	Factor	$J^6/3$	$J^4$
Solve	$N^2$	$N(2p+1)$	Solve	$J^4$	$2J^3$

Table 9.1.1: Approximate operation counts for solving full and banded  $N \times N$  linear systems having a bandwidth of  $2p + 1$  by Gaussian elimination (left). Approximate operation counts when solving a Dirichlet problem for Laplace's equation on a  $J \times J$  square mesh (right) which has  $N = (J-1)^2$  and  $p = J-1$ .

The factorization of  $\mathbf{A}$  creates nonzero entries within the band. Hence, for the Laplacian operator, the storage needed for  $\mathbf{L}$  and  $\mathbf{U}$  using banded Gaussian elimination is approximately  $2J^3$ , while the nonzero entries of  $\mathbf{A}$  only require  $5J^2$  memory locations.

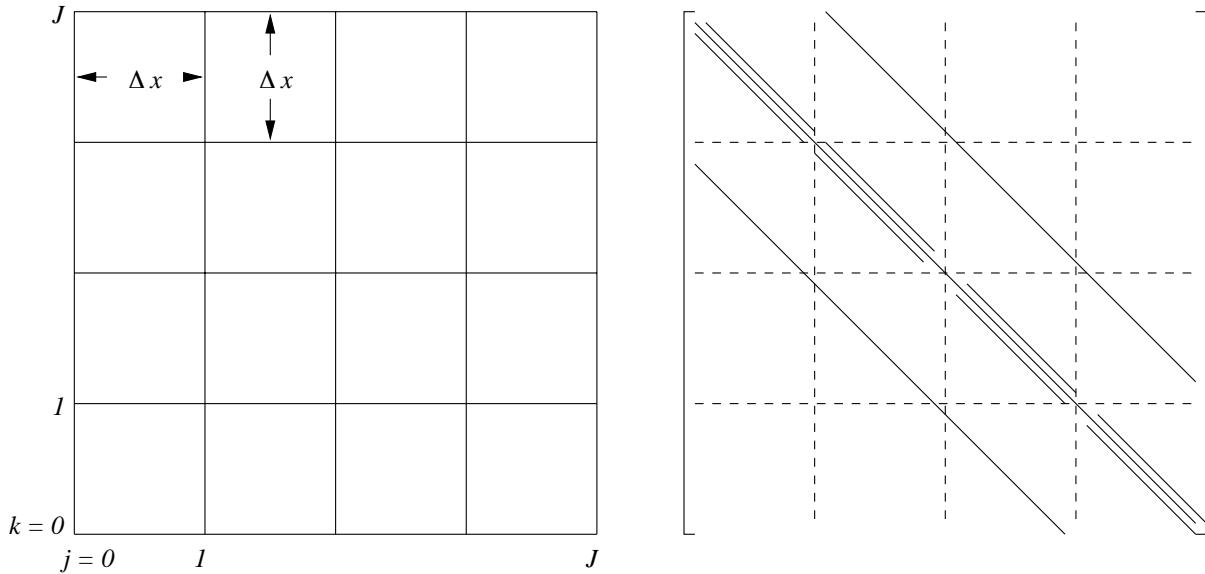


Figure 9.1.2: Uniform square mesh (left) and structure of the corresponding matrix  $\mathbf{A}$  for the solution of Laplace's equation using centered finite differences (8.1.2).

We will have to use iterative methods in order to take full advantage of the sparsity in  $\mathbf{A}$ . Nevertheless, some additional time and space savings are possible. For example, a symmetric, positive definite matrix  $\mathbf{A}$  may be factored as

$$\mathbf{A} = \mathbf{LDL}^T \quad (9.1.8a)$$

where

$$\mathbf{L} = \begin{bmatrix} 1 & & & \\ l_{21} & 1 & & \\ l_{31} & l_{32} & & \\ \vdots & \vdots & \ddots & \\ l_{N1} & l_{N2} & \cdots & 1 \end{bmatrix}, \quad \mathbf{D} = \begin{bmatrix} d_1 & & & \\ & d_2 & & \\ & & \ddots & \\ & & & d_N \end{bmatrix}. \quad (9.1.8b)$$

Computing the product in (9.1.8a) using (9.1.8b) yields

$$d_i = a_{ii} - \sum_{k=1}^{i-1} d_k l_{ik}^2, \quad (9.1.9a)$$

$$l_{ji} = \frac{1}{d_i} (a_{ji} - \sum_{k=1}^{i-1} d_k l_{jk} l_{ik}), \quad j = i+1, i+2, \dots, N, \quad i = 1, 2, \dots, N. \quad (9.1.9b)$$

The solution phase follows by substituting (9.1.8a) into (9.1.1) to get

$$\mathbf{Ax} = \mathbf{LDL}^T \mathbf{x} = \mathbf{b}.$$

Letting

$$\mathbf{L}^T \mathbf{x} = \mathbf{y}, \quad \mathbf{D}\mathbf{y} = \mathbf{z}, \quad \mathbf{L}\mathbf{z} = \mathbf{b}. \quad (9.1.10)$$

The solution is obtained after forward, diagonal, and backward substitution steps, which have the scalar form

$$z_i = b_i - \sum_{k=1}^{i-1} l_{ik} z_k, \quad i = 1, 2, \dots, N, \quad (9.1.11a)$$

$$y_i = z_i/d_i, \quad i = 1, 2, \dots, N, \quad (9.1.11b)$$

$$x_i = y_i - \sum_{k=i+1}^N l_{ki} x_k, \quad i = N, N-1, \dots, 1. \quad (9.1.11c)$$

Banded versions of the factorization and solution steps can also be developed.

The block tridiagonal algorithm [3] exploits the fact that  $\mathbf{A}$  has a tridiagonal structure with entries that are matrices (*cf.* (8.1.4)). This too has (approximately) the same number of operations as banded Gaussian elimination (9.1.6, 9.1.7). A different ordering of the equations and unknowns, however, can significantly reduce fill-in of the band and, hence, the order of operations. Nested dissection, developed by George [1, 2], is known to be optimal in certain situations. The dissection process is illustrated for a  $4 \times 4$  mesh in Figure 9.1.3. Alternate unknowns are eliminated first to create the coarser mesh of “macro elements” shown at the right of Figure 9.1.3. Midside nodes of these macro elements are eliminated next to leave, in this case, a single unknown at the center of the domain (bottom). Although we will not describe how to do the dissection for a more general mesh and problem, one can visualize the process and essential idea.

The structure of the linear systems obtained by using the row-by-row and nested-dissection orderings are shown in Figures 9.1.4 and 9.1.5, respectively, for the  $4 \times 4$  mesh of Figure 9.1.3. The matrix  $\mathbf{A}$  obtained by the nested-dissection ordering has a larger bandwidth than the one with the row-by-row ordering. However, even with this simple  $4 \times 4$  problem, we can see that the fill in is less with the nested dissection ordering than with the row-by-row ordering (Figures 9.1.4 and 9.1.5).

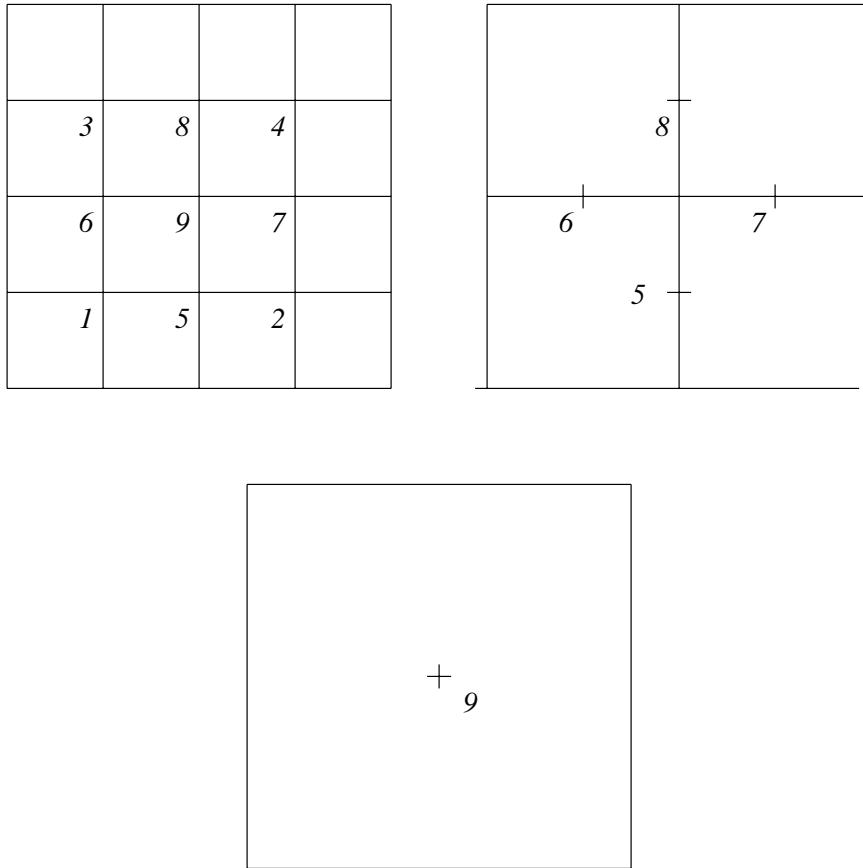


Figure 9.1.3: Nested dissection of a uniform  $4 \times 4$  mesh. Unknowns at the finest level (left) are eliminated first to create the coarser mesh (center). These unknowns are eliminated next to leave a single unknown (bottom).

A banded structure is not necessary for the efficient implementation of a Gaussian elimination procedure. It is just about as simple to implement a “skyline” or “profile” elimination strategy where the local bandwidth is used. This requires an additional vector indicating, *e.g.*, the number of leading zeros in a row or column. Let

$$l_{ij} = 0, \quad 0 \leq j < m_i^* \quad (9.1.12a)$$

and

$$k^* = \max(m_i^*, m_j^*) \quad (9.1.12b)$$

then the skyline form of the symmetric Cholesky decomposition (9.1.9) is

$$d_i = a_{ii} - \sum_{k=k^*}^{i-1} d_k l_{ik}^2, \quad (9.1.13a)$$

$$\left[ \begin{array}{cccccc} x & x & & x & & & \\ x & x & x & & x & & \\ & x & x & & & x & \\ x & & x & x & & x & \\ x & & x & x & x & & x \\ & x & & x & x & & \\ & x & & & x & x & \\ & & x & & x & x & \\ & & & x & x & x & x \\ & & & & x & x & x \end{array} \right] \quad \left[ \begin{array}{cccccc} x & x & & x & & & \\ & x & x & x & x & x & \\ & & x & x & x & x & x \\ & & & x & x & x & x \\ & & & & x & x & x \\ & & & & & x & x \\ & & & & & & x \\ & & & & & & & x \end{array} \right]$$

Figure 9.1.4: Matrix **A** when the finite difference equations for a  $4 \times 4$  mesh are ordered by rows (left) and the resulting fill-in of **U** (right).

$$\left[ \begin{array}{cccccc} x & & x & x & & \\ & x & & x & x & \\ & & x & & x & x \\ & & & x & x & x \\ & x & x & & & x \\ & x & x & & x & x \\ & & x & & x & x \\ & & & x & x & x \\ & & & & x & x \\ & & & & & x \end{array} \right] \quad \left[ \begin{array}{cccccc} x & & x & x & & \\ & x & & x & x & \\ & & x & & x & x \\ & & & x & x & x \\ & & & & x & x \\ & & & & & x \end{array} \right]$$

Figure 9.1.5: Matrix  $\mathbf{A}$  when the finite difference equations have a nested dissection ordering (left) and the resulting fill-in of  $\mathbf{U}$  (right).

$$l_{ji} = \frac{1}{d_i} (a_{ji} - \sum_{k=k^*}^{i-1} d_k l_{jk} l_{ik}), \quad j = m_i^* + 1, i+2, \dots, N, \quad i = 1, 2, \dots, N. \quad (9.1.13b)$$

This procedure ignores any zeros within the profile of  $\mathbf{L}$ . George [1] proved that the profile algorithm (9.1.12, 9.1.13) with the nested dissection ordering could solve a Dirichlet problem for Poisson's equation on a  $J \times J$  square mesh in  $O(J^3)$  operations with  $O(J^2 \log_2 J)$  storage. These should be compared to the  $O(J^4)$  operations and  $O(J^3)$  storage required for the banded algorithm of Table 9.1.1. George [1] additionally showed that the nested ordering is optimal in the sense that all orderings of the mesh must yield an operation count of at least  $O(J^3)$ .

## 9.2 Basic Iterative Solution Methods

The direct methods of Section 9.1 require storage within the band or profile which could be significant for very large problems (*e.g.*, in excess of 10,000 equations). Storage and, perhaps, computer time can be reduced through the use of iterative techniques. As in Section 9.1, we'll focus on techniques for  $N \times N$  linear system having the form (9.1.1). Using a “fixed-point” strategy, we rewrite (9.1.1) in the form

$$\mathbf{x} = \mathbf{M}\mathbf{x} + \hat{\mathbf{b}} \quad (9.2.1)$$

and consider the iteration

$$\mathbf{x}^{(\nu+1)} = \mathbf{M}\mathbf{x}^{(\nu)} + \hat{\mathbf{b}}, \quad \nu = 0, 1, \dots \quad (9.2.2)$$

The iteration must be designed so that

$$\lim_{\nu \rightarrow \infty} \mathbf{x}^{(\nu)} = \mathbf{x}.$$

*Example 9.2.1.* Write

$$\mathbf{A} = \mathbf{A} + \mathbf{I} - \mathbf{I},$$

where  $\mathbf{I}$  is the  $N \times N$  identity matrix, and rewrite (9.1.1) as

$$\mathbf{I}\mathbf{x} = (\mathbf{I} - \mathbf{A})\mathbf{x} + \mathbf{b}.$$

This system has the form of (9.2.1) with

$$\mathbf{M} = \mathbf{I} - \mathbf{A}, \quad \hat{\mathbf{b}} = \mathbf{b}.$$

Here are two fundamental convergence criteria.

**Theorem 9.2.1.** *The iteration (9.2.2) converges to a fixed point  $\mathbf{x}$  of (9.2.1) when*

$$\|\mathbf{M}\| < 1. \quad (9.2.3)$$

*Proof.* Define

$$\mathbf{e}^{(\nu)} = \mathbf{x}^{(\nu)} - \mathbf{x} \quad (9.2.4)$$

and subtract (9.2.1) from (9.2.2) to obtain

$$\mathbf{e}^{(\nu+1)} = \mathbf{M}\mathbf{e}^{(\nu)}.$$

Thus,

$$\mathbf{e}^{(\nu+1)} = \mathbf{M}\mathbf{e}^{(\nu)} = \mathbf{M}^2\mathbf{e}^{(\nu-1)} = \dots = \mathbf{M}^{\nu+1}\mathbf{e}^{(0)}$$

or

$$\mathbf{e}^{(\nu)} = \mathbf{M}^\nu \mathbf{e}^{(0)}.$$

Taking a norm

$$\|\mathbf{e}^{(\nu)}\| = \|\mathbf{M}^\nu \mathbf{e}^{(0)}\| \leq \|\mathbf{M}^\nu\| \|\mathbf{e}^{(0)}\| \leq \|\mathbf{M}\|^\nu \|\mathbf{e}^{(0)}\|. \quad (9.2.5)$$

When  $\|\mathbf{M}\| < 1$ , we see that  $\|\mathbf{e}^{(\nu)}\| \rightarrow 0$  as  $\nu \rightarrow \infty$  and, hence, the iteration converges.  $\square$

**Theorem 9.2.2.** *The iteration (9.2.2) converges from any initial guess if and only if the spectral radius*

$$\rho(\mathbf{M}) \equiv \max_{1 \leq i \leq N} |\lambda_i(\mathbf{M})| < 1 \quad (9.2.6)$$

where  $\lambda_i$ ,  $i = 1, 2, \dots, N$ , are eigenvalues of the  $N \times N$  matrix  $\mathbf{M}$ .

*Proof.* From Lemma 3.3.1 we know

$$\rho^\nu(\mathbf{M}) = \rho(\mathbf{M}^\nu) \leq \|\mathbf{M}^\nu\|.$$

If the iteration (9.2.2) converges from any initial guess  $\mathbf{e}^{(0)}$ , then the results of Theorem 9.2.1 imply  $\|\mathbf{M}^\nu\| \rightarrow 0$  as  $\nu \rightarrow \infty$ ; hence,  $\rho(\mathbf{M}) < 1$ .

Proving that (9.2.2) converges when  $\rho(\mathbf{M}) < 1$  is slightly more involved. We'll establish the result when  $\mathbf{M}$  is diagonalizable. Isaacson and Keller [3], Section 1.1, establish the result under more general conditions.

If  $\mathbf{M}$  is diagonalizable, then there is a matrix  $\mathbf{P}$  such that

$$\mathbf{P}\mathbf{M}\mathbf{P}^{-1} = \mathbf{\Lambda}$$

where  $\mathbf{\Lambda}$  is a diagonal matrix. Now,

$$\mathbf{M}^\nu = (\mathbf{P}^{-1}\mathbf{\Lambda}\mathbf{P})(\mathbf{P}^{-1}\mathbf{\Lambda}\mathbf{P}) \dots (\mathbf{P}^{-1}\mathbf{\Lambda}\mathbf{P}) = \mathbf{P}^{-1}\mathbf{\Lambda}^\nu\mathbf{P}.$$

If  $|\lambda_i| < 1$ ,  $i = 1, 2, \dots, N$ , then

$$\lim_{\nu \rightarrow \infty} \|\mathbf{P}^{-1} \mathbf{\Lambda}^\nu \mathbf{P}\| = 0$$

and the iteration (9.2.2) converges.  $\square$

Theorems 9.2.1 and 9.2.2 prescribe convergence conditions. We also want an indication of the convergence rate of the iteration. Many measures are possible and we'll settle on the following.

**Definition 9.2.1.** *The average convergence rate of the iteration (9.2.2) is*

$$R_\nu(\mathbf{M}) \equiv -\frac{\ln \|\mathbf{M}^\nu\|}{\nu}. \quad (9.2.7a)$$

Using (9.2.5)

$$\|\mathbf{e}^{(\nu)}\| \leq \|\mathbf{M}^\nu\| \|\mathbf{e}^{(0)}\| = e^{-\nu R_\nu} \|\mathbf{e}^{(0)}\|.$$

Thus, convergence is fast when  $R_\nu$  is large or, equivalently, when  $\|\mathbf{M}\|$  is small. Additionally, since  $\rho(\mathbf{M}^\nu) \leq \|\mathbf{M}^\nu\|$  and  $\|\mathbf{M}^\nu\| \leq 1$  for a converging iteration,

$$R_\nu(\mathbf{M}) \leq -\ln \rho(\mathbf{M}). \quad (9.2.7b)$$

Thus, we may take  $-\ln \rho(\mathbf{M})$  as a measure of the convergence rate. Although the spectral radius is more difficult to compute than a matrix norm, this rate is independent of  $\nu$  and the particular norm.

Many iterative procedures partition  $\mathbf{A}$  as

$$\mathbf{A} = \mathbf{D} - \mathbf{L} - \mathbf{U} \quad (9.2.8a)$$

where

$$\mathbf{D} = \begin{bmatrix} a_{11} & & & \\ & a_{22} & & \\ & & \ddots & \\ & & & a_{NN} \end{bmatrix}, \quad \mathbf{L} = \begin{bmatrix} 0 & & & \\ -a_{21} & 0 & & \\ \vdots & \vdots & \ddots & \\ -a_{N1} & -a_{N2} & \cdots & 0 \end{bmatrix}, \quad (9.2.8b)$$

$$\mathbf{U} = \begin{bmatrix} 0 & -a_{12} & \cdots & -a_{1N} \\ 0 & \cdots & -a_{2N} \\ \ddots & & \vdots \\ & & 0 \end{bmatrix}. \quad (9.2.8c)$$

### 9.2.1 Jacobi and Gauss-Seidel Iteration

Three classical iterative methods have the splitting defined by (9.2.8). With the *Jacobi method*, we solve for the diagonal terms of (9.1.1). Thus, using (9.2.8a), we write (9.1.1) in the form

$$(\mathbf{D} - \mathbf{L} - \mathbf{U})\mathbf{x} = \mathbf{b} \quad (9.2.9)$$

and consider the iteration

$$\mathbf{x}^{(\nu+1)} = \mathbf{D}^{-1}(\mathbf{L} + \mathbf{U})\mathbf{x}^{(\nu)} + \mathbf{D}^{-1}\mathbf{b} \quad (9.2.10a)$$

which has the form of (9.2.2) with

$$\mathbf{M}_J = \mathbf{D}^{-1}(\mathbf{L} + \mathbf{U}), \quad \hat{\mathbf{b}}_J = \mathbf{D}^{-1}\mathbf{b}. \quad (9.2.10b)$$

The scalar form (9.2.10) is

$$x_i^{(\nu+1)} = - \sum_{j=1, j \neq i}^N \frac{a_{ij}}{a_{ii}} x_j^{(\nu)} + \frac{b_i}{a_{ii}}, \quad i = 1, 2, \dots, N. \quad (9.2.11)$$

*Example 9.2.2.* The Jacobi method for the Poisson equation (8.1.2a) is

$$\begin{aligned} U_{jk}^{(\nu+1)} &= \theta_x(U_{j-1,k}^{(\nu)} + U_{j+1,k}^{(\nu)}) + \theta_y(U_{j,k-1}^{(\nu)} + U_{j,k+1}^{(\nu)}) + \theta_{xy} f_{jk}, \\ j &= 1, 2, \dots, J-1, \quad k = 1, 2, \dots, K-1. \end{aligned} \quad (9.2.12)$$

Updates to the solution at  $(j, k)$  are computed as a weighted average of solutions at its four neighboring points. Contrary to solutions obtained by direct methods, parallel computational techniques are easily used with Jacobi's method since the solution state at iteration  $\nu + 1$  is explicit.

It's easy to show that Jacobi iteration converges when  $\mathbf{A}$  satisfies some rather restrictive properties.

**Definition 9.2.2.** A matrix  $\mathbf{A}$  is strictly diagonally dominant if

$$\sum_{j=1, j \neq i}^N |a_{ij}| < |a_{ii}|, \quad i = 1, 2, \dots, N. \quad (9.2.13)$$

**Theorem 9.2.3.** *The Jacobi iteration converges in the maximum norm when  $\mathbf{A}$  is strictly diagonally dominant.*

*Proof.* If  $\mathbf{A}$  is strictly diagonally dominant, we may use (9.2.8) and (9.2.10) to obtain

$$\|\mathbf{M}_J\|_\infty = \max_{1 \leq i \leq N} \sum_{j=1, j \neq i}^N \frac{|a_{ij}|}{|a_{ii}|} < 1.$$

□

Convergence of Jacobi's method is too slow for practical serial computation, although it may be used for parallel computation. Gauss-Seidel iteration uses the latest solution information as soon as it becomes available. Thus, when computing  $x_i^{(\nu+1)}$  according to (9.2.11), we could use the latest iterates  $x_j^{(\nu+1)}$ ,  $j = 1, 2, \dots, i-1$ , on the right to obtain

$$x_i^{(\nu+1)} = - \sum_{j=1}^{i-1} \frac{a_{ij}}{a_{ii}} x_j^{(\nu+1)} - \sum_{j=i}^N \frac{a_{ij}}{a_{ii}} x_j^{(\nu)} + \frac{b_i}{a_{ii}}, \quad i = 1, 2, \dots, N. \quad (9.2.14)$$

In the matrix form of (9.2.9), this is equivalent to

$$(\mathbf{D} - \mathbf{L})\mathbf{x}^{(\nu+1)} = \mathbf{U}\mathbf{x}^{(\nu)} + \mathbf{b} \quad (9.2.15a)$$

which has the form of (9.2.2) with

$$\mathbf{M}_{GS} = (\mathbf{D} - \mathbf{L})^{-1}\mathbf{U}, \quad \hat{\mathbf{b}}_{GS} = (\mathbf{D} - \mathbf{L})^{-1}\mathbf{b}. \quad (9.2.15b)$$

*Example 9.2.3.* The Gauss-Seidel iteration for the Poisson equation (8.1.2a) is

$$\begin{aligned} U_{jk}^{(\nu+1)} &= \theta_x(U_{j-1,k}^{(\nu+1)} + U_{j+1,k}^{(\nu)}) + \theta_y(U_{j,k-1}^{(\nu+1)} + U_{j,k+1}^{(\nu)}) + \theta_{xy}f_{jk}, \\ j &= 1, 2, \dots, J-1, \quad k = 1, 2, \dots, K-1. \end{aligned} \quad (9.2.16)$$

The solution process depends on the order in which the equations are written. As described above and as shown in Figure 9.2.1, row ordering has been assumed.

*Example 9.2.4.* Consider the boundary value problem for Laplace's equation on a unit square  $\Omega$

$$\Delta u = 0, \quad (x, y) \in \Omega,$$

$$u(x, y) = \begin{cases} 0, & \text{if } x = 0, y = 1 \\ 1, & \text{if } x = 1, y = 0 \end{cases}, \quad (x, y) \in \partial\Omega.$$

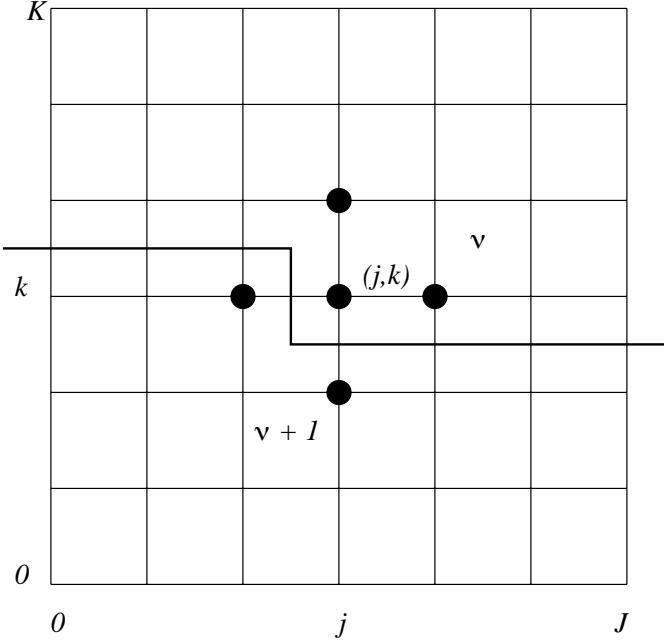


Figure 9.2.1: Gauss-Seidel iteration with row ordering.

Let us solve this problem on a  $3 \times 3$  mesh using Jacobi and Gauss-Seidel iteration with  $\Delta x = \Delta y = 1/3$ ; hence, using (8.1.2) with  $\theta_x = \theta_y = 1/4$  and  $f_{jk} = 0$ , we have

$$U_{jk} = \frac{1}{4}(U_{j+1,k} + U_{j-1,k} + U_{j,k+1} + U_{j,k-1}), \quad j, k = 1, 2.$$

The Jacobi iteration is

$$U_{jk}^{(\nu+1)} = \frac{1}{4}(U_{j+1,k}^{(\nu)} + U_{j-1,k}^{(\nu)} + U_{j,k+1}^{(\nu)} + U_{j,k-1}^{(\nu)}), \quad j, k = 1, 2, \quad \nu = 0, 1, \dots.$$

Starting with the trivial initial guess  $U_{jk}^{(0)} = 0$ ,  $j, k = 1, 2$ , we present solutions after the one and five iterations in Table 9.2.1. The exact solution and differences between the exact and Jacobi solutions after five iterations are shown in Table 9.2.2.

The Gauss-Seidel method for this problem is

$$U_{jk}^{(\nu+1)} = \frac{1}{4}(U_{j+1,k}^{(\nu)} + U_{j-1,k}^{(\nu+1)} + U_{j,k+1}^{(\nu)} + U_{j,k-1}^{(\nu+1)}), \quad \nu = 0, 1, \dots.$$

Its solution after five iterations is shown in Table 9.2.3.

The maximum error after five Jacobi iterations is 0.01656 and after five Gauss-Seidel iterations is 0.00146. Thus, as expected, Gauss-Seidel iteration is converging faster than Jacobi iteration

0	0	0	0/1	0	0	0	0/1
0	0	0.25	1	0	0.23438	0.48344	1
0	0.25	0.5	1	0	0.48344	0.73438	1
0/1	1	1	1	0/1	1	1	1

Table 9.2.1: Solution of Example 9.2.4 after one iteration ( $\nu = 0$ , left) and after five iterations ( $\nu = 4$ , right) using Jacobi's method.

0	0	0	0/1	0	0	0	0
0	0.25	0.5	1	0	0.01562	0.01656	0
0	0.5	0.75	1	0	0.01656	0.01562	0
0/1	1	1	1	0	0	0	0

Table 9.2.2: Exact solution of Example 9.2.4 (left) and the errors in the Jacobi solution after five iterations (right).

0	0	0	0/1	0	0	0	0
0	0.24927	0.49963	1	0	0.00073	0.00037	0
0	0.49854	0.74927	1	0	0.00146	0.00073	0
0/1	1	1	1	0	0	0	0

Table 9.2.3: Solution of Example 4 after five iterations ( $\nu = 4$ ) using the Gauss-Seidel method (left) and errors in this solution (right).

*Example 9.2.5.* We'll try to quantify the differences in the convergence rates of Jacobi and Gauss-Seidel iteration for Poisson's equation on a rectangle. Jacobi's method satisfies (9.2.12) and we let  $\mathbf{p}$  be an eigenvector of  $\mathbf{M}_J$  with corresponding eigenvalue  $\mu$ ; thus,

$$\mathbf{M}_J \mathbf{p} = \mu \mathbf{p}. \quad (9.2.17a)$$

Using (9.2.12), we see that the component form of this relation is

$$\begin{aligned} \mu p_{jk} &= \theta_x(p_{j-1,k} + p_{j+1,k}) + \theta_y(p_{j,k-1} + p_{j,k+1}), & j &= 1, 2, \dots, J-1, \\ k &= 1, 2, \dots, K-1, \end{aligned} \quad (9.2.17b)$$

where  $p_{jk}$  is a component of  $\mathbf{p}$ . (The double subscript notation for a vector component is non-standard, but convenient in this case since it corresponds to a position in the finite difference mesh.) One may easily verify that

$$p_{jk} = \sin \frac{mj\pi}{J} \sin \frac{nk\pi}{K} \quad (9.2.17c)$$

and

$$\mu = 1 - 4\theta_x \sin^2 \frac{m\pi}{2J} - 4\theta_y \sin^2 \frac{n\pi}{2K}, \quad m = 1, 2, \dots, J-1, \quad n = 1, 2, \dots, K-1. \quad (9.2.17d)$$

*Remark 1.* The  $(J-1)(K-1)$  eigenvectors and eigenvalues of  $\mathbf{M}_J$  are indexed by  $m$  and  $n$ .

*Remark 2.* The eigenvector  $\mathbf{p}$  is the eigenfunction of the Laplacian sampled at the mesh points  $j = 1, 2, \dots, J-1, k = 1, 2, \dots, K-1$ . The eigenvalue  $\mu$  is, however, not an eigenvalue of the Laplacian.

The largest eigenvalue and, hence, the spectral radius of  $\mathbf{M}_J$  may be obtained by setting  $m = n = 1$  in (9.2.17d) to obtain

$$\rho(\mathbf{M}_J) = 1 - 4\theta_x \sin^2 \frac{\pi}{2J} - 4\theta_y \sin^2 \frac{\pi}{2K}. \quad (9.2.17e)$$

In the special case of a square,  $\theta_x = \theta_y = 1/4$  and  $J = K$ ; thus,

$$\rho(\mathbf{M}_J) = 1 - 2 \sin^2 \frac{\pi}{2J}.$$

For large values of  $J$ , we may approximate this as

$$\rho(\mathbf{M}_J) \approx 1 - \frac{\pi^2}{2J^2} = 1 - C\Delta x^2,$$

since  $J = a/\Delta x$  for an  $a \times a$  square region. Thus, the spectral radius approaches unity and the convergence rate slows as  $J$  increases (or as  $\Delta x$  decreases).

In a similar manner, Let  $\mathbf{q}$  and  $\lambda$  be an eigenvector-eigenvalue pair of the Gauss-Seidel iteration matrix  $\mathbf{M}_{GS}$  for Poisson's equation on a rectangle (9.2.16). Thus, using (9.2.15b)

$$\mathbf{M}_{GS}\mathbf{q} = (\mathbf{D} - \mathbf{L})^{-1}\mathbf{U}\mathbf{q} = \lambda\mathbf{q}, \quad (9.2.18a)$$

or, using (9.2.15b), in component form

$$\begin{aligned} \lambda q_{jk} - \lambda(\theta_x q_{j-1,k} + \theta_y q_{j,k-1}) &= (\theta_x q_{j+1,k} + \theta_y q_{j,k+1}), \quad j = 1, 2, \dots, J-1, \\ k &= 1, 2, \dots, K-1. \end{aligned} \quad (9.2.18b)$$

This problem appears more difficult to analyze than the eigenvalue problem (9.2.17b) for the Jacobi method; however, there is a transformation that simplifies things considerably. Let

$$q_{jk} = \lambda^{(j+k)/2} r_{jk}$$

and substitute this relationship into (9.2.18b) to obtain

$$\lambda^{(j+k+2)/2} r_{jk} - \lambda^{(j+k+1)/2} (\theta_x r_{j-1,k} + \theta_y r_{j,k-1}) = \lambda^{(j+k+1)/2} (\theta_x r_{j+1,k} + \theta_y r_{j,k+1}).$$

Dividing by the common factor yields

$$\lambda^{1/2} r_{jk} = \theta_x (r_{j-1,k} + r_{j+1,k}) + \theta_y (r_{j,k-1} + r_{j,k+1}).$$

This is the same eigenvalue problem as (9.2.17b) for Jacobi's method with  $\mu$  replaced by  $\lambda^{1/2}$ ; thus, using (9.2.17d)

$$\begin{aligned} \lambda = \mu^2 = [1 - 4\theta_x \sin^2 \frac{m\pi}{2J} - 4\theta_y \sin^2 \frac{n\pi}{2K}]^2, \quad m &= 1, 2, \dots, J-1, \\ n &= 1, 2, \dots, K-1. \end{aligned} \quad (9.2.18c)$$

In particular,

$$\rho(\mathbf{M}_{GS}) = \rho^2(\mathbf{M}_J). \quad (9.2.18d)$$

Thus, according to (9.2.7b), Gauss-Seidel iterations converge twice as fast as Jacobi iterations.

In the special case of Laplace's equation on a square mesh with large  $J$  we obtain the asymptotic approximation

$$\rho(\mathbf{M}_{GS}) \approx 1 - \frac{\pi^2}{J^2}. \quad (9.2.19)$$

The results of Example 9.2.5 generalize as indicated by the following theorem.

**Theorem 9.2.4.** *Suppose that  $\mathbf{A}$  satisfies  $a_{ij}/a_{ii} < 0$ ,  $i \neq j$ , then one and only one of the following conditions can occur:*

1.  $\rho(\mathbf{M}_J) = \rho(\mathbf{M}_{GS}) = 0$ ,
2.  $0 < \rho(\mathbf{M}_{GS}) < \rho(\mathbf{M}_J) < 1$ ,

3.  $\rho(\mathbf{M}_J) = \rho(\mathbf{M}_{GS}) = 1$ , or

4.  $1 < \rho(\mathbf{M}_J) < \rho(\mathbf{M}_{GS})$ .

*Proof.* cf. [6], p. 70. □

In the important Case 2, Gauss-Seidel iterations always converge faster than Jacobi iterations.

### 9.2.2 Successive Over Relaxation

“Relaxation” is a procedure that can accelerate the convergence rate of virtually any iteration. At present, it suits our purposes to apply it to the Gauss-Seidel method. The process begins by using the Gauss-Seidel method (9.2.14) to compute a “provisional” iterate

$$\hat{x}_i^{(\nu+1)} = - \sum_{j=1}^{i-1} \frac{a_{ij}}{a_{ii}} x_j^{(\nu+1)} - \sum_{j=i+1}^n \frac{a_{ij}}{a_{ii}} x_j^{(\nu)} + \frac{b_i}{a_{ii}} \quad (9.2.20a)$$

and concludes with the final iterate

$$x_i^{(\nu+1)} = \omega \hat{x}_i^{(\nu+1)} + (1 - \omega) x_i^{(\nu)}, \quad i = 1, 2, \dots, N. \quad (9.2.20b)$$

The *acceleration parameter*  $\omega$  is to be chosen so that the iteration (9.2.20) converges as fast as possible. In particular, (9.2.20) is called Gauss-Seidel iteration when  $\omega = 1$ , *successive under relaxation* when  $\omega < 1$ , and *successive over relaxation (SOR)* when  $\omega > 1$ . Over relaxation is the important method with elliptic problems.

Using (9.2.8), we can write (9.2.20) in the vector form

$$\mathbf{D}\hat{\mathbf{x}}^{(\nu+1)} = \mathbf{L}x^{(\nu+1)} + \mathbf{U}x^{(\nu)} + \mathbf{b} \quad (9.2.21a)$$

$$\mathbf{x}^{(\nu+1)} = \omega \hat{\mathbf{x}}^{(\nu+1)} + (1 - \omega) \mathbf{x}^{(\nu)}. \quad (9.2.21b)$$

We can further eliminate the provisional iterate and write (9.2.21a, 9.2.21b) in the form of (9.2.2)

$$\mathbf{x}^{(\nu+1)} = \mathbf{M}_\omega \mathbf{x}^{(\nu)} + \hat{\mathbf{b}}_\omega \quad (9.2.21c)$$

with

$$\mathbf{M}_\omega = (\mathbf{D} - \omega \mathbf{L})^{-1}[(1 - \omega)\mathbf{D} + \omega \mathbf{U}], \quad \hat{\mathbf{b}}_\omega = \omega(\mathbf{D} - \omega \mathbf{L})^{-1}\mathbf{b}. \quad (9.2.21d)$$

Our goal is to find the value of  $\omega$  that minimizes  $\rho(\mathbf{M}_\omega)$  and, hence, maximizes the convergence rate. There is a wealth of theory on this subject and let us begin with some preliminary considerations.

**Definition 9.2.3.** A matrix  $\mathbf{A}$  is two cyclic if there is a permutation of its rows and columns that reduce it to the form

$$\begin{bmatrix} \mathbf{D}_1 & \mathbf{F} \\ \mathbf{G} & \mathbf{D}_2 \end{bmatrix}$$

where  $\mathbf{D}_1$  and  $\mathbf{D}_2$  are diagonal.

**Definition 9.2.4.** A matrix  $\mathbf{A}$  is weakly two cyclic if  $\mathbf{D}_1$  and  $\mathbf{D}_2$  are zero.

*Example 9.2.6.* The matrix shown in Figure 9.2.2 is two cyclic as revealed by an interchange of its second and third rows and columns.

$$\begin{bmatrix} 1 & c & 0 \\ a & 1 & c \\ 0 & a & 1 \end{bmatrix} \quad \begin{bmatrix} 1 & c & 0 \\ 0 & a & 1 \\ a & 1 & c \end{bmatrix} \quad \begin{bmatrix} 1 & 0 & c \\ 0 & 1 & a \\ a & c & 1 \end{bmatrix}$$

Figure 9.2.2: Matrix (left) whose second and third rows are interchanged (center) and whose second and third columns are interchanged to obtain a two-cyclic form (right).

*Example 9.2.7.* Consider the Laplacian operator on a  $4 \times 4$  grid as shown in Figure 9.2.3. Instead of ordering the equations and unknowns by rows, order them in “checkerboard” or “red-black” fashion by listing unknowns and equations at every other point. The resulting matrix has the two-cyclic form

$$\begin{bmatrix} \times & & & \times & \times \\ \times & & & \times & \times \\ & \times & & \times & \times & \times \\ & & \times & & \times & \times \\ & & & \times & & \times & \times \\ \times & \times & \times & & \times & & \\ \times & & \times & & & \times & \\ \times & \times & & \times & & & \times \\ & \times & \times & \times & & & \end{bmatrix}.$$

4	9	5	
7	3	8	
1	6	2	

Figure 9.2.3: Red-black ordering of the Laplacian operator on a  $4 \times 4$  square mesh

**Definition 9.2.5.** A two-cyclic matrix of the form of (9.2.8) is consistently ordered if the eigenvalues of

$$\mathbf{D}^{-1}(\alpha\mathbf{L} + \frac{1}{\alpha}\mathbf{U})$$

are independent of  $\alpha$  for all real  $\alpha \neq 0$ .

We're now ready to search for the optimal choice of  $\omega$ .

**Theorem 9.2.5.** If a matrix  $\mathbf{A}$  is consistently ordered then the eigenvalues  $\lambda$  of  $\mathbf{M}_\omega$  and  $\mu$  of  $\mathbf{M}_J$  are related by

$$\mu = \frac{\lambda + \omega - 1}{\lambda^{1/2}\omega}. \quad (9.2.22)$$

*Proof.* From (9.2.21d), we see that the eigenvalues of  $\mathbf{M}_\omega$  satisfy

$$(\mathbf{D} - \omega\mathbf{L})^{-1}[(1 - \omega)\mathbf{D} + \omega\mathbf{U}]\mathbf{q} = \lambda\mathbf{q}$$

where  $\mathbf{q}$  is the eigenvector of  $\mathbf{M}_\omega$  corresponding to  $\lambda$ . Multiplying by  $(\mathbf{D} - \omega\mathbf{L})$ , we have

$$[(1 - \omega)\mathbf{D} + \omega\mathbf{U} - \lambda(\mathbf{D} - \omega\mathbf{L})]\mathbf{q} = \mathbf{0}.$$

Multiplying by  $\mathbf{D}^{-1}/\omega$

$$[\mathbf{D}^{-1}(\mathbf{U} + \lambda\mathbf{L}) - \frac{\lambda + \omega - 1}{\omega}\mathbf{I}]\mathbf{q} = \mathbf{0}.$$

Finally, multiplying by  $\lambda^{-1/2}$  yields

$$[\mathbf{D}^{-1}(\lambda^{1/2}\mathbf{L} + \lambda^{-1/2}\mathbf{U}) - \mu\mathbf{I}]\mathbf{q} = \mathbf{0}$$

where  $\mu$  satisfies (9.2.22). Thus  $\mu$  is an eigenvalue of

$$\mathbf{D}^{-1}(\lambda^{1/2}\mathbf{L} + \lambda^{-1/2}\mathbf{U}).$$

If  $\mathbf{A}$  is consistently ordered then the eigenvalues of this matrix are independent of the parameter  $\lambda^{1/2}$ . Thus, we can choose any convenient value of  $\lambda$  to find the eigenvalues  $\mu$ . In particular, if we choose  $\lambda = 1$  then  $\mu$  is an eigenvalue of  $\mathbf{M}_J$  (*cf.* (9.2.10b)).  $\square$

*Remark 3.* Setting  $\omega = 1$  for the Gauss-Seidel method and using (9.2.22), we see that  $\mu = \lambda^{1/2}$ , confirming the relationship between the eigenvalues of  $\mathbf{M}_J$  and  $\mathbf{M}_{GS}$  that we found in Example 9.2.5 for the discrete Laplacian.

*Remark 4.* The transformation used in Example 9.2.5 could have also been used to prove this theorem for the discrete Laplacian operator.

Let us assume that the eigenvalues  $\mu$  of  $\mathbf{M}_J$  are real. (It suffices to assume that  $\mathbf{A}$  is symmetric.) Let us also assume that  $\rho(\mathbf{M}_J) < 1$ . Then, using Theorem 9.2.4,  $\rho(\mathbf{M}_{GS}) < 1$ . Let

$$f(\lambda, \omega) = \frac{\lambda + \omega - 1}{\omega}, \quad g(\lambda, \mu) = \mu \lambda^{1/2}. \quad (9.2.23)$$

We sketch  $f(\lambda, \omega)$  and  $g(\lambda, \mu)$  as functions of  $\lambda$  in Figure 9.2.4. Both halves of  $g(\lambda, \mu)$  are shown since the eigenvalues of  $\mathbf{M}_J$  occur in pairs. Thus, if  $\mu$  is an eigenvalue of  $\mathbf{M}_J$ , so is  $-\mu$ . This may be shown for Laplace's equation using the results of Example 9.2.5, but we won't do it here.

Let's list several properties of  $f(\lambda, \omega)$  and  $g(\lambda, \mu)$  that can be discerned from (9.2.23) and Figure 9.2.23.

1.  $f(1, \omega) = 1, \forall \omega$ .
2. The function  $g(\lambda, \mu)$  increases linearly with  $|\mu|$ . Its largest amplitude occurs when  $\mu = \rho(\mathbf{M}_J)$ .
3. For fixed  $\mu$  and  $\omega$ , the eigenvalues of  $\mathbf{M}_\omega$  are given by the values of  $\lambda$  where  $f(\lambda, \omega) = g(\lambda, \mu)$  (*cf.* (9.2.22)). With  $\mu = \rho(\mathbf{M}_J)$ , these eigenvalues are shown as points  $A$  and  $B$  on Figure 9.2.4. The larger value of  $\lambda$  at the points labeled  $A$  corresponds to  $\rho(\mathbf{M}_\omega)$  (Item 2).

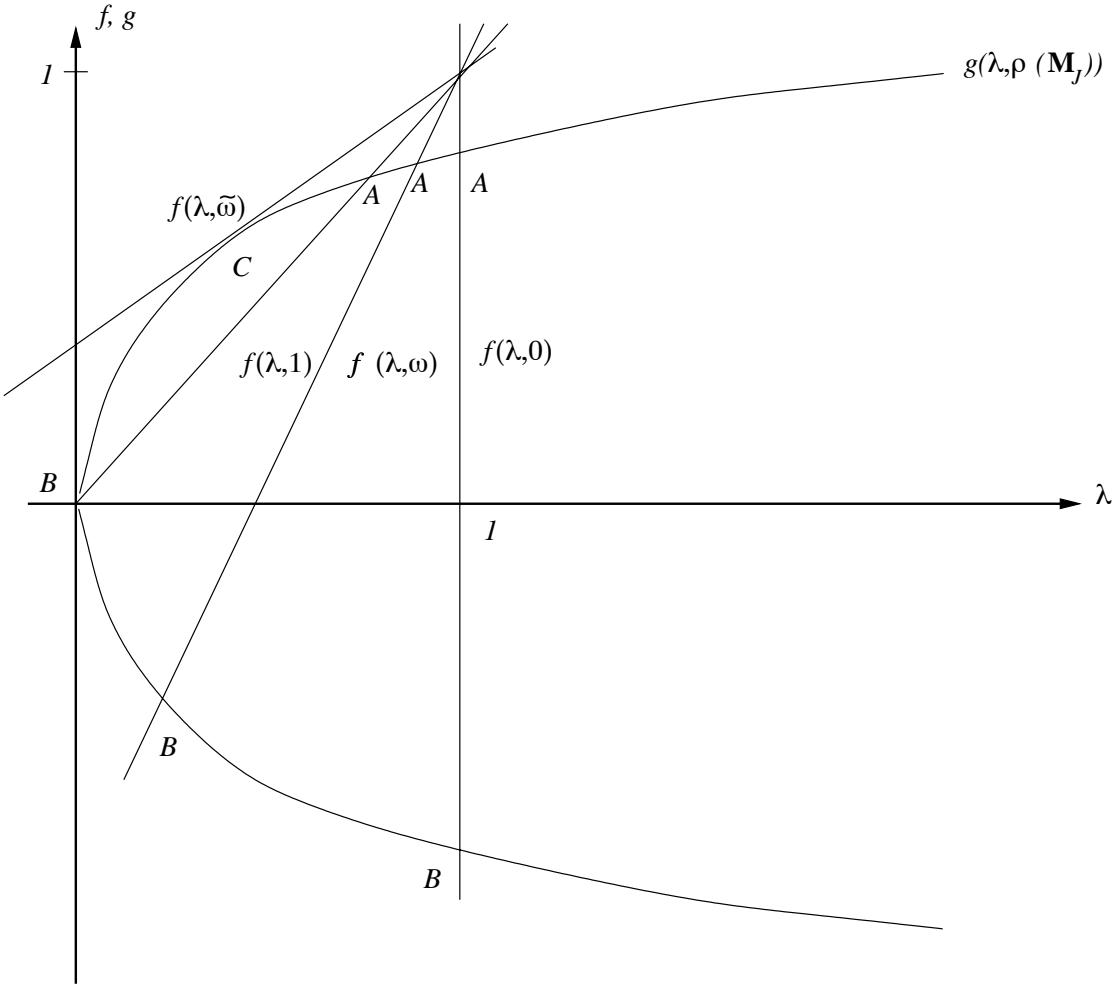


Figure 9.2.4: Functions  $f(\lambda, \omega)$  and  $g(\lambda, \rho(\mathbf{M}_J))$  vs.  $\lambda$ .

4. Setting  $\omega = 0$  gives  $f(\lambda, 0)$  as the line  $\lambda = 1$ . If  $\omega < 0$ , then the values of  $\lambda$  at the intersection points (not shown in Figure 9.2.4) would exceed unity, and the iteration (9.2.21) would diverge.
5. Setting  $\omega = 1$  in (9.2.23) gives  $f(\lambda, 1) = \lambda$ , which is the Gauss-Seidel method.
6. As seen in Figure 9.2.4, the maximum eigenvalue  $\lambda$  of  $\mathbf{M}_\omega$  can be reduced further by choosing  $\omega > 1$ . The minimum real solution for  $\lambda$  occurs at  $\omega = \tilde{\omega}$  when  $f(\lambda, \tilde{\omega})$  is tangent (at point  $C$ ) to  $g(\lambda, \rho(\mathbf{M}_J))$ .

Let us rewrite (9.2.22) as

$$\lambda - \omega \mu \lambda^{1/2} + \omega - 1 = 0.$$

This is a quadratic equation in  $\lambda^{1/2}$ ; thus,

$$\lambda = \left[ \omega\mu/2 \pm \sqrt{(\omega\mu/2)^2 + 1 - \omega} \right]^2. \quad (9.2.24a)$$

The point of tangency ( $C$ ) occurs at  $\tilde{\omega}$  when

$$(\tilde{\omega}\mu/2)^2 + 1 - \tilde{\omega} = 0$$

or

$$\tilde{\omega}_{\pm} = \frac{2}{1 \pm \sqrt{1 - \mu^2}}.$$

For  $0 \leq \mu \leq 1$ , we see that  $1 \leq \tilde{\omega}_+ \leq 2$  and  $2 \leq \tilde{\omega}_- < \infty$ . We'll show in a moment that the interesting range of  $\omega$  is  $[0, 2]$ ; thus, the appropriate value of  $\tilde{\omega}$  is  $\tilde{\omega}_+$ . We'll drop the  $+$  subscript and simply let

$$\tilde{\omega} = \frac{2}{1 + \sqrt{1 - \mu^2}}. \quad (9.2.24b)$$

Substituting (9.2.24b) into (9.2.24a) yields

$$\tilde{\lambda} = \lambda(\tilde{\omega}) = (\tilde{\omega}\mu/2)^2 = \tilde{\omega} - 1. \quad (9.2.24c)$$

Values of  $\omega < \tilde{\omega}$  produce real values of  $\lambda$  while values of  $\omega > \tilde{\omega}$  yield complex values of  $\lambda$ . In this latter case, we may use (9.2.24a) to show that

$$|\lambda| = \omega - 1, \quad \omega > \tilde{\omega}.$$

We're now able to state our main result.

**Theorem 9.2.6.** *Let  $\mathbf{A}$  be a symmetric and consistently-ordered matrix, then the optimal relaxation parameter satisfies*

$$\omega_{OPT} = \frac{2}{1 + \sqrt{1 - \rho^2(\mathbf{M}_J)}} \quad (9.2.25a)$$

and

$$\rho(\mathbf{M}_\omega) = \begin{cases} \left[ \omega\rho(\mathbf{M}_J)/2 + \sqrt{(\omega\rho(\mathbf{M}_J)/2)^2 + 1 - \omega} \right]^2, & \text{if } 0 \leq \omega \leq \omega_{OPT} \\ \omega - 1, & \text{if } \omega_{OPT} < \omega \end{cases}. \quad (9.2.25b)$$

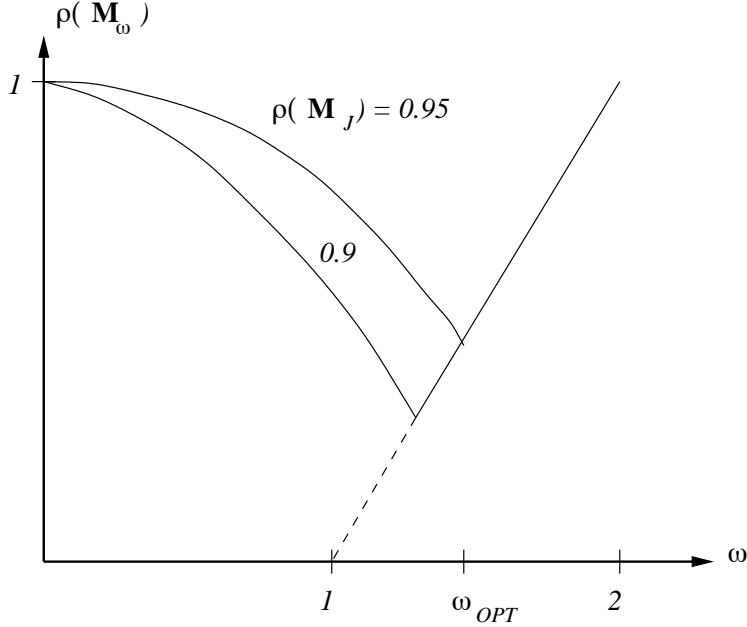


Figure 9.2.5: Spectral radius  $\rho(\mathbf{M}_\omega)$  as a function of  $\omega$  for SOR iteration.

*Proof.* With the various symmetries (*cf.* Strikwerda [5], Section 13.3), it suffices to consider positive values of  $\mu$ . With this choice and our interest in calculating the spectral radius of  $\mathbf{M}_\omega$ , we clearly want the positive sign in (9.2.24a). Thus, in summary, we have

1.  $\lambda = 1$  when  $\omega = 0$ ,
2.  $\lambda = \left[ \omega\mu/2 + \sqrt{(\omega\mu/2)^2 + 1 - \omega} \right]^2$ ,  $0 < \omega \leq \tilde{\omega}$ , and
3.  $|\lambda| = \omega - 1$ ,  $\tilde{\omega} < \omega$ .

The values of  $\lambda$  increase with  $\mu$ ; thus, the spectral radius  $\rho(\mathbf{M}_\omega)$  occurs when  $\mu = \rho(\mathbf{M}_J)$ , as given by (9.2.25b). The minimum value of  $\rho(\mathbf{M}_\omega)$  occurs at  $\tilde{\omega}$  corresponding to  $\mu = \rho(\mathbf{M}_J)$ , as given by (9.2.25a). The spectral radius  $\rho(\mathbf{M}_\omega)$  is displayed as a function of  $\omega$  in Figure 9.2.5.  $\square$

Several corollaries follow from Theorem 9.2.6 and we'll list one.

**Corollary 9.2.1.** *Under the conditions of Theorem 9.2.6, the SOR method converges for  $\omega \in (0, 2)$  when  $\rho(\mathbf{M}_J) < 1$ .*

*Proof.* From (9.2.25)

$$\rho(\mathbf{M}_{\omega_{OPT}}) = \omega_{OPT} - 1 = \frac{2}{1 + \sqrt{1 - \rho^2(\mathbf{M}_J)}} - 1.$$

Thus,  $0 < \rho(\mathbf{M}_{\omega_{OPT}}) < 1$  when  $0 < \rho(\mathbf{M}_J) < 1$ .

Additional inspection of (9.2.25b) reveals that  $d\rho(\mathbf{M}_\omega)/d\omega$  is non-positive when  $\omega < \omega_{OPT}$  and unity when  $\omega > \omega_{OPT}$  (Figure 9.2.5). Since  $\rho(\mathbf{M}_\omega) = 1$  at  $\omega = 0, 2$ , we conclude that  $\rho(\mathbf{M}_\omega) < 1$  for  $0 < \omega < 2$ .  $\square$

An examination of (9.2.25) and Figure 9.2.5 reveals that  $\omega_{OPT} \in (1, 2)$ . Increasing  $\rho(\mathbf{M}_J)$  increases  $\omega_{OPT}$  and, in particular,  $\omega_{OPT} \rightarrow 2$  as  $\rho(\mathbf{M}_J) \rightarrow 1$ . From Figure 9.2.5, we also see that overestimating  $\omega_{OPT}$  by a given amount increases  $\rho(\mathbf{M}_\omega)$  less than underestimating it by the same amount.

*Example 9.2.8.* Let us solve the problem of Example 9.2.4 using SOR iteration. With  $J = K = 3$  and  $\theta_x = \theta_y = 1/4$ , we obtain the Gauss-Seidel method (9.2.16) as

$$\hat{U}_{jk}^{(\nu+1)} = \frac{1}{4}(U_{j+1,k}^{(\nu)} + U_{j-1,k}^{(\nu+1)} + U_{j,k+1}^{(\nu)} + U_{j,k-1}^{(\nu+1)}).$$

With relaxation, we compute

$$U_{jk}^{(\nu+1)} = \omega \hat{U}_{jk}^{(\nu+1)} + (1 - \omega) U_{jk}^{(\nu)}.$$

We'll explicitly eliminate the intermediate variable to obtain the SOR method as

$$U_{jk}^{(\nu+1)} = \frac{\omega}{4}(U_{j+1,k}^{(\nu)} + U_{j-1,k}^{(\nu+1)} + U_{j,k+1}^{(\nu)} + U_{j,k-1}^{(\nu+1)}) + (1 - \omega) U_{jk}^{(\nu)}.$$

Using (9.2.17e) with  $J = K = 3$ , the spectral radius of the Jacobi method is

$$\rho(\mathbf{M}_J) = 1 - \sin^2 \frac{\pi}{6} - \sin^2 \frac{\pi}{6} = \frac{1}{2};$$

thus, using (9.2.25a)

$$\omega_{OPT} = \frac{2}{1 + \sqrt{1 - 1/4}} \approx 1.07.$$

The SOR solution and errors after five iterations are shown in Table 9.2.4. After five iterations, the percentage errors at point  $(1, 1)$  are 1.2, 0.29, and 0.014, respectively, for the Jacobi, Gauss-Seidel, and SOR methods (*cf.* Example 9.2.4).

0	0	0	0/1	0	0	0	0
0	0.24997	0.49999	1	0	0.00003	0.00001	0
0	0.49993	0.74998	1	0	0.00007	0.00002	0
0/1	1	1	1	0	0	0	0

Table 9.2.4: Solution of Example 9.2.8 after five iterations ( $\nu = 4$ ) using the SOR method with  $\omega = 1.07$  (left) and errors in this solution (right).

*Example 9.2.9.* Let us examine the convergence rate of SOR a bit more closely for Laplace's equation. Using (9.2.17e), the spectral radius of Jacobi's method on a square is

$$\rho(\mathbf{M}_J) = 1 - 2 \sin^2 \frac{\pi}{2J} = \cos \frac{\pi}{J}.$$

Using (9.2.25a)

$$\omega_{OPT} = \frac{2}{1 + \sqrt{1 - \cos^2 \pi/J}} = \frac{2}{1 + \sin \pi/J}.$$

Now, using (9.2.25b)

$$\rho(\mathbf{M}_{\omega_{OPT}}) = \omega_{OPT} - 1 = \frac{1 - \sin \pi/J}{1 + \sin \pi/J}$$

or, for large values of  $J$ ,

$$\rho(\mathbf{M}_{\omega_{OPT}}) \approx 1 - \frac{2\pi}{J}.$$

Recall (Example 9.2.5) that the spectral radius of Jacobi and Gauss-Seidel iteration under the same conditions is  $1 - O(1/J^2)$ . Thus, SOR iteration is considerably better. We'll emphasize this by computing the convergence rate according to (9.2.7b). For the Jacobi method, we find

$$R_\nu(\mathbf{M}_J) \leq -\ln \rho(\mathbf{M}_J) \approx -\ln(1 - \frac{\pi^2}{2J^2}) \approx \frac{\pi^2}{2J^2}.$$

Similarly, for the Gauss-Seidel method, we have

$$R_\nu(\mathbf{M}_{GS}) \approx \frac{\pi^2}{J^2},$$

and for SOR, we have

$$R_\nu(\mathbf{M}_{\omega_{OPT}}) \approx \frac{2\pi}{J}.$$

Thus, typically, the Jacobi or Gauss-Seidel methods would require  $O(J^2)$  iterations to obtain an answer having a specified accuracy while the SOR would obtain the same accuracy in only  $O(J)$  iterations.

The optimal relaxation parameter is not known for realistic elliptic problems because the eigenvalues of  $\mathbf{M}_J$  are typically unavailable. Strikwerda [5], Section 13.3, describes a way of calculating approximate values of  $\omega_{OPT}$ . The optimal relaxation parameter for many elliptic problems is close to 2 and may be approximated by an expression of the form

$$\omega_{OPT} = \frac{2}{1 + Ch} \quad (9.2.26)$$

where  $h$  is some measure of the grid spacing, *e.g.*,  $h = \max(\Delta x, \Delta y)$ . The value of the constant  $C$  can be determined by calculating  $\omega_{OPT}$  on some coarse grids and then extrapolating to finer grids. The values of  $\omega_{OPT}$  on a coarse grid is determined experimentally by making several computations on the grid with different values of  $\omega$ . The value that produces the fewest iterations for a given level of accuracy is assumed to be  $\omega_{OPT}$ .

Some common variations of SOR iteration follow:

1. *Red-Black (Checkerboard) ordering.* We presented this ordering in Example 9.2.7.

As usual, let a point in a rectangular mesh be denoted as  $(j, k)$ . With red-black ordering, we number all equations and unknowns at, *e.g.*, odd values of  $j + k$  before those with even values of  $j + k$  (Figure 9.2.3). Recall, that this gave us a system of the form

$$\mathbf{Ax} = \begin{bmatrix} \mathbf{D}_1 & \mathbf{F} \\ \mathbf{G} & \mathbf{D}_2 \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{bmatrix}$$

where  $\mathbf{D}_1$  and  $\mathbf{D}_2$  are diagonal and  $\mathbf{x}_1$  and  $\mathbf{x}_2$  correspond to unknowns at odd-and even-numbered points, respectively. SOR iteration is performed first on the odd points and then on the even points. Note that the updating of an unknown at each odd point is independent of that at any other odd point; hence, they may be done in parallel without a need for synchronization. Similarly, unknowns at all even points may be updated in parallel.

2. *Symmetric ordering (SSOR).* Generally, the matrix  $\mathbf{M}_\omega$  is not symmetric even when the original matrix  $\mathbf{A}$  is. There are instances when it is important to maintain symmetry, *e.g.*, when using SOR iteration as a preconditioning for the conjugate gradient method (Section 9.3). A symmetric iteration matrix can be obtained by

performing a standard SOR sweep with, say, row ordering followed by one with the reverse of this ordering.

3. *Line or block procedures.* Order the unknowns by rows, but gather all of the unknowns in a row into a vector to obtain a “block tridiagonal” system. For Poisson’s equation, this system was given as (8.1.4). It is partially reproduced here for convenience as

$$\mathbf{A}\mathbf{x} = \begin{bmatrix} \mathbf{C}_1 & \mathbf{D}_1 & & \\ \mathbf{D}_2 & \mathbf{C}_2 & \mathbf{D}_2 & \\ & \ddots & & \\ & & \mathbf{D}_{K-1} & \mathbf{C}_{K-1} \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_{K-1} \end{bmatrix} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \vdots \\ \mathbf{b}_{K-1} \end{bmatrix},$$

The matrices  $\mathbf{C}_k$ , etc. were defined by (8.1.4c-8.1.4d). The SOR procedure is applied to an entire row, *i.e.*, we compute

$$\begin{aligned} \mathbf{C}_k \hat{\mathbf{x}}_k^{(\nu+1)} &= -\mathbf{D}_k \mathbf{x}_{k-1}^{(\nu+1)} - \mathbf{D}_k \mathbf{x}_{k+1}^{(\nu)} + \mathbf{b}_k, \\ \mathbf{x}_k^{(\nu+1)} &= \omega \hat{\mathbf{x}}_k^{(\nu+1)} + (1 - \omega) \mathbf{x}_k^{(\nu)}, \quad k = 1, 2, \dots, K-1, \quad \nu = 0, 1, \dots. \end{aligned}$$

Thus, we have to solve a tridiagonal system at each step of the process. This procedure converges faster than point SOR iteration by a factor of  $\sqrt{2}$ .

4. *Alternating direction implicit (ADI) methods.* By considering an elliptic problem as the steady state limit of a transient parabolic problem, we can use some methods for time-dependent problems to solve them. In particular, the ADI method (*cf.* Section 5.2) has been adapted to the solution of elliptic problems. The goal, when using this approach, is to select the “time step” so that the ADI scheme converges to steady state as fast as possible. The ADI method with a single acceleration parameter (artificial time step) has the same convergence rate as SOR. Further acceleration is possible by choosing a sequence of artificial time steps, changing them after each predictor-corrector sweep, and applying them cyclically. Wachspress [7] has shown how to select nearly optimal acceleration parameters.

The results shown in Table 9.2.5 summarize the convergence rates of the methods that we have studied in this section. They are all obtained by solving a Dirichlet problem on a

square mesh with uniform spacing  $h = \Delta x = \Delta y$ . The convergence rates of all methods decline as the mesh spacing decreases. Degradation in performance is least with the ADI method; however, computing optimal parameters can be problematical in realistic situations.

Method	Conv. Rate
Jacobi	$h^2$
Gauss-Seidel	$2h^2$
SOR (with $\omega_{OPT}$ )	$2h$
ADI ( with $m$ parameters)	$\frac{8}{m}(\frac{h}{2})^{1/m}$

Table 9.2.5: Convergence rates for various iterative methods as a function of mesh spacing  $h$  for a Dirichlet problem on a square.

### Problems

1. Consider a problem for Laplace's equation

$$u_{xx} + u_{yy} = 0, \quad (x, y) \in \Omega,$$

where  $\Omega$  is the region between a  $4 \times 4$  square and a concentric  $2 \times 2$  square (Figure 9.2.6). The Dirichlet boundary conditions are  $u = 1$  on the outside of the  $4 \times 4$  square and  $u = 0$  on the edge of the  $2 \times 2$  square. Due to symmetry, this problem need only be solved on the one octant shown on the right of Figure 9.2.6. The subscript  $\mathbf{n}$  denotes differentiation in the outer normal direction.

- 1.1. Construct a discrete approximation of the above problem on the region shown on the right of Figure 9.2.6. Use the five point difference approximation for Laplace's equation and, for simplicity, assume that the mesh spacing in the  $x$  and  $y$  directions is the same, say,  $\Delta x = \Delta y = 1/N$ . Take appropriate steps to ensure that the finite difference approximations at the symmetry boundary and the interior have  $O(1/N^2)$  accuracy.
- 1.2. For the special discretization when  $\Delta x = 1/2$  the discrete problem has only four unknowns. Write down an SOR procedure for determining these unknowns. Calculate the Jacobi iteration matrix  $\mathbf{M}_J$ . Find an expression for

the spectral radius  $\rho(\mathbf{M}_{SOR})$  of the SOR matrix. Plot  $\rho(\mathbf{M}_{SOR})$  and determine the optimal relaxation parameter  $\omega$ . (This problem should be done symbolically.)

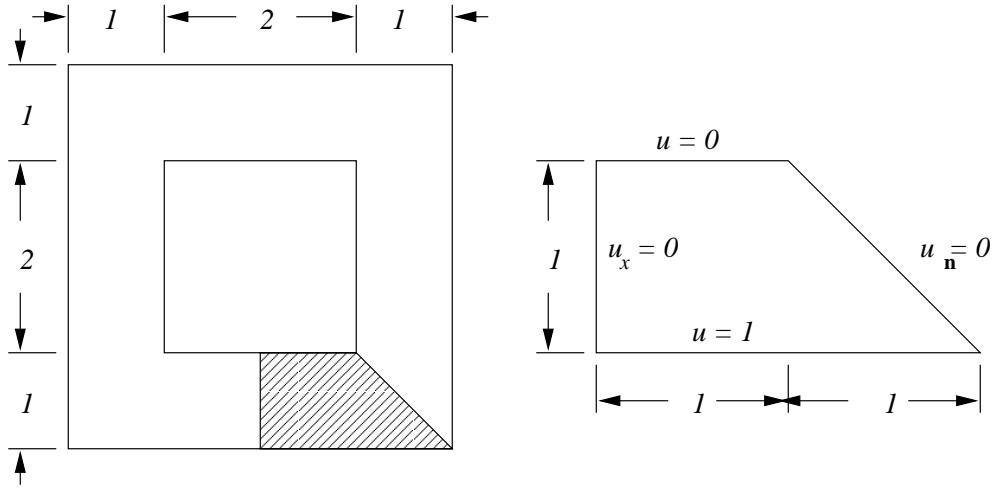


Figure 9.2.6: Domain for Problem 1 (left). Due to symmetry, the problem need only be solved on the octant shown on the right.

### 9.3 Conjugate Gradient Methods

The fixed-point iterative methods of the previous section deteriorate in performance as the dimension  $N$  of the linear system increases. The faster SOR and ADI techniques depend on acceleration parameters that may be difficult to estimate. We seek to overcome these deficiencies without raising the storage requirements to the level of a direct method. Solving the linear system (9.1.1) when  $\mathbf{A}$  is symmetric positive definite matrix is equivalent to minimizing the quadratic functional

$$E(\mathbf{y}) = \frac{1}{2}\mathbf{y}^T \mathbf{A} \mathbf{y} - \mathbf{b}^T \mathbf{y}. \quad (9.3.1)$$

The necessary condition for a minimum,

$$\mathbf{E}'(\mathbf{y}) = \mathbf{A}\mathbf{y} - \mathbf{b} = \mathbf{0}, \quad (9.3.2a)$$

implies that  $\mathbf{y} = \mathbf{x}$ , the solution of (9.1.1).

If we define the residual

$$\mathbf{r}(\mathbf{y}) = \mathbf{b} - \mathbf{A}\mathbf{y} \quad (9.3.2b)$$

then (9.3.2a) may be written as

$$E'(\mathbf{y}) = -\mathbf{r}(\mathbf{y}). \quad (9.3.2c)$$

The level surfaces  $E(\mathbf{y}) = C$  (a constant) of (9.3.1) are ellipsoids in  $\Re^N$  with a common center at  $\mathbf{x}$ . Since the gradient of a function is in the direction of steepest increase, to minimize a  $E(\mathbf{y})$  at a point  $\mathbf{x}^{(0)}$ , we could move in a direction opposite to the gradient of the level surface through  $\mathbf{x}^{(0)}$ . From (9.3.2c), the gradient at  $\mathbf{x}^{(0)}$  is

$$E'(\mathbf{x}^{(0)}) = -\mathbf{r}(\mathbf{x}^{(0)}) = -\mathbf{r}^{(0)}.$$

Let our subsequent guess  $\mathbf{x}^{(1)}$  for the minimum  $\mathbf{x}$  be

$$\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \alpha_0 \mathbf{r}^{(0)},$$

and let us calculate the distance  $\alpha_0$  moved in the negative gradient direction  $\mathbf{r}^{(0)}$  so as to minimize  $E(\mathbf{x}^{(1)})$ . Using (9.3.1), we have

$$E(\mathbf{x}^{(1)}) = E(\mathbf{x}^{(0)} + \alpha_0 \mathbf{r}^{(0)}) = \frac{1}{2}(\mathbf{x}^{(0)} + \alpha_0 \mathbf{r}^{(0)})^T \mathbf{A}(\mathbf{x}^{(0)} + \alpha_0 \mathbf{r}^{(0)}) - \mathbf{b}^T(\mathbf{x}^{(0)} + \alpha_0 \mathbf{r}^{(0)}).$$

Differentiating with respect to  $\alpha_0$

$$\frac{d}{d\alpha_0} E(\mathbf{x}^{(0)} + \alpha_0 \mathbf{r}^{(0)}) = [(\mathbf{x}^{(0)})^T \mathbf{A} - \mathbf{b}^T + \alpha_0 (\mathbf{r}^{(0)})^T \mathbf{A}] \mathbf{r}^{(0)} = 0.$$

Using (9.3.2b)

$$\alpha_0 = \frac{(\mathbf{r}^{(0)})^T \mathbf{r}^{(0)}}{(\mathbf{r}^{(0)})^T \mathbf{A} \mathbf{r}^{(0)}}.$$

With subsequent iterates computed in the same manner, the process is called *the method of steepest descent*. A pseudocode algorithm of the method appears in Figure 9.3.1. Some comments on the procedure and method follow.

1. The calculation of  $\mathbf{r}^{(\nu+1)}$  shown in the algorithm follows definition (9.3.2b); thus,

$$\mathbf{r}^{(\nu+1)} = \mathbf{b} - \mathbf{A}\mathbf{x}^{(\nu+1)} = \mathbf{b} - \mathbf{A}(\mathbf{x}^{(\nu)} + \alpha_\nu \mathbf{r}^{(\nu)}) = \mathbf{r}^{(\nu)} - \alpha_\nu \mathbf{A} \mathbf{r}^{(\nu)}. \quad (9.3.3)$$

Formula (9.3.3) is less susceptible to the accumulation of round-off error than direct computation using (9.3.2b).

```

procedure steepest_descent
   $\mathbf{r}^{(0)} = \mathbf{b} - \mathbf{A}\mathbf{x}^{(0)}$ 
   $\nu = 0$ 
  while not converged do
     $\alpha_\nu = (\mathbf{r}^{(\nu)})^T \mathbf{r}^{(\nu)} / (\mathbf{r}^{(\nu)})^T \mathbf{A} \mathbf{r}^{(\nu)}$ 
     $\mathbf{x}^{(\nu+1)} = \mathbf{x}^{(\nu)} + \alpha_\nu \mathbf{r}^{(\nu)}$ 
     $\mathbf{r}^{(\nu+1)} = \mathbf{r}^{(\nu)} - \alpha_\nu \mathbf{A} \mathbf{r}^{(\nu)}$ 
     $\nu = \nu + 1$ 
  end while

```

Figure 9.3.1: A steepest-descent algorithm.

2. The algorithm only has one matrix multiplication and two vector multiplications per step. When solving partial differential equations, it is not necessary to store the matrix  $\mathbf{A}$ . The product  $\mathbf{A}\mathbf{r}^{(\nu)}$  can be obtained directly from the difference scheme. For example, when solving a problem for Poisson's equation using centered differences (8.1.2), we could compute  $\mathbf{A}\mathbf{r}^{(\nu)}$  at grid point  $(j, k)$  as

$$\mathbf{A}\mathbf{r}_{jk}^{(\nu)} = r_{jk}^{(\nu)} - \theta_x(r_{j+1,k}^{(\nu)} + r_{j-1,k}^{(\nu)}) - \theta_y(r_{j,k+1}^{(\nu)} + r_{j,k-1}^{(\nu)}).$$

The matrix  $\mathbf{A}$  is not stored and the vector  $\mathbf{r}^{(\nu)}$  is stored in the mesh coordinates.

3. Using (9.3.3) and the definition of  $\alpha_\nu$  given in the algorithm of Figure 9.3.1, we have

$$(\mathbf{r}^{(\nu+1)})^T \mathbf{r}^{(\nu)} = (\mathbf{r}^{(\nu)})^T \mathbf{r}^{(\nu)} - \alpha_\nu (\mathbf{r}^{(\nu)})^T \mathbf{A} \mathbf{r}^{(\nu)} = 0.$$

Thus, the search directions  $\mathbf{r}^{(\nu)}$ ,  $k = 0, 1, \dots$ , are “orthogonal” in the sense that

$$(\mathbf{r}^{(\nu+1)})^T \mathbf{r}^{(\nu)} = 0.$$

4. The steepest descent method converges from any initial guess  $\mathbf{x}^{(0)}$  when  $\mathbf{A}$  and  $\mathbf{A}^T \mathbf{A}^{-1}$  are positive definite; thus, symmetry of  $\mathbf{A}$  is not necessary ([5], Section 14.1).
5. Let us introduce the *strain energy norm*

$$\|\mathbf{x}\|_{\mathbf{A}}^2 := \mathbf{x}^T \mathbf{A} \mathbf{x}. \quad (9.3.4a)$$

Then ([4], Section 5.3)

$$\|\mathbf{x}^{(\nu+1)} - \mathbf{x}\|_{\mathbf{A}}^2 \leq \left(1 - \frac{1}{\kappa_2(\mathbf{A})}\right) \|\mathbf{x}^{(\nu)} - \mathbf{x}\|_{\mathbf{A}}^2 \quad (9.3.4b)$$

where  $\kappa_2$  is the *condition number* of  $\mathbf{A}$  in the  $\mathcal{L}^2$  norm as given by

$$\kappa_2(\mathbf{A}) = \lambda_{max}/\lambda_{min}, \quad (9.3.4c)$$

and  $\lambda_{max}$  and  $\lambda_{min}$  are the maximum and minimum eigenvalues of  $\mathbf{A}$ . Convergence is slow when  $\kappa(\mathbf{A})$  is large and  $\mathbf{A}$  is said to be ill conditioned in this case. When this occurs, the level surfaces of  $E(\mathbf{y})$  are often elongated ellipses with the minimum  $\mathbf{x}$  lying at the bottom of a narrow valley with steep walls. Successive iterates tend to wander back and forth across the valley making little progress towards the minimum.

*Example 9.3.1.* Let's solve a simple problem with

$$\mathbf{A} = \begin{bmatrix} 2 & 1 \\ 1 & 10 \end{bmatrix}$$

to illustrate some geometrical aspects of the method of steepest descent. With  $\mathbf{b} = \mathbf{0}$ , the minimum of (9.3.1) is at the origin. We select the initial guess

$$\mathbf{x}^{(0)} = [0.1972, 0.4443]^T$$

to be such that  $E(\mathbf{x}^{(0)}) = 1$  (Figure 9.3.2). Since  $\mathbf{b} = \mathbf{0}$ , we obtain

$$\mathbf{r}^{(0)} = [-0.8387, -4.6406], \quad \alpha_0 = 0.0990, \quad \mathbf{x}^{(1)} = [0.1141, -0.0153]^T.$$

The second iteration produces

$$\mathbf{r}^{(1)} = [-0.2130, 0.0385]^T, \quad \alpha_1 = 0.5255, \quad \mathbf{x}^{(2)} = [0.0022, 0.0050]^T.$$

The iteration seems to be converging to the minimum at the origin. These two iterates and the level surfaces  $E(\mathbf{y}) = 1, 0.75, 0.5, 0.25$  are shown in Figure 9.3.2. The initial iteration proceeds “downhill” in a direction opposite to the gradient at  $\mathbf{x}^{(0)}$  until a local minimum is reached. The second iterate proceeds from there.

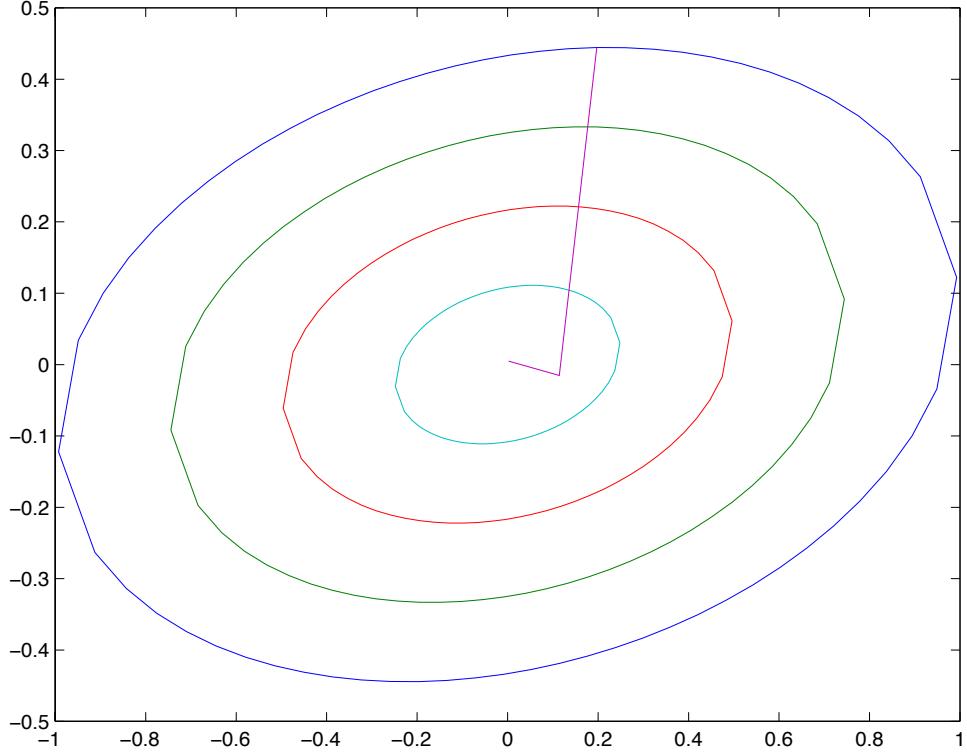


Figure 9.3.2: Convergence of the method of steepest descent for the problem of Example 9.3.1. Ellipses have level values of 1, 0.75, 0.5, and 0.25 (outer to inner). The first two iterations are shown to be converging to the solution at  $(0, 0)$ .

### 9.3.1 The Conjugate Gradient Method

The remedy for the slow convergence encountered with the method of steepest descent is to choose other search directions. With the conjugate gradient method, we choose

$$\mathbf{x}^{(\nu+1)} = \mathbf{x}^{(\nu)} + \alpha_\nu \mathbf{p}^{(\nu)} \quad (9.3.5a)$$

with

$$\mathbf{p}^{(\nu)} = \mathbf{r}^{(\nu)} + \gamma_\nu (\mathbf{x}^{(\nu)} - \mathbf{x}^{(\nu-1)}). \quad (9.3.5b)$$

Thus, the new search direction is a linear combination of the steepest descent search direction and the previous “correction”  $\mathbf{x}^{(\nu)} - \mathbf{x}^{(\nu-1)}$ . The parameters  $\alpha_\nu$  and  $\gamma_\nu$  are to be determined.

Using (9.3.5a), we rewrite (9.3.5b) in the form

$$\mathbf{p}^{(\nu)} = \mathbf{r}^{(\nu)} + \gamma_\nu \alpha_{\nu-1} \mathbf{p}^{(\nu-1)} = \mathbf{r}^{(\nu)} + \beta_{\nu-1} \mathbf{p}^{(\nu-1)} \quad (9.3.5c)$$

with the goal of specifying  $\alpha_\nu$  and  $\beta_{\nu-1}$  so that convergence is as fast as possible. As with steepest descent, we choose  $\alpha_\nu$  to minimize  $E(\mathbf{x}^{(\nu+1)})$ . In this case,

$$E(\mathbf{x}^{(\nu+1)}) = E(\mathbf{x}^{(\nu)} + \alpha_\nu \mathbf{p}^{(\nu)}) = \frac{1}{2}(\mathbf{x}^{(\nu)} + \alpha_\nu \mathbf{p}^{(\nu)})^T \mathbf{A}(\mathbf{x}^{(\nu)} + \alpha_\nu \mathbf{p}^{(\nu)}) - \mathbf{b}^T(\mathbf{x}^{(\nu)} + \alpha_\nu \mathbf{p}^{(\nu)}). \quad (9.3.6)$$

Differentiating with respect to  $\alpha_\nu$

$$E'(\mathbf{x}^{(\nu)} + \alpha_\nu \mathbf{p}^{(\nu)}) = (\mathbf{x}^{(\nu)} + \alpha_\nu \mathbf{p}^{(\nu)})^T \mathbf{A} \mathbf{p}^{(\nu)} - \mathbf{b}^T \mathbf{p}^{(\nu)} = 0$$

or, using (9.3.2b)

$$(-\mathbf{r}^{(\nu)} + \alpha_\nu \mathbf{p}^{(\nu)} \mathbf{A})^T \mathbf{p}^{(\nu)} = 0.$$

Thus,

$$\alpha_\nu = \frac{(\mathbf{r}^{(\nu)})^T \mathbf{p}^{(\nu)}}{(\mathbf{p}^{(\nu)})^T \mathbf{A} \mathbf{p}^{(\nu)}}, \quad \nu = 0, 1, \dots. \quad (9.3.7)$$

Let's develop a few properties and relationships that will be interesting in their own right and useful for calculating  $\beta_\nu$ . First, we'll use (9.3.2b) and (9.3.5a) to write

$$\mathbf{r}^{(\nu+1)} = \mathbf{b} - \mathbf{A}\mathbf{x}^{(\nu+1)} = \mathbf{b} - \mathbf{A}(\mathbf{x}^{(\nu)} + \alpha_\nu \mathbf{p}^{(\nu)}) = \mathbf{r}^{(\nu)} - \alpha_\nu \mathbf{A} \mathbf{p}^{(\nu)}. \quad (9.3.8)$$

Taking an inner product with  $\mathbf{p}^{(\nu)}$  to obtain

$$(\mathbf{r}^{(\nu+1)})^T \mathbf{p}^{(\nu)} = (\mathbf{r}^{(\nu)})^T \mathbf{p}^{(\nu)} - \alpha_\nu (\mathbf{p}^{(\nu)})^T \mathbf{A} \mathbf{p}^{(\nu)}.$$

Using (9.3.7) to eliminate  $\alpha_\nu$  reveals the orthogonality condition

$$(\mathbf{r}^{(\nu+1)})^T \mathbf{p}^{(\nu)} = 0. \quad (9.3.9a)$$

Next, take the inner product of (9.3.5c) with  $\mathbf{r}^{(\nu+1)}$  to obtain

$$(\mathbf{r}^{(\nu+1)})^T \mathbf{p}^{(\nu+1)} = (\mathbf{r}^{(\nu+1)})^T \mathbf{r}^{(\nu+1)} + \beta_\nu (\mathbf{r}^{(\nu+1)})^T \mathbf{p}^{(\nu)}$$

or, using (9.3.9a),

$$(\mathbf{r}^{(\nu+1)})^T \mathbf{p}^{(\nu+1)} = (\mathbf{r}^{(\nu+1)})^T \mathbf{r}^{(\nu+1)}.$$

If we select  $\mathbf{p}^{(0)} = \mathbf{r}^{(0)}$ , then we may write the above expression as

$$(\mathbf{r}^{(\nu)})^T \mathbf{p}^{(\nu)} = (\mathbf{r}^{(\nu)})^T \mathbf{r}^{(\nu)}, \quad \nu = 0, 1, \dots. \quad (9.3.9b)$$

Let us next expand (9.3.6) while using (9.3.1) and (9.3.2b) to show that

$$E(\mathbf{x}^{(\nu+1)}) = E(\mathbf{x}^{(\nu)}) - \alpha_\nu (\mathbf{r}^{(\nu)})^T \mathbf{p}^{(\nu)} + (1/2) \alpha_\nu^2 (\mathbf{p}^{(\nu)})^T \mathbf{A} \mathbf{p}^{(\nu)}.$$

Using (9.3.7) and (9.3.9b)

$$E(\mathbf{x}^{(\nu+1)}) = E(\mathbf{x}^{(\nu)}) - \frac{1}{2} \frac{[(\mathbf{r}^{(\nu)})^T \mathbf{p}^{(\nu)}]^2}{(\mathbf{p}^{(\nu)})^T \mathbf{A} \mathbf{p}^{(\nu)}} = E(\mathbf{x}^{(\nu)}) - \frac{1}{2} \frac{[(\mathbf{r}^{(\nu)})^T \mathbf{r}^{(\nu)}]^2}{(\mathbf{p}^{(\nu)})^T \mathbf{A} \mathbf{p}^{(\nu)}}. \quad (9.3.10)$$

Having (9.3.10), we see that the error  $E(\mathbf{x}^{(\nu+1)})$  is decreased most rapidly when  $(\mathbf{p}^{(\nu)})^T \mathbf{A} \mathbf{p}^{(\nu)}$  is minimal. This will be our criterion for determining  $\beta_\nu$ . Using (9.3.5c), we have

$$(\mathbf{p}^{(\nu)})^T \mathbf{A} \mathbf{p}^{(\nu)} = (\mathbf{r}^{(\nu)} + \beta_{\nu-1} \mathbf{p}^{(\nu-1)})^T \mathbf{A} (\mathbf{r}^{(\nu)} + \beta_{\nu-1} \mathbf{p}^{(\nu-1)}).$$

Minimizing with respect to  $\beta_{\nu-1}$  gives

$$\beta_{\nu-1} = -\frac{(\mathbf{r}^{(\nu)})^T \mathbf{A} \mathbf{p}^{(\nu-1)}}{(\mathbf{p}^{(\nu-1)})^T \mathbf{A} \mathbf{p}^{(\nu-1)}}$$

or reindexing

$$\beta_\nu = -\frac{(\mathbf{r}^{(\nu+1)})^T \mathbf{A} \mathbf{p}^{(\nu)}}{(\mathbf{p}^{(\nu)})^T \mathbf{A} \mathbf{p}^{(\nu)}}, \quad \nu = 0, 1, \dots . \quad (9.3.11)$$

Using (9.3.5c)

$$(\mathbf{p}^{(\nu+1)})^T \mathbf{A} \mathbf{p}^{(\nu)} = (\mathbf{r}^{(\nu+1)})^T \mathbf{A} \mathbf{p}^{(\nu)} + \beta_\nu (\mathbf{p}^{(\nu)})^T \mathbf{A} \mathbf{p}^{(\nu)},$$

which, upon use of (9.3.11), reveals

$$(\mathbf{p}^{(\nu+1)})^T \mathbf{A} \mathbf{p}^{(\nu)} = 0, \quad \nu = 0, 1, \dots . \quad (9.3.12a)$$

Thus, the search directions are orthogonal with respect to a *strain energy inner product*. We usually call this a conjugacy condition and say that the search directions are *conjugate*.

Using (9.3.12a) with (9.3.5c), we have

$$(\mathbf{p}^{(\nu)})^T \mathbf{A} \mathbf{p}^{(\nu)} = (\mathbf{r}^{(\nu)})^T \mathbf{A} \mathbf{p}^{(\nu)} + \beta_{\nu-1} (\mathbf{p}^{(\nu-1)})^T \mathbf{A} \mathbf{p}^{(\nu)} = (\mathbf{r}^{(\nu)})^T \mathbf{A} \mathbf{p}^{(\nu)}.$$

Combining this result with (9.3.8) yields

$$(\mathbf{r}^{(\nu+1)})^T \mathbf{r}^{(\nu)} = (\mathbf{r}^{(\nu)})^T \mathbf{r}^{(\nu)} - \alpha_\nu (\mathbf{p}^{(\nu)})^T \mathbf{A} \mathbf{r}^{(\nu)} = (\mathbf{r}^{(\nu)})^T \mathbf{r}^{(\nu)} - \alpha_\nu (\mathbf{p}^{(\nu)})^T \mathbf{A} \mathbf{p}^{(\nu)}.$$

Using (9.3.7) and (9.3.9b), we find the orthogonality relation

$$(\mathbf{r}^{(\nu+1)})^T \mathbf{r}^{(\nu)} = 0. \quad (9.3.12b)$$

Equation (9.3.11) can be put in a slightly simpler form by using (9.3.8) and (9.3.12b) to obtain

$$(\mathbf{r}^{(\nu+1)})^T \mathbf{r}^{(\nu+1)} = (\mathbf{r}^{(\nu+1)})^T \mathbf{r}^{(\nu)} - \alpha_\nu (\mathbf{r}^{(\nu+1)})^T \mathbf{A} \mathbf{p}^{(\nu)} = -\alpha_\nu (\mathbf{r}^{(\nu+1)})^T \mathbf{A} \mathbf{p}^{(\nu)}.$$

Using this with (9.3.11) and (9.3.7)

$$\beta_\nu = \frac{1}{\alpha_\nu} \frac{(\mathbf{r}^{(\nu+1)})^T \mathbf{r}^{(\nu+1)}}{(\mathbf{p}^{(\nu)})^T \mathbf{A} \mathbf{p}^{(\nu)}} = \frac{(\mathbf{r}^{(\nu+1)})^T \mathbf{r}^{(\nu+1)}}{(\mathbf{p}^{(\nu)})^T \mathbf{r}^{(\nu)}}.$$

Finally, using (9.3.9b)

$$\beta_\nu = \frac{(\mathbf{r}^{(\nu+1)})^T \mathbf{r}^{(\nu+1)}}{(\mathbf{r}^{(\nu)})^T \mathbf{r}^{(\nu)}}. \quad (9.3.12c)$$

We summarize our findings as a theorem.

**Theorem 9.3.1.** *The residuals and search directions of the conjugate gradient method satisfy*

$$(\mathbf{r}^{(\mu)})^T \mathbf{r}^{(\nu)} = (\mathbf{p}^{(\mu)})^T \mathbf{A} \mathbf{p}^{(\nu)} = \mathbf{0}, \quad \mu \neq \nu. \quad (9.3.13)$$

*Proof.* This has essentially been proven by the prior developments.  $\square$

An algorithm for the conjugate gradient method is presented in Figure 9.3.3. Some comments on the algorithm follow:

1. Equation (9.3.9b) was used to modify the expression (9.3.7) for  $\alpha_\nu$ .
2. The procedure requires storage for the nonzero elements of  $\mathbf{A}$  and for  $\mathbf{x}^{(\nu)}$ ,  $\mathbf{p}^{(\nu)}$ , and  $\mathbf{r}^{(\nu)}$ . An additional vector is needed to store the product  $\mathbf{A} \mathbf{p}^{(\nu)}$ . Thus, storage costs remain modest relative to the direct methods of Section 9.1.
3. The procedure requires a matrix multiplication ( $\mathbf{A} \mathbf{p}^{(\nu)}$ ) and computation of two inner products ( $(\mathbf{r}^{(\nu)})^T \mathbf{r}^{(\nu)}$  and  $(\mathbf{p}^{(\nu)})^T \mathbf{A} \mathbf{p}^{(\nu)}$ ) per step.
4. Unlike SOR methods, there are no acceleration parameters to determine.

```

procedure conjugate_gradient
   $\mathbf{p}^{(0)} = \mathbf{r}^{(0)} = \mathbf{b} - \mathbf{Ax}^{(0)}$ 
   $\nu = 0$ 
  while not converged do
     $\alpha_\nu = (\mathbf{r}^{(\nu)})^T \mathbf{r}^{(\nu)} / (\mathbf{p}^{(\nu)})^T \mathbf{Ap}^{(\nu)}$ 
     $\mathbf{x}^{(\nu+1)} = \mathbf{x}^{(\nu)} + \alpha_\nu \mathbf{p}^{(\nu)}$ 
     $\mathbf{r}^{(\nu+1)} = \mathbf{r}^{(\nu)} - \alpha_\nu \mathbf{Ap}^{(\nu)}$ 
     $\beta_\nu = (\mathbf{r}^{(\nu+1)})^T \mathbf{r}^{(\nu+1)} / (\mathbf{r}^{(\nu)})^T \mathbf{r}^{(\nu)}$ 
     $\mathbf{p}^{(\nu+1)} = \mathbf{r}^{(\nu+1)} + \beta_\nu \mathbf{p}^{(\nu)}$ 
     $\nu = \nu + 1$ 
  end while

```

Figure 9.3.3: Conjugate gradient algorithm.

The conjugate gradient method is both a direct and an iterative method as indicated by the following theorem.

**Theorem 9.3.2.** *Let  $\mathbf{A}$  be a positive definite, symmetric,  $N \times N$  matrix. Then the conjugate gradient method converges to the exact solution in no more than  $N$  steps.*

*Proof.* By Theorem 9.3.1, the residuals  $\mathbf{r}^{(\nu)}$ ,  $\nu = 0, 1, \dots, N-1$ , are mutually orthogonal. Since the space is  $N$ -dimensional, the residual  $\mathbf{r}^{(N)}$  must be zero; hence, the method converges in  $N$  steps.  $\square$

While convergence is achieved in  $N$  steps, the hope is to produce acceptable approximations of  $\mathbf{x}$  in far fewer than  $N$  steps when  $N$  is large. Practically, convergence may not be achieved in  $N$  steps when round-off errors are present.

*Example 9.3.2* ([5], Section 14.2). Consider the solution of Laplace's equation on a square region with  $\Delta x = \Delta y = h$ . Let the Dirichlet boundary conditions be prescribed so that the exact solution is

$$u(x, y) = e^x \sin y.$$

Solutions were calculated using SOR and conjugate gradient iterations until the change in the solution in the  $\mathcal{L}^2$  norm was less than  $10^{-7}$ . The value of  $\omega = 2/(1 + \pi h)$  was used with the SOR method. Results are shown in Table 9.3.1. The two methods are comparable. Apparent convergence is at a linear rate in  $h$ , i.e., doubling  $h$  approximately

doubles the number of SOR and conjugate gradient iterations to convergence. Since an SOR iteration is less costly than a conjugate gradient step, we may infer that the SOR procedure is the faster.

$1/h$	SOR	CG
10	31	27
20	64	54
40	122	107

Table 9.3.1: Number of SOR and conjugate gradient iterations to convergence for Example 9.3.2.

The following theorem confirms the findings of the previous example.

**Theorem 9.3.3.** *Let  $\mathbf{A}$  be a symmetric and positive definite matrix, then iterates of the conjugate gradient method satisfy*

$$\|\mathbf{x}^{(\nu)} - \mathbf{x}\|_{\mathbf{A}} \leq 2 \left( \frac{\sqrt{\kappa_2} - 1}{\sqrt{\kappa_2} + 1} \right)^\nu \|\mathbf{x}^{(0)} - \mathbf{x}\|_{\mathbf{A}} \quad (9.3.14)$$

where  $\|\cdot\|_{\mathbf{A}}$  and  $\kappa_2$  were defined in (9.3.4a, 9.3.4c).

*Proof.* cf. [4], Section 6.11. □

Examining (9.3.14) and (9.3.4c), we see that convergence is fastest when the eigenvalues of  $\mathbf{A}$  are clustered together, i.e., when  $\kappa_2(\mathbf{A}) \approx 1$ .

*Example 9.3.3.* The factor

$$R = \frac{\sqrt{\kappa_2} - 1}{\sqrt{\kappa_2} + 1}$$

determines the convergence rate of the conjugate gradient method. Since the condition number  $\kappa_2$  depends on the eigenvalues of  $\mathbf{A}$ , it would seem that we have to examine the eigenvalue problem

$$\mathbf{A}\mathbf{q} = \lambda\mathbf{q}.$$

Using (9.2.8a), let us write this relation as

$$(\mathbf{D} - \mathbf{L} - \mathbf{U})\mathbf{q} = \lambda\mathbf{q}$$

where  $\mathbf{D}$ ,  $\mathbf{L}$ , and  $\mathbf{U}$  were defined by (9.2.8b-9.2.8c). Multiplying by  $\mathbf{D}^{-1}$  and using (9.2.10b), we have

$$\mathbf{M}_J \mathbf{q} = (\mathbf{I} - \lambda \mathbf{D}^{-1}) \mathbf{q}$$

where  $\mathbf{M}_J$  is the Jacobi iteration matrix. For the Laplacian operator,  $\mathbf{D} = \mathbf{I}$  and we have  $\mu = 1 - \lambda$ , where  $\mu$  is an eigenvalue of  $\mathbf{M}_J$ . Still confining our attention to the Laplacian operator, we may use (9.2.17d) to evaluate  $\mu$  and, hence, obtain

$$\lambda = 1 - \mu = 4\theta_x \sin^2 \frac{m\pi}{2J} + 4\theta_y \sin^2 \frac{n\pi}{2K}, \quad m = 1, 2, \dots, J-1, \quad n = 1, 2, \dots, K-1.$$

For simplicity, let us focus on a square grid ( $J = K$ ) where

$$\lambda = \sin^2 \frac{m\pi}{2J} + \sin^2 \frac{n\pi}{2J}, \quad m, n = 1, 2, \dots, J-1.$$

The smallest eigenvalue occurs with  $m = n = 1$  and the largest occurs with  $m = n = J-1$ ; thus,

$$\lambda_{min} = 2 \sin^2 \frac{\pi}{2J}, \quad \lambda_{max} = 2 \sin^2 \frac{(J-1)\pi}{2J}.$$

Hence, using (9.3.4c)

$$\kappa_2 = \frac{\sin^2(J-1)\pi/2J}{\sin^2 \pi/2J}.$$

When  $J \gg 1$  we may approximate this as

$$\sqrt{\kappa_2} = \frac{\sin(J-1)\pi/2J}{\sin \pi/2J} \approx \frac{2J}{\pi}.$$

Thus,

$$R \approx \frac{1 - \pi/2J}{1 + \pi/2J} \approx 1 - \frac{\pi}{J}.$$

Convergence is, therefore, at the same rate as the SOR method (Example 9.2.9).

### 9.3.2 Preconditioned Conjugate Gradient Iterations

From Theorem 9.3.3, we see that the performance of the conjugate gradient method improves when the eigenvalues of  $\mathbf{A}$  are clustered about a point. This suggests the possibility of preconditioning  $\mathbf{A}$  by a positive definite matrix  $\mathbf{M}$  and solving

$$\mathbf{M}^{-1} \mathbf{A} \mathbf{x} = \mathbf{M}^{-1} \mathbf{b}. \quad (9.3.15)$$

If the eigenvalues of  $\mathbf{M}^{-1}\mathbf{A}$  were clustered, the conjugate gradient procedure may converge at a faster rate. The preconditioner  $\mathbf{M}$  should be chosen to minimize the solution time. There are, however, competing priorities. Thus, for example, the optimal choice of  $\mathbf{M}$  as far as clustering eigenvalues is concerned is  $\mathbf{M} = \mathbf{A}$ . This choice requires a direct solution of the original system and, thus, has an extreme cost. The optimal choice of  $\mathbf{M}$  as far as computational effort is concerned is  $\mathbf{M} = \mathbf{I}$ . This is the conjugate gradient algorithm and, thus, no improvement has been provided. The search for the best preconditioning is still an active area of research with optimality dependent on many factors including sparsity and intended computer architecture.

The preconditioning shown in (9.3.15) is called a *left preconditioning*. A *right preconditioning* is

$$\mathbf{A}\mathbf{M}^{-1}\mathbf{w} = \mathbf{b}, \quad \mathbf{x} = \mathbf{M}^{-1}\mathbf{w}. \quad (9.3.16a)$$

A *symmetric preconditioning* is

$$\mathbf{C}^{-1}\mathbf{A}\mathbf{C}^{-T}\mathbf{w} = \mathbf{C}^{-1}\mathbf{b}, \quad \mathbf{x} = \mathbf{C}^{-T}\mathbf{w} \quad (9.3.16b)$$

where  $\mathbf{C}^{-T}$  denotes the transpose of  $\mathbf{C}^{-1}$ . The preconditioning matrix  $\mathbf{C}$  need not be symmetric since  $\mathbf{C}^{-1}\mathbf{A}\mathbf{C}^{-T}$  is symmetric and positive definite when  $\mathbf{A}$  is. The matrix  $\mathbf{C}$  may be a Cholesky factor of  $\mathbf{M}$ , *i.e.*,

$$\mathbf{M} = \mathbf{C}\mathbf{C}^T. \quad (9.3.16c)$$

In this case,  $\mathbf{C}$  would be lower triangular and it could be obtained from the symmetric factorization of  $\mathbf{M}$  given by (9.1.8, 9.1.9) or (9.1.12, 9.3.11).

The preconditioned conjugate gradient (PCG) algorithm with the symmetric preconditioning may be implemented by applying the conjugate gradient procedure to

$$\tilde{\mathbf{A}}\mathbf{w} = \tilde{\mathbf{b}} \quad (9.3.17a)$$

where

$$\tilde{\mathbf{A}} = \mathbf{C}^{-1}\mathbf{A}\mathbf{C}^{-T}, \quad \tilde{\mathbf{b}} = \mathbf{C}^{-1}\mathbf{b}, \quad \mathbf{w} = \mathbf{C}^T\mathbf{x} \quad (9.3.17b)$$

```

 $\nu = 0$ 
while not converged do
     $\alpha_\nu = (\tilde{\mathbf{r}}^{(\nu)})^T \tilde{\mathbf{r}}^{(\nu)} / (\tilde{\mathbf{p}}^{(\nu)})^T \tilde{\mathbf{A}} \tilde{\mathbf{p}}^{(\nu)}$ 
     $\mathbf{w}^{(\nu+1)} = \mathbf{w}^{(\nu)} + \alpha_\nu \tilde{\mathbf{p}}^{(\nu)}$ 
     $\tilde{\mathbf{r}}^{(\nu+1)} = \tilde{\mathbf{r}}^{(\nu)} - \alpha_\nu \tilde{\mathbf{A}} \tilde{\mathbf{p}}^{(\nu)}$ 
     $\beta_\nu = (\tilde{\mathbf{r}}^{(\nu+1)})^T \tilde{\mathbf{r}}^{(\nu+1)} / (\tilde{\mathbf{r}}^{(\nu)})^T \tilde{\mathbf{r}}^{(\nu)}$ 
     $\tilde{\mathbf{p}}^{(\nu+1)} = \tilde{\mathbf{r}}^{(\nu+1)} + \beta_\nu \tilde{\mathbf{p}}^{(\nu)}$ 
     $\nu = \nu + 1$ 
end while

```

Figure 9.3.4: Main loop of the conjugate gradient algorithm applied to (9.3.17).

The main loop of the conjugate gradient algorithm of Figure 9.3.3 is reproduced in Figure 9.3.4 for the system (9.3.17). In this algorithm

$$\tilde{\mathbf{r}}^{(\nu)} = \tilde{\mathbf{b}} - \tilde{\mathbf{A}} \mathbf{w}^{(\nu)} = \mathbf{C}^{-1}(\mathbf{b} - \mathbf{A} \mathbf{x}^{(\nu)}) = \mathbf{C}^{-1} \mathbf{r}^{(\nu)} \quad (9.3.18a)$$

Also

$$\tilde{\mathbf{p}}^{(\nu)} = \mathbf{C}^T \mathbf{p}^{(\nu)}. \quad (9.3.18b)$$

Using (9.3.18), let us rewrite the conjugate gradient algorithm of Figure 9.3.4 in terms of the original variables as

$$(\tilde{\mathbf{r}}^{(\nu)})^T \tilde{\mathbf{r}}^{(\nu)} = (\mathbf{r}^{(\nu)})^T \mathbf{C}^{-T} \mathbf{C}^{-1} \mathbf{r}^{(\nu)} = (\mathbf{r}^{(\nu)})^T \mathbf{M}^{-1} \mathbf{r}^{(\nu)}, \quad (9.3.19a)$$

$$\tilde{\mathbf{A}} \tilde{\mathbf{p}}^{(\nu)} = \mathbf{C}^{-1} \mathbf{A} \mathbf{C}^{-T} \mathbf{C}^T \mathbf{p}^{(\nu)} = \mathbf{C}^{-1} \mathbf{A} \mathbf{p}^{(\nu)} \quad (9.3.19b)$$

$$(\tilde{\mathbf{p}}^{(\nu)})^T \tilde{\mathbf{A}} \tilde{\mathbf{p}}^{(\nu)} = (\mathbf{p}^{(\nu)})^T \mathbf{C} \mathbf{C}^{-1} \mathbf{A} \mathbf{p}^{(\nu)} = (\mathbf{p}^{(\nu)})^T \mathbf{A} \mathbf{p}^{(\nu)} \quad (9.3.19c)$$

The PCG algorithm written in terms of the original variables appears in Figure 9.3.5. Some comments follow:

1. The PCG algorithm does not involve  $\mathbf{C}$  or a Cholesky factorization of  $\mathbf{M}$ . Thus, a PCG algorithm with the left preconditioning (9.3.15) would be identical. The PCG algorithm with the right preconditioning also gives the same sequence of operations ([4], Section 9.2).

```

procedure pcg
   $\mathbf{r}^{(0)} = \mathbf{b} - \mathbf{Ax}^{(0)}$ 
  Solve  $\mathbf{Mz}^{(0)} = \mathbf{r}^{(0)}$ 
   $\mathbf{p}^{(0)} = \mathbf{z}^{(0)}$ 
   $\nu = 0$ 
  while not converged do
     $\alpha_\nu = (\mathbf{r}^{(\nu)})^T \mathbf{z}^{(\nu)} / (\mathbf{p}^{(\nu)})^T \mathbf{Ap}^{(\nu)}$ 
     $\mathbf{x}^{(\nu+1)} = \mathbf{x}^{(\nu)} + \alpha_\nu \mathbf{p}^{(\nu)}$ 
     $\mathbf{r}^{(\nu+1)} = \mathbf{r}^{(\nu)} - \alpha_\nu \mathbf{Ap}^{(\nu)}$ 
    Solve  $\mathbf{Mz}^{(\nu+1)} = \mathbf{r}^{(\nu+1)}$ 
     $\beta_\nu = (\mathbf{r}^{(\nu+1)})^T \mathbf{z}^{(\nu+1)} / (\mathbf{r}^{(\nu)})^T \mathbf{z}^{(\nu)}$ 
     $\mathbf{p}^{(\nu+1)} = \mathbf{z}^{(\nu+1)} + \beta_\nu \mathbf{p}^{(\nu)}$ 
     $\nu = \nu + 1$ 
  end while

```

Figure 9.3.5: PCG algorithm.

2. Rather than compute  $\mathbf{M}^{-1}$ , it is more efficient to solve  $\mathbf{Mz}^{(\nu)} = \mathbf{r}^{(\nu)}$  for  $\mathbf{z}^{(\nu)}$ .
3. The original conjugate gradient algorithm of Figure 9.3.3 would set  $\tilde{\mathbf{p}}^{(0)} = \tilde{\mathbf{r}}^{(0)}$ .

Using (9.3.18a, 9.3.18b)

$$\mathbf{C}^T \mathbf{p}^{(0)} = \mathbf{C}^{-1} \mathbf{r}^{(0)}$$

or, using (9.3.16c)

$$\mathbf{p}^{(0)} = \mathbf{C}^{-T} \mathbf{C}^{-1} \mathbf{r}^{(0)} = \mathbf{M}^{-1} \mathbf{r}^{(0)} = \mathbf{z}^{(0)}.$$

This explains the choice of  $\mathbf{p}^{(0)}$  in the initial stages of the algorithm. The subsequent calculation of  $\mathbf{p}^{(\nu+1)}$  uses the same manipulations.

4. Storage is needed for  $\mathbf{A}$ ,  $\mathbf{M}$  (possibly in factored form),  $\mathbf{x}^{(\nu)}$ ,  $\mathbf{p}^{(\nu)}$ ,  $\mathbf{r}^{(\nu)}$ ,  $\mathbf{z}^{(\nu)}$ , and  $\mathbf{Ap}^{(\nu)}$ . Relative to the conjugate gradient procedure, additional storage is needed for  $\mathbf{M}$  and  $\mathbf{z}^{(\nu)}$ . Storage costs are still modest relative to those of direct methods.
5. In addition to the matrix multiplication ( $\mathbf{Ap}^{(\nu)}$ ) and the two inner products per step required by the conjugate gradient method, a linear equations solution ( $\mathbf{Mz}^{(\nu+1)} = \mathbf{r}^{(\nu+1)}$ ) is required

6. Using (9.3.13) for the conjugate gradient method

$$(\tilde{\mathbf{r}}^{(\mu)})^T \tilde{\mathbf{r}}^{(\nu)} = 0, \quad (\tilde{\mathbf{p}}^{(\mu)})^T \tilde{\mathbf{A}} \tilde{\mathbf{p}}^{(\nu)} = 0, \quad \nu \neq \mu.$$

Using (9.3.18a, 9.3.18b), we obtain the “orthogonality conditions”

$$(\mathbf{r}^{(\mu)})^T \mathbf{M}^{-1} \mathbf{r}^{(\nu)} = 0, \quad (\mathbf{p}^{(\mu)})^T \mathbf{A} \mathbf{p}^{(\nu)} = 0, \quad \nu \neq \mu \quad (9.3.20)$$

7. When  $\mathbf{M}$  is positive definite

$$(\mathbf{r}^{(\nu)})^T \mathbf{z}^{(\nu)} = (\mathbf{r}^{(\nu)})^T \mathbf{M}^{-1} \mathbf{r}^{(\nu)} > 0.$$

Thus, values of  $\beta_\nu$  can always be obtained and the procedure does not fail.

Let us select some preconditionings, beginning with some choices based on iterative strategies. It will be convenient to write the basic fixed-point strategy (9.2.2) in the form

$$\hat{\mathbf{M}}\mathbf{x}^{(\nu+1)} = \hat{\mathbf{N}}\mathbf{x}^{(\nu)} + \mathbf{b} \quad (9.3.21a)$$

where

$$\mathbf{A} = \hat{\mathbf{M}} - \hat{\mathbf{N}} \quad (9.3.21b)$$

Comparing (9.3.21) with (9.2.10b), (9.2.15b), and (9.2.21d), we have

- *Jacobi iteration:*

$$\hat{\mathbf{M}}_J = \mathbf{D}, \quad \hat{\mathbf{N}}_J = \mathbf{L} + \mathbf{U} \quad (9.3.22)$$

- *Gauss-Seidel iteration:*

$$\hat{\mathbf{M}}_{GS} = \mathbf{D} - \mathbf{L}, \quad \hat{\mathbf{N}}_{GS} = \mathbf{U} \quad (9.3.23)$$

- SOR Iteration:

$$\hat{\mathbf{M}}_\omega = \frac{1}{\omega}(\mathbf{D} - \omega \mathbf{L}), \quad \hat{\mathbf{N}}_\omega = \frac{1-\omega}{\omega} \mathbf{D} + \mathbf{U}. \quad (9.3.24)$$

Recall that  $\mathbf{D}$  is the diagonal part,  $\mathbf{L}$  is the negative of the lower triangular part, and  $\mathbf{U}$  is the negative of the upper triangular part of  $\mathbf{A}$  (*cf.* (9.2.8)). Let us also include symmetric successive over relaxation (SSOR) in our study. As discussed in Section 9.2, SSOR takes two SOR sweeps with the unknowns placed in reverse order on the second sweep. Using (9.3.24) and (9.3.21), the first step of the SOR procedure is

$$(\mathbf{D} - \omega\mathbf{L})\mathbf{x}^{(\nu+1/2)} = [(1 - \omega)\mathbf{D} + \omega\mathbf{U}]\mathbf{x}^{(\nu)} + \omega\mathbf{b}. \quad (9.3.25a)$$

Reversing the sweep direction on the second step yields

$$(\mathbf{D} - \omega\mathbf{U})\mathbf{x}^{(\nu+1)} = [(1 - \omega)\mathbf{D} + \omega\mathbf{L}]\mathbf{x}^{(\nu+1/2)} + \omega\mathbf{b}. \quad (9.3.25b)$$

The intermediate solution  $\mathbf{x}^{(\nu+1/2)}$  can be eliminated to obtain a scheme of the form (9.3.21) with (*cf.* Problem 2 at the end of this section)

$$\hat{\mathbf{M}}_{SSOR} = \frac{1}{\omega(2 - \omega)}(\mathbf{D} - \omega\mathbf{L})\mathbf{D}^{-1}(\mathbf{D} - \omega\mathbf{U}), \quad (9.3.26a)$$

$$\hat{\mathbf{N}}_{SSOR} = \frac{1}{\omega}\left[\mathbf{I} - \frac{1}{2 - \omega}(\mathbf{D} - \omega\mathbf{L})\mathbf{D}^{-1}\right][\omega\mathbf{U} + (1 - \omega)\mathbf{D}]. \quad (9.3.26b)$$

At the moment, the iteration matrix  $\hat{\mathbf{M}}$  of (9.3.21) and the preconditioning matrix  $\mathbf{M}$  of (9.3.15) are unrelated; however, observe that the exact solution of (9.3.21a) satisfies

$$\hat{\mathbf{M}}\mathbf{x} = \hat{\mathbf{N}}\mathbf{x} + \mathbf{b}.$$

Multiplying by  $\hat{\mathbf{M}}^{-1}$

$$(\mathbf{I} - \hat{\mathbf{M}}^{-1}\hat{\mathbf{N}})\mathbf{x} = \hat{\mathbf{M}}^{-1}\mathbf{b}.$$

Using (9.3.21b) to eliminate  $\hat{\mathbf{N}}$

$$\hat{\mathbf{M}}^{-1}\mathbf{A}\mathbf{x} = \hat{\mathbf{M}}^{-1}\mathbf{b}.$$

This has the same form as the left preconditioning (9.3.15); thus,  $\hat{\mathbf{M}}$  serves as a preconditioner.

Examining (9.3.22 - 9.3.24), (9.3.26), however, we see that  $\hat{\mathbf{M}}_{GS}$  and  $\hat{\mathbf{M}}_\omega$  are not symmetric. Thus, only the Jacobi and SSOR methods will furnish acceptable preconditionings and, of the two, we focus on the SSOR preconditioner (9.3.26a).

At each PCG iteration (Figure 9.3.5) we must solve

$$\hat{\mathbf{M}}_{SSOR} \mathbf{z}^{(\nu)} = \mathbf{r}^{(\nu)}. \quad (9.3.27a)$$

If  $\mathbf{A}$  is symmetric then  $\mathbf{U} = \mathbf{L}^T$  and (9.3.26a) becomes

$$\hat{\mathbf{M}}_{SSOR} = \frac{1}{\omega(2-\omega)} (\mathbf{D} - \omega \mathbf{L}) \mathbf{D}^{-1} (\mathbf{D} - \omega \mathbf{L}^T), \quad (9.3.27b)$$

Thus, (9.3.27a) may be solved with a forward, diagonal, and backward substitution as

$$(\mathbf{D} - \omega \mathbf{L}) \bar{\mathbf{z}} = \omega(2-\omega) \mathbf{r}^{(\nu)}, \quad (9.3.27c)$$

$$\mathbf{D}^{-1} \tilde{\mathbf{z}} = \bar{\mathbf{z}}, \quad (9.3.27d)$$

and

$$(\mathbf{D} - \omega \mathbf{L}^T) \mathbf{z}^{(\nu)} = \tilde{\mathbf{z}}. \quad (9.3.27e)$$

The choice of  $\omega$  doesn't appear to be critical and may, *e.g.*, be selected as unity.

*Example 9.3.4* ([5], Section 14.5). Consider the solution of Poisson's equation on a square with uniform spacing  $\Delta x = \Delta y = h$ . Suppose that the forcing and boundary data is such that the exact solution is

$$u(x, y) = \cos x \sin y.$$

The initial iterate was trivial inside the square and all other numerical parameters were selected as for Example 9.3.2. Comparisons of results obtained using SOR, CG, and PCG are presented in Table 9.3.2. The number of iterations of the SOR and CG algorithms is increasing as  $1/h$  while that of the PCG algorithm is increasing as  $1/h^{1/2}$ . The work of the conjugate gradient method is about twice that of the SOR method and that of the PCG method is about four times the SOR method. Thus, for small systems, the SOR and conjugate gradient method will be superior, but the SSOR-PCG method overtakes these methods for larger systems.

One of the most successful preconditioning techniques utilizes incomplete factorization by Gaussian elimination. Thus, let

$$\mathbf{M} = \mathbf{L}_0 \mathbf{D}_0 \mathbf{L}_0^T \quad (9.3.28)$$

$1/h$	SOR	CG	PCG
10	33	26	12
20	60	52	16
40	115	103	22

Table 9.3.2: Number of SOR, conjugate gradient, and SSOR-PCG iterations for Example 9.3.4.

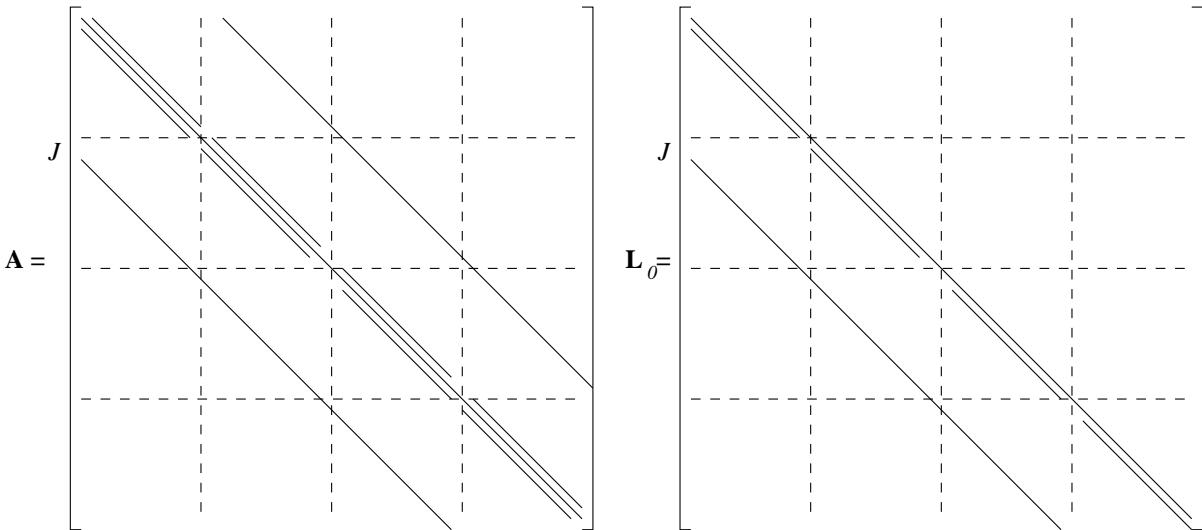


Figure 9.3.6: Nonzero structure of a discrete Poisson operator  $\mathbf{A}$  (left) and an approximate lower triangular factor  $\mathbf{L}_0$  (right).

where  $\mathbf{L}_0$  is determined to have a particular sparsity structure.

*Example 9.3.5.* Consider a Dirichlet problem for Poisson's equation, which has the nonzero structure shown in Figure 9.3.6. Ordering the equations by rows, we have

$$(\mathbf{Ax})_n = -x_n + \theta_x(x_{n-1} + x_{n+1}) + \theta_y(x_{n-J} + x_{n+J}). \quad (9.3.29a)$$

If equation  $n$  corresponds to mesh point  $(j, k)$ , we may also write (9.3.29a) in the (double-subscripted) mesh notation as

$$(\mathbf{Ax})_{jk} = -x_{jk} + \theta_x(x_{j-1,k} + x_{j+1,k}) + \theta_y(x_{j,k-1} + x_{j,k+1}). \quad (9.3.29b)$$

We insist that

$$(\mathbf{L}_0 \mathbf{x})_n = x_n + b_n x_{n-1} + c_n x_{n-J}. \quad (9.3.30a)$$

Taking a transpose

$$(\mathbf{L}_0^T \mathbf{x})_n = x_n + b_{n+1} x_{n+1} + c_{n+J} x_{n+J}. \quad (9.3.30b)$$

Likewise

$$(\mathbf{D}_0 \mathbf{x})_n = d_n x_n. \quad (9.3.30c)$$

Multiplying (9.3.30b) and (9.3.30c)

$$(\mathbf{D}_0 \mathbf{L}_0^T \mathbf{x})_n = d_n(x_n + b_{n+1}x_{n+1} + c_{n+J}x_{n+J}).$$

Now, using (9.3.30a)

$$(\mathbf{L}_0 \mathbf{D}_0 \mathbf{L}_0^T \mathbf{x})_n = (\mathbf{D}_0 \mathbf{L}_0^T \mathbf{x})_n + b_n(\mathbf{D}_0 \mathbf{L}_0^T \mathbf{x})_{n-1} + c_n(\mathbf{D}_0 \mathbf{L}_0^T \mathbf{x})_{n-J}.$$

Combining the above two equations

$$\begin{aligned} (\mathbf{L}_0 \mathbf{D}_0 \mathbf{L}_0^T \mathbf{x})_n &= d_n(x_n + b_{n+1}x_{n+1} + c_{n+J}x_{n+J}) + b_n d_{n-1}(x_{n-1} + b_n x_n + c_{n+J-1}x_{n+J-1}) + \\ &\quad c_n d_{n-J}(x_{n-J} + b_{n+1-J}x_{n+1-J} + c_n x_n). \end{aligned}$$

Regrouping terms

$$\begin{aligned} (\mathbf{L}_0 \mathbf{D}_0 \mathbf{L}_0^T \mathbf{x})_n &= (d_n + d_{n-1}b_n^2 + d_{n-J}c_n^2)x_n + d_{n-1}b_n x_{n-1} + d_n b_{n+1} x_{n+1} + \\ &\quad d_n c_{n+J} x_{n+J} + d_{n-J} c_n x_{n-J} + b_{n+1-J} d_{n-J} c_n x_{n+1-J} + b_n d_{n-1} c_{n+J-1} x_{n+J-1}. \end{aligned} \quad (9.3.31)$$

Thus,  $\mathbf{L}_0 \mathbf{D}_0 \mathbf{L}_0^T$  has seven non-trivial bands as shown in Figure 9.3.7.

Using incomplete factorization, we equate the coefficients of  $x_n$ ,  $x_{n+1}$ , and  $x_{n+J}$  in (9.3.31) to those of  $(\mathbf{A}\mathbf{x})_n$  in (9.3.29a). This yields

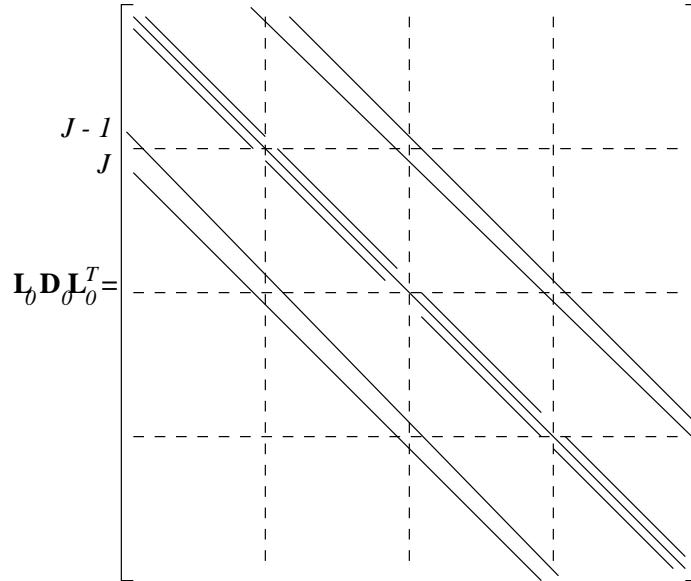
$$d_n = -1 - d_{n-1}b_n^2 - d_{n-J}c_n^2, \quad n = 1, 2, \dots, N. \quad (9.3.32a)$$

$$b_{n+1} = \theta_x/d_n, \quad n = 1, 2, \dots, N-1. \quad (9.3.32b)$$

$$c_{n+J} = \theta_y/d_n, \quad n = 1, 2, \dots, N-J. \quad (9.3.32c)$$

*Example 9.3.6.* ([5], Section 14.5). Consider the solution of Laplace's equation on the unit square using a square mesh ( $\theta_x = \theta_y = 1/4$ ) with  $h \times h$  spacing by the 9-point approximation of the Laplacian

$$\frac{1}{6}(U_{j+1,k+1} + U_{j+1,k-1} + U_{j-1,k+1} + U_{j-1,k-1})$$

Figure 9.3.7: The seven-band structure of  $\mathbf{L}_0 \mathbf{D}_0 \mathbf{L}_0^T$ .

$$+\frac{2}{3}(U_{j+1,k} + U_{j-1,k} + U_{j,k+1} + U_{j,k-1}) - \frac{10}{3}U_{jk} = 0.$$

The 9-point approximation of the Laplacian is accurate to  $O(h^4)$  compared to the  $O(h^2)$  accuracy of the 5-point formula. The Dirichlet boundary conditions are established so that the exact solution is

$$u(x, y) = e^{3x} \sin 3y.$$

The problem is solved using the PCG method with the following preconditioners:

- *None*. No preconditioning,
- *SSOR-5*. SSOR iteration using the 5-point approximation of the Laplacian,
- *SSOR-9*. SSOR iteration using the 9-point approximation of the Laplacian, and
- *ILU(0)-5*. Incomplete Cholesky factorization of the 5-point Laplacian with no fill in.

Each method was started with a trivial solution in the interior and one that satisfied the exact solution on the boundary. The iteration was terminated when the difference between successive iterates in the  $\mathcal{L}^2$  norm was less than  $10^{-10}$ . Each SSOR method used  $\omega = 2/(1 + \pi h)$ . The number of iterations to reach convergence are recorded in Table 9.3.2.

$1/h$	None	ILU(0)-5	SSOR-5	SSOR-9
10	28	16	18	16
20	57	28	25	23
40	112	52	34	32

Table 9.3.3: Number of iterations to convergence for the PCG solution of Example 9.3.6 using the indicated preconditioners.

The number of iterations for the two SSOR preconditioners increases as  $O(h^{-1/2})$ . The more accurate SSOR-9 iterative preconditioning differs little from the SSOR-5 preconditioning. The number of iterations for the incomplete factorization preconditioner is increasing slower than  $O(1/h)$  but faster than  $O(h^{-1/2})$ . All preconditioners in the table are better than using the conjugate gradient method without preconditioning.

Thus far, we have used incomplete factorization without allowing any fill in. In Example 9.3.6, we referred to this preconditioning as ILU(0). The ILU(0) preconditioning may not provide enough acceleration to the basic conjugate gradient algorithm. Thus, we consider continuing the factorization to additional levels by allowing some fill in. We will call a preconditioner ILU( $p$ ) if there are  $p$  levels of fill in. The next preconditioner beyond ILU(0) allows one level of fill in. The structure of  $\mathbf{L}_1$  for this ILU(1) algorithm is shown on the right of Figure 9.3.8. One row nearest the bandwidth has been filled in. It's simplest to imagine  $\mathbf{A}$  as having the two bands corresponding to the filled band of  $\mathbf{L}_1$  as also being filled in. This situation is shown on the upper right of Figure 9.3.8. The product  $\mathbf{L}_1 \mathbf{D} \mathbf{L}_1^T$  fills in yet another band (*cf.* the lower portion of Figure 9.3.8), which is ignored in the elimination.

An algorithm for performing incomplete factorization to level  $p$  for elliptic problems on rectangular meshes follows the procedure described in Example 9.3.5. Let's, instead, illustrate the method for a more general sparse matrix. This is typically done by defining a *level of fill* for each element created by Gaussian elimination. The assumption is that elements get smaller as the fill in progresses. A size  $\epsilon^k$ ,  $0 < \epsilon < 1$ , is assigned to an element at fill level  $k$ . Initially, a nonzero element has a unit level of fill and a zero element has an infinite level of fill. Rather than work with the  $\mathbf{LDL}^T$  factorization, let's simplify our task and work with a more general  $\mathbf{LU}$  factorization such as (9.1.3). The

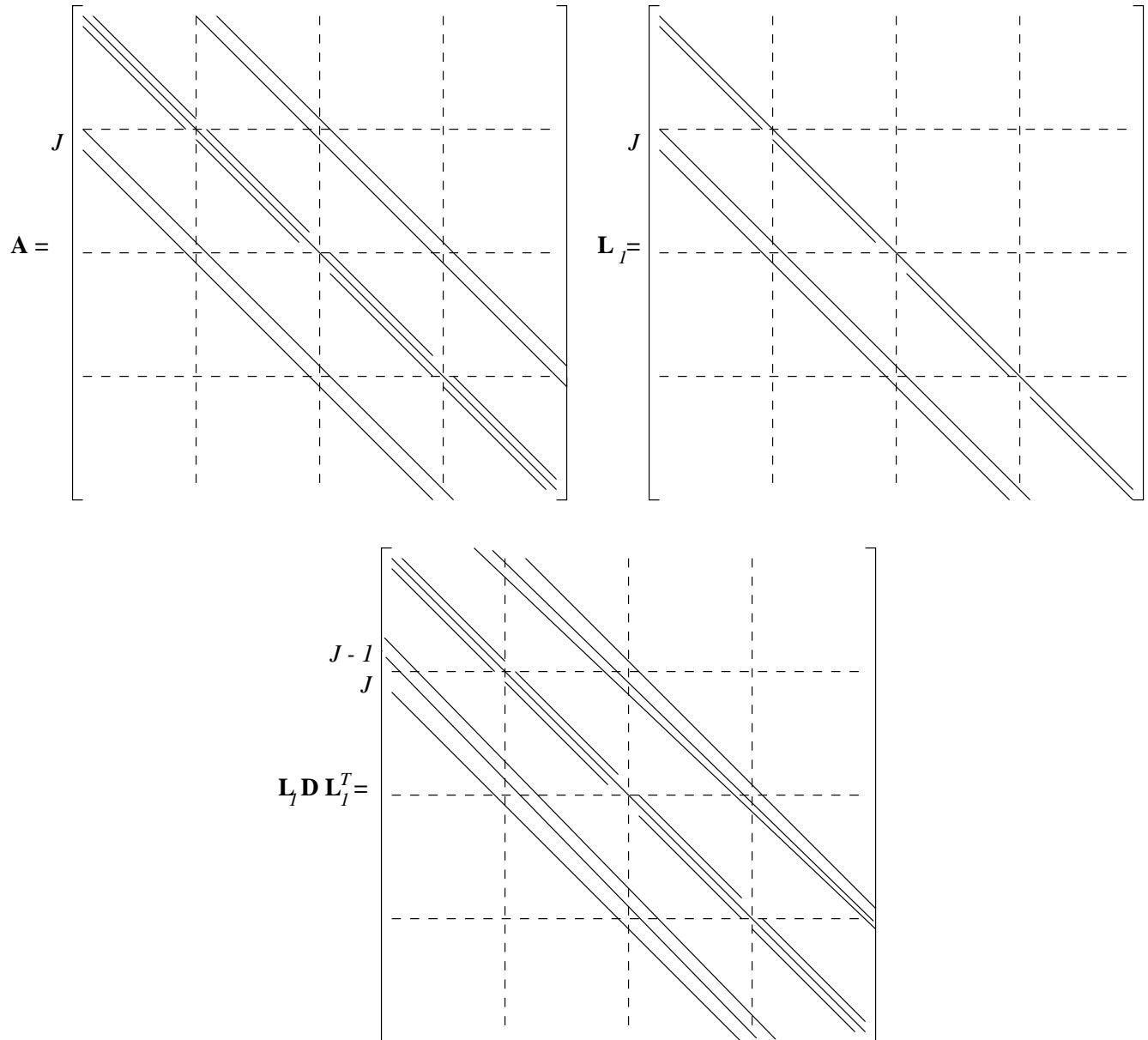


Figure 9.3.8: Imagined structure of  $\mathbf{A}$  (top left) and structure of  $\mathbf{L}_I$  (top right) for an ILU(1) preconditioner. The fill in when calculating  $\mathbf{L}_I \mathbf{D} \mathbf{L}_I^T$  is shown at the bottom.

computation is generated within a loop and, typically, the elements of  $\mathbf{L}$  replace the elements in the lower triangular portion of  $\mathbf{A}$  and the elements of  $\mathbf{U}$  replace the elements in the upper triangular portion of  $\mathbf{A}$ . A sample loop structure for the elimination is shown in Figure 9.3.9. This is very basic Gaussian elimination. No pivoting is performed and sparsity is not included.

```

for  $i = 2$  to  $N$  do
  for  $k = 1$  to  $i - 1$  do
     $a_{ik} = a_{ik}/a_{kk}$ 
    for  $j = k + 1$  to  $N$  do
       $a_{ij} = a_{ij} - a_{ik}a_{kj}$ 
    end for
  end for
end for

```

Figure 9.3.9: Gaussian elimination loop.

The Gaussian elimination update in Figure 9.3.9 is

$$a_{ij} := a_{ij} - a_{ik}a_{kj}.$$

Let the current level of fill of  $a_{ij}$  be denoted as  $\text{lev}_{ij}$ . Then the size of updated element  $a_{ij}$  should be

$$\epsilon^{\text{lev}_{ij}} - \epsilon^{\text{lev}_{ik}}\epsilon^{\text{lev}_{kj}} = \epsilon^{\text{lev}_{ij}} - \epsilon^{(\text{lev}_{ik} + \text{lev}_{kj})}.$$

Roughly speaking, the size of the updated element  $a_{ij}$  will be the maximum of the two sizes  $\epsilon^{\text{lev}_{ij}}$  and  $\epsilon^{(\text{lev}_{ik} + \text{lev}_{kj})}$ . Thus, it makes sense to define the new level of fill as

$$\text{lev}_{ij} := \min(\text{lev}_{ij}, \text{lev}_{ik} + \text{lev}_{kj}).$$

It is common (*cf.* [4], Section 10.3) to shift all levels by -1 from this definition. Thus, the initial level of fill of  $a_{ij}$  is set as

$$\text{lev}_{ij} := \begin{cases} 0, & \text{if } a_{ij} \neq 0, \text{ or } i = j \\ \infty, & \text{otherwise} \end{cases}. \quad (9.3.33a)$$

The updated level of fill is

$$\text{lev}_{ij} := \min(\text{lev}_{ij}, \text{lev}_{ik} + \text{lev}_{kj} + 1). \quad (9.3.33b)$$

We see that  $\text{lev}_{ij}$  cannot increase during the factorization. Thus, if  $a_{ij} \neq 0$  in the original matrix  $\mathbf{A}$ ,  $\text{lev}_{ij}$  never becomes nonzero. Equations (9.3.33) suggest a natural way of discarding elements during the incomplete factorization. With an ILU( $p$ ) algorithm, we'll keep all elements whose level of fill does not exceed  $p$ . The *zero pattern* of  $\mathbf{A}$  for the ILU( $p$ ) procedure may be defined as the set

$$\mathcal{Z}_p(\mathbf{A}) := \{(i, j) | \text{lev}_{ij} > p\}. \quad (9.3.33c)$$

This definition is consistent with the ILU(0) preconditioning described in Example 9.3.5.

```

Define  $lev_{ij}$  according to (9.3.33a)
for  $i = 2$  to  $N$  do
    for  $k = 1$  to  $i - 1$  do
        if  $lev_{ik} \leq p$  then
             $a_{ik} = a_{ik}/a_{kk}$ 
            for  $j = k + 1$  to  $N$  do
                if  $lev_{ij} \leq p$  then
                     $a_{ij} = a_{ij} - a_{ik}a_{kj}$ 
                    Update  $lev_{ij}$  according to (9.3.33b)
                end if
            end for
            for  $j = k + 1$  to  $N$  do
                if  $lev_{ij} > p$  then
                     $a_{ij} = 0;$ 
                     $lev_{ij} = \infty$ 
                end if
            end for
        end if
    end for
end for

```

Figure 9.3.10: Incomplete ILU( $p$ ) factorization.

A pseudocode ILU( $p$ ) algorithm appears in Figure 9.3.10. It is intended to give the general idea of the procedure and, as stated, is not very efficient. It ignores sparsity and does too much searching for levels. These conditions can be remedied. For example, the search for levels does not depend on the values of the elements of  $\mathbf{A}$  and can be separated from the elimination process. It can, thus, be done in advance and be used for several matrices having the same structure. However, because the algorithm is based on levels of fill rather than numerical information, some large elements of  $\mathbf{A}$  may be dropped because of their locations rather than their sizes. This may result in a poor preconditioning and, hence, require more iterations than necessary for convergence.

A *threshold elimination procedure* (ILUT) would identify small elements of  $\mathbf{A}$  and set them to zero during the factorization. At its simplest, this can be done by modifying the algorithm of Figure 9.3.10 as follows:

1. eliminate the definition and updates of  $lev_{ij}$ ,
2. replace the tests on  $lev_{ik} \leq p$  and  $lev_{ij} \leq p$  by tests on  $a_{ik} \neq 0$  and  $a_{ij} \neq 0$ , respectively,
3. insert a rule for dropping  $a_{ik}$  following its calculation ( $a_{ik} = a_{ik}/a_{kk}$ ), and
4. replace the statements within the second  $j$  loop by rules for dropping  $a_{ij}$ .

Rules for dropping elements are described by Saad [4], Section 10.4. Briefly, he suggests

1. dropping elements  $a_{ik}$  (Item 3 above) and  $a_{ij}$  (Item 4 above) that are less than a tolerance  $\tau$  relative to the size (e.g., the  $L^2$  norm) of row  $i$ , and
2. only retaining the  $p$  largest elements  $a_{ij}$  (Item 4 above) of row  $i$  without dropping the diagonal element.

The goal of the second dropping rule is to control the number of elements per row (or column) of  $\mathbf{L}$ . The first rule avoids retaining unnecessarily small elements in the factorization.

### Problems

1. Consider the method of steepest descent and, using (9.3.1) and (9.3.2b), show that

$$E(\mathbf{y}) = \frac{1}{2}(\mathbf{y} - \mathbf{x})^T \mathbf{A}(\mathbf{y} - \mathbf{x}) - \frac{1}{2}\mathbf{x}^T \mathbf{A}\mathbf{x}.$$

2. Show that the SSOR procedure (9.3.25) has the form of (9.3.21) with  $\hat{\mathbf{M}}$  and  $\hat{\mathbf{N}}$  given by (9.3.26).
3. Indicate the algorithm for solving (9.3.27c - 9.3.27e) when  $\mathbf{D}$  and  $\mathbf{L}$  correspond to centered differencing of the two-dimensional Laplacian.

## 9.4 Krylov Subspace Methods

The conjugate gradient method is applicable to symmetric and positive definite linear systems. While many elliptic problems satisfy these conditions, there are important

problems that do not. Convection-diffusion equations, studied in Section 4.4 are a good example. The use of upwind differencing of the convective (first-derivative) terms will produce an unsymmetric algebraic system. Of course, if our goal is to solve (9.1.1), we could modify the problem to

$$\mathbf{A}^T \mathbf{A} \mathbf{x} = \mathbf{A}^T \mathbf{b}. \quad (9.4.1)$$

This system is symmetric and positive definite; hence, we can solve it using the steepest descent or conjugate gradient method. However, the condition number (9.3.4c) of  $\mathbf{A}^T \mathbf{A}$  is the square of that of  $\mathbf{A}$ , and this may lead to ill conditioning. Thus, we can expect slow convergence when applying this procedure.

An alternate possibility is to seek an approximate solution by *projection methods*. With such an approach, our goal is to find approximations

$$\tilde{\mathbf{x}} = \mathbf{x}^{(0)} + \mathbf{v} \quad (9.4.2a)$$

where  $\mathbf{x}^{(0)}$  is an initial guess and  $\mathbf{v}$  is an element of an  $M$ -dimensional ( $M \leq N$ ) subspace  $\mathcal{V}$  of  $\mathbb{R}^N$ . The subspace  $\mathcal{V}$  is called the *trial*, *candidate*, or *search space*. In order to determine  $\tilde{\mathbf{x}}$  we must specify  $M$  conditions or constraints for  $\mathbf{v}$  to satisfy. Using the *Petrov-Galerkin method*, we introduce a second  $M$ -dimensional subspace  $\mathcal{W}$  of  $\mathbb{R}^N$ , and determine  $\tilde{\mathbf{x}}$  such that

$$(\tilde{\mathbf{r}}, \mathbf{w}) = (\mathbf{b} - \mathbf{A}\tilde{\mathbf{x}}, \mathbf{w}) = 0, \quad \forall \mathbf{w} \in \mathcal{W}. \quad (9.4.2b)$$

Thus, the residual  $\mathbf{r}$  is orthogonal to all ( $M$  linearly independent) vectors  $\mathbf{w} \in \mathcal{W}$ . The subspace  $\mathcal{W}$  is called the *test* or *constraint space*. Orthogonality can involve the usual  $\mathcal{L}^2$  inner product

$$(\mathbf{v}, \mathbf{w}) := \mathbf{w}^T \mathbf{v}; \quad (9.4.2c)$$

however, we'll consider other inner products (*e.g.*, strain energy (9.3.12a)) as well.

Using (9.4.2a) in (9.4.2b)

$$(\mathbf{b} - \mathbf{A}(\mathbf{x}^{(0)} + \mathbf{v}), \mathbf{w}) = (\mathbf{r}^{(0)} - \mathbf{A}\mathbf{v}, \mathbf{w}) = 0, \quad \forall \mathbf{w} \in \mathcal{W}. \quad (9.4.3)$$

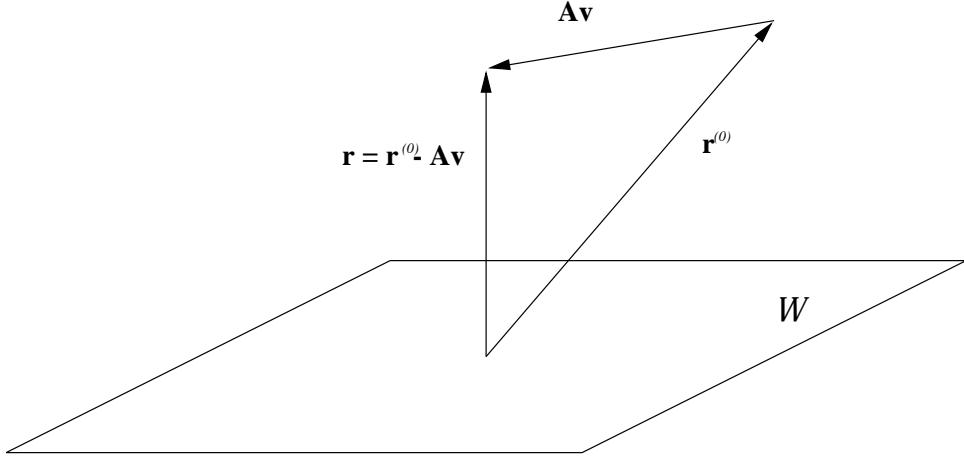


Figure 9.4.1: Orthogonal projection in three dimensions.

The orthogonality condition is now imposed on the “new residual”  $\mathbf{r}^{(0)} - \mathbf{A}\mathbf{v}$  and all vectors  $\mathbf{w} \in \mathcal{W}$ . This is illustrated for two- and three-dimensional spaces in Figure 9.4.1.

Two choices of the test space  $\mathcal{W}$  are important:  $\mathcal{W} = \mathbf{A}\mathcal{V}$  and  $\mathcal{W} = \mathcal{V}$ . In the first case, the solution of (9.4.2b) (or (9.4.3)) is optimal in the sense of minimizing  $\|\tilde{\mathbf{r}}\|_2$ ,  $\forall \mathbf{w} \in \mathbf{A}\mathcal{V}$ . To show this, let  $\tilde{\mathbf{x}}$  satisfy (9.4.2b), add a perturbation  $\delta\mathbf{x}$ , and consider the residual

$$\|\mathbf{r}\|_2^2 = (\mathbf{b} - \mathbf{A}(\tilde{\mathbf{x}} + \delta\mathbf{x}))^T(\mathbf{b} - \mathbf{A}(\tilde{\mathbf{x}} + \delta\mathbf{x})).$$

Expanding

$$\|\mathbf{r}\|_2^2 = \|\tilde{\mathbf{r}}\|_2^2 - 2(\mathbf{A}\delta\mathbf{x})^T(\mathbf{b} - \mathbf{A}\tilde{\mathbf{x}}) + \delta\mathbf{x}\mathbf{A}^T\mathbf{A}\delta\mathbf{x}.$$

With  $\delta\mathbf{x} \in \mathcal{V}$  and  $\mathbf{w} = \mathbf{A}\delta\mathbf{x} \in \mathcal{W}$ , we have from (9.4.2b)

$$(\mathbf{A}\delta\mathbf{x})^T(\mathbf{b} - \mathbf{A}\tilde{\mathbf{x}}) = \mathbf{0}. \quad (9.4.4)$$

Thus,

$$\|\mathbf{r}\|_2^2 = \|\tilde{\mathbf{r}}\|_2^2 + \delta\mathbf{x}\mathbf{A}^T\mathbf{A}\delta\mathbf{x}.$$

Since  $\delta\mathbf{x}\mathbf{A}^T\mathbf{A}\delta\mathbf{x}$  is non-negative definite, we establish  $\|\tilde{\mathbf{r}}\|_2^2$  as the minimum. The converse, *i.e.*, that minimizers of  $\|\mathbf{r}\|$  satisfy (9.4.2b), is also true and its proof follows similar arguments.

Using (9.4.2)

$$\tilde{\mathbf{r}} = \mathbf{b} - \mathbf{A}(\mathbf{x}^{(0)} - \mathbf{v}) = \mathbf{r}^{(0)} - \mathbf{A}\mathbf{v}.$$

Condition (9.4.4) enforces orthogonality between  $\tilde{\mathbf{r}}$  and all vectors in  $\mathbf{A}\mathcal{V}$ . Using the above relation, we say that  $\mathbf{Av}$  is the *orthogonal projection* of  $\mathbf{r}^{(0)}$  onto the subspace  $\mathcal{V}$  (Figure 9.4.1). Methods of this type are called *residual projection methods*.

When  $\mathcal{W} = \mathcal{V}$ , the method (9.4.2b) is called *Galerkin's method*. When  $\mathbf{A}$  is symmetric and positive definite, Galerkin's method is optimal in the sense of minimizing

$$\|\mathbf{x} - \mathbf{y}\|_A^2 = (\mathbf{x} - \mathbf{y})^T \mathbf{A}(\mathbf{x} - \mathbf{y}) \quad (9.4.5a)$$

for all  $\mathbf{y} = \mathbf{x}^{(0)} + \mathbf{v}$ ,  $\mathbf{v} \in \mathcal{V}$ . The appropriate inner product for use with this strain energy norm is

$$(\mathbf{v}, \mathbf{w}) := \mathbf{w}^T \mathbf{A} \mathbf{v}. \quad (9.4.5b)$$

Thus, if  $\tilde{\mathbf{x}}$  satisfies (9.4.2b), we have

$$\mathbf{w}^T \mathbf{A}^T (\mathbf{b} - \mathbf{Ax}) = \mathbf{0}, \quad \forall \mathbf{w} \in \mathcal{V}. \quad (9.4.5c)$$

Proving that  $\tilde{\mathbf{x}}$  also minimizes (9.4.5a), and conversely, follows the lines of the proof for residual projection methods and is left as an exercise (*cf.* Problem 1 at the end of this section).

If our aim is to address nonsymmetric systems, we will primarily be interested in residual projection methods; however, let us proceed in a more general manner for the moment and examine solution schemes for (9.4.2b). Let

$$\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_M], \quad \mathbf{W} = [\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_M] \quad (9.4.6a)$$

be  $N \times M$  matrices whose columns form bases for  $\mathcal{V}$  and  $\mathcal{W}$ , respectively. Then, letting

$$\mathbf{v} = \mathbf{V}\mathbf{c}, \quad \mathbf{w} = \mathbf{W}\mathbf{d}, \quad (9.4.7)$$

we write (9.4.3) as

$$(\mathbf{r}^{(0)} - \mathbf{AVc}, \mathbf{Wd}) = 0.$$

The requirement that this result hold for all  $\mathbf{w} \in \mathcal{W}$  is equivalent to it holding for all possible choices of  $\mathbf{d}$ . For this to occur, we must have

$$(\mathbf{r}^{(0)} - \mathbf{AVc}, \mathbf{W}) = \mathbf{0}.$$

When the inner product corresponds to (9.4.2c), we have

$$\mathbf{W}^T(\mathbf{r}^{(0)} - \mathbf{AVc}) = \mathbf{0}.$$

Thus,  $\mathbf{c}$  satisfies

$$\mathbf{W}^T \mathbf{AVc} = \mathbf{W}^T \mathbf{r}^{(0)}.$$

Assuming that the  $M \times M$  matrix  $\mathbf{W}^T \mathbf{AV}$  is invertible, we find

$$\mathbf{c} = (\mathbf{W}^T \mathbf{AV})^{-1} \mathbf{W}^T \mathbf{r}^{(0)} \quad (9.4.8a)$$

and, using (9.4.2a) and (9.3.5c),

$$\tilde{\mathbf{x}} = \mathbf{x}^{(0)} + \mathbf{V}(\mathbf{W}^T \mathbf{AV})^{-1} \mathbf{W}^T \mathbf{r}^{(0)}. \quad (9.4.8b)$$

Let's look at two methods where the subspaces  $\mathcal{V}$  and  $\mathcal{W}$  are one dimensional.

*Example 9.4.1.* Let  $\mathcal{V} = \mathcal{W} = \text{span}\{\mathbf{v}_1\}$  where  $\mathbf{v}_1$  is an  $N$ -vector and choose  $\mathbf{v}_1 = \mathbf{r}^{(0)}$ .

Then, using (9.4.8)

$$\mathbf{c} = \frac{(\mathbf{r}^{(0)})^T \mathbf{r}^{(0)}}{(\mathbf{r}^{(0)})^T \mathbf{A} \mathbf{r}^{(0)}}$$

and

$$\tilde{\mathbf{x}} = \mathbf{x}^{(0)} + c \mathbf{r}^{(0)}.$$

We recognize this as being one step of the steepest descent algorithm (Figure 9.3.1).

*Example 9.4.2.* In minimum residual (MR) iteration, we select  $\mathcal{V} = \text{span}\{\mathbf{v}_1\} = \mathbf{r}^{(0)}$  and  $\mathcal{W} = \text{span}\{\mathbf{w}_1\} = \mathbf{A} \mathbf{r}^{(0)}$ . Using (9.4.8)

$$c = \frac{(\mathbf{r}^{(0)})^T \mathbf{A}^T \mathbf{r}^{(0)}}{(\mathbf{r}^{(0)})^T \mathbf{A}^T \mathbf{A} \mathbf{r}^{(0)}}$$

and

$$\tilde{\mathbf{x}} = \mathbf{x}^{(0)} + c \mathbf{r}^{(0)}.$$

The matrix  $\mathbf{A}$  need not be symmetric but only be positive definite for MR to converge ([4], Section 5.3).

Many iterative techniques for symmetric and non-symmetric matrices utilize Krylov subspaces for the trial space  $\mathcal{V}$ . A *Krylov subspace* of dimension  $M$  is

$$\mathcal{K}_M(\mathbf{A}, \mathbf{r}^{(0)}) = \text{span}\{\mathbf{r}^{(0)}, \mathbf{A} \mathbf{r}^{(0)}, \mathbf{A}^2 \mathbf{r}^{(0)}, \dots, \mathbf{A}^{M-1} \mathbf{r}^{(0)}\}. \quad (9.4.9)$$

It seems clear that vectors  $\mathbf{v} \in \mathcal{K}_M$  have the form  $\mathbf{v} = p(\mathbf{A})\mathbf{r}^{(0)}$  where  $p(\mathbf{A})$  is a polynomial of degree not exceeding  $M - 1$  in  $\mathbf{A}$ . Methods using Krylov subspaces as trial spaces are called *Krylov subspace methods* or simply *Krylov* methods. Different approaches are distinguished by their choice of the test space  $\mathcal{W}$ , with the two most popular choices being  $\mathcal{K}_M$  and  $\mathbf{A}\mathcal{K}_M$ . The *generalized minimal residual* (GMRES) method, in particular, uses the latter choice. Thus, if we define the Krylov matrix

$$\mathbf{K}_M = [\mathbf{r}^{(0)}, \mathbf{A}\mathbf{r}^{(0)}, \mathbf{A}^2\mathbf{r}^{(0)}, \dots, \mathbf{A}^M\mathbf{r}^{(0)}],$$

we could select

$$\mathbf{V}_M = \mathbf{K}_M, \quad \mathbf{W}_M = \mathbf{A}\mathbf{K}_M,$$

choose the  $M$  *th* solution iterate according to (9.4.2a) and (9.3.5c) as

$$\mathbf{x}^{(M)} = \mathbf{x}^{(0)} + \mathbf{V}_M \mathbf{c}^{(M)}, \quad (9.4.10)$$

and determine the  $M$  *th*-GMRES iterate from (9.4.8a) as

$$\mathbf{c}^{(M)} = (\mathbf{K}_M^T \mathbf{A}^T \mathbf{A} \mathbf{K}_M)^{-1} \mathbf{K}_M^T \mathbf{A}^T \mathbf{r}^{(0)}.$$

We see that:

1. The GMRES procedure is the extension of the MR procedure to higher-dimensional spaces.
2. The dimension of the Krylov space increases by one after each iteration.
3. As may be expected from our earlier work,  $\mathbf{x}^{(M)}$  minimizes

$$\|\mathbf{r}^{(M)}\|_2 = \|\mathbf{b} - \mathbf{A}\mathbf{x}^{(M)}\|_2$$

Unfortunately, the procedure just outlined is unstable (sensitive to round-off error accumulation). It will be necessary to select a more well conditioned basis for  $\mathcal{K}_M$  and we'll do this by constructing a set of mutually orthogonal vectors. This orthogonal projection onto  $\mathcal{K}_M$  is usually done by Arnoldi's method and a pseudocode procedure for its construction appears in Figure 9.4.2. The procedure begins with an initial vector  $\mathbf{v}_1$

which is the initial residual  $\mathbf{r}^{(0)}$  normalized to have a unit Euclidean length. Successive vectors  $\mathbf{v}_j$  are chosen to be orthogonal to the previous ones and to have unit lengths. Indeed, the algorithm of Figure 9.4.2 is just classical Gram-Schmidt orthogonalization modified by the presence of  $\mathbf{A}$  to produce an orthonormal basis for the Krylov subspace

$$\mathcal{K}_M(\mathbf{A}, \mathbf{v}_1) = \text{span}\{\mathbf{v}_1, \mathbf{A}\mathbf{v}_1, \mathbf{A}^2\mathbf{v}_1, \dots, \mathbf{A}^{M-1}\mathbf{v}_1\}. \quad (9.4.11)$$

The algorithm will halt prematurely if any unnormalized vector  $\mathbf{w}_j$  has a zero Euclidean length. The Boolean variable *quit* is set to **true** should this happen. Proving these assertions may be done with an induction argument ([4], Section 6.3), but we'll proceed less formally. At the first ( $j = 1$ ) step of the algorithm, we obtain

```

procedure arnoldi
    quit = ( $\|\mathbf{r}^{(0)}\|_2 = 0$ )
    if not quit then
         $\mathbf{v}_1 = \mathbf{r}^{(0)} / \|\mathbf{r}^{(0)}\|_2$ 
    end if
     $j = 1$ 
    while ( $j \leq M$ ) and (not quit) do
         $\mathbf{w}_j = \mathbf{A}\mathbf{v}_j$ 
        for  $i = 1$  to  $j$  do
             $h_{ij} = \mathbf{v}_i^T \mathbf{A}\mathbf{v}_j$ 
             $\mathbf{w}_j = \mathbf{w}_j - h_{ij}\mathbf{v}_i$ 
        end for
         $h_{j+1,j} = \|\mathbf{w}_j\|_2$ 
        quit = ( $h_{j+1,j} = 0$ )
        if not quit then
             $\mathbf{v}_{j+1} = \mathbf{w}_j / h_{j+1,j}$ 
        end if
         $j = j + 1$ 
    end while

```

Figure 9.4.2: Arnoldi Gram-Schmidt orthogonal basis construction for  $\mathcal{K}_M$ .

$$h_{21}\mathbf{v}_2 = \mathbf{w}_1 = \mathbf{A}\mathbf{v}_1 - h_{11}\mathbf{v}_1.$$

Taking an inner product with  $\mathbf{v}_1$

$$h_{21}\mathbf{v}_1^T \mathbf{v}_2 = \mathbf{v}_1^T \mathbf{A}\mathbf{v}_1 - h_{11}\mathbf{v}_1^T \mathbf{v}_1.$$

However,  $\mathbf{v}_1^T \mathbf{v}_1 = 1$  and  $h_{11} = \mathbf{v}_1^T \mathbf{A} \mathbf{v}_1$ , so  $\mathbf{v}_1$  is orthogonal to  $\mathbf{v}_2$ . It is also clear that  $\mathbf{v}_2$  is a linear combination of  $\mathbf{v}_1$  and  $\mathbf{A}\mathbf{v}_1$  and, hence, an element of  $\mathcal{K}_2(\mathbf{A}, \mathbf{v}_1)$ . The extension of the induction argument to higher values of  $j$  is similar.

From the **for** loop within Arnoldi's algorithm, we may infer

$$h_{j+1,j} \mathbf{v}_{j+1} = \mathbf{A} \mathbf{v}_j - \sum_{i=1}^j h_{ij} \mathbf{v}_i, \quad j = 1, 2, \dots, M, \quad (9.4.12a)$$

or

$$\mathbf{A} \mathbf{v}_j = \sum_{i=1}^{j+1} h_{ij} \mathbf{v}_i, \quad j = 1, 2, \dots, M. \quad (9.4.12b)$$

Let  $\mathbf{V}_M$  be the  $N \times M$  matrix whose columns are  $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_M$  and let  $\bar{\mathbf{H}}_M$  be the  $(M+1) \times M$  Hessenberg matrix

$$\bar{\mathbf{H}}_M = \begin{bmatrix} h_{11} & h_{12} & \cdots & h_{1M} \\ h_{21} & h_{22} & \cdots & h_{2M} \\ & h_{32} & \cdots & h_{3M} \\ & & \ddots & \vdots \\ & & & h_{M+1,M} \end{bmatrix}. \quad (9.4.13)$$

Then (9.4.12) can be written in matrix form as

$$\mathbf{A} \mathbf{V}_M = \mathbf{V}_{M+1} \bar{\mathbf{H}}_M. \quad (9.4.14a)$$

Additionally, let  $\mathbf{H}_M$  be the matrix obtained by deleting the last row of  $\bar{\mathbf{H}}_M$ . Then, either by using (9.4.12a) or (9.4.14a), we have

$$\mathbf{A} \mathbf{V}_M = \mathbf{V}_M \mathbf{H}_M + \mathbf{w}_M \mathbf{e}_M^T \quad (9.4.14b)$$

where, from Arnoldi's algorithm,  $\mathbf{w}_M = h_{M+1,M} \mathbf{v}_{M+1}$  and  $\mathbf{e}_M$  is the  $M$ th column of the identity matrix. Finally, multiplying (9.4.14b) by  $\mathbf{V}_M^T$  and using the orthogonality of its columns, we obtain

$$\mathbf{V}_M^T \mathbf{A} \mathbf{V}_M = \mathbf{H}_M. \quad (9.4.14c)$$

The Arnoldi algorithm as stated in Figure 9.4.2 is subject to round-off error accumulation. The modified Gram-Schmidt procedure, as illustrated in Figure 9.4.3, is much

more stable and less sensitive to round-off error difficulties. Mathematically, the steps in the two Arnoldi procedures are identical; however, the modified Gram-Schmidt version avoids cancellations of nearly equal quantities. In cases of severe round-off error accumulation, Householder transformations can be used to construct the orthogonal basis. We will not do this here, but interested readers may consult Saad [4], Section 6.3.

```

procedure marnoldi
  quit = ( $\|\mathbf{r}^{(0)}\|_2 = 0$ )
  if not quit then
     $\mathbf{v}_1 = \mathbf{r}^{(0)}/\|\mathbf{r}^{(0)}\|_2$ 
  end if
   $j = 1$ 
  while ( $j \leq M$ ) and (not quit) do
     $\mathbf{w}_j = \mathbf{A}\mathbf{v}_j$ 
    for  $i = 1$  to  $j$  do
       $h_{ij} = \mathbf{v}_i^T \mathbf{w}_j$ 
       $\mathbf{w}_j = \mathbf{w}_j - h_{ij} \mathbf{v}_i$ 
    end for
     $h_{j+1,j} = \|\mathbf{w}_j\|_2$ 
    quit = ( $h_{j+1,j} = 0$ )
    if not quit then
       $\mathbf{v}_{j+1} = \mathbf{w}_j/h_{j+1,j}$ 
    end if
     $j = j + 1$ 
  end while

```

Figure 9.4.3: Arnoldi modified Gram-Schmidt orthogonal basis construction for  $\mathcal{K}_M$ .

We are now in a position to improve the stability of the GMRES procedure. Again, choose  $\mathbf{x}^{(M)}$  according to (9.4.10) and calculate the residual

$$\mathbf{b} - \mathbf{Ax}^{(M)} = \mathbf{b} - \mathbf{A}(\mathbf{x}^{(0)} - \mathbf{V}_M \mathbf{c}^{(M)}) = \mathbf{r}^{(0)} - \mathbf{AV}_M \mathbf{c}^{(M)}.$$

Using (9.4.14)

$$\mathbf{b} - \mathbf{Ax}^{(M)} = \beta \mathbf{v}_1 - \mathbf{V}_{M+1} \bar{\mathbf{H}}_M \mathbf{c}^{(M)} = \mathbf{V}_{M+1} (\beta \mathbf{e}_1 - \bar{\mathbf{H}}_M \mathbf{c}^{(M)}) \quad (9.4.15a)$$

where  $\mathbf{e}_1$  is the first column of the identity matrix and (Figure 9.4.3)

$$\beta = \|\mathbf{r}^{(0)}\|_2. \quad (9.4.15b)$$

Since the GMRES approximation minimizes  $\|\mathbf{b} - \mathbf{Ax}^{(M)}\|_2$ , our task is to find  $\mathbf{c}^{(M)}$  as the minimizer of  $\|\mathbf{V}_{M+1}(\beta\mathbf{e}_1 - \bar{\mathbf{H}}_M \mathbf{c}^{(M)})\|_2$ . Actually, since  $\mathbf{V}_M$  is orthogonal, it suffices to minimize  $\|\beta\mathbf{e}_1 - \bar{\mathbf{H}}_M \mathbf{c}^{(M)}\|_2$ . The approximate solution is then given by (9.4.10).

The minimizer  $\mathbf{c}^{(M)}$  is computationally inexpensive. It requires the solution of an  $(M + 1) \times M$  least-squares problem where  $M$  is typically small relative to  $N$ . An algorithm for the GMRES procedure appears in Figure 9.4.4. The procedure is basically the Arnoldi-modified Gram-Schmidt algorithm with a least-squares procedure. Some additional comments follow.

**procedure** GMRES

```

 $\bar{\mathbf{H}}_M = \mathbf{0}$ 
 $\mathbf{r}^{(0)} = \mathbf{b} - \mathbf{Ax}^{(0)}$ 
 $\beta = \|\mathbf{r}^{(0)}\|_2$ 
 $quit = (\beta = 0)$ 
if not  $quit$  then
     $\mathbf{v}_1 = \mathbf{r}^{(0)} / \beta$ 
end if
 $j = 1$ 
while ( $j \leq M$ ) and (not  $quit$ ) do
     $\mathbf{w}_j = \mathbf{Av}_j$ 
    for  $i = 1$  to  $j$  do
         $h_{ij} = \mathbf{v}_i^T \mathbf{w}_j$ 
         $\mathbf{w}_j = \mathbf{w}_j - h_{ij} \mathbf{v}_i$ 
    end for
     $h_{j+1,j} = \|\mathbf{w}_j\|_2$ 
     $quit = (h_{j+1,j} = 0)$ 
    if not  $quit$  then
         $\mathbf{v}_{j+1} = \mathbf{w}_j / h_{j+1,j}$ 
    end if
     $j = j + 1$ 
end while
 $m = j - 1$ 
Determine  $\mathbf{c}^{(M)}$  as the minimizer of  $\|\beta\mathbf{e}_1 - \bar{\mathbf{H}}_M \mathbf{c}^{(M)}\|_2$ 
 $\mathbf{x}^{(M)} = \mathbf{x}^{(0)} + \mathbf{V}_M \mathbf{c}^{(M)}$ 

```

Figure 9.4.4: Generalized minimum residual algorithm.

- As stated, the GMRES procedure does not calculate the solution at each step. Thus, it is difficult to know when to stop. It would be better to calculate  $\mathbf{x}^{(j)}$

during the procedure and to check for convergence by, *e.g.*, monitoring the size of the residuals.

2. If the procedure terminates before completing  $M$  steps ( $quit = \text{true}$ ), then  $\mathbf{x}^{(j)}$  is the exact solution. This is the only way that the GMRES procedure can terminate prematurely. Although shown, the least-squares solution need not be calculated in this case.
3. The GMRES procedure becomes impractical because of the growth of memory and computational requirements when  $M$  becomes too large. One remedy is to restart the orthogonalization after a fixed number ( $M$ ) steps using the last computed solution  $\mathbf{x}^{(M)}$  as an initial guess. This procedure tends to converge slowly when  $\mathbf{A}$  is not positive definite.
4. A second technique for restricting the size of  $M$  is to truncate the Arnoldi process. With this approach, only the last  $k$  vectors are made orthogonal. The first  $M - k$  vectors can be discarded and need not be stored.

The only remaining detail is the solution of the least-squares problem

$$\min_{\mathbf{c}} \|\beta \mathbf{e}_1 - \bar{\mathbf{H}}_M \mathbf{c}\|_2.$$

Since  $\bar{\mathbf{H}}$  is a Hessenberg matrix, it seems reasonable to transform it to upper triangular form. We'll do this by using plane rotations (Givens matrices) of the form

$$\Omega_i = \begin{bmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & & c_i & s_i \\ & & & -s_i & c_i \\ & & & & 1 \\ & & & & & \ddots \\ & & & & & & 1 \end{bmatrix} \quad \begin{array}{l} \text{row } i \\ \text{row } i+1 \end{array} \quad (9.4.16a)$$

with

$$c_i^2 + s_i^2 = 1. \quad (9.4.16b)$$

We multiply  $\beta\mathbf{e}_1 - \bar{\mathbf{H}}_M \mathbf{c}$  on the left by a sequence of rotations  $\Omega_i$  with the coefficients  $c_i$  and  $s_i$  designed to eliminate  $h_{i+1,i}$ ,  $i = 1, 2, \dots, M$ . If  $M$  were five, we would have

$$\bar{\mathbf{H}}_M = \begin{bmatrix} h_{11} & h_{12} & h_{13} & h_{14} & h_{15} \\ h_{21} & h_{22} & h_{23} & h_{24} & h_{25} \\ h_{32} & h_{33} & h_{34} & h_{35} & \\ h_{43} & h_{44} & h_{45} & & \\ h_{54} & h_{55} & & & \\ h_{65} & & & & \end{bmatrix}, \quad \bar{\mathbf{g}} = \begin{bmatrix} \beta \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

We multiply  $\bar{\mathbf{H}}_M$  and  $\bar{\mathbf{g}}$  by

$$\Omega_1 = \begin{bmatrix} c_1 & s_1 & & & \\ -s_1 & c_1 & & & \\ & & 1 & & \\ & & & 1 & \\ & & & & 1 \end{bmatrix}$$

with

$$c_1 = \frac{h_{11}}{\sqrt{h_{11}^2 + h_{21}^2}} \quad s_1 = \frac{h_{21}}{\sqrt{h_{11}^2 + h_{21}^2}}$$

to obtain

$$\bar{\mathbf{H}}_M^{(1)} = \begin{bmatrix} h_{11}^{(1)} & h_{12}^{(1)} & h_{13}^{(1)} & h_{14}^{(1)} & h_{15}^{(1)} \\ h_{22}^{(1)} & h_{23}^{(1)} & h_{24}^{(1)} & h_{25}^{(1)} & \\ h_{32} & h_{33} & h_{34} & h_{35} & \\ h_{43} & h_{44} & h_{45} & & \\ h_{54} & h_{55} & & & \\ h_{65} & & & & \end{bmatrix}, \quad \bar{\mathbf{g}}^{(1)} = \begin{bmatrix} c_1\beta \\ -s_1\beta \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

The process continues with the parameters for rotation  $i$  satisfying

$$c_i = \frac{h_{ii}^{(i-1)}}{\sqrt{(h_{ii}^{(i-1)})^2 + h_{i+1,i}^2}} \quad s_i = \frac{h_{i+1,i}}{\sqrt{(h_{ii}^{(i-1)})^2 + h_{i+1,i}^2}}. \quad (9.4.16c)$$

At the end of  $M$  ( $= 5$ ) rotations, we have

$$\bar{\mathbf{H}}_M^{(5)} = \begin{bmatrix} h_{11}^{(5)} & h_{12}^{(5)} & h_{13}^{(5)} & h_{14}^{(5)} & h_{15}^{(5)} \\ h_{22}^{(5)} & h_{23}^{(5)} & h_{24}^{(5)} & h_{25}^{(5)} & \\ h_{33}^{(5)} & h_{34}^{(5)} & h_{35}^{(5)} & & \\ h_{44}^{(5)} & h_{45}^{(5)} & & & \\ h_{55}^{(5)} & & & & \\ 0 & & & & \end{bmatrix}, \quad \bar{\mathbf{g}}^{(5)} = \begin{bmatrix} \gamma_1 \\ \gamma_2 \\ \vdots \\ \gamma_6 \end{bmatrix}.$$

In the general case, let

$$\mathbf{Q}_M = \Omega_M \Omega_{M-1} \dots \Omega_1 \quad (9.4.17a)$$

and

$$\bar{\mathbf{R}}_M = \bar{\mathbf{H}}_M^{(M)} = \mathbf{Q}_M \bar{\mathbf{H}}_M, \quad \bar{\mathbf{g}}_M = \bar{\mathbf{g}}_M^{(M)} = \mathbf{Q}_M \beta \mathbf{e}_1. \quad (9.4.17b)$$

Since the matrices  $\Omega_i$ ,  $i = 1, 2, \dots, M$ , and  $\mathbf{Q}_M$  are orthogonal,

$$\min_{\mathbf{c}} \|\beta \mathbf{e}_1 - \bar{\mathbf{H}}_M \mathbf{c}\|_2 = \min_{\mathbf{c}} \|\bar{\mathbf{g}}_M - \bar{\mathbf{R}}_M \mathbf{c}\|_2.$$

The solution of this problem is obtained by deleting the last row of  $\bar{\mathbf{R}}_M$  and  $\bar{\mathbf{g}}_M$  and solving the resulting upper triangular linear system for  $\mathbf{c}^{(M)}$ . Thus, let  $\mathbf{R}_M$  be the  $M \times M$  matrix obtained by deleting the last row of  $\bar{\mathbf{R}}_M$  and  $\mathbf{g}_M$  be the  $M$ -vector obtained by deleting the last element of  $\bar{\mathbf{g}}_M$ , then

$$\mathbf{c}^{(M)} = \mathbf{R}_M^{-1} \mathbf{g}_M. \quad (9.4.18)$$

*Example 9.4.3.* Saad [4] solves the partial differential equation

$$-u_{xx} - u_{yy} - u_{zz} + 10(e^{xy}u)_x + 10(e^{-xy}u)_y = f(x, y, z)$$

on a unit square with trivial Dirichlet boundary conditions. He chooses a  $17 \times 17 \times 17$  mesh and uses centered differencing. The function  $f(x, y, z)$  is chosen so that the right side  $\mathbf{b}$  of the discrete problem (9.1.1) satisfies

$$\mathbf{b} = \mathbf{A}[1, 1, \dots, 1]^T.$$

With Dirichlet boundary conditions, the dimension of the discrete system is  $(16)^3 = 4096$ . Saad [4] solves this system by GMRES procedures with and without preconditioning. The results of some of his calculations are reported in Table 9.4. The column labeled *Iter* reports the number of matrix-by-vector multiplications to reduce the initial residual by a factor of  $10^7$  in the  $\mathcal{L}^2$  norm. The parameter *Kflops* reports the number of floating point operations performed divided by 1000. The parameters *Residual* and *Error* record the residual and error in the  $\mathcal{L}^2$  norm. The first row of Table 9.4 reports data for restarted GMRES with a Krylov space of dimension  $M = 10$ . The second row displays data for GMRES calculations performed with a truncated orthogonalization using  $k = 10$  columns. The third and fourth rows contain results of preconditioned GMRES calculations using, respectively, SSOR and ILU(0) preconditioners. The acceleration parameter

$\omega$  was set to unity for the SSOR technique. The preconditioned versions increased the dimension of the Krylov space until convergence was attained. All procedures were initiated with a random initial guess.

Method	Iter	Kflops	Residual	Error
Restarted GMRES	67	11862	0.37(-3)	0.28(-3)
Truncated GMRES	75	22798	0.64(-3)	0.32(-3)
SSOR-GMRES	20	4870	0.14(-2)	0.30(-3)
ILU(0)-GMRES	17	4004	0.52(-3)	0.30(-3)

Table 9.4.1: Number of iterations (Iter) to convergence, number of floating point operations (Kflops), and the residual and error in  $\mathcal{L}^2$  for GMRES procedures with and without preconditioners (Example 9.4.3).

The results with preconditioning are much better than those without. The restarted and incomplete orthogonalization versions of GMRES have comparable performance. Likewise, the SSOR and ILU(0) preconditionings are comparable.

### Problems

- With  $\mathbf{A}$  being positive definite and symmetric, show that solutions  $\tilde{\mathbf{x}}$  of (9.4.5c) minimizes (9.4.5a), and conversely.



# Bibliography

- [1] J.A. George. Nested dissection of a regular finite element mesh. *SIAM Journal on Numerical Analysis*, 10:345–363, 1973.
- [2] J.A. George and J.W. Liu. *Computational Solution of Large Sparse Positive Definite Systems*. Prentice-Hall Series in Automatic Computation. Prentice-Hall, Englewood Cliffs, 1981.
- [3] E. Isaacson and H.B. Keller. *Analysis of Numerical Methods*. John Wiley and Sons, New York, 1966.
- [4] Y. Saad. *Iterative Methdos for Sparse Linear Systems*. PWS, Boston, 1996.
- [5] J.C. Strikwerda. *Finite Difference Schemes and Partial Differential Equations*. Chapman and Hall, Pacific Grove, 1989.
- [6] R.S. Varga. *Matrix Iterative Analysis*. Prentice-Hall, Englewood Cliffs, 1963.
- [7] E. L. Wachpress. *Iterative Solution of Elliptic Systems*. Prentice-Hall, Englewood Cliffs, 1966.