CHAPTER

FOUR

APPLICATION OF NUMERICAL METHODS TO SELECTED MODEL EQUATIONS

In this chapter we examine in detail various numerical schemes that can be used to solve simple model partial differential equations (PDEs). These model equations include the first-order wave equation, the heat equation, Laplace's equation, and Burgers' equation. These equations are called model equations because they can be used to "model" the behavior of more complicated PDEs. For example, the heat equation can serve as a model equation for other parabolic PDEs such as the boundary-layer equations. All of the present model equations have exact solutions for certain boundary and initial conditions. We can use this knowledge to quickly evaluate and compare numerical methods that we might wish to apply to more complicated PDEs. The various methods discussed in this chapter were selected because they illustrate the basic properties of numerical algorithms. Each of the methods exhibits certain distinctive features that are characteristic of a class of methods. Some of these features may not be desirable, but the method is included anyway for pedagogical reasons. Other very useful methods have been omitted because they are similar to those that are included. Space does not permit a discussion of all possible methods that could be used.

4.1 WAVE EQUATION

The one-dimensional (1-D) wave equation is a second-order hyperbolic PDE given by

$$\frac{\partial^2 u}{\partial t^2} = c^2 \frac{\partial^2 u}{\partial x^2} \tag{4.1}$$

This equation governs the propagation of sound waves traveling at a wave speed c in a uniform medium. A first-order equation that has properties similar to those of Eq. (4.1) is given by

$$\frac{\partial u}{\partial t} + c\frac{\partial u}{\partial x} = 0 \qquad c > 0 \tag{4.2}$$

Note that Eq. (4.1) can be obtained from Eq. (4.2). We will use Eq. (4.2) as our model equation and refer to it as the first-order 1-D wave equation, or simply the "wave equation." This linear hyperbolic equation describes a wave propagating in the x direction with a velocity c, and it can be used to "model" in a rudimentary fashion the nonlinear equations governing inviscid flow. Although we will refer to Eq. (4.2) as the wave equation, the reader is cautioned to be aware of the fact that Eq. (4.1) is the classical wave equation. More appropriately, Eq. (4.2) is often called the 1-D linear convection equation.

The exact solution of the wave equation [Eq. (4.2)] for the pure initial value problem with initial data

$$u(x,0) = F(x) \qquad -\infty < x < \infty \tag{4.3}$$

is given by

$$u(x,t) = F(x - ct) \tag{4.4}$$

Let us now examine some schemes that could be used to solve the wave equation.

4.1.1 Euler Explicit Methods

The following simple explicit one-step methods,

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} + c\frac{u_{j+1}^n - u_j^n}{\Delta x} = 0 \qquad c > 0 \tag{4.5}$$

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} + c\frac{u_{j+1}^n - u_{j-1}^n}{2\,\Delta x} = 0 \tag{4.6}$$

have truncation errors (T.E.s) of $O[\Delta t, \Delta x]$ and $O[\Delta t, (\Delta x)^2]$, respectively. We refer to these schemes as being first-order accurate, since the lowest-order term in the T.E. is first order, i.e., Δt and Δx for Eq. (4.5) and Δt for Eq. (4.6). These schemes are explicit, since only one unknown u_j^{n+1} appears in each equation. Unfortunately, when the von Neumann stability analysis is applied to these schemes, we find that they are unconditionally unstable. These simple schemes, therefore, prove to be worthless in solving the wave equation. Let us now proceed to look at methods that have more utility.

4.1.2 Upstream (First-Order Upwind or Windward) Differencing Method

The simple Euler method, Eq. (4.5), can be made stable by replacing the forward space difference by a backward space difference, provided that the wave speed c is positive. If the wave speed is negative, a forward difference must be used to assure stability. This point is discussed further at the end of the present section. For a positive wave speed, the following algorithm results:

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} + c\frac{u_j^n - u_{j-1}^n}{\Delta x} = 0 \qquad c > 0 \tag{4.7}$$

This is a first-order accurate method with T.E. of $O[\Delta t, \Delta x]$. The von Neumann stability analysis shows that this method is stable, provided that

$$0 \leqslant \nu \leqslant 1 \tag{4.8}$$

where $\nu = c\,\Delta t/\Delta x$.

Let us substitute Taylor-series expansions into Eq. (4.7) for u_j^{n+1} and u_{j-1}^n. The following equation results:

$$\frac{1}{\Delta t}\left\{\left[u_j^n + \Delta t\,u_t + \frac{(\Delta t)^2}{2}u_{tt} + \frac{(\Delta t)^3}{6}u_{ttt} + \cdots\right] - u_j^n\right\}$$
$$+\frac{c}{\Delta x}\left\{u_j^n - \left[u_j^n - \Delta x\,u_x + \frac{(\Delta x)^2}{2}u_{xx} - \frac{(\Delta x)^3}{6}u_{xxx} + \cdots\right]\right\} = 0 \tag{4.9}$$

Equation (4.9) simplifies to

$$u_t + cu_x = -\frac{\Delta t}{2}u_{tt} + \frac{c\,\Delta x}{2}u_{xx} - \frac{(\Delta t)^2}{6}u_{ttt} - c\frac{(\Delta x)^2}{6}u_{xxx} + \cdots \tag{4.10}$$

Note that the left-hand side of this equation corresponds to the wave equation and the right-hand side is the T.E., which is generally not zero. The significance of terms in the T.E. can be more easily interpreted if the time-derivative terms are replaced by spatial derivatives. In order to replace u_{tt} by a spatial-derivative term, we take the partial derivative of Eq. (4.10) with respect to time, to obtain

$$u_{tt} + cu_{xt} = -\frac{\Delta t}{2}u_{ttt} + \frac{c\,\Delta x}{2}u_{xxt} - \frac{(\Delta t)^2}{6}u_{tttt} - \frac{c(\Delta x)^2}{6}u_{xxxt} + \cdots \tag{4.11}$$

and take the partial derivative of Eq. (4.10) with respect to x and multiply by $-c$:

$$-cu_{tx} - c^2u_{xx} = \frac{c\,\Delta t}{2}u_{ttx} - \frac{c^2\,\Delta x}{2}u_{xxx} + \frac{c(\Delta t)^2}{6}u_{tttx} + \frac{c^2(\Delta x)^2}{6}u_{xxxx} + \cdots \tag{4.12}$$

Adding Eqs. (4.11) and (4.12) gives

$$u_{tt} = c^2u_{xx} + \Delta t\left(\frac{-u_{ttt}}{2} + \frac{c}{2}u_{ttx} + O[\Delta t]\right) + \Delta x\left(\frac{c}{2}u_{xxt} - \frac{c^2}{2}u_{xxx} + O[\Delta x]\right) \tag{4.13}$$

In a similar manner, we can obtain the following expressions for u_{ttt}, u_{ttx}, and u_{xxt}:

$$\begin{aligned} u_{ttt} &= -c^3u_{xxx} + O[\Delta t, \Delta x] \\ u_{ttx} &= c^2u_{xxx} + O[\Delta t, \Delta x] \\ u_{xxt} &= -cu_{xxx} + O[\Delta t, \Delta x] \end{aligned} \tag{4.14}$$

Combining Eqs. (4.10), (4.13), and (4.14) leaves

$$\begin{aligned} u_t + cu_x = {} & \frac{c\,\Delta x}{2}(1 - \nu)u_{xx} - \frac{c(\Delta x)^2}{6}(2\nu^2 - 3\nu + 1)u_{xxx} \\ & + O[(\Delta x)^3, (\Delta x)^2\,\Delta t, \Delta x(\Delta t)^2, (\Delta t)^3] \end{aligned} \tag{4.15}$$

An equation, such as Eq. (4.15), is called a *modified equation* (Warming and Hyett, 1974). It can be thought of as the PDE that is actually solved (if sufficient boundary conditions were available) when a finite-difference method is applied to a PDE. It is important to emphasize that the equation obtained after substitution of the Taylor-series expansions, i.e., Eq. (4.10), must be used to eliminate the higher-order time derivatives rather than the original PDE, Eq. (4.2). This is due to the fact that a solution of the original PDE does not in general satisfy the difference equation, and since the modified equation represents the difference equation, it is obvious that the original PDE should not be used to eliminate the time derivatives.

The process of eliminating time derivatives can be greatly simplified if a table is constructed (Table 4.1). The coefficients of each term in Eq. (4.10) are placed in the first row of the table. Note that all terms have been moved to the left-hand side of the equation. The u_{tt} term is then eliminated by multiplying Eq. (4.10) by the operator

$$-\frac{\Delta t}{2}\frac{\partial}{\partial t}$$

Table 4.1 Procedure for determining modified equation

	u_t	u_x	u_{tt}	u_{tx}	u_{xx}	u_{ttt}	u_{ttx}	u_{txx}	u_{xxx}	u_{tttt}	u_{tttx}	u_{ttxx}	u_{txxx}	u_{xxxx}
Coefficients of Eq. (4.10)	1	c	$\frac{\Delta t}{2}$	0	$-\frac{c\,\Delta x}{2}$	$\frac{\Delta t^2}{6}$	0	0	$\frac{c\,\Delta x^2}{6}$	$\frac{\Delta t^3}{24}$	0	0	0	$-\frac{c\,\Delta x^3}{24}$
$-\frac{\Delta t}{2}\frac{\partial}{\partial t}$ Eq. (4.10)			$-\frac{\Delta t}{2}$	$-\frac{c\,\Delta t}{2}$	0	$-\frac{\Delta t^2}{4}$	0	$\frac{c\,\Delta t\,\Delta x}{4}$	0	$-\frac{\Delta t^3}{12}$	0	0	$-\frac{c\,\Delta t\,\Delta x^2}{12}$	0
$\frac{c}{2}\Delta t\frac{\partial}{\partial x}$ Eq. (4.10)				$\frac{c\,\Delta t}{2}$	$\frac{c^2}{2}\Delta t$	0	$\frac{c\,\Delta t^2}{4}$	0	$-\frac{c^2\,\Delta t\,\Delta x}{4}$	0	$\frac{c\,\Delta t^3}{12}$	0	0	$\frac{c^2\,\Delta t\,\Delta x^2}{12}$
$\frac{1}{12}\Delta t^2\frac{\partial^2}{\partial t^2}$ Eq. (4.10)						$\frac{1}{12}\Delta t^2$	$\frac{c\,\Delta t^2}{12}$	0	0	$\frac{1}{24}\Delta t^3$	0	$-\frac{c}{24}\Delta x\,\Delta t^2$	0	0
$-\frac{1}{3}c\,\Delta t^2\frac{\partial^2}{\partial t\,\partial x}$ Eq. (4.10)							$-\frac{1}{3}c\,\Delta t^2$	$-\frac{1}{3}c^2\,\Delta t^2$	0	0	$-\frac{1}{6}c\,\Delta t^3$	0	$+\frac{c^2}{6}\Delta x\,\Delta t^2$	0
$\left(\frac{1}{3}c^2\,\Delta t^2-\frac{c\,\Delta t\,\Delta x}{4}\right)\frac{\partial^2}{\partial x^2}$ Eq. (4.10)								$\frac{1}{3}c^2\,\Delta t^2$ $-\frac{c\,\Delta t\,\Delta x}{4}$	$\frac{1}{3}c^3\,\Delta t^2$ $-\frac{c^2\,\Delta t\,\Delta x}{4}$	0	0	$\frac{1}{6}c^2\,\Delta t^3$ $-\frac{c\,\Delta t^2\,\Delta x}{8}$	0	$-\frac{1}{6}c^3\,\Delta t^2\,\Delta x$ $+\frac{c^2}{8}\Delta t\,\Delta x^2$
$\frac{1}{12}c\,\Delta t^3\frac{\partial^3}{\partial t^2\,\partial x}$ Eq. (4.10)											$\frac{1}{12}c\,\Delta t^3$	$\frac{c^2}{12}\Delta t^3$	0	0
$\left(\frac{1}{6}c\,\Delta x\,\Delta t^2\right.$												$\frac{1}{6}c\,\Delta x\,\Delta t^2$	$\frac{1}{6}c^2\Delta x\,\Delta t^2$	0
$\left.-\frac{1}{4}c^2\,\Delta t^3\right)\frac{\partial^3}{\partial t\,\partial x^2}$ Eq. (4.10)												$-\frac{1}{4}c^2\,\Delta t^3$	$-\frac{1}{4}c^3\,\Delta t^3$	
$\left(\frac{c}{12}\Delta t\,\Delta x^2-\frac{1}{3}c^2\,\Delta x\,\Delta t^2\right.$ $\left.+\frac{1}{4}c^3\,\Delta t^3\right)\frac{\partial^3}{\partial x^3}$ Eq. (4.10)													$\frac{c}{12}\Delta t\,\Delta x^2$ $-\frac{1}{3}c^2\,\Delta x\,\Delta t^2$ $+\frac{1}{4}c^3\,\Delta t^3$	$\frac{c^2}{12}\Delta t\,\Delta x^2$ $-\frac{1}{3}c^3\,\Delta x\,\Delta t^2$ $+\frac{1}{4}c^4\,\Delta t^3$
…														
Sum of coefficients	1	c	0	0	$\frac{c\,\Delta x}{2}(\nu-1)$	0	0	0	$\frac{c\,\Delta x^2}{6}(2\nu^2-3\nu+1)$	0	0	0	0	$\frac{c\,\Delta x^3}{24}(6\nu^3-12\nu^2+7\nu-1)$

and adding the result to the first row, i.e., Eq. (4.10). This introduces the term $-(c\,\Delta t/2)u_{tx}$, which is eliminated by multiplying Eq. (4.10) by the operator

$$\frac{c\,\Delta t}{2}\frac{\partial}{\partial x}$$

and adding the result to the first two rows of the table. This procedure is continued until the desired time derivatives are eliminated. Each coefficient in the modified equation is then obtained by simply adding the coefficients in the corresponding column of the table. The algebra required to derive the modified equation can be programmed on a digital computer using an algebraic manipulation code.

The right-hand side of the modified equation [Eq. (4.15)] is the T.E., since it represents the difference between the original PDE and the finite-difference approximation to it. Consequently, the lowest order term on the right-hand side of the modified equation gives the order of the method. In the present case, the method is first-order accurate, since the lowest order term is $O[\Delta t, \Delta x]$. If $\nu = 1$, the right-hand side of the modified equation becomes zero, and the wave equation is solved exactly. In this case, the upstream differencing scheme reduces to

$$u_j^{n+1} = u_{j-1}^n$$

which is equivalent to solving the wave equation exactly using the method of characteristics. Finite-difference algorithms that exhibit this behavior are said to satisfy the *shift condition* (Kutler and Lomax, 1971).

The lowest order term of the T.E. in the present case contains the partial derivative u_{xx}, which makes this term similar to the viscous term in 1-D fluid flow equations. For example, the viscous term in the 1-D Navier-Stokes equation (see Chapter 5) may be written as

$$\frac{\partial}{\partial x}(\tau_{xx}) = \frac{4}{3}\mu u_{xx} \tag{4.16}$$

if a constant coefficient of viscosity is assumed. Thus, when $\nu \neq 1$, the upstream differencing scheme introduces an *artificial viscosity* into the solution. This is often called implicit artificial viscosity, as opposed to explicit artificial viscosity, which is purposely added to a difference scheme. Artificial viscosity tends to reduce all gradients in the solution whether physically correct or numerically induced. This effect, which is the direct result of even derivative terms in the T.E., is called *dissipation*.

Another quasi-physical effect of numerical schemes is called *dispersion*. This is the direct result of the odd derivative terms that appear in the T.E. As a result of dispersion, phase relations between various waves are distorted. The combined effect of dissipation and dispersion is sometimes referred to as *diffusion*. Diffusion tends to spread out sharp dividing lines that may appear in the computational region. Figure 4.1 illustrates the effects of dissipation and dispersion on the computation of a discontinuity. In general, if the lowest order

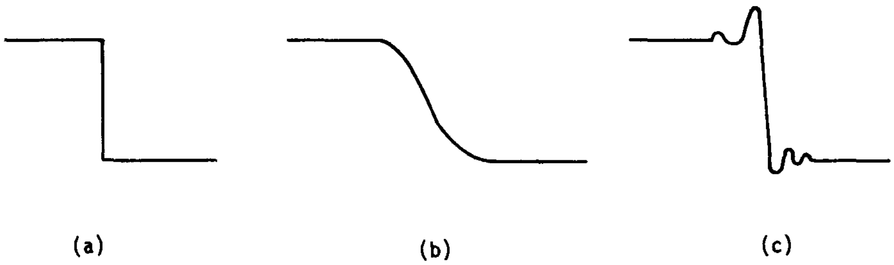

Figure 4.1 Effects of dissipation and dispersion. (a) Exact solution. (b) Numerical solution distorted primarily by dissipation errors (typical of first-order methods). (c) Numerical solution distorted primarily by dispersion errors (typical of second-order methods).

term in the T.E. contains an even derivative, the resulting solution will predominantly exhibit dissipative errors. On the other hand, if the leading term is an odd derivative, the resulting solution will predominantly exhibit dispersive errors.

In Chapter 3 we discussed a technique for finding the relative errors in both amplitude (dissipation) and phase (dispersion) from the amplification factor. At this point it seems natural to ask if the amplification factor is related to the modified equation. The answer is definitely yes! Warming and Hyett (1974) have developed a "heuristic" stability theory based on the even derivative terms in the modified equation and have determined the phase shift error by examining the odd derivative terms. However, the analysis of Warming and Hyett has been shown by Chang (1987) to be restricted to schemes involving only two time levels $(n, n+1)$. Before showing the correspondence between the modified equation and the amplification factor, let us first examine the amplification factor of the present upstream differencing scheme:

$$G = (1 - \nu + \nu\cos\beta) - i(\nu\sin\beta) \tag{4.17}$$

The modulus of this amplification factor,

$$|G| = \left[(1 - \nu + \nu\cos\beta)^2 + (-\nu\sin\beta)^2\right]^{1/2}$$

is plotted in Fig. 4.2 for several values of ν. It is clear from this plot that ν must be less than or equal to 1 if the von Neumann stability condition $|G| \leqslant 1$ is to be met.

The amplification factor, Eq. (4.17), can also be expressed in the exponential form for a complex number:

$$G = |G|e^{i\phi}$$

where ϕ is the phase angle given by

$$\phi = \tan^{-1}\left[\frac{\operatorname{Im}(G)}{\operatorname{Re}(G)}\right] = \tan^{-1}\left(\frac{-\nu\sin\beta}{1 - \nu + \nu\cos\beta}\right)$$

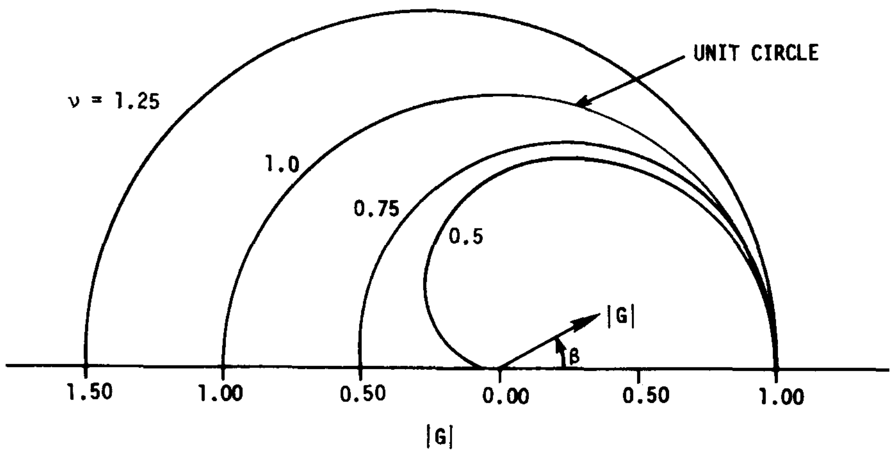

Figure 4.2 Amplification factor modulus for upstream differencing scheme.

The phase angle for the exact solution of the wave equation (ϕ_e) is determined in a similar manner once the amplification factor of the exact solution is known. In order to find the exact amplification factor we substitute the elemental solution

$$u = e^{\alpha t} e^{ik_m x}$$

into the wave equation and find that $\alpha = -ik_m c$, which gives

$$u = e^{ik_m(x-ct)}$$

The exact amplification factor is then

$$G_e = \frac{u(t+\Delta t)}{u(t)} = \frac{e^{ik_m[x-c(t+\Delta t)]}}{e^{ik_m(x-ct)}}$$

which reduces to

$$G_e = e^{-ik_m c\,\Delta t} = e^{i\phi_e}$$

where

$$\phi_e = -k_m c\,\Delta t = -\beta\nu$$

and

$$|G_e| = 1$$

Thus the total dissipation (amplitude) error that accrues from applying the upstream differencing method to the wave equation for N steps is given by

$$(1 - |G|^N)A_0$$

where A_0 is the initial amplitude of the wave. Likewise, the total dispersion (phase) error can be expressed as $N(\phi_e - \phi)$. The relative phase shift error after one time step is given by

$$\frac{\phi}{\phi_e} = \frac{\tan^{-1}[(-\nu\sin\beta)/(1-\nu+\nu\cos\beta)]}{-\beta\nu} \tag{4.18}$$

and is plotted in Fig. 4.3 for several values of ν. For small wave numbers (i.e., small β) the relative phase error reduces to

$$\frac{\phi}{\phi_e} \cong 1 - \frac{1}{6}(2\nu^2 - 3\nu + 1)\beta^2 \tag{4.19}$$

If the relative phase error exceeds 1 for a given value of β, the corresponding Fourier component of the numerical solution has a wave speed greater than the exact solution, and this is a *leading phase error*. If the relative phase error is less than 1, the wave speed of the numerical solution is less than the exact wave speed, and this is a *lagging phase error*. The upstream differencing scheme has a leading phase error for $0.5 < \nu < 1$ and a lagging phase error for $\nu < 0.5$.

Example 4.1 Suppose the upstream differencing scheme is used to solve the wave equation ($c = 0.75$) with the initial condition

$$u(x,0) = \sin(6\pi x) \qquad 0 \leqslant x \leqslant 1$$

and periodic boundary conditions. Determine the amplitude and phase errors after 10 steps if $\Delta t = 0.02$ and $\Delta x = 0.02$.

Solution In this problem a unique value of β can be determined because the exact solution of the wave equation (for the present initial condition) is

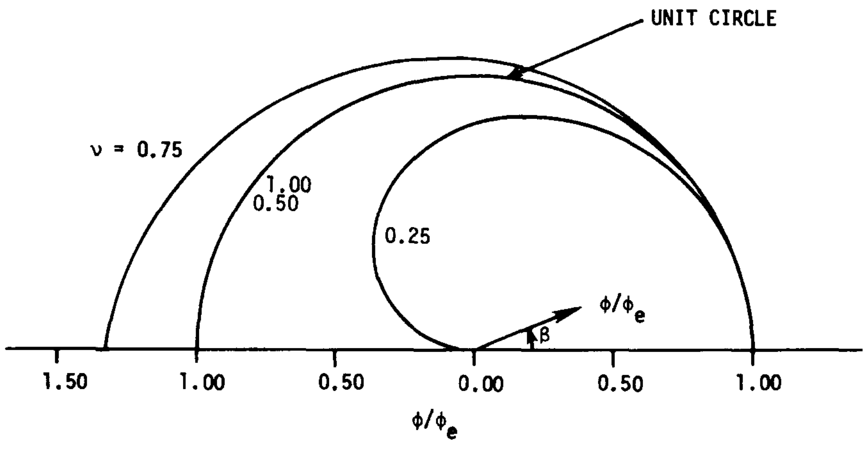

Figure 4.3 Relative phase error of upstream differencing scheme.

represented by a single term of a Fourier series. Since the amplification factor is also determined using a single term of a Fourier series that satisfies the wave equation, the frequency of the exact solution is identical to the frequency associated with the amplification factor, i.e., $f_m = k_m/2\pi$. Thus the wave number for the present problem is given by

$$k_m = \frac{2m\pi}{2L} = \frac{6\pi}{1} = 6\pi$$

and β can be calculated as

$$\beta = k_m\,\Delta x = (6\pi)(0.02) = 0.12\pi$$

Using the Courant number,

$$\nu = \frac{c\,\Delta t}{\Delta x} = \frac{(0.75)(0.02)}{(0.02)} = 0.75$$

the modulus of the amplification factor becomes

$$|G| = \left[(1 - \nu + \nu\cos\beta)^2 + (-\nu\sin\beta)^2\right]^{1/2} = 0.986745$$

and the resulting amplitude error after 10 steps is

$$(1 - |G|^N)A_0 = (1 - |G|^{10})(1) = 1 - 0.8751 = 0.1249$$

The phase angle (ϕ) after one step,

$$\phi = \tan^{-1}\left[\frac{-\nu\sin\beta}{1 - \nu + \nu\cos\beta}\right] = -0.28359$$

can be compared with the exact phase angle (ϕ_e) after one step,

$$\phi_e = -\beta\nu = -0.28274$$

to give the phase error after 10 steps:

$$10(\phi_e - \phi) = 0.0084465$$

Let us now compare the exact and numerical solutions after 10 steps where the time is

$$t = 10\,\Delta t = 0.2$$

The exact solution is given by

$$u(x, 0.2) = \sin[6\pi(x - 0.15)]$$

and the numerical solution that results after applying the upstream differencing scheme for 10 steps is

$$u(x, 0.2) = (0.8751)\sin[6\pi(x - 0.15) - 0.0084465]$$

In order to show the correspondence between the amplification factor and the modified equation, we write the modified equation [Eq. (4.15)] in the following form:

$$u_t + cu_x = \sum_{n=1}^{\infty} \left(C_{2n} \frac{\partial^{2n} u}{\partial x^{2n}} + C_{2n+1} \frac{\partial^{2n+1} u}{\partial x^{2n+1}} \right) \tag{4.20}$$

where C_{2n} and C_{2n+1} represent the coefficients of the even and odd spatial-derivative terms, respectively. Warming and Hyett (1974) have shown that a necessary condition for stability is

$$(-1)^{l-1} C_{2l} > 0 \tag{4.21}$$

where C_{2l} represents the coefficient of the lowest order even derivative term. This is analogous to the requirement that the coefficient of viscosity in viscous flow equations be greater than zero. In Eq. (4.15) the coefficient of the lowest order even derivative term is

$$C_2 = \frac{c\,\Delta x}{2}(1 - \nu) \tag{4.22}$$

and therefore the stability condition becomes

$$\frac{c\,\Delta x}{2}(1 - \nu) > 0 \tag{4.23}$$

or $\nu < 1$, which was obtained earlier from the amplification factor. It should be remembered that the "heuristic" stability analysis, i.e., Eq. (4.21), can only provide a necessary condition for stability. Thus, for some finite-difference algorithms, only partial information about the complete stability bound is obtained, and for others (such as algorithms for the heat equation) a more complete theory must be employed.

Warming and Hyett have also shown that the relative phase error for difference schemes applied to the wave equation is given by

$$\frac{\phi}{\phi_e} = 1 - \frac{1}{c} \sum_{n=1}^{\infty} (-1)^n (k_m)^{2n} C_{2n+1} \tag{4.24}$$

where $k_m = \beta/\Delta x$ is the wave number. For small wave numbers, we need only retain the lowest order term. For the upstream differencing scheme, we find that

$$\frac{\phi}{\phi_e} \cong 1 - \frac{1}{c}(-1)\left(\frac{\beta}{\Delta x}\right)^2 C_3 = 1 - \frac{1}{6}(2\nu^2 - 3\nu + 1)\beta^2 \tag{4.25a}$$

which is identical to Eq. (4.19). Thus we have demonstrated that the amplification factor and the modified equation are directly related.

The upstream method given by Eq. (4.7) may be written in a more general form to account for either positive or negative wave speeds. The method is normally written separately for these two cases as

$$u_j^{n+1} = u_j^n - c\frac{\Delta t}{\Delta x}(u_j^n - u_{j-1}^n) \qquad c > 0$$

$$u_j^{n+1} = u_j^n - c\frac{\Delta t}{\Delta x}(u_{j+1}^n - u_j^n) \qquad c < 0$$

However, if we make use of the following definitions,

$$c^+ = \tfrac{1}{2}(c + |c|)$$

$$c^- = \tfrac{1}{2}(c - |c|)$$

the upstream scheme may be written as the single expression

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

Upon substituting for the values of c^+ and c^-, the final form becomes

$$u_j^{n+1} = u_j^n - c\frac{\Delta t}{2\,\Delta x}(u_{j+1}^n - u_{j-1}^n) + \frac{|c|\,\Delta t}{2\,\Delta x}(u_{j+1}^n - 2u_j^n + u_{j-1}^n) \qquad (4.25b)$$

It is interesting to note that this form of the upstream scheme gives the impression that it is a centered method. We recognize the first difference term as a central-difference approximation and interpret the last term as an artificial viscosity term. The function of this last term is to add the appropriate dissipation to produce the upstream scheme when c is either positive or negative.

4.1.3 Lax Method

The Euler method, Eq. (4.6), can be made stable by replacing u_j^n with the averaged term $(u_{j+1}^n + u_{j-1}^n)/2$. The resulting algorithm is the well-known Lax method (Lax, 1954), which was presented earlier:

$$\frac{u_j^{n+1} - (u_{j+1}^n + u_{j-1}^n)/2}{\Delta t} + c\frac{u_{j+1}^n - u_{j-1}^n}{2\,\Delta x} = 0 \qquad (4.26)$$

This explicit one-step scheme is first-order accurate with T.E. of $O[\Delta t, (\Delta x)^2/\Delta t]$ and is stable if $|\nu| \leqslant 1$. The modified equation is given by

$$u_t + cu_x = \frac{c\,\Delta x}{2}\left(\frac{1}{\nu} - \nu\right)u_{xx} + \frac{c(\Delta x)^2}{3}(1 - \nu^2)u_{xxx} + \cdots \qquad (4.27)$$

Note that this method is not uniformly consistent, since $(\Delta x)^2/\Delta t$ may not approach zero in the limit as Δt and Δx go to zero. However, if ν is held constant as Δt and Δx approach zero, the method is consistent. The Lax method is known for its large dissipation error when $\nu \neq 1$. This large dissipation is readily apparent when we compare the coefficient of the u_{xx} term in Eq. (4.27) with the same coefficient in the modified equation of the upstream differencing scheme for various values of ν. The large dissipation can also be observed in the amplification factor

$$G = \cos\beta - i\nu\sin\beta \tag{4.28}$$

which is described in Section 3.6.1. The modulus of the amplification factor is plotted in Fig. 4.4(a). The relative phase error is given by

$$\frac{\phi}{\phi_e} = \frac{\tan^{-1}(-\nu\tan\beta)}{-\beta\nu}$$

which produces a leading phase error, as seen in Fig. 4.4(b).

4.1.4 Euler Implicit Method

The algorithms discussed previously for the wave equation have all been explicit. The following implicit scheme,

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} + \frac{c}{2\,\Delta x}\left(u_{j+1}^{n+1} - u_{j-1}^{n+1}\right) = 0 \tag{4.29}$$

is first-order accurate with T.E. of $O[\Delta t, (\Delta x)^2]$ and, according to a Fourier stability analysis, is unconditionally stable for all time steps. However, a system of algebraic equations must be solved at each new time level. To illustrate this, let us rewrite Eq. (4.29) so that the unknowns at time level $(n + 1)$ appear on the left-hand side of the equation and the known quantity u_j^n appears on the right-hand side. This gives

$$\frac{\nu}{2}u_{j+1}^{n+1} + (1)u_j^{n+1} - \frac{\nu}{2}u_{j-1}^{n+1} = u_j^n \tag{4.30}$$

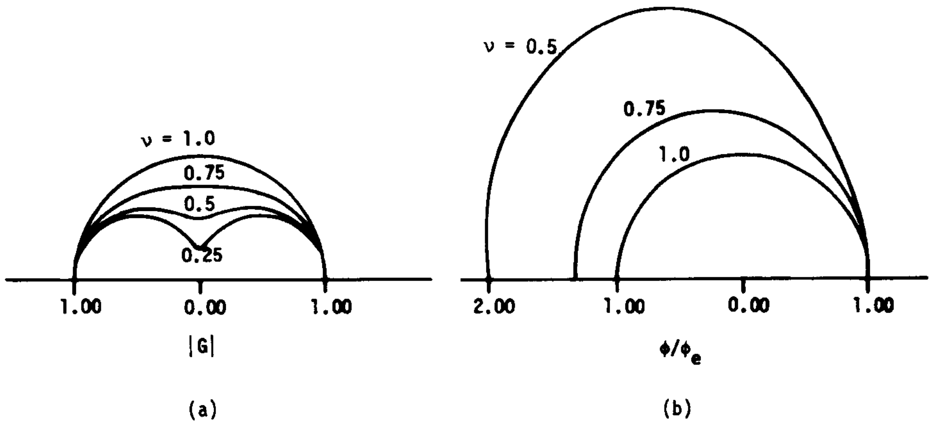

Figure 4.4 Lax method. (a) Amplification factor modulus. (b) Relative phase error.

or

$$a_j u_{j+1}^{n+1} + d_j u_j^{n+1} + b_j u_{j-1}^{n+1} = C_j \tag{4.31}$$

where $a_j = \nu/2$, $d_j = 1$, $b_j = -\nu/2$, and $C_j = u_j^n$. Consider the computational mesh shown in Fig. 4.5, which contains $M + 2$ grid points in the x direction and known initial conditions at $n = 0$. Along the left boundary, u_0^{n+1} has a fixed value of u_0. Along the right boundary, u_{M+1}^{n+1} can be computed as part of the solution using characteristic theory. For example, if $\nu = 1$, then $u_{M+1}^{n+1} = u_M^n$. Applying Eq. (4.31) to the grid shown in Fig. 4.5, we find that the following system of M linear algebraic equations must be solved at each $(n + 1)$ time level:

$$\begin{array}{ccccc} [A] & & [u] & & [C] \\ \begin{bmatrix} d_1 & a_1 & 0 & \cdot & \cdot & \cdot & \cdot & 0 \\ b_2 & d_2 & a_2 & & & & & \cdot \\ 0 & b_3 & d_3 & a_3 & & & & \cdot \\ \cdot & & \ddots & \ddots & \ddots & & & \cdot \\ \cdot & & & \ddots & \ddots & \ddots & & \cdot \\ \cdot & & & & \ddots & \ddots & \ddots & 0 \\ \cdot & & & & & b_{M-1} & d_{M-1} & a_{M-1} \\ 0 & \cdot & \cdot & \cdot & \cdot & 0 & b_M & d_M \end{bmatrix} & & \begin{bmatrix} u_1^{n+1} \\ u_2^{n+1} \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ u_{M-1}^{n+1} \\ u_M^{n+1} \end{bmatrix} & = & \begin{bmatrix} C_1 \\ C_2 \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ C_{M-1} \\ C_M \end{bmatrix} \end{array} \tag{4.32}$$

In Eq. (4.32), C_1 and C_M are given by

$$\begin{aligned} C_1 &= u_1^n - b u_0^{n+1} \\ C_M &= u_M^n - a u_{M+1}^{n+1} \end{aligned} \tag{4.33}$$

where u_0^{n+1} and u_{M+1}^{n+1} are the known boundary conditions.

Matrix $[A]$ in Eq. (4.32) is a tridiagonal matrix. A technique for rapidly solving a tridiagonal system of linear algebraic equations is due to Thomas (1949) and is called the Thomas algorithm. In this algorithm, the system of equations is first put into upper triangular form by replacing the diagonal

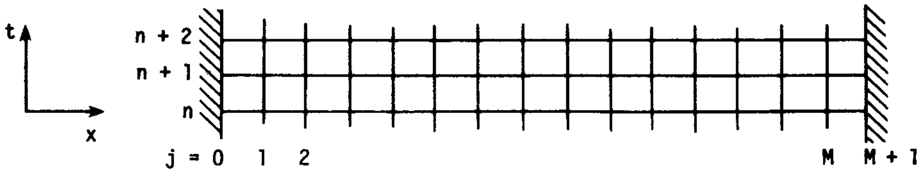

Figure 4.5 Computational mesh.

elements d_i with

$$d_i - \frac{b_i}{d_{i-1}} a_{i-1} \qquad i = 2,3,\ldots,M$$

and the C_i with

$$C_i - \frac{b_i}{d_{i-1}} C_{i-1} \qquad i = 2,3,\ldots,M$$

The unknowns are then computed using back substitution starting with

$$u_M^{n+1} = \frac{C_M}{d_M}$$

and continuing with

$$u_j^{n+1} = \frac{C_j - a_j u_{j+1}^{n+1}}{d_j} \qquad j = M-1, M-2, \ldots, 1$$

Further details of the Thomas algorithm are given in Section 4.3.3.

In general, implicit schemes require more computation time per time step but, of course, permit a larger time step, since they are usually unconditionally stable. However, the solution may become meaningless if too large a time step is taken. This is due to the fact that a large time step produces large T.E.s. The modified equation for the Euler implicit scheme is

$$u_t + cu_x = \left(\tfrac{1}{2}c^2 \Delta t\right)u_{xx} - \left[\tfrac{1}{6}c(\Delta x)^2 + \tfrac{1}{3}c^3(\Delta t)^2\right]u_{xxx} + \cdots \tag{4.34}$$

which does not satisfy the shift condition. The amplification factor

$$G = \frac{1 - i\nu \sin\beta}{1 + \nu^2 \sin^2\beta} \tag{4.35}$$

and the relative phase error

$$\frac{\phi}{\phi_e} = \frac{\tan^{-1}(-\nu\sin\beta)}{-\beta\nu} \tag{4.36}$$

are plotted in Fig. 4.6. The Euler implicit scheme is very dissipative for intermediate wave numbers and has a large lagging phase error for high wave numbers.

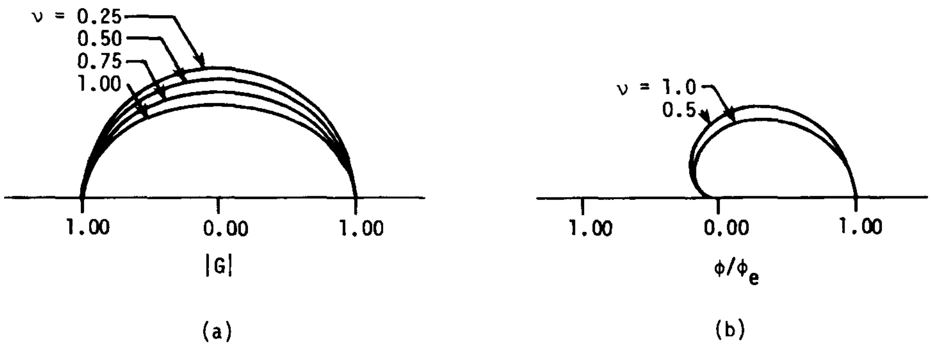

Figure 4.6 Euler implicit method. (a) Amplification factor modulus. (b) Relative phase error.

4.1.5 Leap Frog Method

The numerical schemes presented so far in this chapter for solving the linear wave equation are all first-order accurate. In most cases, first-order schemes are not used to solve PDEs because of their inherent inaccuracy. The leap frog method is the simplest second-order accurate method. When applied to the first-order wave equation, this explicit one-step three-time-level scheme becomes

$$\frac{u_j^{n+1} - u_j^{n-1}}{2\,\Delta t} + c\frac{u_{j+1}^n - u_{j-1}^n}{2\,\Delta x} = 0 \tag{4.37}$$

The leap frog method is referred to as a three-time-level scheme, since u must be known at time levels n and $n - 1$ in order to find u at time level $n + 1$. This method has a T.E. of $O[(\Delta t)^2, (\Delta x)^2]$ and is stable whenever $|\nu| \leqslant 1$. The modified equation is given by

$$u_t + cu_x = \frac{c(\Delta x)^2}{6}(\nu^2 - 1)u_{xxx} - \frac{c(\Delta x)^4}{120}(9\nu^4 - 10\nu^2 + 1)u_{xxxxx} + \cdots \tag{4.38}$$

The leading term in the T.E. contains the odd derivative u_{xxx}, and hence the solution will predominantly exhibit dispersive errors. This is typical of second-order accurate methods. In this case, however, there are no even derivative terms in the modified equation, so that the solution will not contain any dissipation error. As a consequence, the leap frog algorithm is neutrally stable, and errors caused by improper boundary conditions or computer round-off will not be damped (assuming periodic boundary conditions and $|\nu| \leqslant 1$). The amplification factor

$$G = \pm(1 - \nu^2 \sin^2 \beta)^{1/2} - i\nu \sin \beta \tag{4.39}$$

and the relative phase error

$$\frac{\phi}{\phi_e} = \frac{\tan^{-1}\left[-\nu \sin \beta / \pm(1 - \nu^2 \sin^2 \beta)^{1/2}\right]}{-\beta\nu} \tag{4.40}$$

are plotted in Fig. 4.7.

The leap frog method, while being second-order accurate with no dissipation error, does have its disadvantages. First, initial conditions must be specified at two-time levels. This difficulty can be circumvented by using a two-time-level scheme for the first time step. A second disadvantage is due to the "leap frog" nature of the differencing (i.e., u_j^{n+1} does not depend on u_j^n), so that two independent solutions develop as the calculation proceeds. And finally, the leap frog method may require additional computer storage because it is a three-time-level scheme. The required computer storage is reduced considerably if a simple overwriting procedure is employed, whereby u_j^{n-1} is overwritten by u_j^{n+1}.

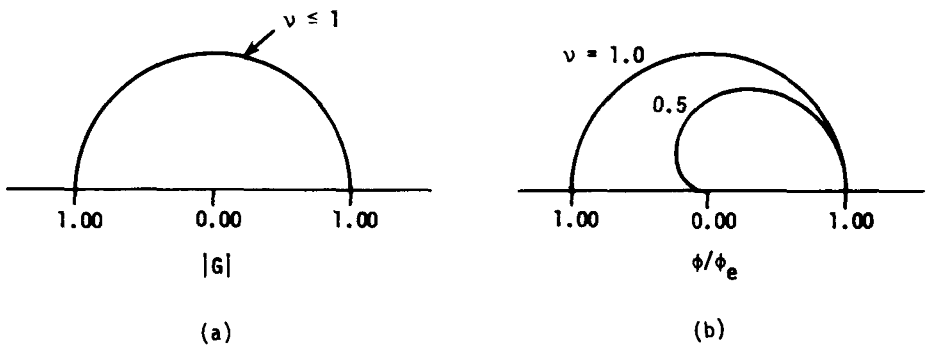

Figure 4.7 Leap frog method. (a) Amplification factor modulus. (b) Relative phase error.

4.1.6 Lax-Wendroff Method

The Lax-Wendroff finite-difference scheme (Lax and Wendroff, 1960) can be derived from a Taylor-series expansion in the following manner:

$$u_j^{n+1} = u_j^n + \Delta t u_t + \tfrac{1}{2}(\Delta t)^2 u_{tt} + O[(\Delta t)^3] \tag{4.41}$$

Using the wave equations

$$\begin{aligned} u_t &= -cu_x \\ u_{tt} &= c^2 u_{xx} \end{aligned} \tag{4.42}$$

Equation (4.41) may be written as

$$u_j^{n+1} = u_j^n - c\,\Delta t u_x + \tfrac{1}{2}c^2(\Delta t)^2 u_{xx} + O[(\Delta t)^3] \tag{4.43}$$

And finally, if u_x and u_{xx} are replaced by second-order accurate central-difference expressions, the well-known Lax-Wendroff scheme is obtained:

$$u_j^{n+1} = u_j^n - \frac{c\,\Delta t}{2\,\Delta x}(u_{j+1}^n - u_{j-1}^n) + \frac{c^2(\Delta t)^2}{2(\Delta x)^2}(u_{j+1}^n - 2u_j^n + u_{j-1}^n) \tag{4.44}$$

This explicit one-step scheme is second-order accurate with a T.E. of $O[(\Delta x)^2, (\Delta t)^2]$ and is stable whenever $|\nu| \leqslant 1$. The modified equation for this method is

$$u_t + cu_x = -c\frac{(\Delta x)^2}{6}(1 - \nu^2)u_{xxx} - \frac{c(\Delta x)^3}{8}\nu(1 - \nu^2)u_{xxxx} + \cdots \tag{4.45}$$

The amplification factor

$$G = 1 - \nu^2(1 - \cos\beta) - i\nu\sin\beta \tag{4.46}$$

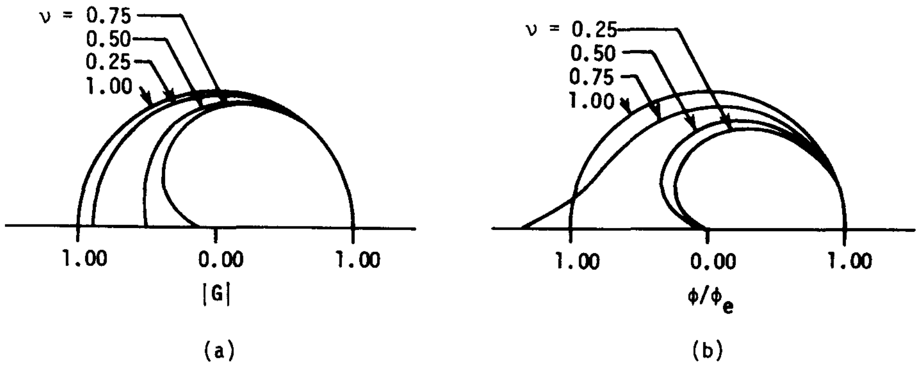

Figure 4.8 Lax-Wendroff method. (a) Amplification factor modulus. (b) Relative phase error.

and the relative phase error

$$\frac{\phi}{\phi_e} = \frac{\tan^{-1}\{-\nu\sin\beta/[1-\nu^2(1-\cos\beta)]\}}{-\beta\nu} \tag{4.47}$$

are plotted in Fig. 4.8. The Lax-Wendroff scheme has a predominantly lagging phase error except for large wave numbers with $\sqrt{0.5} < \nu < 1$.

4.1.7 Two-Step Lax-Wendroff Method

For nonlinear equations such as the inviscid flow equations, a two-step variation of the original Lax-Wendroff method can be used. When applied to the wave equation, this explicit two-step three-time-level method becomes

$$\text{Step 1:} \qquad \frac{u_{j+1/2}^{n+1/2} - (u_{j+1}^n + u_j^n)/2}{\Delta t/2} + c\frac{u_{j+1}^n - u_j^n}{\Delta x} = 0 \tag{4.48}$$

$$\text{Step 2:} \qquad \frac{u_j^{n+1} - u_j^n}{\Delta t} + c\frac{u_{j+1/2}^{n+1/2} - u_{j-1/2}^{n+1/2}}{\Delta x} = 0 \tag{4.49}$$

This scheme is second-order accurate with a T.E. of $O[(\Delta x)^2, (\Delta t)^2]$ and is stable whenever $|\nu| \leqslant 1$. Step 1 is the Lax method applied at the midpoint $j + \frac{1}{2}$ for a half time step, and step 2 is the leap frog method for the remaining half time step. When applied to the linear wave equation, the two-step Lax-Wendroff scheme is equivalent to the original Lax-Wendroff scheme. This can be readily shown by substituting Eq. (4.48) into Eq. (4.49). Since the two schemes are equivalent, it follows that the modified equation and the amplification factor are the same for the two methods.

4.1.8 MacCormack Method

The MacCormack method (MacCormack, 1969) is a widely used scheme for solving fluid flow equations. It is a variation of the two-step Lax-Wendroff scheme that removes the necessity of computing unknowns at the grid points $j + \frac{1}{2}$ and $j - \frac{1}{2}$. Because of this feature, the MacCormack method is particularly useful when solving nonlinear PDEs, as is shown in Section 4.4.3. When applied to the linear wave equation, this explicit, predictor-corrector method becomes

$$\text{Predictor:} \qquad \overline{u_j^{n+1}} = u_j^n - c\frac{\Delta t}{\Delta x}(u_{j+1}^n - u_j^n) \tag{4.50}$$

$$\text{Corrector:} \qquad u_j^{n+1} = \frac{1}{2}\left[u_j^n + \overline{u_j^{n+1}} - c\frac{\Delta t}{\Delta x}\left(\overline{u_j^{n+1}} - \overline{u_{j-1}^{n+1}}\right)\right] \tag{4.51}$$

The term $\overline{u_j^{n+1}}$ is a temporary "predicted" value of u at the time level $n + 1$. The corrector equation provides the final value of u at the time level $n + 1$. Note that in the predictor equation a forward difference is used for $\partial u/\partial x$, while in the corrector equation a backward difference is used. This differencing can be reversed, and in some problems it is advantageous to do so. This is particularly true for problems involving moving discontinuities. For the present linear wave equation, the MacCormack scheme is equivalent to the original Lax-Wendroff scheme. Hence the truncation error, stability limit, modified equation, and amplification factor are identical with those of the Lax-Wendroff scheme.

4.1.9 Second-Order Upwind Method

The second-order upwind method (Warming and Beam, 1975) is a variation of the MacCormack method, which uses backward (upwind) differences in both the predictor and corrector steps for $c > 0$:

$$\text{Predictor:} \qquad \overline{u_j^{n+1}} = u_j^n - \frac{c\,\Delta t}{\Delta x}(u_j^n - u_{j-1}^n) \tag{4.52}$$

Corrector:

$$u_j^{n+1} = \frac{1}{2}\left[u_j^n + \overline{u_j^{n+1}} - \frac{c\,\Delta t}{\Delta x}\left(\overline{u_j^{n+1}} - \overline{u_{j-1}^{n+1}}\right) - \frac{c\,\Delta t}{\Delta x}(u_j^n - 2u_{j-1}^n + u_{j-2}^n)\right] \tag{4.53}$$

The addition of the second backward difference in Eq. (4.53) makes this scheme second-order accurate with T.E. of $O[(\Delta t)^2, (\Delta t)(\Delta x), (\Delta x)^2]$. If Eq. (4.52) is substituted into Eq. (4.53), the following one-step algorithm is obtained:

$$u_j^{n+1} = u_j^n - \nu(u_j^n - u_{j-1}^n) + \tfrac{1}{2}\nu(\nu - 1)(u_j^n - 2u_{j-1}^n + u_{j-2}^n) \tag{4.54}$$

The modified equation for this scheme is

$$u_t + cu_x = \frac{c(\Delta x)^2}{6}(1-\nu)(2-\nu)u_{xxx} - \frac{(\Delta x)^4}{8\,\Delta t}\nu(1-\nu)^2(2-\nu)u_{xxxx} + \cdots \tag{4.55}$$

The second-order upwind method satisfies the shift condition for both $\nu = 1$ and $\nu = 2$. The amplification factor is

$$G = 1 - 2\nu\left(\nu + 2(1-\nu)\sin^2\frac{\beta}{2}\right)\sin^2\frac{\beta}{2} - i\nu\sin\beta\left(1 + 2(1-\nu)\sin^2\frac{\beta}{2}\right) \tag{4.56}$$

and the resulting stability condition becomes $0 \leqslant \nu \leqslant 2$. The modulus of the amplification factor and the relative phase error are plotted in Fig. 4.9. The second-order upwind method has a predominantly leading phase error for $0 < \nu < 1$ and a predominantly lagging phase error for $1 < \nu < 2$. We observe that the second-order upwind method and the Lax-Wendroff method have opposite phase errors for $0 < \nu < 1$. This suggests that a considerable reduction in dispersive error would occur if a linear combination of the two methods were used. Fromm's method of zero-average phase error (Fromm, 1968) is based on this observation.

4.1.10 Time-Centered Implicit Method (Trapezoidal Differencing Method)

A second-order accurate implicit scheme can be obtained if the two Taylor-series expansions

$$\begin{aligned} u_j^{n+1} &= u_j^n + \Delta t(u_t)_j^n + \frac{(\Delta t)^2}{2}(u_{tt})_j^n + \frac{(\Delta t)^3}{6}(u_{ttt})_j^n + \cdots \\ u_j^n &= u_j^{n+1} - \Delta t(u_t)_j^{n+1} + \frac{(\Delta t)^2}{2}(u_{tt})_j^{n+1} - \frac{(\Delta t)^3}{6}(u_{ttt})_j^{n+1} + \cdots \end{aligned} \tag{4.57}$$

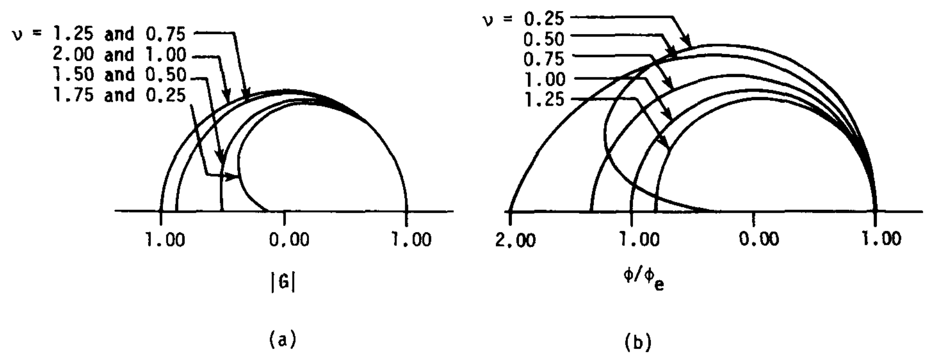

Figure 4.9 Second-order upwind method. (a) Amplification factor modulus. (b) Relative phase error.

are subtracted and $(u_{tt})_j^{n+1}$ is replaced with

$$(u_{tt})_j^{n+1} = (u_{tt})_j^n + \Delta t (u_{ttt})_j^n + \cdots$$

The resulting expression becomes

$$u_j^{n+1} = u_j^n + \frac{\Delta t}{2}\left[(u_t)^n + (u_t)^{n+1}\right]_j + O[(\Delta t)^3] \tag{4.58}$$

The time differencing in this equation is known as trapezoidal differencing or Crank-Nicolson differencing. Upon substituting the linear wave equation $u_t = -cu_x$, we obtain

$$u_j^{n+1} = u_j^n - \frac{c\,\Delta t}{2}\left[(u_x)^n + (u_x)^{n+1}\right]_j + O[(\Delta t)^3] \tag{4.59}$$

And finally, if the u_x terms are replaced by second-order central differences, the time-centered implicit method results:

$$u_j^{n+1} = u_j^n - \frac{\nu}{4}\left(u_{j+1}^{n+1} + u_{j+1}^n - u_{j-1}^{n+1} - u_{j-1}^n\right) \tag{4.60}$$

This method has second-order accuracy with T.E. of $O[(\Delta x)^2, (\Delta t)^2]$ and is unconditionally stable for all time steps. However, a tridiagonal matrix must be solved at each new time level. The modified equation for this scheme is

$$\begin{aligned} u_t + cu_x = &-\left[\frac{c^3(\Delta t)^2}{12} + \frac{c(\Delta x)^2}{6}\right]u_{xxx} \\ &-\left[\frac{c(\Delta x)^4}{120} + \frac{c^3(\Delta t)^2(\Delta x)^2}{24} + \frac{c^4(\Delta t)^4}{80}\right]u_{xxxxx} + \cdots \end{aligned} \tag{4.61}$$

Note that the modified equation contains no even derivative terms, so that the scheme has no implicit artificial viscosity. When this scheme is applied to the nonlinear fluid dynamic equations, it often becomes necessary to add some explicit artificial viscosity to prevent the solution from "blowing up." The addition of explicit artificial viscosity (i.e., "smoothing" term) to this scheme will be discussed in Section 4.4.7. The modulus of the amplification factor,

$$G = \frac{1 - (i\nu/2)\sin\beta}{1 + (i\nu/2)\sin\beta} \tag{4.62}$$

and the relative phase error are plotted in Fig. 4.10.

The time-centered implicit method can be made fourth-order accurate in space if the difference approximation given by Eq. (3.31) is used for u_x:

$$(u_x)_j = \frac{1}{2\,\Delta x}\frac{\bar{\delta}_x}{1 + \delta_x^2/6}u_j + O[(\Delta x)^4] \tag{4.63}$$

The modified equation and phase error diagram for the resulting scheme can be found in the work by Beam and Warming (1976).

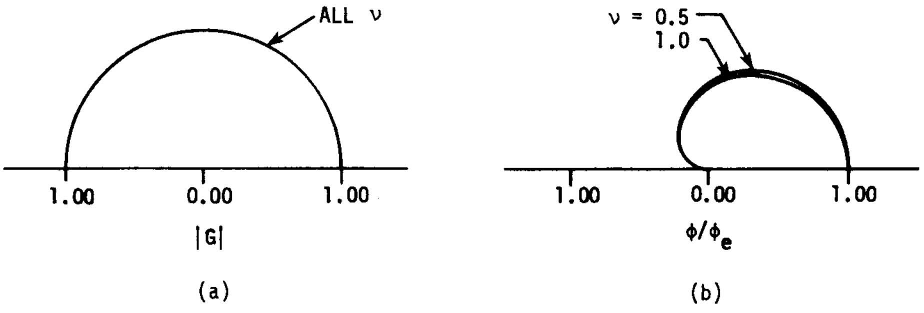

Figure 4.10 Time-centered implicit method. (a) Amplification factor modulus. (b) Relative phase error.

4.1.11 Rusanov (Burstein-Mirin) Method

The methods presented thus far for solving the wave equation have either been first-order or second-order accurate. Only a small number of third-order methods have appeared in the literature. Rusanov (1970) and Burstein and Mirin (1970) simultaneously developed the following explicit three-step method:

Step 1: $$u_{j+1/2}^{(1)} = \tfrac{1}{2}(u_{j+1}^n + u_j^n) - \tfrac{1}{3}\nu(u_{j+1}^n - u_j^n)$$

Step 2: $$u_j^{(2)} = u_j^n - \tfrac{2}{3}\nu\left(u_{j+1/2}^{(1)} - u_{j-1/2}^{(1)}\right)$$

Step 3: $$\begin{aligned} u_j^{n+1} = u_j^n &- \frac{1}{24}\nu(-2u_{j+2}^n + 7u_{j+1}^n - 7u_{j-1}^n + 2u_{j-2}^n) \\ &- \frac{3}{8}\nu\left(u_{j+1}^{(2)} - u_{j-1}^{(2)}\right) \\ &- \frac{\omega}{24}(u_{j+2}^n - 4u_{j+1}^n + 6u_j^n - 4u_{j-1}^n + u_{j-2}^n) \end{aligned} \tag{4.64}$$

Step 3 contains the fourth-order difference term

$$\delta_x^4 u_j^n = u_{j+2}^n - 4u_{j+1}^n + 6u_j^n - 4u_{j-1}^n + u_{j-2}^n$$

which is multiplied by a free parameter ω. This term has been added to make the scheme stable. The need for this term is apparent when we examine the stability requirements for the scheme:

$$|\nu| \leqslant 1$$

$$4\nu^2 - \nu^4 \leqslant \omega \leqslant 3 \tag{4.65}$$

If the fourth-order difference term were not present (i.e., $\omega = 0$), we could not satisfy Eq. (4.65) for $0 < \nu \leqslant 1$. The modified equation for this method is

$$\begin{aligned} u_t + cu_x = &-\frac{c(\Delta x)^3}{24}\left(\frac{\omega}{\nu} - 4\nu + \nu^3\right)u_{xxxx} \\ &+ \frac{c(\Delta x)^4}{120}(-5\omega + 4 + 15\nu^2 - 4\nu^4)u_{xxxxx} + \cdots \end{aligned} \tag{4.66}$$

In order to reduce the dissipation of this scheme, we can make the coefficient of the fourth derivative equal to zero by letting

$$\omega = 4\nu^2 - \nu^4 \tag{4.67}$$

In a like manner, we can reduce the dispersive error by setting the coefficient of the fifth derivative to zero, which gives

$$\omega = \frac{(4\nu^2 + 1)(4 - \nu^2)}{5} \tag{4.68}$$

The amplification factor for this method is

$$G = 1 - \frac{\nu^2}{2}\sin^2\beta - \frac{2\omega}{3}\sin^4\frac{\beta}{2} - i\nu\sin\beta\left[1 + \frac{2}{3}(1 - \nu^2)\sin^2\frac{\beta}{2}\right] \tag{4.69}$$

The modulus of the amplification factor and the relative phase error are plotted in Fig. 4.11. This figure shows that the Rusanov method has a leading or a lagging phase error, depending on the value of the free parameter ω.

4.1.12 Warming-Kutler-Lomax Method

Warming et al. (1973) developed a third-order method that uses MacCormack's method for the first two steps and has the same third step as the Rusanov method. This so-called WKL method is given by

Step 1: $$u_j^{(1)} = u_j^n - \tfrac{2}{3}\nu(u_{j+1}^n - u_j^n)$$

Step 2: $$u_j^{(2)} = \tfrac{1}{2}\left[u_j^n + u_j^{(1)} - \tfrac{2}{3}\nu\left(u_j^{(1)} - u_{j-1}^{(1)}\right)\right]$$

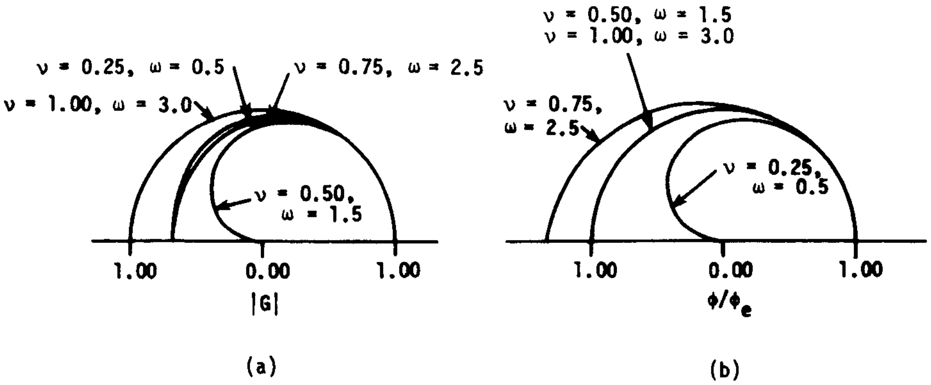

Figure 4.11 Rusanov method. (a) Amplification factor modulus. (b) Relative phase error.

Step 3:
$$u_j^{n+1} = u_j^n - \frac{1}{24}\nu(-2u_{j+2}^n + 7u_{j+1}^n - 7u_{j-1}^n + 2u_{j-2}^n) \qquad (4.70a)$$
$$-\frac{3}{8}\nu\left(u_{j+1}^{(2)} - u_{j-1}^{(2)}\right)$$
$$-\frac{\omega}{24}(u_{j+2}^n - 4u_{j+1}^n + 6u_j^n - 4u_{j-1}^n + u_{j-2}^n)$$

This method has the same stability bounds as the Rusanov method. In addition, the modified equation is identical to Eq. (4.66) for the present linear wave equation. The WKL method has the same advantage over the Rusanov method that the MacCormack method has over the two-step Lax-Wendroff method.

4.1.13 Runge-Kutta Methods

Runge-Kutta methods are frequently employed to solve ordinary differential equations (ODEs). They can also be applied to solve PDEs (Lomax et al., 1970; and Jameson et al., 1981, 1983). In fact, several of the methods described previously in this section can be derived using Runge-Kutta methodology. The first step in this process is to convert the PDE into a "pseudo-ODE." This is accomplished by separating out a partial derivative with respect to a single independent variable in the marching direction and placing the remaining partial derivatives into a term that is a function of the dependent variable. For example, the linear wave equation can be written as

$$u_t = R(u) \qquad (4.70b)$$

where $R(u) = -cu_x$. This pseudo-ODE is a time-continuous equation, and any integration scheme applicable to ODEs, including Runge-Kutta methods, may be used. Once the time differencing is completed, the partial derivatives contained in $R(u)$ can be differenced using appropriate spatial differences. To illustrate this approach, let us apply the second-order Runge-Kutta method, also referred to as the improved Euler's method (Carnahan et al., 1969), to Eq. (4.70*b*), which gives

Step 1:
$$u^{(1)} = u^n + \Delta t R^n$$

Step 2:
$$u^{n+1} = u^n + \frac{\Delta t}{2}(R^n + R^{(1)})$$

where

$$R^n \equiv R(u^n) = -cu_x^n$$

The term $R^{(1)}$ in step 2 can be evaluated by making use of step 1 in the following manner:

$$\begin{aligned} R^{(1)} &= -cu_x^{(1)} \\ &= -c(u_x^n + \Delta t R_x^n) \\ &= -cu_x^n + c^2 \Delta t u_{xx}^n \end{aligned}$$

Substituting this expression for $R^{(1)}$ into step 2 yields

$$u^{n+1} = u^n + \frac{\Delta t}{2}(-2cu_x^n + c^2 \Delta t u_{xx}^n)$$

If second-order accurate central differences are then used to approximate the spatial derivatives, the resulting scheme becomes

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

which is the second-order accurate Lax-Wendroff scheme, Eq. (4.44).

Procedures and equations for obtaining *n*th-order Runge-Kutta methods can be found in the works by Carnahan et al. (1969), Luther (1966), and Yu et al. (1992). A fourth-order Runge-Kutta method, attributed to Kutta, is given by

Step 1: $$u^{(1)} = u^n + \frac{\Delta t}{2} R^n$$

Step 2: $$u^{(2)} = u^n + \frac{\Delta t}{2} R^{(1)}$$

Step 3: $$u^{(3)} = u^n + \Delta t R^{(2)}$$

Step 4: $$u^{n+1} = u^n + \frac{\Delta t}{6}(R^n + 2R^{(1)} + 2R^{(2)} + R^{(3)})$$

where $R^{(\ )} = -cu_x^{(\ )}$ for the linear wave equation. If second-order accurate spatial differences are inserted into this algorithm, the resulting scheme will have a T.E. of $O[(\Delta t)^4, (\Delta x)^2]$. In order to obtain higher-order spatial accuracy, it is convenient to employ compact difference schemes (Yu et al., 1992) with the Runge-Kutta time stepping.

4.1.14 Additional Comments

The improved accuracy of higher-order methods is at the expense of added computer time and additional complexity. These factors must be considered carefully when choosing a scheme to solve a PDE. In general, second-order accurate methods provide enough accuracy for most practical problems.

For the 1-D linear wave equation, the second-order accurate explicit schemes such as the Lax-Wendroff scheme give excellent results with a minimum of computational effort. An implicit scheme may not be the optimum choice in this case because the solution is unsteady and intermediate results are typically desired at relatively small time intervals.

4.2 HEAT EQUATION

The 1-D heat equation (diffusion equation),

$$\frac{\partial u}{\partial t} = \alpha \frac{\partial^2 u}{\partial x^2} \tag{4.71}$$

is a parabolic PDE. In its present form, it is the governing equation for heat conduction or diffusion in a 1-D isotropic medium. It can be used to "model" in a rudimentary fashion the parabolic boundary-layer equations. The exact solution of the heat equation for the initial condition

$$u(x,0) = f(x)$$

and boundary conditions

$$u(0,t) = u(1,t) = 0$$

is

$$u(x,t) = \sum_{n=1}^{\infty} A_n e^{-\alpha k^2 t} \sin(kx) \tag{4.72}$$

where

$$A_n = 2\int_0^1 f(x) \sin(kx)\, dx$$

and $k = n\pi$. Let us now examine some of the more important finite-difference algorithms that can be used to solve the heat equation.

4.2.1 Simple Explicit Method

The following explicit one-step method,

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} = \alpha \frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{(\Delta x)^2} \tag{4.73}$$

is first-order accurate with T.E. of $O[\Delta t, (\Delta t)^2]$. At steady-state the accuracy is $O[(\Delta x)^2]$. As we have shown earlier, this scheme is stable whenever

$$0 \leqslant r \leqslant \tfrac{1}{2} \tag{4.74}$$

where

$$r = \frac{\alpha\, \Delta t}{(\Delta x)^2} \tag{4.75}$$

The modified equation is given by

$$u_t - \alpha u_{xx} = \left[-\frac{1}{2}\alpha^2\,\Delta t + \frac{\alpha(\Delta x)^2}{12}\right]u_{xxxx} + \left[\frac{1}{3}\alpha^3(\Delta t)^2 - \frac{1}{12}\alpha^2\,\Delta t(\Delta x)^2 + \frac{1}{360}\alpha(\Delta x)^4\right]u_{xxxxxx} + \cdots \tag{4.76}$$

We note that if $r = \frac{1}{6}$, the T.E. becomes of $O[(\Delta t)^2, (\Delta x)^4]$. It is also interesting to note that no odd derivative terms appear in the T.E. As a consequence, this scheme, as well as almost all other schemes for the heat equation, has no dispersive error. This fact can also be ascertained by examining the amplification factor for this scheme:

$$G = 1 + 2r(\cos\beta - 1) \tag{4.77}$$

which has no imaginary part and hence no phase shift. The amplification factor is plotted in Fig. 4.12 for two values of r and is compared with the exact amplification factor of the solution. The exact amplification (decay) factor is obtained by substituting the elemental solution

$$u = e^{-\alpha k_m^2 t} e^{i k_m x}$$

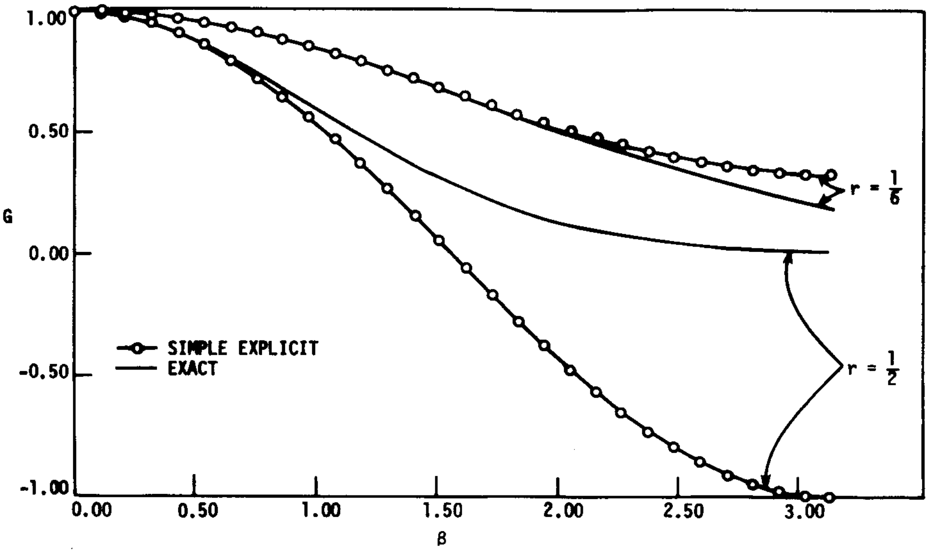

Figure 4.12 Amplification factor for simple explicit method.

into

$$G_e = \frac{u(t+\Delta t)}{u(t)}$$

which gives

$$G_e = e^{-\alpha k_m^2 \Delta t} \tag{4.78}$$

or

$$G_e = e^{-r\beta^2} \tag{4.79}$$

where $\beta = k_m \Delta x$. Hence the amplitude of the exact solution decreases by the factor $e^{-r\beta^2}$ during one time step, assuming no boundary condition influence.

In Fig. 4.12, we observe that the simple explicit method is highly dissipative for large values of β when $r = \frac{1}{2}$. As expected, the amplification factor is in closer agreement with the exact decay factor when $r = \frac{1}{6}$.

The present simple explicit scheme marches the solution outward from the initial data line in much the same manner as the explicit schemes of the previous section. This is illustrated in Fig. 4.13. In this figure we see that the unknown u can be calculated at point P without any knowledge of the boundary conditions along AB and CD. We know, however, that point P should depend on the boundary conditions along AB and CD, since the parabolic heat equation has the characteristic t = const. From this we conclude that the present explicit scheme (with a finite Δt) does not properly model the physical behavior of the parabolic PDE. It would appear that an implicit method would be the more appropriate method for solving a parabolic PDE, since an implicit method normally assimilates information from all grid points located on or below the characteristic t = const. On the other hand, explicit schemes seem to provide a

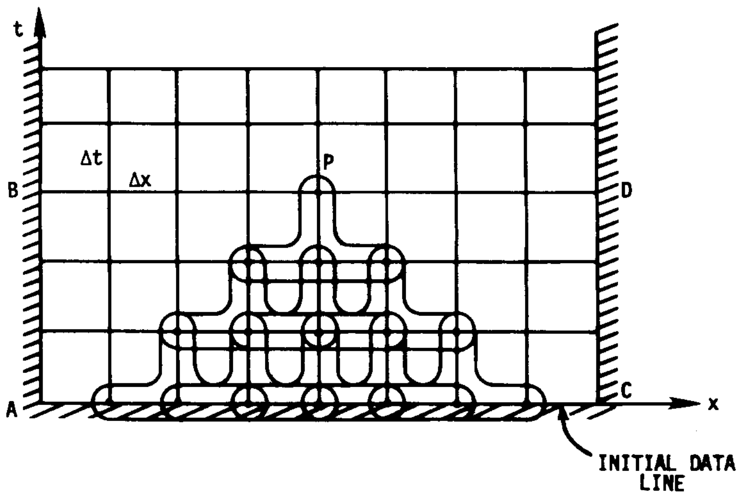

Figure 4.13 Zone of influence of simple explicit scheme.

more natural finite-difference approximation for hyperbolic PDEs that possess limited zones of influence.

Example 4.2 Suppose the simple explicit method is used to solve the heat equation ($\alpha = 0.05$) with the initial condition

$$u(x,0) = \sin(2\pi x) \qquad 0 \leqslant x \leqslant 1$$

and periodic boundary conditions. Determine the amplitude error after 10 steps if $\Delta t = 0.1$ and $\Delta x = 0.1$.

Solution A unique value of β can be determined in this problem for the same reason that was given in Example 4.1. Thus the value of β becomes

$$\beta = k_m \, \Delta x = (2\pi)(0.1) = 0.2\pi$$

After computing r,

$$r = \frac{\alpha \, \Delta t}{(\Delta x)^2} = \frac{(0.05)(0.1)}{(0.1)^2} = 0.5$$

the amplification factor for the simple explicit method is given by

$$G = 1 + 2r(\cos\beta - 1) = 0.809017$$

while the exact amplification factor becomes

$$G_e = e^{-r\beta^2} = 0.820869$$

As a result, the amplitude error is

$$A_0|G_e^{10} - G^{10}| = (1)(0.1389 - 0.1201) = 0.0188$$

Using Eq. (4.72), the exact solution after 10 steps ($t = 1.0$) is given by

$$u(x,1) = e^{-\alpha 4\pi^2} \sin(2\pi x) = 0.1389 \sin(2\pi x)$$

which can be compared to the numerical solution:

$$u(x,1) = 0.1201 \sin(2\pi x)$$

4.2.2 Richardson's Method

Richardson (1910) proposed the following explicit one-step three-time-level scheme for solving the heat equation:

$$\frac{u_j^{n+1} - u_j^{n-1}}{2\,\Delta t} = \alpha \frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{(\Delta x)^2} \tag{4.80}$$

This scheme is second-order accurate with T.E. of $O[(\Delta t)^2, (\Delta x)^2]$. Unfortunately, this method proves to be unconditionally unstable and cannot be used to solve the heat equation. It is presented here for historic purposes only.

4.2.3 Simple Implicit (Laasonen) Method

A simple implicit scheme for the heat equation was proposed by Laasonen (1949). The algorithm for this scheme is

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} = \alpha \frac{u_{j+1}^{n+1} - 2u_j^{n+1} + u_{j-1}^{n+1}}{(\Delta x)^2} \tag{4.81}$$

If we make use of the central-difference operator

$$\delta_x^2 u_j^n = u_{j+1}^n - 2u_j^n + u_{j-1}^n$$

we can rewrite Eq. (4.81) in the simpler form:

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} = \alpha \frac{\delta_x^2 u_j^{n+1}}{(\Delta x)^2} \tag{4.82}$$

This scheme has first-order accuracy with a T.E. of $O[\Delta t, (\Delta x)^2]$ and is unconditionally stable. Upon examining Eq. (4.82), it is apparent that a tridiagonal system of linear algebraic equations must be solved at each time level $n + 1$.

The modified equation for this scheme is

$$u_t - \alpha u_{xx} = \left[\frac{1}{2}\alpha^2 \Delta t + \frac{\alpha(\Delta x)^2}{12}\right] u_{xxxx} + \left[\frac{1}{3}\alpha^3(\Delta t)^2 + \frac{1}{12}\alpha^2 \Delta t(\Delta x)^2 + \frac{1}{360}\alpha(\Delta x)^4\right] u_{xxxxxx} + \cdots \tag{4.83}$$

It is interesting to observe that in this modified equation, the terms in the coefficient of u_{xxxx} are of the same sign, whereas they are of opposite sign in the modified equation for the simple explicit scheme, Eq. (4.76). This observation can explain why the simple explicit scheme is generally more accurate than the simple implicit scheme when used within the appropriate stability limits. The amplification factor for the simple implicit scheme,

$$G = [1 + 2r(1 - \cos \beta)]^{-1} \tag{4.84}$$

is plotted in Fig. 4.14 for $r = \frac{1}{2}$ and is compared with the exact decay factor.

4.2.4 Crank-Nicolson Method

Crank and Nicolson (1947) used the following implicit algorithm to solve the heat equation:

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} = \alpha \frac{\delta_x^2 u_j^n + \delta_x^2 u_j^{n+1}}{2(\Delta x)^2} \tag{4.85}$$

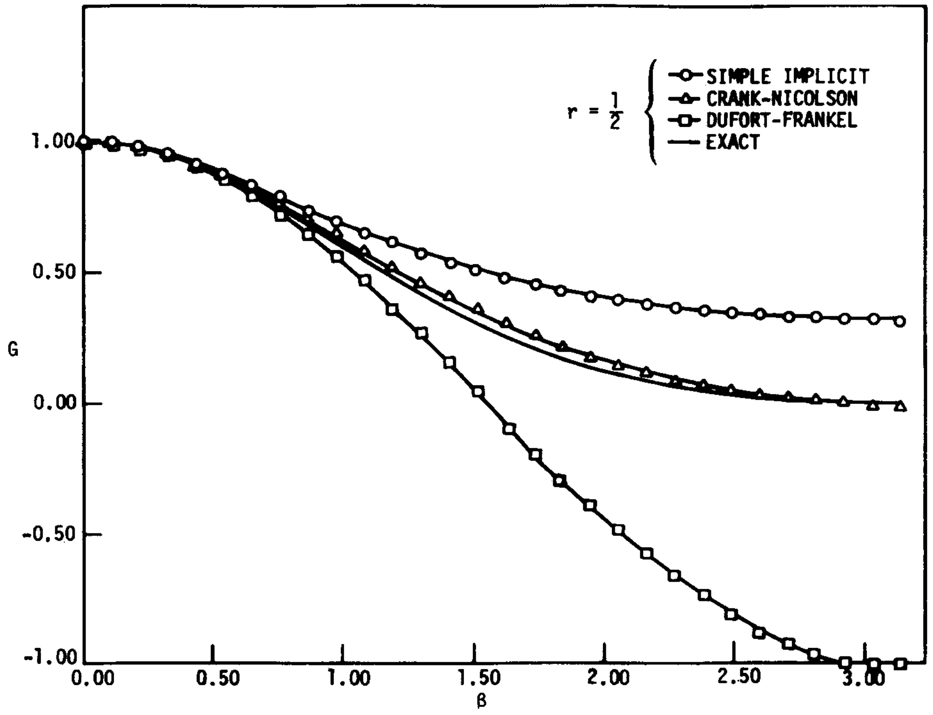

Figure 4.14 Amplification factors for several methods.

This unconditionally stable algorithm has become very well known and is referred to as the Crank-Nicolson scheme. This scheme makes use of trapezoidal differencing to achieve second-order accuracy with a T.E. of $O[(\Delta t)^2, (\Delta x)^2]$. Once again, a tridiagonal system of linear algebraic equations must be solved at each time level $n + 1$. The modified equation for the Crank-Nicolson method is

$$u_t - \alpha u_{xx} = \frac{\alpha(\Delta x)^2}{12} u_{xxxx} + \left[\frac{1}{12}\alpha^3(\Delta t)^2 + \frac{1}{360}\alpha(\Delta x)^4\right] u_{xxxxxx} + \cdots \tag{4.86}$$

The amplification factor

$$G = \frac{1 - r(1 - \cos\beta)}{1 + r(1 - \cos\beta)} \tag{4.87}$$

is plotted in Fig. 4.14 for $r = \frac{1}{2}$.

4.2.5 Combined Method A

The simple explicit, the simple implicit, and the Crank-Nicolson methods are special cases of a general algorithm given by

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} = \alpha \frac{\theta\, \delta_x^2 u_j^{n+1} + (1-\theta)\, \delta_x^2 u_j^n}{(\Delta x)^2} \tag{4.88}$$

where θ is a constant $(0 \leqslant \theta \leqslant 1)$. The simple explicit method corresponds to $\theta = 0$, the simple implicit method corresponds to $\theta = 1$, and the Crank-Nicolson method corresponds to $\theta = \frac{1}{2}$. This combined method has first-order accuracy with T.E. of $O[\Delta t, (\Delta x)^2]$ except for special cases such as

1. $\theta = \frac{1}{2}$(Crank-Nicolson method) $\quad$ T.E. $= O[(\Delta t)^2, (\Delta x)^2]$

2. $\theta = \dfrac{1}{2} - \dfrac{(\Delta x)^2}{12\alpha\,\Delta t}$ $\quad$ T.E. $= O[(\Delta t)^2, (\Delta x)^4]$

3. $\theta = \dfrac{1}{2} - \dfrac{(\Delta x)^2}{12\alpha\,\Delta t}$ and $\dfrac{(\Delta x)^2}{\alpha\,\Delta t} = \sqrt{20}$ $\quad$ T.E. $= O[(\Delta t)^2, (\Delta x)^6]$

The T.E.s of these special cases can be obtained by examining the modified equation

$$u_t - \alpha u_{xx} = \left[\left(\theta - \frac{1}{2}\right)\alpha^2\,\Delta t + \frac{\alpha(\Delta x)^2}{12}\right]u_{xxxx} + \left[\left(\theta^2 - \theta + \frac{1}{3}\right)\alpha^3(\Delta t)^2 \right.$$
$$\left. + \frac{1}{6}\left(\theta - \frac{1}{2}\right)\alpha^2\,\Delta t(\Delta x)^2 + \frac{1}{360}\alpha(\Delta x)^4\right]u_{xxxxxx} + \cdots \tag{4.89}$$

The present combined method is unconditionally stable if $\frac{1}{2} \leqslant \theta \leqslant 1$. However, when $0 \leqslant \theta < \frac{1}{2}$, the method is stable only if

$$0 \leqslant r \leqslant \frac{1}{2 - 4\theta} \tag{4.90}$$

4.2.6 Combined Method B

Richtmyer and Morton (1967) present the following general algorithm for a three-time-level implicit scheme:

$$(1+\theta)\frac{u_j^{n+1} - u_j^n}{\Delta t} - \theta\frac{u_j^n - u_j^{n-1}}{\Delta t} = \alpha\frac{\delta_x^2 u_j^{n+1}}{(\Delta x)^2} \tag{4.91}$$

This general algorithm has first-order accuracy with T.E. of $O[\Delta t, (\Delta x)^2]$ except for special cases:

1. $\theta = \frac{1}{2}$ $\quad$ T.E. $= O[(\Delta t)^2, (\Delta x)^2]$

2. $\theta = \dfrac{1}{2} + \dfrac{(\Delta x)^2}{12\alpha\,\Delta t}$ $\quad$ T.E. $= O[(\Delta t)^2, (\Delta x)^4]$

which can be verified by examining the modified equation

$$u_t - \alpha u_{xx} = \left[-(\theta - \tfrac{1}{2})\alpha^2 \Delta t + \tfrac{1}{12}\alpha(\Delta x)^2\right] u_{xxxx} + \cdots$$

4.2.7 DuFort-Frankel Method

The unstable Richardson method [Eq. (4.80)] can be made stable by replacing u_j^n with the time-averaged expression $(u_j^{n+1} + u_j^{n-1})/2$. The resulting explicit three-time-level scheme,

$$\frac{u_j^{n+1} - u_j^{n-1}}{2\,\Delta t} = \alpha \frac{u_{j+1}^n - u_j^{n+1} - u_j^{n-1} + u_{j-1}^n}{(\Delta x)^2} \tag{4.92}$$

was first proposed by DuFort and Frankel (1953). Note that Eq. (4.92) can be rewritten as

$$u_j^{n+1}(1 + 2r) = u_j^{n-1} + 2r\left(u_{j+1}^n - u_j^{n-1} + u_{j-1}^n\right) \tag{4.93}$$

so that only one unknown, u_j^{n+1}, appears in the scheme, and therefore it is explicit. The T.E. for the DuFort-Frankel method is $O[(\Delta t)^2, (\Delta x)^2, (\Delta t/\Delta x)^2]$. Consequently, if this method is to be consistent, then $(\Delta t/\Delta x)^2$ must approach zero as Δt and Δx approach zero. As pointed out in Chapter 3, if $\Delta t/\Delta x$ approaches a constant value γ, instead of zero, the DuFort-Frankel scheme is consistent with the hyperbolic equation

$$\frac{\partial u}{\partial t} + \alpha\gamma^2 \frac{\partial^2 u}{\partial t^2} = \alpha \frac{\partial^2 u}{\partial x^2}$$

If we let r remain constant as Δt and Δx approach zero, the term $(\Delta t/\Delta x)^2$ becomes formally a first-order term of $O(\Delta t)$. The modified equation is given by

$$\begin{aligned} u_t - \alpha u_{xx} &= \left[\frac{1}{12}\alpha(\Delta x)^2 - \alpha^3 \frac{(\Delta t)^2}{(\Delta x)^2}\right] u_{xxxx} \\ &\quad + \left[\frac{1}{360}\alpha(\Delta x)^4 - \frac{1}{3}\alpha^3(\Delta t)^2 + 2\alpha^5 \frac{(\Delta t)^4}{(\Delta x)^4}\right] u_{xxxxxx} + \cdots \end{aligned}$$

The amplification factor

$$G = \frac{2r\cos\beta \pm \sqrt{1 - 4r^2\sin^2\beta}}{1 + 2r}$$

is plotted in Fig. 4.14 for $r = \frac{1}{2}$. The explicit DuFort-Frankel scheme has the unusual property of being unconditionally stable for $r \geqslant 0$. In passing, we note that the DuFort-Frankel method can be extended to two and three dimensions without any unexpected penalties. The scheme remains unconditionally stable.

4.2.8 Keller Box and Modified Box Methods

The Keller box method (Keller, 1970) for parabolic PDEs is an implicit scheme with second-order accuracy in both space and time. This formulation allows for the spatial and temporal steps to vary without causing deterioration in the formal second-order accuracy. The scheme differs from others considered thus far, in that second and higher derivatives are replaced by first derivatives through the introduction of additional variables as discussed in Section 2.5. Thus a system of first-order equations results. For the 1-D heat equation,

$$\frac{\partial u}{\partial t} = \alpha \frac{\partial^2 u}{\partial x^2}$$

we can define

$$v = \frac{\partial u}{\partial x}$$

so that the second-order heat equation can be written as a system of two first-order equations:

$$\frac{\partial u}{\partial x} = v$$

$$\frac{\partial u}{\partial t} = \alpha \frac{\partial v}{\partial x}$$

Now we endeavor to approximate these equations using only central differences, making use of the four points at the corners of a "box" about $(n + \frac{1}{2}, j - \frac{1}{2})$ (see Fig. 4.15). The resulting difference equations are

$$\frac{u_j^{n+1} - u_{j-1}^{n+1}}{\Delta x_j} = v_{j-\frac{1}{2}}^{n+1} \tag{4.94a}$$

$$\frac{u_{j-\frac{1}{2}}^{n+1} - u_{j-\frac{1}{2}}^{n}}{\Delta t_{n+1}} = \frac{\alpha\left(v_j^{n+\frac{1}{2}} - v_{j-1}^{n+\frac{1}{2}}\right)}{\Delta x_j} \tag{4.94b}$$

where the difference molecules are shown in Figs. 4.16 and 4.17. The mesh functions that contain a subscript or superscript $\frac{1}{2}$ are defined as averages, as for example,

$$u_{j-\frac{1}{2}}^{n+1} = \frac{u_j^{n+1} + u_{j-1}^{n+1}}{2}$$

$$v_j^{n+\frac{1}{2}} = \frac{v_j^n + v_j^{n+1}}{2}$$

After substituting the averaged expressions into Eqs. (4.94*a*) and (4.94*b*), the new difference equations become

$$\frac{u_j^{n+1} - u_{j-1}^{n+1}}{\Delta x_j} = \frac{v_j^{n+1} + v_{j-1}^{n+1}}{2} \tag{4.95a}$$

$$\frac{u_j^{n+1} + u_{j-1}^{n+1}}{\Delta t_{n+1}} = \alpha \frac{v_j^{n+1} - v_{j-1}^{n+1}}{\Delta x_j} + \frac{u_j^n + u_{j-1}^n}{\Delta t_{n+1}} + \alpha \frac{v_j^n - v_{j-1}^n}{\Delta x_j} \tag{4.95b}$$

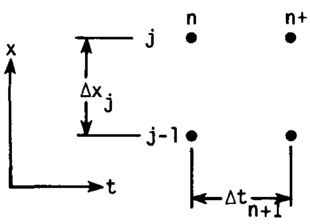

Figure 4.15 Grid for box scheme.

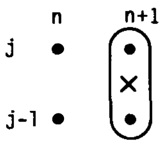

Figure 4.16 Difference molecule for evaluation of $v_{j-1/2}^{n+1}$.

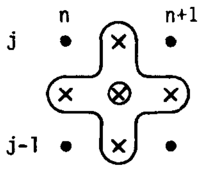

Figure 4.17 Difference molecule for Eq. (4.94*b*).

The unknowns (u^{n+1}, v^{n+1}) in the above equations are located at grid points j and $j - 1$. However, when boundary conditions are included, the unknowns may also occur at grid point $j + 1$. Thus the algebraic system resulting from the Keller box scheme for the general point can be represented in matrix form as

$$[B]\mathbf{F}_{j-1}^{n+1} + [D]\mathbf{F}_j^{n+1} + [A]\mathbf{F}_{j+1}^{n+1} = \mathbf{C}$$

where

$$\mathbf{F} = [u, v]^T$$

and $[B]$, $[D]$, and $[A]$ are 2×2 matrices and $\mathbf{C}$ is a two-component vector. When the entire system of equations for a given problem is assembled and boundary conditions are taken into account, the algebraic problem can be expressed in the general form $[M]\mathbf{x} = \mathbf{c}$, where the "elements" of the coefficient matrix $[M]$ are now 2×2 matrices, and each "component" of the column vectors becomes the two components of $\mathbf{F}_j^{n+1}$ and $\mathbf{C}_j$ associated with point j. This system of equations is a block tridiagonal system and can be solved with the general block tridiagonal algorithm given in Appendix B or with a special-purpose algorithm specialized to take advantage of zeros that may be present in the coefficient matrices. The block algorithm actually proceeds with the same operations as for the scalar tridiagonal algorithm with matrix and matrix-vector multiplications replacing scalar operations. When division by a matrix would be indicated by this analogy, premultiplication by the inverse of the matrix is carried out.

The work required to solve the algebraic system resulting from the box-difference stencil can be reduced by combining the difference representations at two adjacent grid points to eliminate one variable. This system can then be solved with the simple scalar Thomas algorithm. This revision of the box method, which simplifies the final algebraic formulation, will be referred to as the *modified box method.*

Modified box method. The strategy in the development of the modified box method is to express the v's in terms of u's. The term v_{j-1}^{n+1} can be eliminated from Eq. (4.95*b*) by a simple substitution using Eq. (4.95*a*). Similarly, v_{j-1}^{n} can be eliminated through substitution by evaluating Eq. (4.95*a*) at time level n. This gives

$$\frac{u_j^{n+1}+u_{j-1}^{n+1}}{\Delta t_{n+1}} = 2\alpha\frac{v_j^{n+1}}{\Delta x_j} - 2\alpha\frac{u_j^{n+1}-u_{j-1}^{n+1}}{(\Delta x_j)^2} + \frac{u_j^n+u_{j-1}^n}{\Delta t_{n+1}} + 2\alpha\frac{v_j^n}{\Delta x_j} - 2\alpha\frac{u_j^n-u_{j-1}^n}{(\Delta x_j)^2} \tag{4.96a}$$

To eliminate v_j^{n+1} and v_j^n, Eqs. (4.95*a*) and (4.95*b*) can first be rewritten with the j index advanced by 1 and combined. The result is

$$\frac{u_{j+1}^{n+1}+u_j^{n+1}}{\Delta t_{n+1}} = \frac{-2\alpha v_j^{n+1}}{\Delta x_{j+1}} + \frac{2\alpha\left(u_{j+1}^{n+1}-u_j^{n+1}\right)}{(\Delta x_{j+1})^2} + \frac{u_{j+1}^n+u_j^n}{\Delta t_{n+1}} + \frac{-2\alpha v_j^n}{\Delta x_{j+1}} + 2\alpha\frac{u_{j+1}^n-u_j^n}{(\Delta x_{j+1})^2} \tag{4.96b}$$

The terms v_j^{n+1} and v_j^n can then be eliminated by multiplying Eq. (4.96*a*) by Δx_j and Eq. (4.96*b*) by Δx_{j+1} and adding the two products. The result can be written in the tridiagonal format

$$B_j u_{j-1}^{n+1} + D_j u_j^{n+1} + A_j u_{j+1}^{n+1} = C_j$$

where

$$B_j = \frac{\Delta x_j}{\Delta t_{n+1}} - \frac{2\alpha}{\Delta x_j} \qquad A_j = \frac{\Delta x_{j+1}}{\Delta t_{n+1}} - \frac{2\alpha}{\Delta x_{j+1}}$$

$$D_j = \frac{\Delta x_j}{\Delta t_{n+1}} + \frac{\Delta x_{j+1}}{\Delta t_{n+1}} + \frac{2\alpha}{\Delta x_j} + \frac{2\alpha}{\Delta x_{j+1}}$$

$$C_j = 2\alpha\frac{u_{j-1}^n-u_j^n}{\Delta x_j} + 2\alpha\frac{u_{j+1}^n-u_j^n}{\Delta x_{j+1}} + (u_j^n+u_{j-1}^n)\frac{\Delta x_j}{\Delta t_{n+1}} + (u_{j+1}^n+u_j^n)\frac{\Delta x_{j+1}}{\Delta t_{n+1}}$$

The above equations can be simplified somewhat if the spacing in the x direction is uniform. Even then, a few more algebraic operations per time step are required than for the Crank-Nicolson scheme, which is also second-order accurate for a uniformly spaced mesh. A conceptual advantage of schemes based on the box-difference molecule is that formal second-order accuracy is achieved even when the mesh is nonuniform. The Crank-Nicolson scheme can be extended to cases of nonuniform grid spacing by representing the second derivative term as indicated for Laplace's equation in Eq. (3.97). Formally, the T.E. for that representation is reduced to first order for arbitrary grid spacing. Blottner (1974) has shown that if the variable grid spacing used is one that could be established through a coordinate stretching transformation, then the Crank-Nicolson scheme is also second-order accurate for that variable grid arrangement.

4.2.9 Methods for the Two-Dimensional Heat Equation

The 2-D heat equation is given by

$$\frac{\partial u}{\partial t} = \alpha\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) \tag{4.97}$$

Since this PDE is different from the 1-D equation, caution must be exercised when attempting to apply the previous finite-difference methods to this equation. The following two examples illustrate some of the difficulties. If we apply the simple explicit method to the 2-D heat equation, the following algorithm results:

$$\frac{u_{i,j}^{n+1} - u_{i,j}^n}{\Delta t} = \alpha\left[\frac{u_{i+1,j}^n - 2u_{i,j}^n + u_{i-1,j}^n}{(\Delta x)^2} + \frac{u_{i,j+1}^n - 2u_{i,j}^n + u_{i,j-1}^n}{(\Delta y)^2}\right] \tag{4.98}$$

where $x = i\,\Delta x$ and $y = j\,\Delta y$. As shown in Chapter 3, the stability condition is

$$\alpha\,\Delta t\left[\frac{1}{(\Delta x)^2} + \frac{1}{(\Delta y)^2}\right] \leqslant \frac{1}{2}$$

If $(\Delta x)^2 = (\Delta y)^2$, the stability condition reduces to $r \leqslant \frac{1}{4}$, which is twice as restrictive as the 1-D constraint $r \leqslant \frac{1}{2}$ and makes this method even more impractical.

When we apply the Crank-Nicolson scheme to the 2-D heat equation, we obtain

$$\frac{u_{i,j}^{n+1} - u_{i,j}^n}{\Delta t} = \frac{\alpha}{2}\left(\hat{\delta}_x^2 + \hat{\delta}_y^2\right)\left(u_{i,j}^{n+1} + u_{i,j}^n\right) \tag{4.99}$$

where the 2-D central-difference operators $\hat{\delta}_x^2$ and $\hat{\delta}_y^2$ are defined by

$$\begin{aligned}\hat{\delta}_x^2 u_{i,j}^n &= \frac{u_{i+1,j}^n - 2u_{i,j}^n + u_{i-1,j}^n}{(\Delta x)^2} = \frac{\delta_x^2 u_{i,j}^n}{(\Delta x)^2}\\ \hat{\delta}_y^2 u_{i,j}^n &= \frac{u_{i,j+1}^n - 2u_{i,j}^n + u_{i,j-1}^n}{(\Delta y)^2} = \frac{\delta_y^2 u_{i,j}^n}{(\Delta y)^2}\end{aligned} \tag{4.100}$$

As with the 1-D case, the Crank-Nicolson scheme is unconditionally stable when applied to the 2-D heat equation with periodic boundary conditions. Unfortunately, the resulting system of linear algebraic equations is no longer tridiagonal because of the five unknowns $u_{i,j}^{n+1}$, $u_{i+1,j}^{n+1}$, $u_{i-1,j}^{n+1}$, $u_{i,j+1}^{n+1}$, and $u_{i,j-1}^{n+1}$. The same is true for all the implicit schemes we have studied previously. In order to examine this further, let us rewrite Eq. (4.99) as

$$au_{i,j-1}^{n+1} + bu_{i-1,j}^{n+1} + cu_{i,j}^{n+1} + bu_{i+1,j}^{n+1} + au_{i,j+1}^{n+1} = d_{i,j}^{n} \tag{4.101}$$

where

$$a = -\frac{\alpha\,\Delta t}{2(\Delta y)^2} = -\frac{1}{2}r_y$$

$$b = -\frac{\alpha\,\Delta t}{2(\Delta x)^2} = -\frac{1}{2}r_x$$

$$c = 1 + r_x + r_y$$

$$d_{i,j}^{n} = u_{i,j}^{n} + \frac{\alpha\,\Delta t}{2}\left(\hat{\delta}_x^2 + \hat{\delta}_y^2\right)u_{i,j}^{n}$$

If we apply Eq. (4.101) to the 2-D (6 × 6) computational mesh shown in Fig. 4.18, the following system of 16 linear algebraic equations must be solved at each (n + 1) time level:

$$\begin{bmatrix}
c & b & 0 & 0 & a & 0 & & & & & & & & & & 0 \\
b & c & b & & & a & & & & & & & & & & \\
0 & b & c & b & & & a & & & & & & & & & \\
0 & & b & c & 0 & & & a & & & & & & & & \\
a & & & 0 & c & b & & & a & & & & & & & \\
0 & a & & & b & c & b & & & a & & & & & & \\
& & a & & & b & c & b & & & a & & & & & \\
& & & a & & & b & c & 0 & & & a & & & & \\
& & & & a & & & 0 & c & b & & & a & & & \\
& & & & & a & & & b & c & b & & & a & & \\
& & & & & & a & & & b & c & b & & & a & 0 \\
& & & & & & & a & & & b & c & 0 & & & a \\
& & & & & & & & a & & & 0 & c & b & & 0 \\
& & & & & & & & & a & & & b & c & b & 0 \\
& & & & & & & & & & a & & & b & c & b \\
0 & & & & & & & & & & 0 & a & 0 & 0 & b & c
\end{bmatrix}
\begin{bmatrix}
u_{2,2}^{n+1} \\ u_{3,2}^{n+1} \\ u_{4,2}^{n+1} \\ u_{5,2}^{n+1} \\ u_{2,3}^{n+1} \\ u_{3,3}^{n+1} \\ u_{4,3}^{n+1} \\ u_{5,3}^{n+1} \\ u_{2,4}^{n+1} \\ u_{3,4}^{n+1} \\ u_{4,4}^{n+1} \\ u_{5,4}^{n+1} \\ u_{2,5}^{n+1} \\ u_{3,5}^{n+1} \\ u_{4,5}^{n+1} \\ u_{5,5}^{n+1}
\end{bmatrix}
=
\begin{bmatrix}
d'''_{2,2} \\ d'_{3,2} \\ d'_{4,2} \\ d'''_{5,2} \\ d''_{2,3} \\ d_{3,3} \\ d_{4,3} \\ d''_{5,3} \\ d''_{2,4} \\ d_{3,4} \\ d_{4,4} \\ d''_{5,4} \\ d'''_{2,5} \\ d'_{3,5} \\ d'_{4,5} \\ d'''_{5,5}
\end{bmatrix}
\tag{4.102}$$

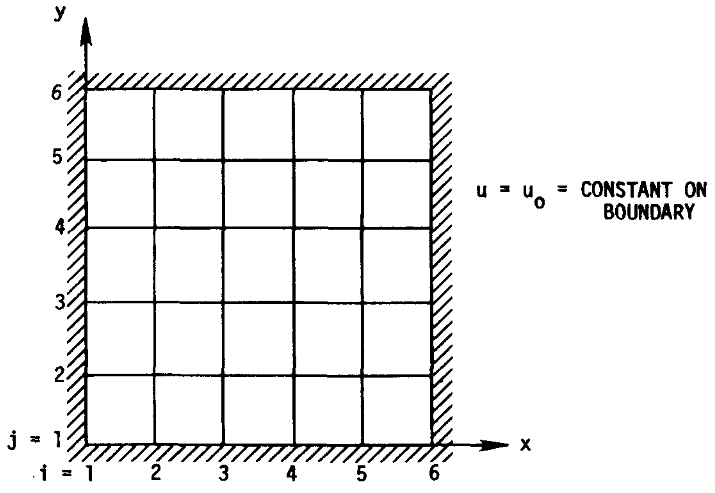

Figure 4.18 Two-dimensional computational mesh.

$$\begin{aligned} \text{where } d' &= d - au_0 \\ d'' &= d - bu_0 \\ d''' &= d - (a+b)u_0 \end{aligned}$$

A system of equations like Eq. (4.102) requires substantially more computer time to solve than does a tridiagonal system. In fact, equations of this type are often solved by iterative methods. These methods are discussed in Section 4.3.

4.2.10 ADI Methods

The difficulties described above, which occur when attempting to solve the 2-D heat equation by conventional algorithms, led to the development of alternating-direction implicit (ADI) methods by Peaceman and Rachford (1955) and Douglas (1955). The usual ADI method is a two-step scheme given by

$$\begin{aligned} &\text{Step 1:} \quad \frac{u_{i,j}^{n+1/2} - u_{i,j}^{n}}{\Delta t/2} = \alpha\left(\hat{\delta}_x^2 u_{i,j}^{n+1/2} + \hat{\delta}_y^2 u_{i,j}^{n}\right) \\ &\text{Step 2:} \quad \frac{u_{i,j}^{n+1} - u_{i,j}^{n+1/2}}{\Delta t/2} = \alpha\left(\hat{\delta}_x^2 u_{i,j}^{n+1/2} + \hat{\delta}_y^2 u_{i,j}^{n+1}\right) \end{aligned} \tag{4.103}$$

As a result of the "splitting" that is employed in this algorithm, only tridiagonal systems of linear algebraic equations must be solved. During step 1, a tridiagonal matrix is solved for each j row of grid points, and during step 2, a tridiagonal matrix is solved for each i row of grid points. This procedure is illustrated in Fig. 4.19. The ADI method is second-order accurate with a T.E. of

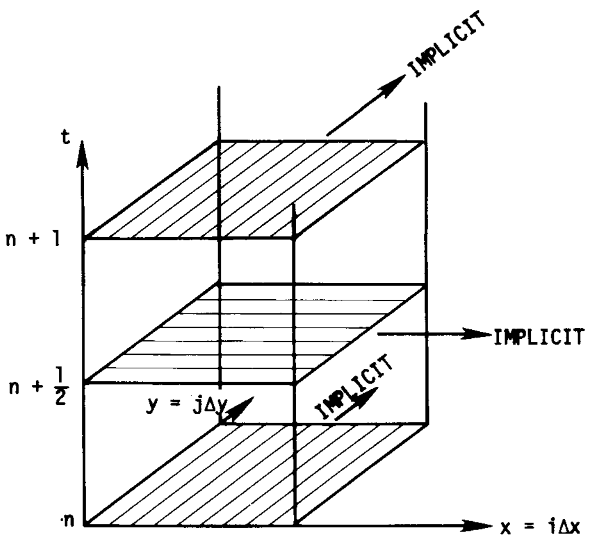

Figure 4.19 ADI calculation procedure.

$O[(\Delta t)^2, (\Delta x)^2, (\Delta y)^2]$. Upon examining the amplification factor

$$G = \frac{[1 - r_x(1 - \cos\beta_x)][1 - r_y(1 - \cos\beta_y)]}{[1 + r_x(1 - \cos\beta_x)][1 + r_y(1 - \cos\beta_y)]}$$

where

$$r_x = \alpha\,\Delta t/(\Delta x)^2 \qquad \beta_x = k_m\,\Delta x$$
$$r_y = \alpha\,\Delta t/(\Delta y)^2 \qquad \beta_y = k_m\,\Delta y$$

we find this method to be unconditionally stable. The obvious extension of this method to three dimensions (making use of the time levels $n, n + \frac{1}{3}, n + \frac{2}{3}, n + 1$) leads to a conditionally stable method with T.E. of $O[(\Delta t, (\Delta x)^2, (\Delta y)^2, (\Delta z)^2]$.

In order to circumvent these problems, Douglas and Gunn (1964) developed a general method for deriving ADI schemes that are unconditionally stable and retain second-order accuracy. Their method of derivation is commonly called *approximate factorization.* When an implicit procedure such as the Crank-Nicolson scheme is cast into residual or *delta* form, the motivation for factoring becomes evident. The delta form is obtained by defining $\Delta u_{i,j}$ as

$$\Delta u_{i,j} \equiv u_{i,j}^{n+1} - u_{i,j}^{n}$$

Substituting this into the 2-D Crank-Nicolson scheme [Eq. (4.99)] gives

$$\Delta u_{i,j} = \frac{\alpha\,\Delta t}{2}\left[\frac{\delta_x^2\,\Delta u_{i,j}}{(\Delta x)^2} + \frac{\delta_y^2\,\Delta u_{i,j}}{(\Delta y)^2} + \frac{2\,\delta_x^2 u_{i,j}^n}{(\Delta x)^2} + \frac{2\,\delta_y^2 u_{i,j}^n}{(\Delta y)^2}\right]$$

After rearranging this equation to put all of the unknowns on the left-hand side of the equation and inserting r_x and r_y, we obtain

$$\left(1 - \frac{r_x}{2}\delta_x^2 - \frac{r_y}{2}\delta_y^2\right)\Delta u_{i,j} = \left(r_x\delta_x^2 + r_y\delta_y^2\right)u_{i,j}^n$$

If the quantity in parentheses on the left can be arranged as the product of two operators, one involving x-direction differences and the other involving y-direction differences, then the algorithm can proceed in two steps. One such *factorization* is

$$\left(1 - \frac{r_x}{2}\delta_x^2\right)\left(1 - \frac{r_y}{2}\delta_y^2\right)\Delta u_{i,j} = \left(r_x\delta_x^2 + r_y\delta_y^2\right)u_{i,j}^n$$

In order to achieve this factorization, the quantity $r_x r_y\,\delta_x^2\delta_y^2\,\Delta u_{i,j}/4$ must be added to the left-hand side. The T.E. is thus augmented by the same amount. The factored equation can now be solved in the two steps:

Step 1: $$\left(1 - \frac{r_x}{2}\delta_x^2\right)\Delta u_{i,j}^* = \left(r_x\delta_x^2 + r_y\delta_y^2\right)u_{i,j}^n$$

Step 2: $$\left(1 - \frac{r_y}{2}\delta_y^2\right)\Delta u_{i,j} = \Delta u_{i,j}^*$$

where the superscript asterisk denotes an intermediate value. The unknown $u_{i,j}^{n+1}$ is then obtained from

$$u_{i,j}^{n+1} = u_{i,j}^n + \Delta u_{i,j}$$

Using the preceding factorization approach and starting with the three-dimensional (3-D) Crank-Nicolson scheme, Douglas and Gunn also developed an algorithm to solve the 3-D heat equation:

$$\begin{aligned}
&\text{Step 1:} && \left(1 - \frac{r_x}{2}\delta_x^2\right)\Delta u^* = \left(r_x\delta_x^2 + r_y\delta_y^2 + r_z\delta_z^2\right)u^n \\
&\text{Step 2:} && \left(1 - \frac{r_y}{2}\delta_y^2\right)\Delta u^{**} = \Delta u^* \qquad (4.104)\\
&\text{Step 3:} && \left(1 - \frac{r_z}{2}\delta_z^2\right)\Delta u = \Delta u^{**}
\end{aligned}$$

where the superscript asterisks and double asterisks denote intermediate values and the subscripts i, j, k have been dropped from each term.

4.2.11 Splitting or Fractional-Step Methods

The ADI methods are closely related and in some cases identical to the method of fractional steps or methods of splitting, which were developed by Soviet mathematicians at about the same time as the ADI methods were developed in the United States. The basic idea of these methods is to split a finite-difference algorithm into a sequence of 1-D operations. For example, the simple implicit scheme applied to the 2-D heat equation could be split in the following manner:

$$\text{Step 1:} \qquad \frac{u_{i,j}^{n+1/2} - u_{i,j}^{n}}{\Delta t} = \alpha\, \hat{\delta}_x^2 u_{i,j}^{n+1/2}$$

$$\text{Step 2:} \qquad \frac{u_{i,j}^{n+1} - u_{i,j}^{n+1/2}}{\Delta t} = \alpha\, \hat{\delta}_y^2 u_{i,j}^{n+1} \tag{4.105}$$

to give a first-order accurate method with a T.E. of $O[\Delta t, (\Delta x)^2, (\Delta y)^2]$. For further details on the method of fractional steps, the reader is urged to consult the book by Yanenko (1971).

4.2.12 ADE Methods

Another way of solving the 2-D heat equation is by means of an alternating-direction explicit (ADE) method. Unlike the ADI methods, the ADE methods do not require tridiagonal matrices to be "inverted." Since the ADE methods can also be used to solve the 1-D heat equation, we will apply the ADE algorithms to this equation, for simplicity.

The first ADE method was proposed by Saul'yev (1957). His two-step scheme is given by

$$\text{Step 1:} \qquad \frac{u_j^{n+1} - u_j^{n}}{\Delta t} = \alpha \frac{u_{j-1}^{n+1} - u_j^{n+1} - u_j^{n} + u_{j+1}^{n}}{(\Delta x)^2}$$

$$\text{Step 2:} \qquad \frac{u_j^{n+2} - u_j^{n+1}}{\Delta t} = \alpha \frac{u_{j-1}^{n+1} - u_j^{n+1} - u_j^{n+2} + u_{j+1}^{n+2}}{(\Delta x)^2} \tag{4.106}$$

In the application of this method, step 1 marches the solution from the left boundary to the right boundary. By marching in this direction, u_{j-1}^{n+1} is always known, and consequently, u_j^{n+1} can be determined "explicitly." In a like manner, step 2 marches the solution from the right boundary to the left boundary, again resulting in an "explicit" formulation, since u_{j+1}^{n+2} is always known. We assume that u is known on the boundaries. Although this scheme involves three time levels, only one storage array is required for u because of the unique way in which the calculation procedure sweeps through the mesh. This scheme is unconditionally stable, and the T.E. is $O[(\Delta t)^2, (\Delta x)^2, (\Delta t/\Delta x)^2]$. The scheme is formally first-order accurate (if r is constant) owing to the presence of the inconsistent term $(\Delta t/\Delta x)^2$ in the T.E.

Another ADE method was proposed by Barakat and Clark (1966). In this method the calculation procedure is simultaneously "marched" in both directions, and the resulting solutions (p_j^{n+1} and q_j^{n+1}) are averaged to obtain the final value of u_j^{n+1}:

$$\frac{p_j^{n+1} - p_j^n}{\Delta t} = \alpha \frac{p_{j-1}^{n+1} - p_j^{n+1} - p_j^n + p_{j+1}^n}{(\Delta x)^2}$$
$$\frac{q_j^{n+1} - q_j^n}{\Delta t} = \alpha \frac{q_{j-1}^n - q_j^n - q_j^{n+1} + q_{j+1}^{n+1}}{(\Delta x)^2} \tag{4.107}$$
$$u_j^{n+1} = \tfrac{1}{2}\left(p_j^{n+1} + q_j^{n+1}\right)$$

This method is unconditionally stable, and the T.E. is approximately $O[(\Delta t)^2, (\Delta x)^2]$ because the simultaneous marching tends to cancel the $(\Delta t/\Delta x)^2$ terms. It has been observed that this method is about 18/16 times faster than the ADI method for the 2-D heat equation.

Larkin (1964) proposed a slightly different algorithm, which replaces the p and q with u whenever possible. His algorithm is

$$\frac{p_j^{n+1} - u_j^n}{\Delta t} = \alpha \frac{p_{j-1}^{n+1} - p_j^{n+1} - u_j^n + u_{j+1}^n}{(\Delta x)^2}$$
$$\frac{q_j^{n+1} - u_j^n}{\Delta t} = \alpha \frac{u_{j-1}^n - u_j^n - q_j^{n+1} + q_{j+1}^{n+1}}{(\Delta x)^2} \tag{4.108}$$
$$u_j^{n+1} = \tfrac{1}{2}\left(p_j^{n+1} + q_j^{n+1}\right)$$

Numerical tests indicate that this method is usually less accurate than the Barakat and Clark scheme.

4.2.13 Hopscotch Method

As our final algorithm for solving the 2-D heat equation, let us examine the hopscotch method. This method is an explicit procedure that is unconditionally stable. The calculation procedure, illustrated in Fig. 4.20, involves two sweeps through the mesh. For the first sweep, $u_{i,j}^{n+1}$ is computed at each grid point (for which $i + j + n$ is even) by the simple explicit scheme

$$\frac{u_{i,j}^{n+1} - u_{i,j}^n}{\Delta t} = \alpha\left(\hat{\delta}_x^2 u_{i,j}^n + \hat{\delta}_y^2 u_{i,j}^n\right) \tag{4.109}$$

For the second sweep, $u_{i,j}^{n+1}$ is computed at each grid point (for which $i + j + n$ is odd) by the simple implicit scheme

$$\frac{u_{i,j}^{n+1} - u_{i,j}^n}{\Delta t} = \alpha\left(\hat{\delta}_x^2 u_{i,j}^{n+1} + \hat{\delta}_y^2 u_{i,j}^{n+1}\right) \tag{4.110}$$

The second sweep appears to be implicit, but no simultaneous algebraic equations must be solved because $u_{i+1,j}^{n+1}$, $u_{i-1,j}^{n+1}$, $u_{i,j+1}^{n+1}$, and $u_{i,j-1}^{n+1}$ are known from the first sweep; hence the algorithm is explicit. The T.E. for the hopscotch method is of $O[\Delta t, (\Delta x)^2, (\Delta y)^2]$.

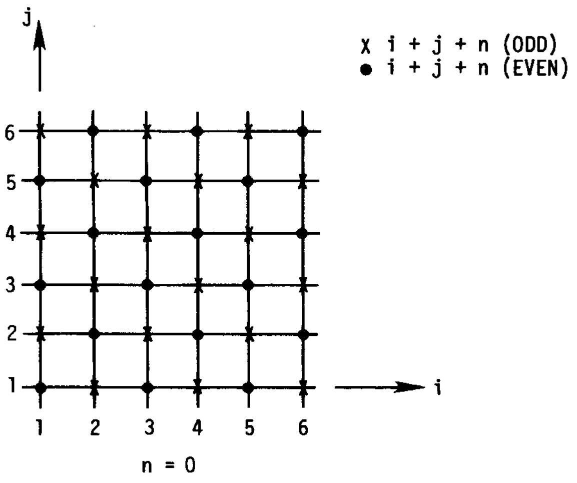

Figure 4.20 Hopscotch calculation procedure.

4.2.14 Additional Comments

The selection of a best method for solving the heat equation is made difficult by the large variety of acceptable methods. In general, implicit methods are considered more suitable than explicit methods. For the 1-D heat equation, the Crank-Nicolson method is highly recommended because of its second-order temporal and spatial accuracy. For the 2-D and 3-D heat equations, both the ADI schemes of Douglas and Gunn and the modified Keller box method give excellent results.

4.3 LAPLACE'S EQUATION

Laplace's equation is the model form for elliptic PDEs. For 2-D problems in Cartesian coordinates, Laplace's equation is

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0 \tag{4.111}$$

Some of the important practical problems governed by a single elliptic equation include the steady-state temperature distribution in a solid and the incompressible irrotational ("potential") flow of a fluid.

The incompressible Navier-Stokes equations are an example of a more complicated system of equations that has an elliptic character. The steady incompressible Navier-Stokes equations are elliptic but in a coupled and complicated fashion, since the pressure derivatives as well as velocity derivatives are sources of elliptic behavior. The elliptic equation arising in many physical problems is a Poisson equation of the form

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f(x, y) \tag{4.112}$$

Thus elliptic PDEs will be found to frequently govern important problems in heat transfer and fluid mechanics. For this reason, we will give serious attention to ways of solving a model elliptic equation.

4.3.1 Finite-Difference Representations for Laplace's Equation

The differences between "methods" for Laplace's equation and elliptic equations in general are not so much differences in the finite-difference representations, (although these will vary) but more often, differences in the techniques used for solving the resulting system of linear algebraic equations.

Five-point formula. By far the most common difference scheme for the 2-D Laplace equation is the five-point formula first used by Runge in 1908:

$$\frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{(\Delta x)^2} + \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{(\Delta y)^2} = 0 \tag{4.113}$$

which has a T.E. of $O[(\Delta x)^2, (\Delta y)^2]$. The modified equation is

$$u_{xx} + u_{yy} = -\tfrac{1}{12}\left[u_{xxxx}(\Delta x)^2 + u_{yyyy}(\Delta y)^2\right] + \cdots$$

Nine-point formula. The nine-point formula appears to be a logical choice when greater accuracy is desired for Laplace's equation in the Cartesian coordinate system. Letting $\Delta x = h$ and $\Delta y = k$, the formula becomes

$$u_{i+1,j+1} + u_{i-1,j+1} + u_{i+1,j-1} + u_{i-1,j-1} - 2\frac{h^2 - 5k^2}{h^2 + k^2}(u_{i+1,j} + u_{i-1,j})$$
$$+ 2\frac{5h^2 - k^2}{h^2 + k^2}(u_{i,j+1} + u_{i,j-1}) - 20u_{i,j} = 0 \tag{4.114}$$

The T.E. for this scheme is $O(h^2, k^2)$ but becomes $O(h^6)$ on a square mesh. Details of the T.E. and modified equation for this scheme are left as an exercise. Although the nine-point formula appears to be very attractive for Laplace's equation because of the favorable T.E., this error may be only $O(h^2, k^2)$ when applied to a more general elliptic equation (including the Poisson equation) containing other terms. More details on the nine-point scheme can be found in the work by Lapidus and Pinder (1982).

Other finite-difference schemes for Laplace's equation can be found in the literature (see, for example, Thom and Apelt, 1961), but none seems to offer significant advantages over the five- and nine-point schemes given here. To obtain smaller formal T.E. in these schemes, more grid points must be used in the difference molecules. High accuracy is difficult to maintain near boundaries with such schemes.

Residual form of the difference equations. In some solution schemes it is advantageous to solve the difference equations in delta or residual form. We will illustrate the residual form by way of an example based on the five-point stencil given in Eq. (4.113). We let L be a difference operator giving the five-point difference representation. Thus, $Lu_{i,j} = 0$ is equivalent to Eq. (4.113). A *delta* (change in the variable) is defined by $u_{i,j} = \tilde{u}_{i,j} + \Delta u_{i,j}$, where $\tilde{u}_{i,j}$ represents a provisional solution such as might occur at some point in an iterative process before convergence, and $u_{i,j}$ represents the exact numerical solution to the difference equation. (Readers should note that all deltas that arise in computational fluid dynamics are not defined in the same way. The deltas may have slightly different meanings, depending upon the algorithm or application. Deltas denote a change in something, but be alert to exactly how the delta is defined.) We can substitute $\tilde{u}_{i,j} + \Delta u_{i,j}$ for $u_{i,j}$ in the difference equation $Lu_{i,j} = 0$ and obtain

$$Lu_{i,j} = L\tilde{u}_{i,j} + L\,\Delta u_{i,j} = 0 \tag{4.115}$$

The *residual* is defined as the number that results when the difference equation, written in a form giving zero on the right-hand side, is evaluated for an intermediate or provisional solution. For Laplace's equation the residual can be evaluated as $R_{i,j} = L\tilde{u}_{i,j}$. If the provisional solution satisfies the difference equation exactly, the residual vanishes. With this definition, Eq. (4.115) can be written as

$$L\,\Delta u_{i,j} = -R_{i,j} \tag{4.116}$$

Equation (4.116) is an alternate and equivalent form of the difference equation for Laplace's equation. Starting with any provisional solution or, in fact, a simple guess, allows the residual to be computed at each grid point. From there, Eq. (4.116) can be solved for the deltas that are then added to the provisional solution. In some iterative schemes, the residuals are updated at each iteration, and new deltas are computed. To update the solution for u, the delta values are added to the provisional solution used to compute the residual. An example below in connection with the multigrid method will help clarify these ideas.

4.3.2 Simple Example for Laplace's Equation

Consider how we might determine a function satisfying

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

on the square domain

$$0 \leqslant x \leqslant 1 \qquad 0 \leqslant y \leqslant 1$$

subject to Dirichlet boundary conditions. Series solutions can be obtained for this problem (most readily by separation of variables) satisfying certain distributions of u at the boundaries. These are available in most textbooks that cover conduction heat transfer (Chapman, 1974) and can be used as test cases to verify the finite-difference formulation. In this example, we will use the five-point scheme, Eq. (4.113), and let $\Delta x = \Delta y = 0.1$, resulting in a uniform 11×11 grid over the square problem domain (see Fig. 4.21). With $\Delta x = \Delta y$, the difference equation can be written as

$$u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{i,j} = 0 \tag{4.117}$$

for each point where u is unknown. In this example problem with Dirichlet boundary conditions, we have 81 grid points where u is unknown. For each one of those points, we can write the difference equation so that our problem is one of solving the system of 81 simultaneous linear algebraic equations for the 81 unknown $u_{i,j}$. Mathematically, our problem can be expressed as

$$\begin{array}{llll} a_{11}u_1 + a_{12}u_2 + \cdots\cdots\cdots & a_{1n}u_n & = c_1 \\ a_{21}u_1 + a_{22}u_2 + \cdots\cdots\cdots & a_{2n}u_n & = c_2 \\ \vdots & \vdots & \vdots \\ a_{n1}u_1 + \cdots\cdots\cdots\cdots\cdots\cdots & a_{nn}u_n & = c_n \end{array} \tag{4.118}$$

or more compactly as $[A]\mathbf{u} = \mathbf{C}$, where $[A]$ is the matrix of known coefficients, $\mathbf{u}$ is the column vector of unknowns, and $\mathbf{C}$ is a column vector of known quantities. It is worth noting that the matrix of coefficients will be very sparse, since about 76 of the 81 a's in each row will be zero. To make our example algebraically as simple as possible, we have let $\Delta x = \Delta y$. If $\Delta x \neq \Delta y$, the

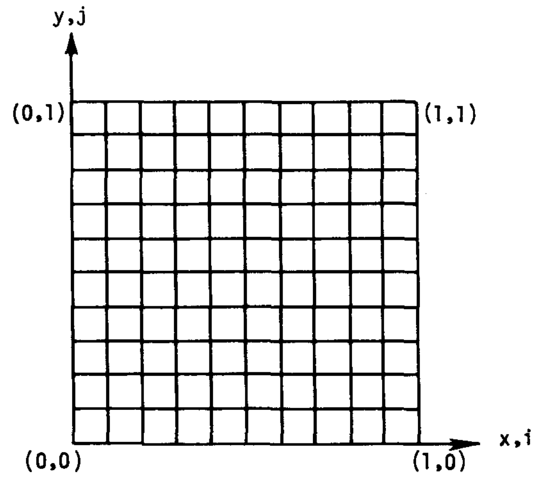

Figure 4.21 Finite-difference grid for Laplace's equation.

coefficients will be a little more involved, but the algebraic equations will still be linear and can be represented by the general $[A]\mathbf{u} = \mathbf{C}$ system given above.

Methods for solving systems of linear algebraic equations can be readily classified as either direct or iterative. Direct methods are those that give the solution (exactly if round-off error does not exist) in a finite and predeterminable number of operations using an algorithm that is often quite complicated. Iterative methods consist of a repeated application of an algorithm that is usually quite simple. They yield the exact answer only as a limit of a sequence, but if the iterative procedure converges, we can come within ϵ of the answer in a finite but usually not predeterminable number of operations. Some examples of both types of methods will be given.

4.3.3 Direct Methods for Solving Systems of Linear Algebraic Equations

Cramer's rule. Cramer's rule is one of the most elementary methods. All students have certainly heard of it, and most are familiar with the workings of the procedure. Unfortunately, the algorithm is immensely time consuming, the number of operations being approximately proportional to $(n + 1)!$, where n is the number of unknowns. A number of horror stories have been told about the large computation time required to solve systems of equations by Cramer's rule. The number of operations (multiplications and divisions) required to solve a system of algebraic equations by Gauss elimination (described below) is approximately $n^3/3$. The example problem discussed above for an 11×11 grid involved 81 unknowns. The operation count for solving this problem by Cramer's rule is 82!, which is a very large number indeed (4.75×10^{122}), dwarfing even the national debt. By comparison, solving the problem by Gauss elimination would require only about 177,147 operations. If we applied Cramer's rule to the example problem with 81 unknowns using a machine capable of performing 100 million floating point operations per second (100 megaflops), the calculation would require about 3.20×10^{101} years. This is not worth waiting for! Using Gauss elimination would only require a fraction of a second on the same machine. Cramer's rule should never be used for more than about three unknowns, since it rapidly becomes very inefficient as the number of unknowns increases.

Gaussian elimination. Gaussian elimination is a very useful and efficient tool for solving many systems of algebraic equations, particularly for the special case of a tridiagonal system of equations. However, the method is not as fast as some others to be considered for more general systems of algebraic equations that arise in solving PDEs. Approximately $n^3/3$ multiplications and divisions are required in solving n equations. Also, round-off errors, which can accumulate through the many algebraic operations, sometimes cause deterioration of accuracy when n is large. Actually, the accuracy of the method depends on the specific system of equations, and the matter is too complex to resolve by a simple general statement. Rearranging the equations to the extent possible in order to put the coefficients that are largest in magnitude on the main diagonal (known as "pivoting") will tend to improve accuracy. However, since we will want to use an elimination scheme for tridiagonal systems of equations that arise in implicit difference schemes for marching problems, it would be well to gain some notion of how the basic Gaussian elimination procedure works.

Consider the equations

$$\begin{aligned} a_{11}u_1 + a_{12}u_2 + \cdots\cdots\cdots &= c_1 \\ a_{21}u_1 + a_{22}u_2 + \cdots\cdots\cdots &= c_2 \\ \vdots \qquad\qquad\qquad & \quad \vdots \\ a_{n1}u_n + \cdots\cdots\cdots\cdots\cdots\cdots &= c_n \end{aligned} \tag{4.119}$$

The objective is to transform the system into an upper triangular array by eliminating some of the unknowns from some of the equations by algebraic operations. To illustrate, we choose the first equation (row) as the "pivot" equation and use it to eliminate the u_1 term from each equation below it. This is done by multiplying the first equation by a_{21}/a_{11}[†] and subtracting it from the second equation to eliminate u_1 from the second equation. Multiplying the pivot equation by a_{31}/a_{11} and subtracting it from the third equation eliminates the first term from the third equation. This procedure can be continued to eliminate the u_1 from equations 2 through n. The system now appears in Fig. 4.22.

Next, the second equation (as altered by the above procedure) is used as the pivot equation to eliminate u_2 from all equations below it, leaving the system in the form shown in Fig. 4.23. The third equation in the altered system is then used as the next pivot equation and the process continues until only an upper triangular form remains:

$$\begin{aligned} a_{11}u_1 + a_{12}u_2 + \cdots\cdots\cdots\cdots &= c_1 \\ a'_{22}u_2 + a'_{23}u_3 + \cdots\cdots\cdots &= c'_2 \\ a'_{33}u_3 + \cdots\cdots\cdots &= c'_3 \\ &\;\vdots \\ a'_{nn}u_n &= c'_n \end{aligned} \tag{4.120}$$

[†] We must always interchange rows if necessary to avoid division by zero.

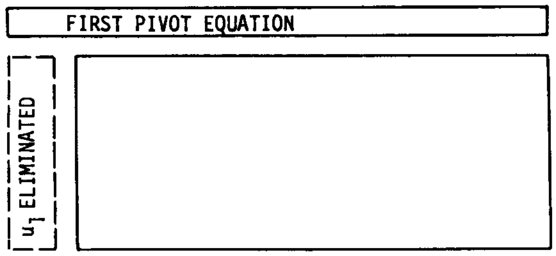

Figure 4.22 Gaussian elimination, u_1 eliminated below main diagonal.

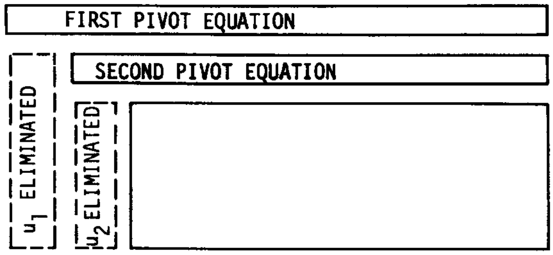

Figure 4.23 Gaussian elimination, u_1 and u_2 eliminated below main diagonal.

At this point, only one unknown appears in the last equation, two in the next to last equation, etc., so a solution can be obtained by back substitution.

Consider the following system of three equations as a specific numerical example:

$$\begin{aligned} U_1 + 4U_2 + U_3 &= 7 \\ U_1 + 6U_2 - U_3 &= 13 \\ 2U_1 - U_2 + 2U_3 &= 5 \end{aligned}$$

Using the top equation as a pivot, we can eliminate U_1 from the lower two equations:

$$\begin{aligned} U_1 + 4U_2 + U_3 &= 7 \\ 2U_2 - 2U_3 &= 6 \\ -9U_2 + 0 &= -9 \end{aligned}$$

Now using the second equation as a pivot, we obtain the upper triangular form:

$$\begin{aligned} U_1 + 4U_2 + U_3 &= 7 \\ 2U_2 - 2U_3 &= 6 \\ -9U_3 &= 18 \end{aligned}$$

Back substitution yields $U_3 = -2$, $U_2 = 1$, $U_1 = 5$.

Block-iterative methods for Laplace's equation (Section 4.3.4) lead to systems of simultaneous algebraic equations which have a tridiagonal matrix of coefficients. This was also observed in Sections 4.1 and 4.2 for implicit formulations of PDEs for marching problems. To illustrate how Gaussian elimination can be efficiently modified to take advantage of the tridiagonal form of the coefficient matrix, we will consider the simple implicit scheme for the heat equation as an example:

$$\frac{\partial u}{\partial t} = \alpha \frac{\partial^2 u}{\partial x^2}$$

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

In terms of the format used before for algebraic equations, this can be rewritten as

$$b_j u_{j-1}^{n+1} + d_j u_j^{n+1} + a_j u_{j+1}^{n+1} = c_j$$

where

$$a_j = b_j = -\frac{\alpha\,\Delta t}{(\Delta x)^2} \qquad c_j = u_j^n \qquad d_j = 1 + \frac{2\alpha\,\Delta t}{(\Delta x)^2}$$

For Dirichlet boundary conditions, u_{j-1}^{n+1} is known at one boundary and u_{j+1}^{n+1} at the other. All known u are collected into the c_j term, so our system looks like

$$\begin{bmatrix} d_1 & a_1 & 0 & \cdot & \cdot & \cdot & \cdot & 0 \\ b_2 & d_2 & a_2 & 0 & \cdot & \cdot & \cdot & \cdot \\ 0 & b_3 & d_3 & a_3 & 0 & \cdot & \cdot & \cdot \\ 0 & 0 & b_4 & d_4 & a_4 & 0 & \cdot & \cdot \\ \cdot & \cdot & \cdot & \ddots & \ddots & \ddots & \cdot & \cdot \\ \cdot & \cdot & \cdot & & \ddots & \ddots & \ddots & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \ddots & \ddots & a_{NJ-1} \\ 0 & \cdot & \cdot & \cdot & \cdot & 0 & b_{NJ} & d_{NJ} \end{bmatrix} \begin{bmatrix} u_1^{n+1} \\ u_2^{n+1} \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ u_{NJ}^{n+1} \end{bmatrix} = \begin{bmatrix} c_1 \\ c_2 \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ c_{NJ} \end{bmatrix}$$

Even when other boundary conditions apply, the system can be cast into the above form, although the first and last equations in the array may result from auxiliary relationships related to the boundary conditions and not to the original difference equation, which applies to nonboundary points.

For this tridiagonal system it is easy to modify the Gaussian elimination procedure to take advantage of the zeros in the matrix of coefficients. This modified procedure, suggested by Thomas (1949), is discussed briefly in Section 4.1.4.

Thomas algorithm. Referring to the tridiagonal matrix of coefficients above, the system is put into an upper triangular form by computing the new d_j by

$$d_j = d_j - \frac{b_j}{d_{j-1}} a_{j-1} \qquad j = 2, 3, \ldots, NJ$$

and the new c_j by

$$c_j = c_j - \frac{b_j}{d_{j-1}} c_{j-1} \qquad j = 2, 3, \ldots, NJ$$

then computing the unknowns from back substitution according to $u_{NJ} = c_{NJ}/d_{NJ}$, and then

$$u_k = \frac{c_k - a_k u_{k+1}}{d_k} \qquad k = NJ - 1, NJ - 2, \ldots, 1$$

In the above equations, the equals sign means "is replaced by," as in the FORTRAN programming language. A FORTRAN program for this procedure is given in Appendix A.

Some flexibility exists in the way in which boundary conditions are handled when the Thomas algorithm is used to solve for the unknowns. It is best that the reader develop an appreciation for these details through experience; however, a comment or two will be offered here by way of illustration. The main purpose of the elimination scheme is to determine the unknowns; therefore, for Dirichlet boundary conditions, the u's at the boundary need not be included in the list of unknowns. That is, u_1 in the elimination algorithm could correspond to the u at the first nonboundary point, and u_{NJ} to the u at the last nonboundary point. However, no harm is done, and programming may be made easier, by specializing a_1, d_1, b_{NJ}, and d_{NJ} to provide a redundant statement of the boundary conditions. That is, if we let $d_1 = 1$, $a_1 = 0$, $d_{NJ} = 1$, $b_{NJ} = 0$, $c_1 = u_1^{n+1}$ (given), and $c_{NJ} = d_{NJ}^{n+1}$ (given), the first and last algebraic equations become just a statement of the boundary conditions. As an example of how other boundary conditions fall easily into the tridiagonal format, consider convective (mixed) boundary conditions for the heat equation:

$$h(u_\infty - u_{\text{bdy}}) = -k\frac{\partial u}{\partial x}\bigg)_{\text{bdy}}$$

A control-volume analysis at the boundary where $j = 1$ leads to a difference equation that can be written as

$$d_1 u_1^{n+1} + a_1 u_2^{n+1} = c_1 \ .$$

where

$$d_1 = 1 + \frac{2\alpha\,\Delta t}{(\Delta x)^2}\left(1 + \frac{h\,\Delta x}{k}\right)$$

$$a_1 = \frac{-2\alpha\,\Delta t}{(\Delta x)^2} \qquad c_1 = \frac{2\alpha(\Delta t)h(\Delta x)}{(\Delta x)^2 k}u_\infty + u_1^n$$

which obviously fits the tridiagonal form for the first row.

Advanced-direct methods. Direct methods for solving systems of algebraic equations that are faster than Gaussian elimination certainly exist. Unfortunately, none of these methods is completely general. That is, they are applicable only to the algebraic equations arising from a special class of difference equations and associated boundary conditions. Many of these methods are "field size limited" (limited in applicability to relatively small systems of algebraic equations) owing to the accumulation of round-off errors. As a class, the algorithms for fast direct procedures tend to be rather complicated and not easily adapted to irregular problem domains or complex boundary conditions. Somewhat more computer storage is usually required than for an iterative method suitable for a given problem. In addition, the best iterative methods developed in recent years (mainly multigrid methods) are actually faster than the direct methods. It seems that the simplest of the direct methods suffer from the field size limitations and are relatively restricted in their range of application, and those that are not field size limited have algorithms with very involved details that are beyond the scope of this text. Consequently, only a few of these methods will be mentioned here, and none will be discussed in detail.

One of the simplest of the advanced, direct methods is the error vector propagation (EVP) method developed for the Poisson equation by Roache (1972). This method is field size limited; however, the concepts are straightforward. Two fast direct methods for the Poisson equation that are not limited, owing to accumulation of round-off errors, are the "odd-even reduction" method of Buneman (1969) and the fast Fourier transform method of Hockney (1965, 1970). Swartztrauber (1977) discusses optimal combinations of the two schemes. More recent developments, including implementation details for supercomputers, are discussed by Hockney and Jesshope (1981). Clearly, the fast direct methods should be considered for problems where the geometry and boundary conditions will permit their use and when computer execution time is an overriding consideration. These methods can be 10–30 times faster than the simpler iterative methods but are not expected to be faster than the multigrid iterative methods, which will be introduced below.

Another type of advanced direct method is known as a *sparse matrix method*, the most well known of which is the Yale sparse matrix package (Eisenstadt et al., 1977). Unlike the "fast" solvers discussed above, the sparse matrix methods can be quite general. The methods are essentially "smart" elimination methods that take advantage of the sparseness of the coefficient matrix. The methods generally require extensive computer memory and are not more efficient than the better iterative methods. Examples of the use of such methods to solve the Navier-Stokes equations can be found in the works of Bender and Khosla (1988) and Venkatakrishnan and Barth (1989).

4.3.4 Iterative Methods for Solving Systems of Linear Algebraic Equations

Iterative methods are also known as "relaxation" methods, a term derived from the residual relaxation method introduced by Southwell many years ago. This class of methods can be further broken down into point- (or explicit) iterative methods and block- (or implicit) iterative methods. In brief, for point-iterative methods, the same simple algorithm is applied to each point where the unknown function is to be determined in successive iterative sweeps, whereas in block-iterative methods, subgroups of points are singled out for solution by elimination (direct) schemes in an overall iterative procedure.

Gauss-Seidel iteration. Although many different iterative methods have been suggested over the years, Gauss-Seidel iteration (often called Liebmann iteration when applied to the algebraic equation resulting from the differencing of an elliptic PDE) is one of the most efficient and useful point-iterative procedures for large systems of equations. The method is extremely simple but only converges under certain conditions related to "diagonal dominance" of the matrix of coefficients. Fortunately, the differencing of many steady-state conservation statements provides this diagonal dominance. The method makes explicit use of the sparseness of the matrix of coefficients.

The simplicity of the procedure will be demonstrated by an example prior to a concise statement regarding the sufficient condition for convergence. When the method can be used, the procedure for a general system of algebraic equations would be to (1) make initial guesses for all unknowns (a guessed value for one unknown will not be needed, as seen in example below); (2) solve each equation for the unknown whose coefficient is largest in magnitude, using guessed values initially and the most recently computed values thereafter for the other unknowns in each equation; (3) repeat iteratively the solution of the equations in this manner until changes in the unknowns become "small," remembering to use the most recently computed value for each unknown when it appears on the right-hand side of an equation. As an example, consider the system

$$\begin{aligned} 4x_1 - x_2 + x_3 &= 4 \\ x_1 + 6x_2 + x_3 &= 9 \\ -x_1 + 2x_2 + 5x_3 &= 2 \end{aligned}$$

We would first rewrite the equations as

$$\begin{aligned} x_1 &= \tfrac{1}{4}(4 + x_2 - x_3) \\ x_2 &= \tfrac{1}{6}(9 - x_1 - x_3) \\ x_3 &= \tfrac{1}{5}(2 + x_1 - 2x_2) \end{aligned}$$

then make initial guesses for x_2 and x_3 (a guess for x_1 is not needed) and compute x_1, x_2, x_3 iteratively as indicated above.

Referring to the five-point stencil for Laplace's equation, we observe that the unknown having the coefficient largest in magnitude is $u_{i,j}$. Letting $\beta = \Delta x/\Delta y$, the grid aspect ratio, the general equation for the Gauss-Seidel procedure for Laplace's equation can be written as

$$u_{i,j}^{k+1} = \frac{u_{i+1,j}^{k} + u_{i-1,j}^{k+1} + \beta^2\left(u_{i,j+1}^{k} + u_{i,j-1}^{k+1}\right)}{2(1+\beta^2)} \tag{4.121}$$

where k denotes the iterative level, i denotes the column, and j the row. In Eq. (4.121), the sweep direction is assumed to be from low values of i and j to large values. Thus, at least two unknowns in each equation would already have been calculated at the $k + 1$ level. In terms of a general system of equations,

$[A]\mathbf{x} = \mathbf{b}$, the Gauss-Seidel scheme can be written as

$$x_i^{k+1} = \frac{1}{a_{ii}}\left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{k+1} - \sum_{j=i+1}^{n} a_{ij}x_i^k\right) \tag{4.122}$$

where it is understood that the system of equations has been reordered, if necessary, so that the coefficients largest in magnitude are on the main diagonal.

In passing, we shall mention that an iterative process can also be performed without continuously updating values on the right-hand side. If the unknowns on the right-hand side are updated only after each iterative sweep through the entire field, the process is known as *Jacobi iteration*. For model problems, Jacobi iteration requires approximately twice as many iterations for convergence as Gauss-Seidel iteration.

Sufficient condition for convergence of the Gauss-Seidel procedure. In order to provide a compact notation, as above, we will order the equations, if possible, so that the coefficient largest in magnitude in each row is on the main diagonal. Then if the system is irreducible (cannot be arranged so that some of the unknowns can be determined by solving less than n equations) and if

$$|a_{ii}| \geqslant \sum_{\substack{j=1\\ j\neq i}}^{n} |a_{ij}| \tag{4.123}$$

for all i and if

$$|a_{ii}| > \sum_{\substack{j=1\\ j\neq i}}^{n} |a_{ij}| \tag{4.124}$$

for at least one i, then the Gauss-Seidel iteration will converge. This is a *sufficient condition*, which means that convergence may sometimes be observed when the above condition is not met. A necessary condition can be stated, but it is impractical to evaluate. Stated in words, the sufficient condition can be interpreted as requiring for each equation that the magnitude of the coefficient on the diagonal be greater than or equal to the sum of the magnitudes of the other coefficients in the equation, with the "greater than" holding for at least one (usually corresponding to a point near a boundary for a physical problem) equation.

Perhaps we should now relate the above iterative convergence criteria to the system of equations that results from differencing Laplace's equation according to Eq. (4.113). First we observe that the coefficient largest in magnitude belongs to $u_{i,j}$. Since we apply Eq. (4.113) to each point where $u_{i,j}$ is unknown, we could clearly arrange all the equations in the system so that the coefficient largest in magnitude appeared on the diagonal. With the exercise of proper care in establishing difference representations, this type of diagonal dominance can normally be achieved for all elliptic equations. In terms of a linear difference equation for u, we would expect the Gauss-Seidel iterative procedure to converge if the finite-difference equation applicable to each point i, j, where $u_{i,j}$ is unknown, is such that the magnitudes of the coefficient of $u_{i,j}$ is greater than or equal to the sum of the magnitudes of the coefficients of the other unknowns in the equation. The "greater than" must hold for at least one equation.

We will not offer a proof for this sufficient condition for the convergence of the Gauss-Seidel iteration, but hopefully, a simple example will suggest why it is true. If we look back to our simple three-equation example for Gauss-Seidel iteration and consider that at any point our intermediate values of x are the exact solution plus some ϵ, i.e., $x_1 = (x_1)_{\text{exact}} + \epsilon_1$, then our condition of diagonal dominance is forcing the ϵ to become smaller and smaller as the iteration is repeated cyclically. For one run through the iteration, we could observe

$$|\epsilon_1^2| \leqslant \tfrac{1}{4}|\epsilon_2^1| + \tfrac{1}{4}|\epsilon_3^1|$$

$$|\epsilon_2^2| \leqslant \tfrac{1}{6}|\epsilon_1^2| + \tfrac{1}{6}|\epsilon_3^1|$$

$$|\epsilon_3^2| \leqslant \tfrac{1}{5}|\epsilon_1^2| + \tfrac{2}{5}|\epsilon_2^2|$$

If ϵ_2^1 and ϵ_3^1 were initially each 10, $|\epsilon_1^2|$ would be $\leqslant 5$ and $|\epsilon_1^3| \leqslant 1.446$. Here, superscripts denote iterative level.

Finally, we note for a general system of equations, the multiplications per iteration could be as great as n^2 but could be much less if the matrix was sparse.

Successive overrelaxation. Successive overrelaxation (SOR) is a technique that can be used in an attempt to accelerate any iterative procedure, but we will propose it here primarily as a refinement to the Gauss-Seidel method. As to the origins of the method, one story (probably inaccurate) being passed around is that the method was suggested by a duck hunter, who finally learned that if he pointed his gun ahead of the duck, he would score more hits than if he pointed the gun right at the duck. The duck is a moving target, and if we anticipate its motion, we are more likely to hit it with the shot pattern. The duck hunter told his story to his neighbor, who was a numerical analyst, and SOR was born—or so the story goes.

As we apply Gauss-Seidel iteration to a system of simultaneous algebraic equations, we expect to make several recalculations or iterations before convergence to an acceptable level is achieved. Suppose that during this process we observe the change in the value of the unknown at a point between two successive iterations, note the direction of change, and anticipate that the same trend will continue on to the next iteration. Why not go ahead and make a correction to the variable in the anticipated direction *before* the next iteration, thereby, hopefully, accelerating the convergence? An arbitrary correction to the intermediate values of the unknowns from *any* iterative procedure (Gauss-Seidel iteration is of most interest to us at this point, so we will use it as the representative iterative scheme), according to the form

$$u_{i,j}^{k+1'} = u_{i,j}^{k'} + \omega\left(u_{i,j}^{k+1} - u_{i,j}^{k'}\right) \tag{4.125}$$

is known as overrelaxation or successive overrelaxation. Here, k denotes iteration level, $u_{i,j}^{k+1}$ is the most recent value of $u_{i,j}$ calculated from the Gauss-Seidel procedure, $u_{i,j}^{k'}$ is the value from the previous iteration as adjusted by previous application of this formula if the overrelaxation is being applied successively (at each iteration), and $u_{i,j}^{k+1'}$ is the newly adjusted or "better guess" for $u_{i,j}$ at the $k+1$ iteration level. That is, we expect $u_{i,j}^{k+1'}$ to be closer to the final solution than the unaltered value $u_{i,j}^{k+1}$ from the Gauss-Seidel calculation. The formula is applied immediately at each point after $u_{i,j}^{k+1}$ has been obtained, and $u_{i,j}^{k+1'}$ replaces $u_{i,j}^{k+1}$ in all subsequent calculations in the cycle. Here, ω is the relaxation parameter, and when $1 < \omega < 2$, *overrelaxation* is being employed. Overrelaxation can be likened to linear extrapolation based on values $u_{i,j}^{k'}$ and $u_{i,j}^{k+1}$. In some problems, *underrelaxation* $0 < \omega < 1$ is employed. Underrelaxation appears to be most appropriate when the convergence at a point is taking on an oscillatory pattern and tending to "overshoot" the apparent final solution. For underrelaxation the adjusted value, $u_{i,j}^{k+1'}$ is between $u_{i,j}^{k'}$ and $u_{i,j}^{k+1}$. Overrelaxation is usually appropriate for numerical solutions to Laplace's equation with Dirichlet boundary conditions. Underrelaxation is sometimes called for in elliptic problems, it seems, when the equations are nonlinear. Occasionally, for nonlinear problems, underrelaxation is even observed to be necessary for convergence.

We note that the relaxation parameter should be restricted to the range $0 < \omega < 2$. For convergence, we require that the magnitude of the changes in u from one iteration to the next become smaller. Use of $\omega \geqslant 2$ forces these changes to remain the same or to increase, in contradiction to convergent behavior.

Two important remaining questions are, how can we properly determine a good or even the best value for ω, and by how much does this procedure accelerate the convergence? No completely general answers to these questions are available, but some guidelines can be drawn.

For Laplace's equation on a rectangular domain with Dirichlet boundary conditions, theories pioneered by Young (1954) and Frankel (1950) lead to an expression for the optimum ω (hereafter denoted by ω_{opt}). First, defining σ as

$$\sigma = \frac{1}{1+\beta^2}\left(\cos\frac{\pi}{p} + \beta^2\cos\frac{\pi}{q}\right) \tag{4.126}$$

the optimum ω is given by

$$\omega_{\text{opt}} = \frac{2}{1+(1-\sigma^2)^{1/2}} \tag{4.127}$$

where β is the grid aspect ratio as defined previously, p is the number of Δx increments, and q is the number of Δy increments along the sides of the rectangular region. The formula can also be used for problems in rectangular regions with certain combinations of Dirichlet and Neumann boundary conditions that permit an equivalent Dirichlet problem to be recognized by identifying the Neumann boundaries as lines of symmetry in a Dirichlet problem. In general, however, for more complex elliptic problems it is not possible to determine ω_{opt} in advance. In these cases, some numerical experimentation should be helpful in identifying useful values for ω. Numerical examples and theory generally indicate that it is better to guess on the high side of ω_{opt} than on the low side. Hageman and Young (1981) discuss considerations in the search for ω_{opt} in some detail.

Is the ω search worthwhile? The answer is emphatically yes. In some problems it is possible to reduce the computation time by a factor of 30. This is significant! Occasionally, SOR may be found not to be of much help in accelerating convergence, but it should always be considered and evaluated. The potential for savings in computation time is too great to ignore.

Since overrelaxation can be viewed as applying a correction to the values obtained from the Gauss-Seidel procedure based on extrapolation from previous iterates, it is natural to wonder if other, perhaps more accurate (in terms of T.E. of the extrapolation formula) extrapolation schemes can be used to accelerate the convergence of iterative procedures. In fact, other schemes such as Aitken and Richardson extrapolation have been used in this application. The details of these extrapolation schemes are covered in standard texts on numerical analysis, but as is perhaps expected, any advantage in accelerating the convergence of the iterative process by using more complex extrapolation schemes has to be weighed against any added computation costs due to requirements of additional storage or algebraic operations. SOR has simplicity in its favor, and it can be programmed so that no additional arrays need to be stored.

In the SOR scheme the calculations normally proceed in a systematic way with sweeps from the lower left-hand corner of the domain to the upper right-hand side (in two dimensions from low values of i and j to high values of i and j). This scheme has a bias in terms of sweep direction that may permit the largest errors to accumulate at the high values of i and j. A modification to SOR known as symmetric successive overrelaxation (SSOR) attempts to improve upon this condition. In SSOR, one alternates the sweep direction. A pass from low values of i and j to high values of i and j is followed by a sweep from high i and j to low i and j.

Coloring schemes. Supercomputers in common use today employ *vector processing* to obtain greater execution speeds. The *vectorization* occurs in FORTRAN DO loops (only in the innermost loop if the loops are nested) and can be thought of as a simultaneous execution of the statements in the DO loop for all values of the DO parameter. If the statements in the DO loop are recursive in nature, i.e., the right-hand side of the statement contains results previously computed in the loop, then the compiler rejects that loop for vectorization because an apparent error would occur if the statements were executed simultaneously. Vectorization naturally speeds up the algorithm and is a desirable feature. An example of a vectorizable FORTRAN DO loop is

```
      do 10 j = 1, nj
      a(j) = b(j)*c + d
10    continue
```

and an example of a recursive loop that cannot be performed as given in a vector manner is

```
      do 10 j = 1, nj
      a(j) = a(j - 1)*c + d
10    continue
```

The Gauss-Seidel algorithm is recursive in appearance because of the preference to use the most recent (updated) values of the unknown function on the right-hand side. Thus the algorithm is not vectorizable. Simple Jacobi iteration is vectorizable but converges more slowly than Gauss-Seidel iteration. A variation of the Gauss-Seidel procedure known as the *red-black* or *checkerboard* scheme has approximately the same convergence properties as the Gauss-Seidel procedure but is vectorizable. Imagine that the nodes are colored like a checkerboard, every other point red, alternate points black, as indicated in Fig. 4.24. The red-black scheme updates the variables in two sweeps, much as was done for the hopscotch scheme. This can be thought of as performing a Jacobi iteration on every other point. Sweep 1 updates red points (points for which $i + j$ is even in two dimensions). At this point, black points are surrounded by nodes for which the unknown has been updated (see Fig. 4.24). Sweep 2 updates the black points (points for which $i + j$ is odd). The two sweeps constitute one iteration. The DO loops proceed in strides of 2 (every other point), and a compiler directive may be needed to confirm the vectorization because the appearance of the right-hand sides may give the impression that the

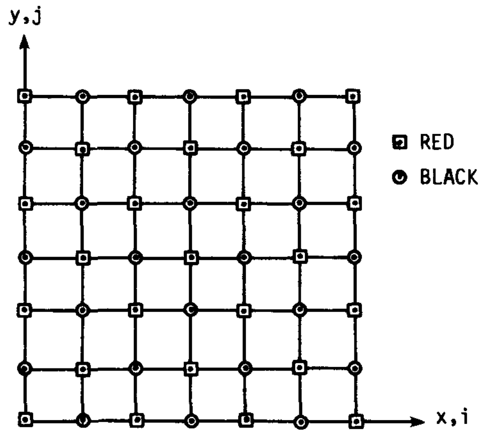

Figure 4.24 Red-black (checkerboard) ordering.

scheme is recursive. The favorable convergence properties arise because some of the updates use information that has been obtained within the same iteration (but not the same DO loop).

For Laplace's equation in two dimensions in the Cartesian coordinate system, only two subdivisions of points (red and black) are necessary to recover the convergence properties of the Gauss-Seidel scheme on a vector machine. The question might arise as to whether more than two colors (or two subdivisions of points) are ever needed. This question is most likely to arise in the use of unstructured grids, for which the cell shape and the ordering of nodes in the solution algorithm may vary greatly. Unstructured grids will be discussed in Chapters 6 and 10. If we restrict our concern to the nearest neighbors and wish to color the domain so that adjacent nodes (or cells) are of a different color, a famous theorem from graph theory states that four colors are sufficient to do this in two dimensions. This fact has been generally accepted for more than 100 years but was only proven by Appel and Haken in 1976. Four colors mean that four DO loops would be used to "visit" every node or cell. Clearly, less than four colors is sometimes adequate, as is the case for a 2-D grid formed in the Cartesian coordinate system.

Block-iterative methods. The Gauss-Seidel iteration method with SOR stands as the best all-around method for the finite-difference solution of elliptic equations discussed in detail thus far in this chapter. The number of iterations can usually be reduced even further by use of block-iterative concepts, but the number of algebraic operations required per iterative cycle generally increases, and whether the reduction in number of required iterative cycles compensates for the extra computation time per cycle is a matter that must be studied for each problem. However, several cases can be cited where the use of block-iterative methods has resulted in a net saving of computation time, so that these procedures warrant serious attention. Ames (1977) and Lapidus and Pinder (1982) present useful discussions that compare the rates of convergence for several point- and block-iterative methods.

In block- (or group) iterative methods, subgroups of the unknowns are singled out, and their values modified simultaneously by obtaining a solution to the simultaneous algebraic equations by elimination methods. Thus the block-iterative methods have an implicit nature and are sometimes known as implicit-iterative methods. In the most common block-iterative methods the unknowns in the subgroups (to be modified simultaneously) are set up so that the matrix of coefficients will be tridiagonal in form, permitting the Thomas algorithm to be used. The simplest block procedure is SOR by lines.

SOR by lines (SLOR). Although SLOR is workable with almost any iterative algorithm, it makes the most sense within the framework of the Gauss-Seidel method with SOR. We can choose either rows or columns for grouping with equal ease. To illustrate the procedure, consider again the solution to Laplace's equation on a square domain with Dirichlet boundary conditions using the five-point scheme. If we agree to start at the bottom of the square and sweep up by rows, we could write, for the general point

$$u_{i,j}^{k+1} = \frac{u_{i+1,j}^{k+1} + u_{i-1,j}^{k+1} + \beta^2\left(u_{i,j+1}^{k} + u_{i,j-1}^{k+1}\right)}{2(1+\beta^2)} \tag{4.128}$$

If we study this equation carefully, we observe that only three unknowns are present, since $u_{i,j-1}^{k+1}$ would be known from either the lower boundary conditions, if we were applying the equation to the first row of unknowns, or from the solution already obtained at the $k+1$ level from the row below. We have chosen to evaluate $u_{i,j+1}$ at the k iteration level rather than the $k+1$ level in order to obtain just three unknowns in the equation, so that the efficient Thomas algorithm can be used. This configuration can be seen in Fig. 4.25.

The procedure is then to solve the system of $I-2$ simultaneous algebraic equations for the $I-2$ unknowns representing the values of $u_{i,j}$ at the $k+1$ iteration level. SOR can now be applied in the same manner as indicated previously before moving on to the next row. Some flexibility exists in the way SOR is applied. After the Thomas algorithm is used to solve Eq. (4.128) for each row, the newly calculated values can be simply overrelaxed, as indicated by Eq. (4.125) before the calculation is advanced to the next row.

Alternatively, the overrelaxation parameter ω can be introduced prior to solution of the simultaneous algebraic equations. This is accomplished by substituting the right-hand side of Eq. (4.128) into the right-hand side of Eq. (4.125) to replace $u_{i,j}^{k+1}$. The resulting equation

$$u_{i,j}^{k+1} = (1-\omega)u_{i,j}^{k} + \frac{\omega}{2(1+\beta^2)}\left[u_{i+1,j}^{k+1} + u_{i-1,j}^{k+1} + \beta^2\left(u_{i,j+1}^{k} + u_{i,j-1}^{k+1}\right)\right]$$

is then solved for each row by the Thomas algorithm. The overrelaxation has been accomplished as part of the row solution and not as a separate step. Since

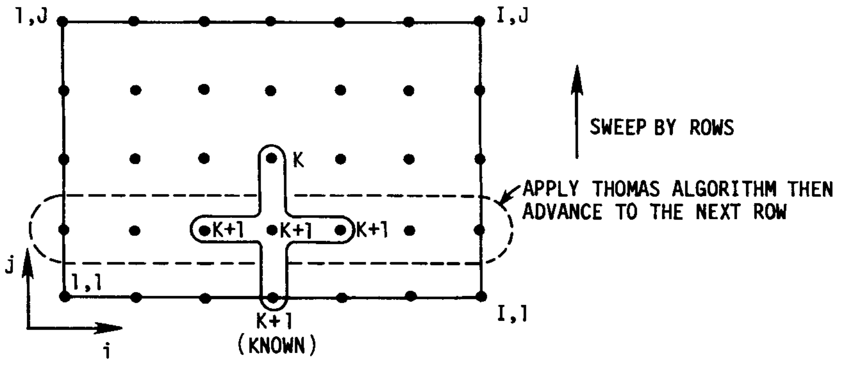

Figure 4.25 SOR by lines.

it is highly desirable to maintain diagonal dominance in the application of the Thomas algorithm, care should be taken when this latter procedure is used to ensure that $\omega \leqslant 1 + \beta^2$.

In the SLOR procedure, one iterative cycle is completed when the tridiagonal inversion has been applied to all the rows. The process is then repeated until convergence has been achieved. In applying the method to a standard example problem with Dirichlet boundary conditions, Ames (1977) indicates that only $1/\sqrt{2}$ as many iterations would be required as for a Gauss-Seidel iteration with SOR to reduce the initial errors by the same amount. On the other hand, use of the Thomas algorithm is expected to increase the computation time per iteration cycle somewhat.

The improved convergence rates observed for block-iterative methods compared with point-iterative methods might be thought of as being due to the greater influence exerted by the boundary values in each iterative pass. For example, in SOR by rows, the unknowns in each row are determined simultaneously, so it is possible for the boundary values to influence all the unknowns in the row in one iteration. This is not the case for point-iterative methods such as the Gauss-Seidel procedure, where in the first pass, at least one of the boundary points (details depend on sequence used in sweeping) only influences adjacent points.

ADI methods. The SLOR method proceeds by taking all lines in the same direction in a repetitive manner. The convergence rate can often be improved by following the sequence by rows, say, by a second sequence in the column direction. Thus a complete iteration cycle would consist of a sweep over all rows followed by a sweep over the columns. Several closely related ADI forms are observed in practice. Perhaps the simplest procedure is to first employ Eq. (4.128) to sweep by rows. We will designate the values so determined by a $k + \frac{1}{2}$ superscript. This is followed by a sweep by columns using

$$u_{i,j}^{k+1} = \frac{u_{i+1,j}^{k+1/2} + u_{i-1,j}^{k+1} + \beta^2\left(u_{i,j+1}^{k+1} + u_{i,j}^{k+1}\right)}{2(1+\beta^2)}$$

This completes one iteration, and overrelaxation can be achieved by applying Eq. (4.125) to all grid points as a second step before another iterative sweep is carried out. Alternatively, we can include the overrelaxation as part of the row and column sweeps using first for rows,

$$u_{i,j}^{k+1/2} = (1-\omega)u_{i,j}^{k} + \frac{\omega}{2(1+\beta^2)}\left[u_{i+1,j}^{k+1/2} + u_{i-1,j}^{k+1/2} + \beta^2\left(u_{i,j+1}^{k} + u_{i,j-1}^{k+1/2}\right)\right] \tag{4.129a}$$

and then for columns

$$u_{i,j}^{k+1} = (1-\omega)u_{i,j}^{k+1/2} + \frac{\omega}{2(1+\beta^2)}\left[u_{i+1,j}^{k+1/2} + u_{i-1,j}^{k+1} + \beta^2\left(u_{i,j+1}^{k+1} + u_{i,j-1}^{k+1}\right)\right] \tag{4.129b}$$

To preserve diagonal dominance in the Thomas algorithm requires $\omega \leqslant 1 + \beta^2$ in the sweep by rows and $\omega \leqslant (1 + \beta^2)/\beta^2$ in the sweep by columns.

Schemes patterned after the ADI procedures for the 2-D heat equation [Eq. (4.97)] are also very commonly used for obtaining solutions to Laplace's equation. If the boundary conditions for an unsteady problem governed by Eq. (4.97) are independent of time, the solution will asymptotically approach a steady-state distribution that satisfies Laplace's equation. Since we are only interested in the "steady-state" solution, the size of the time step can be selected with a view toward speeding convergence of the iterative process. Letting $\alpha \Delta t/2 = \rho_k$ in Eq. (4.103), we can write the Peaceman-Rachford ADI scheme for solving Laplace's equation as the two-step procedure:

Step 1: $$u_{i,j}^{k+1/2} = u_{i,j}^{k} + \rho_k\left(\hat{\delta}_x^2 u_{i,j}^{k+1/2} + \hat{\delta}_y^2 u_{i,j}^{k}\right) \tag{4.130a}$$

Step 2: $$u_{i,j}^{k+1} = u_{i,j}^{k+1/2} + \rho_k\left(\hat{\delta}_x^2 u_{i,j}^{k+1/2} + \hat{\delta}_y^2 u_{i,j}^{k+1}\right) \tag{4.130b}$$

where $\hat{\delta}_x^2$ and $\hat{\delta}_y^2$ are defined by Eq. (4.100).

Step 1 proceeds using the Thomas algorithm by rows, and step 2 completes the iteration cycle by applying the Thomas algorithm by columns. The ρ_k are known as iteration parameters, and Mitchell and Griffiths (1980) show that the Peaceman-Rachford iterative procedure for solving Laplace's equation in a square is convergent for any fixed value of ρ_k. On the other hand, for maximum computational efficiency, the iteration parameters should be varied with k, but the same ρ_k should be used in both steps of the iterative cycle. The key to using the ADI method most efficiently for elliptic problems lies in the proper choice of ρ_k. Peaceman and Rachford (1955) suggested one procedure, and another in common usage was suggested by Wachspress (1966). Although the evidence is not conclusive, some studies have suggested that the Wachspress parameters are superior to those suggested by Peaceman and Rachford. The reader is encouraged to study the literature regarding the selection of ρ_k prior to using the Peaceman-Rachford ADI method.

It is difficult to compare the computation times required by point- and block-iterative methods with SOR because of the difficulty in establishing the optimum value of the overrelaxation factor. Conclusions are also very much dependent upon the specific problem considered, the boundary conditions, and the number of grid points involved. The block-iterative methods as a class require fewer iterations than point-iterative methods, but as was mentioned earlier, more computational effort is required by each iteration. Experience suggests that SLOR will require very close to the same computation time as the Gauss-Seidel procedure with SOR for convergence to the same level for most problems. Use of an ADI procedure with SOR (fixed parameter) often provides a savings in computer time of 20-40% over that required by the Gauss-Seidel procedure with SOR. A greater savings can normally be observed if the iteration parameters are suitably varied in the ADI procedure.

Strongly implicit methods. In recent years, another type of block-iterative procedure has been gaining favor as an efficient method for solving the algebraic equations arising from the numerical solution of elliptic PDEs. To illustrate this approach, let us consider the system of algebraic equations arising from the use of the five-point difference scheme for Laplace's equations as

$$[A]\mathbf{u} = \mathbf{C}$$

where $[A]$ is the relatively sparse matrix of known coefficients, $\mathbf{u}$ is the column vector of unknowns, and $\mathbf{C}$ is a column vector of known quantities. It is well known that if the matrix $[A]$ could be factored into the product of upper and lower triangular matrices, the solution for $\mathbf{u}$ could proceed in two sweeps, involving only forward and backward substitution. To do this exactly, however, requires approximately the same effort as solving the system by Gaussian elimination. On the other hand, a number of investigators have explored the merits of obtaining an approximate (or incomplete) $[L][U]$ factorization, which requires less effort than the complete factorization, and then solving for $\mathbf{u}$ iteratively. The strongly implicit procedure (SIP) proposed by Stone (1968) is one example of this factorization strategy. The objective is to replace the sparse matrix $[A]$ by a modified form $[A + P]$ such that the modified matrix can be decomposed into upper and lower triangular sparse matrices denote by $[U]$ and $[L]$, respectively. If the $[L]$ and $[U]$ matrices are not sparse, then very little will be gained in computational efficiency over the use of Gaussian elimination. Thus the key to any computational advantage of the SIP procedure lies in the manner in which $[P]$ is selected. It is essential that the elements of $[P]$ be small in magnitude and permit the set of equations to remain more strongly implicit than for the ADI procedure. An iterative procedure is defined by writing $[A]\mathbf{u} = \mathbf{C}$ as

$$[A + P]\mathbf{u}^{n+1} = \mathbf{C} + [P]\mathbf{u}^n$$

Decomposing $[B] = [A + P]$ into the upper and lower triangular matrices $[U]$ and $[L]$ permits our system to be written as

$$[L][U]\mathbf{u}^{n+1} = \mathbf{C} + [P]\mathbf{u}^n$$

Defining an intermediate vector as $\mathbf{V}^{n+1} = [U]\mathbf{u}^{n+1}$, we form the following two-step algorithm:

Step 1: $$[L]\mathbf{V}^{n+1} = \mathbf{C} + [P]\mathbf{u}^n \tag{4.131a}$$

Step 2: $$[U]\mathbf{u}^{n+1} = \mathbf{V}^{n+1} \tag{4.131b}$$

which is repeated iteratively. Step 1 consists simply of a forward substitution. This is followed by the backward substitution indicated by step 2.

Stone (1968) selected $[P]$ so that $[L]$ and $[U]$ have only three nonzero diagonals, the principal diagonal of $[U]$ being the unity diagonal. Furthermore, the elements of $[L]$ and $[U]$ were determined such that the coefficients in the $[B]$ matrix in the locations of the nonzero entries of matrix $[A]$ were identical with those in $[A]$. Two additional nonzero diagonals appear in $[B]$. The elements of $[L]$, $[U]$, and $[P]$ can be determined from the defining equations established by forming the $[L][U]$ product. The details of this are given by Stone (1968). The procedure is implicit in both the x and y directions. Studies have indicated that, for a solution to Laplace's equations, the method requires only on the order of 50-60% of the computation time required for ADI schemes.

Schneider and Zedan (1981) proposed an alternative procedure for establishing the $[L][U]$ matrices, which is reported to reduce the computational cost for a converged solution to Laplace's equation by a factor of 2–4 over the procedures proposed by Stone (1968). They refer to their alternative procedure as the modified strongly implicit (MSI) procedure. The basic two-step iterative sequence remains the same as given in Eqs. (4.131a) and (4.131b). The improvement apparently results from extending the approach of Stone to a nine-point formulation. The MSI procedure then easily treats five-point difference representations as a special case, and the great reduction in computational cost mentioned above (a factor of 2–4) applies to use of the five-point representation. This new scheme appears to hold great promise as a very efficient and general procedure. Further details on the MSI procedure are given in Appendix C.

Despite the recursive steps in the SIP procedure, it is possible to vectorize the algorithm by structuring DO loops to move along diagonals. The scheme has also been successfully extended to solve coupled systems of equations, i.e., a "block" version of SIP has been developed (see, for example, Zedan and Schneider, 1985; Chen and Pletcher, 1991).

Other iterative methods. The quest continues for more economical and memory efficient iterative methods. Generally, methods that are economical in terms of iteration count or computer time require a relatively large amount of memory. The cost of computer memory has been decreasing rapidly, but memory can still be a limiting factor for 3-D problems involving the Navier-Stokes equations. Another iterative method that appears promising for use with large systems of equations is the generalized minimum residual (GMRES) algorithm introduced by Saad and Schultz (1986). The GMRES algorithm is closely related to the conjugate gradient (see, for example, Golub and van Loan, 1989) procedure but is applicable to problems in which the coefficient matrix may be nonsymmetric. Examples of the application of the GMRES algorithm to flow problems can be found in the works by Wigton et al. (1985), Venkatakrishnan and Mavriplis (1991), and Hixon and Sankar (1992).

4.3.5 Multigrid Method

The Gauss-Seidel method with and without SOR and the block-iterative methods just discussed provide excellent smoothing of the local error. However, because the difference stencil for Laplace's equation is relatively compact, on fine grids a very large number of iterations is often required for the influence of boundary conditions to propagate throughout the grid. Convergence often becomes painfully slow. This violates the "golden rule" of computational physics: "The amount of computational work should be proportional to the amount of real physical changes in the simulated system."

It is the removal of the low-frequency component of the error that usually slows convergence of iterative schemes on a fixed grid. However, a low-frequency component on a fine grid becomes a high-frequency component on a coarse grid. Therefore it makes good sense to use coarse grids to remove the low-frequency errors and propagate boundary information throughout the domain in combination with fine grids to improve accuracy. The strategy known as *multigrid* can do this (Brandt, 1977).

The multigrid method is one of the most efficient general iterative methods known today. The key word here is "general." More efficient schemes can be found for certain problems or certain choices of grids, but it is difficult to find a method more efficient than multigrid for the general case. The multigrid technique can be applied using any of the iterative schemes discussed in this chapter as the "smoother," although the Gauss-Seidel procedure will be used to illustrate the main points of this technique in the introductory material presented here. The objective of the multigrid technique is to accelerate the convergence of an iterative scheme.

To take full advantage of multigrid, several mesh levels are typically used. Normally, the mesh size is increased by a factor of 2 with each coarsening. For many problems the coarsening may continue until the grid consists of one internal point. It would be instructive, however, to illustrate the method first using a two-level scheme applied to Laplace's equation.

The standard Gauss-Seidel scheme will be used based on the five-point stencil. For convenience, let the operator L be defined such that $Lu_{i,j}$ becomes the standard difference representation for the left-hand side of Laplace's equation. That is,

$$Lu_{i,j} = \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{(\Delta x)^2} + \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{(\Delta y)^2} \tag{4.132}$$

The residual, $R_{i,j}$, has been defined as the number that results when the difference equation, written in a form giving zero on the right-hand side, is evaluated for an intermediate solution. Thus for the present application, $R_{i,j} = Lu_{i,j}$, where it is understood that at convergence, $R_{i,j} = 0$. Let the final converged solution of the difference equations be $u_{i,j}$ and define the corrections, $\Delta u_{i,j}$ by $u_{i,j} = \Delta u_{i,j} + u^k_{i,j}$, where the superscript k denotes iteration level. Thus the correction is the value that must be added to an intermediate solution in order to obtain the final converged solution. Since the difference equation to be solved is $Lu_{i,j} = 0$, we can write

$$L\,\Delta u_{i,j} + Lu^k_{i,j} = 0$$

but

$$Lu^k_{i,j} = R_{i,j}$$

so that

$$L\,\Delta u_{i,j} + R_{i,j} = 0 \tag{4.133}$$

This is known as the residual or delta form of the equation, as discussed earlier. This equation can be solved iteratively for the $\Delta u_{i,j}$ until convergence. If $u_{i,j}$ and $R_{i,j}$ are updated after each iteration, the delta variables vanish upon convergence. Alternatively, if $R_{i,j}$ is held fixed, the iterations will converge, yielding generally finite values for the $\Delta u_{i,j}$, which can be added to the $u_{i,j}^k$ in $R_{i,j}$ to obtain the final value for the solution.

The key idea in multigrid is to improve the fine-grid solution. We do not seek a solution to the original problem on the coarse grid. The coarse grid or grids are only used to obtain corrections to the fine-grid solution. For the present linear PDE (Laplace's equation), we can "transfer the problem" to a coarser grid by interpolating the fine-grid residual to the coarser grid and then solving Eq. (4.133) for the corrections. This form of the multigrid scheme that is applicable to linear PDEs is known as the *coarse-grid correction* scheme or the *correction storage* (CS) scheme. The residuals, of course, would be treated as known, so we would normally rearrange Eq. (4.133) for numerical solution as

$$L\,\Delta u_{i,j} = -R_{i,j} \tag{4.134}$$

and solve for the $\Delta u_{i,j}$ by the Gauss-Seidel procedure. For a two-level scheme, we would proceed through the following steps:

1. Do n iterations on the fine grid, solving Laplace's equation, $Lu_{i,j} = 0$, for $u_{i,j}$ using a "smoother" like the Gauss-Seidel scheme. The value of n would be 3 or 4 in most cases. Do not overrelax—use "pure" Gauss-Seidel. If the solution has not converged after n iterations, compute and store the residual at each fine-grid point.
2. Interpolate the residual onto the coarse grid by using a *restriction operator*. The most common way to do this on a uniform grid is by "injection," which means using the values of the residual at the fine-grid points that coincide with the coarse-grid points. That is, every second fine-grid point will be a coarse-grid point. We have $R_{i,j}$ at these points, so we use it. The chore in practice is in defining or setting up the coarse grid. Solve Eq. (4.134), the residual form of Laplace's equation, for the corrections using zero as the initial guess. [A common mistake here is to neglect to change the grid size in implementing Eq. (4.134); the corrections are being computed on the coarse grid, so the grid increments in the difference equation need to be adjusted accordingly.] The residuals are not updated during the iterations because we want the computed deltas to represent corrections to the fine-grid solution. It is common to iterate the corrections to some predetermined convergence level on the coarsest grid.
3. The corrections are interpolated onto the fine grid using a *prolongation operator*. The simplest procedure is to use bilinear interpolation. This can be

carried out as follows:

(a) Sweep through the coarse-grid rows adding values needed on the fine grid by simply averaging values of the correction existing to the right and left, i.e., simply average the neighboring values of the correction in the row.

(b) Sweep through the coarse-grid columns, adding values needed on the fine grid by averaging values above and below, much as was done for the rows in step 3(a).

(c) Examining Fig. 4.26, we note that to fill out the fine grid, we still need values at the locations marked x. We can obtain these values by sweeping either by rows or columns and filling in with averages of the neighbors above and below or to the right and the left. The result for either method will be identical because of the previous averaging.

The corrections at the fine-grid points are now added to the intermediate solution obtained from step 1. One cycle has now been completed. We would be finished except for errors associated with the interpolation (down to the coarse grid and back up to the fine grid). We now go back to step 1 and iterate n times to obtain an improved solution. If convergence is not indicated, the new residuals are interpolated to the coarse grid, and another cycle is implemented. This continues until convergence is observed on the fine grid.

Most features of the multigrid strategy can be demonstrated by use of the two-level scheme. However, significant improvements in computational economy can be expected by use of additional levels. The extension to additional levels is straightforward in principle but requires careful interpretation of the multigrid concept in order to correctly implement the procedure. Here we will discuss only the simple V cycle, in which the calculation proceeds from the finest grid down to the coarsest and then back up to the finest. Many variations in cycles are possible, and reference to more complete works on multigrid, such as Briggs (1987), Hackbusch and Trottenberg (1982), and Brandt (1977), is recommended in order to obtain more complete information on the multigrid concept. As before, the scheme will be applied to solve Laplace's equation on a square

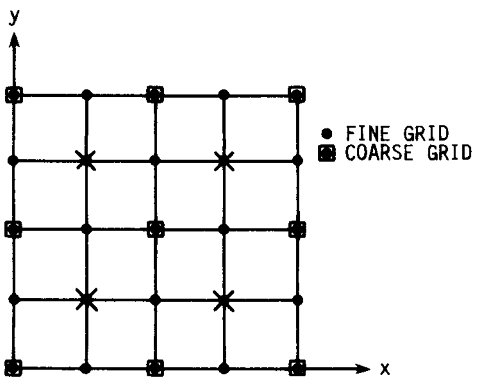

Figure 4.26 Prolongation details.

domain. The main steps are as follows:

1. The general multilevel scheme begins in the same manner as the two-level scheme discussed previously. The difference scheme, $Lu_{i,j} = 0$, is iterated n times (again n is a small number like 3 or 4, or alternatively, a "stalling factor" can be used to determine when to transfer to the coarser grids) on the finest grid.
2. The residual, $Lu^k_{i,j} = R^1_{i,j}$ is computed and stored at each point. This residual is then restricted by injection to the next coarsest grid. The restricted residual is denoted as $I^2_1 R^1_{i,j}$, where I is the transfer operator, the subscript indicates the level of origin, and the superscript the level of the destination. The superscript on the R indicates the grid upon which the residual was computed. The grids will be numbered from the finest (level 1) to coarsest.
3. The equation $L\Delta(u_2)_{i,j} = -I^2_1 R^1_{i,j}$ is iterated ("relaxed") n times on grid level 2 using zero as the initial guesses while keeping the residual fixed at each grid point. The solution after n iterations, $\Delta(u_2)^k_{i,j}$, represents a correction to the fine-grid solution. This solution, as well as the residual used to obtain it, are stored for future use in the prolongation phase. In order to transfer the problem to a coarser grid, an updated residual needs to be computed on grid level 2. It is about at this point that beginners typically lose their concentration and make mistakes. The updated residual at level 2 is $R^2_{i,j} = I^2_1 R^1_{i,j} + L\Delta(u_2)^k_{i,j}$, where $\Delta(u_2)^k_{i,j}$ is the solution obtained on grid level 2 after n iterations. The newly updated residual is then restricted to the next coarsest grid (level 3) as $I^3_2 R^2_{i,j}$.
4. The equation $L\Delta(u_3)_{i,j} = -I^3_2 R^2_{i,j}$ is iterated n times on grid level 3 using zero as the initial guess. The solution after n iterations can be thought of as a correction to the correction obtained on grid level 2, which of course, represents a further correction to the fine-grid solution. This solution and the residual used to obtain it are stored for use in the prolongation phase. The transfer to coarser grids, relaxation sweeps, and creation of new corrections continues following the residual update and restriction steps described above until the coarsest grid is reached. The coarsest grid may consist of one grid point in the interior. The solution is usually iterated to convergence on the coarsest grid. With one grid point, this solution can be obtained analytically, although the iterative scheme will normally reflect convergence in two passes, depending of course, upon how the convergence criterion is applied.
5. The corrections obtained on the coarsest grid are prolongated (interpolated) onto the next finer grid following the steps outlined in the description of the two-level scheme. Let us assume that the coarsest grid is grid level 4, in order to provide specific notation. These prolongated corrections are then $I^3_4 \Delta(u_4)^k_{i,j}$. These are added to the corrections obtained earlier at level 3 in the restriction phase, $\Delta(u_3)^k_{i,j}$. The sum of the two corrections is used as the initial guess at each point, as we continue to solve the problem we started to solve on grid level 3 on the way down in the restriction phase. That is, the problems to be solved as we move up the sequence of finer grids are the

continuation of the problems started on the way down. In other words, on grid level 3 in the prolongation phase, we continue to solve the problem

$$L\Delta(u_3)_{i,j} = -I_2^3 R_{i,j}^2$$

but the computation is started with the corrections identified above as the initial guesses. As in the restriction phase, n sweeps are made at level 3. The solution represents improved corrections.

6. The corrections from level 3 are prolongated onto the next finer grid at level 2. These corrections are added to the values of $\Delta(u_2)_{i,j}^k$ obtained at level 2 in the restriction phase, and the sums are used as the initial guesses for continuing the computation of the same problem solved at level 2 on the way down from finer to coarser grids, $L\Delta(u_2)_{i,j} = -I_1^2 R_{i,j}^1$. Again, n sweeps are made, and the solution after n sweeps represents improved corrections. Note that no new residuals are computed in the prolongation phase of moving up from coarser to finer grids. The solution is being improved as we move up toward finer grids because additional sweeps are being made that start with improved guesses.
7. The corrections from level 2 are prolongated onto grid level 1, the finest grid, and added to the last solution obtained on the fine grid, $u_{i,j}$. The corrected solution is then iterated through n sweeps unless convergence is detected before n sweeps are completed. If convergence has not occurred, new residuals are computed after n sweeps, and the cycle down to the coarsest grid and back up is repeated.

Example using multigrid. Here we will apply the multigrid technique to obtain the solution to Laplace's equation with Dirichlet boundary conditions. The computational effort for the multigrid scheme will be compared with that required for the simple Gauss-Seidel scheme and for the Gauss-Seidel scheme with optimum overrelaxation. A square domain will be utilized. The Gauss-Seidel scheme will also be used as the smoother for the multigrid scheme. Results wil' be presented for both the two-level and the multilevel V-cycle procedure. Fo. simplicity, each side of the square will be set at a fixed value of u, as indicated in Fig. (4.27). Although use of discontinuous boundary conditions is physically somewhat unrealistic, the points of discontinuity do not enter into the finite-difference calculation. The primary purpose of this example is to compare the computational effort required by the four procedures, so detailed results of the solution values themselves will not be given. As a rough check, the reader could confirm that the solution at the center is the arithmetic average of the temperatures of the four boundaries. An analytic series solution can actually be obtained for this problem by superposition and could be used to check the numerical solution.

Use of Dirichlet boundary conditions will make it easy to determine an optimum overrelaxation factor to use with the Gauss-Seidel scheme for purposes of comparison. The overrelaxation factors are obtained from the formula given

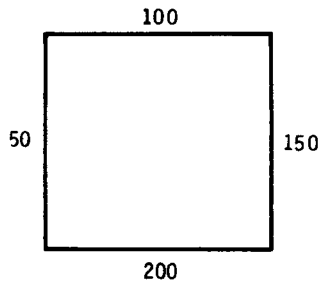

Figure 4.27 Boundary conditions.

previously in this chapter. When the coarsest grid in a multilevel scheme is to be simply one interior grid point (i.e., a 3×3 grid), it is convenient to let 2^n be the number of mesh increments into which each side of the square is divided. Hence grids of 9×9, 17×17, 33×33, 65×65, and 129×129 will be used. The overrelaxation factors computed for these grids are 1.45, 1.67, 1.82, 1.91, and 1.95. These will be used with the Gauss-Seidel scheme without multigrid.

Overrelaxation was not used with multigrid, since it did not seem to improve the convergence rate. The computational effort will be reported in terms of equivalent fine-grid sweeps that are usually referred to as work units. That is, in the multigrid calculations the total number of times the Gauss-Seidel smoother was applied was determined, and the total was then divided by the number of calculation (internal) grid points in the finest grid. As is customary in such comparisons, other operations in the multigrid algorithm such as computation of the residual, the restriction and prolongation operations, and the addition of the corrections were not counted, as such effort is generally considered to be a fairly negligible percentage of the total effort. The convergence parameter used in the calculations was the maximum change in the computed variable (u or Δu) between two successive sweeps divided by the maximum value of the dependent variable on the boundary. In the present example, that maximum boundary value was 200.

Using the same reference in the denominator of the convergence parameter for both u and Δu appeared to be important in order to obtain results that had the property that the number of iterations to convergence was independent of whether the variable itself or a correction was being computed. Convergence was declared when this parameter was less than 10^{-5}.

The number of iterations (in terms of equivalent fine-grid sweeps or work units) required for convergence for the four schemes is given in Table 4.2. For the multigrid results shown, three sweeps were made on the fine and all intermediate grids before a transfer was made and convergence was achieved on the coarsest grid for each cycle. The first column, labeled GS, gives results obtained with the conventional Gauss-Seidel scheme with no overrelaxation. The second column labeled GSω_{opt} gives results obtained with the Gauss-Seidel scheme using the optimum overrelaxation factor. The column labeled MG2 gives results obtained with the two-level multigrid, and the column labeled

Table 4.2 Number of equivalent fine-grid iterations required for convergence

Grid size	GS	GS ω_{opt}	MG2	MGMAX
9×9	62	19	19	17
17×17	215	40	40	19
33×33	715	75	95	20
65×65	2282	137	258	20
129×129	6826	282	732	21

MGMAX provides multigrid results obtained using the maximum number of levels, i.e., taking the calculation down to one internal grid point. This results in use of seven, six, five, four, and three levels for the 129×129, 65×65, 33×33, 17×17, and 9×9 grids, respectively.

A number of interesting points can be made from the results shown in Table 4.2. The number of iterative sweeps required by the standard Gauss-Seidel scheme can be seen to be almost proportional to the number of grid points used. Of course, the computational effort per sweep is also proportional to the number of points used. The use of the optimum overrelaxation factor reduces the computational effort substantially, especially as the number of grid points increases. For the finest grid, use of overrelaxation reduces the computational effort by a factor of about 24. This is significant. The two-level multigrid is seen to provide a significant reduction in computational effort, but it does not perform quite as well as the Gauss-Seidel scheme with optimum overrelaxation. However, it is general, whereas the optimum overrelaxation factor can only be computed in advance for special cases.

The performance of the multigrid with the maximum number of levels (hereafter referred to as the n-level scheme) is truly amazing. The number of sweeps is seen to be nearly independent of the number of grid points used. Only 21 sweeps were required for the finest grid compared to 6826 for the conventional Gauss-Seidel scheme. This is a reduction in effort by a factor of 325! It requires only 1/13th as much effort as the Gauss-Seidel scheme with the optimum overrelaxation. This reduction in effort or "speed-up factor" is shown graphically in Fig. 4.28. Both multigrid schemes required four cycles for convergence for this problem, which utilized only 13 or 14 sweeps through the finest grid.

For the 129×129 grid, it was observed that using four or five levels gave nearly the same performance as using seven levels. Specifically, it was observed that 732, 82, 25, 21, 21, and 21 work units (equivalent fine-grid sweeps) were required when using 2, 3, 4, 5, 6, and 7 levels, respectively. This trend is illustrated in Fig. 4.29. The especially large improvement observed when moving from two to three levels is noteworthy.

The results were fairly insensitive to the number of sweeps for the fine and intermediate grids. The following trend was observed for the n-level scheme on the 129×129 grid. For use of 2, 3, 4, and 5 sweeps, the number or work units

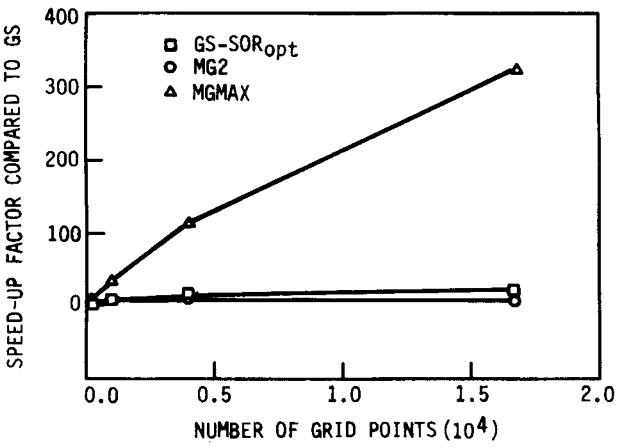

Figure 4.28 Comparison of effort for Dirichlet problem. GS-SOR$_{opt}$, Gauss-Seidel with optimum overrelaxation; MG2, two-level multigrid; MGMAX, multigrid using maximum number of levels.

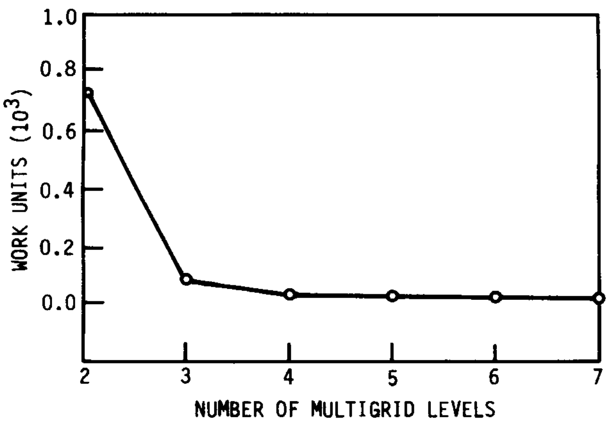

Figure 4.29 Effect of multigrid levels on Dirichlet problem.

required was 21, 21, 21, and 25, respectively. It was assumed that performance would continue to deteriorate as the number of sweeps was increased above 5.

For the two-level scheme, performance was actually improved by not converging on the coarse grid. For the 129 $\times$ 129 grid the number of equivalent fine-grid sweeps required for convergence was reduced to 495, 471, 474, and 479 if the maximum number of coarse-grid sweeps was limited to 75, 100, 125, and 150, respectively. This represents a reduction of about one-third in the computational effort for the two-level scheme. No effort was made to determine a general criterion for determining the optimum number of coarse-grid sweeps.

Multigrid for nonlinear equations. For a linear equation it is possible to "transfer the problem" from one grid to another by merely transferring the residual. This is the correction storage scheme. If the equation is nonlinear, this transfer by residual alone is generally not possible, and we must transfer (restrict and prolongate) the solution as well as the residual. Such a version of multigrid is known as the *full approximation storage* (FAS) method. It is not really much more complicated than the coarse-grid correction scheme, but it may seem like it initially because the problem formulation itself for a nonlinear equation is more complicated. Consider a 1-D problem governed by the following nonlinear equation:

$$\frac{\partial u^2}{\partial x} + \frac{\partial^2 u}{\partial x^2} = 0 \tag{4.135}$$

Choosing a central-difference discretization, we wish to find a way to satisfy the difference equation

$$\frac{u_{i+1}^2 - u_{i-1}^2}{2\,\Delta x} + \frac{u_{i+1} - 2u_i + u_{i-1}}{(\Delta x)^2} = 0 \tag{4.136}$$

Let us introduce a nonlinear difference operator N such that the difference equation above can be represented as $Nu_i = 0$. the residual R_i will be the N operator operating on any intermediate solution Nu_i^k. This definition of the residual is consistent with the terminology used for the linear problem. It will be convenient to solve the problem using a delta form for the equation on all grids. In this nonlinear case the delta will be an iteration delta, $\Delta u_i = u_i^{k+1} - u_i^k$. Suppose we wish Eq. (4.136) to be satisfied at the $k + 1$ level and substitute $u_i^k + \Delta u_i$ for u_i in Eq. (4.136). The result can be written as

$$\frac{2u_{i+1}^k\,\Delta u_{i+1} - 2u_{i-1}^k\,\Delta u_{i-1}}{2\,\Delta x} + \frac{\Delta u_{i+1} - 2\,\Delta u_i + \Delta u_{i-1}}{(\Delta x)^2}$$
$$+ \frac{(\Delta u_{i+1})^2 - (\Delta u_{i-1})^2}{2\,\Delta x} = -\left(\frac{(u_{i+1}^k)^2 - (u_{i-1}^k)^2}{2\,\Delta x} + \frac{u_{i+1}^k - 2u_i^k + u_{i-1}^k}{(\Delta x)^2} \right) \tag{4.137}$$

Observe that the last term on the left-hand side involves the change in u to the second power. Dropping this term, which is small near convergence, linearizes the difference equation and is equivalent to a Newton linearization (see Section 7.3.3), which can be developed through the use of a Taylor-series expansion, neglecting terms involving derivatives of higher order than the first. Other ways of linearizing the algebraic equation can obviously be found. The final working equation for the fine grid can be written

$$\frac{2u_{i+1}^k\,\Delta u_{i+1} - 2u_{i-1}^k\,\Delta u_{i-1}}{2\,\Delta x} + \frac{\Delta u_{i+1} - 2\,\Delta u_i + \Delta u_{i-1}}{(\Delta x)^2}$$
$$= -\left(\frac{(u_{i+1}^k)^2 - (u_{i-1}^k)^2}{2\,\Delta x} + \frac{u_{i+1}^k - 2u_i^k + u_{i-1}^k}{(\Delta x)^2}\right) \qquad (4.138)$$

where the right-hand side can be recognized as the negative of the residual evaluated at the kth iteration level. Notice that as the iterative process converges, the delta terms on the left-hand side vanish, and the nonlinear difference equation is satisfied exactly as the residual goes to zero.

Only a two-level FAS multigrid scheme will be discussed here. The sweeping strategy is the same as before: n sweeps on the fine grid and then a transfer to the coarse grid, where normally, the solution would be iterated to convergence and then the changes transferred to the fine grid. The equation being solved on the fine grid is Eq. (4.138). Note that because the residual or delta form of the governing equation is being solved on the fine grid, the residual in Eq. (4.138) is updated at every iteration. After n sweeps, the most recent residual and the current solution u_i are restricted to the coarse grid.

As before, on the coarse grid we wish to compute changes to the solution that will annihilate the residual on the fine grid. However, because the equation is nonlinear, we continue to solve for the solution itself on the coarse grid even though it will be the changes or corrections to the fine-grid solution that will be prolongated to the fine grid. Thus Eq. (4.138) would be appropriate to use on the coarse grid, provided the right-hand side (residual) would vanish if and only if the residual on the fine grid vanished. This would happen if we modified the residual by adding the difference between the restricted fine-grid residual and the residual computed on the coarse grid using the restricted fine-grid solution. This modification to the residual can be thought of as compensation for the difference in T.E.s associated with the solution on meshes of different sizes. Thus, letting M be the operator that gives the left-hand side of Eq. (4.138), we can write the difference equation solved on the coarse grid as

$$M\,\Delta u_i = -R_i^2 - I_1^2 R_i^1 + R_i^2(I_1^2 u_i^1) \qquad (4.139)$$

where the first term on the right-hand side is the residual computed from the coarse-grid solution [evaluated as indicated by the right-hand side of Eq. (4.138)] at each coarse-grid iteration. The second term on the right is the restricted fine-grid residual. This is the same term that was used in the correction scheme in the linear example. The third term is the residual computed on the coarse grid using the restricted fine-grid solution. The second and third terms are source terms that remain fixed during the iterative process on the coarse grid.

The restricted fine-grid solution is taken as the starting value for u on the coarse grid. Thus we notice that the first and third terms cancel for the first coarse-grid sweep. This leaves the restricted residual from the fine grid as the source term driving the changes. Notice also that no changes would be computed if the residual on the fine grid were zero. This formulation for the right-hand side properly takes into account discrepancies that arise owing to the restriction. If no "errors" were introduced in the restriction, the second and third terms would cancel. The formulation also ensures that it is the residual on the fine grid that drives the multigrid process. At the conclusion of the coarse-grid iterations (usually signaled by convergence on the coarse grid), the *changes* in u computed over the duration of the coarse-grid iterations are prolongated onto the fine grid. An additional n iterations are performed on the fine grid, and if the solution has not converged, the cycle is repeated.

4.4 BURGERS' EQUATION (INVISCID)

We have discussed finite-difference methods and have applied them to simple linear problems. This has provided an understanding of the various techniques and acquainted us with the peculiarities of each approach. Unfortunately, the usual fluid mechanics problem is highly nonlinear. The governing PDEs form a nonlinear system that must be solved for the unknown pressures, densities, temperatures, and velocities.

A single equation that could serve as a nonlinear analog of the fluid mechanics equations would be very useful. This single equation must have terms that closely duplicate the physical properties of the fluid equations, i.e., the model equation should have a convective term, a diffusive or dissipative term, and a time-dependent term. Burgers (1948) introduced a simple nonlinear equation that meets these requirements:

$$\underbrace{\frac{\partial u}{\partial t}}_{\substack{\text{Unsteady}\\ \text{term}}} + \underbrace{u\frac{\partial u}{\partial x}}_{\substack{\text{Convective}\\ \text{term}}} = \underbrace{\mu\frac{\partial^2 u}{\partial x^2}}_{\substack{\text{Viscous}\\ \text{term}}} \tag{4.140}$$

Equation (4.140) is parabolic when the viscous term is included. If the viscous term is neglected, the remaining equation is composed of the unsteady term and a nonlinear convection term. The resulting hyperbolic equation

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = 0 \tag{4.141}$$

may be viewed as a simple analog of the Euler equations for the flow of an inviscid fluid. Equation (4.141) is a nonlinear convection equation and possesses properties that need to be examined in some detail. Methods for solving the inviscid Burgers equation will be presented in this section. Typical results for a number of commonly used finite-difference/finite-volume methods are included, and the effects of the nonlinear terms are discussed. A discussion of the viscous Burgers equation follows in Section 4.5.

Equation (4.141) may be viewed as a nonlinear wave equation, where each point on the wave front can propagate with a different speed. In contrast, the speed of propagation of all signals or waves was constant for the linear, 1-D convection equation, Eq. (4.2). A consequence of the changing wave speed is the coalescence of characteristics and the formation of discontinuous solutions similar to shock waves in fluid mechanics. This means the class of solutions that include discontinuities can be studied with this simple 1-D model.

Nonlinear hyperbolic PDEs exhibit two types of solutions according to Lax (1954). For simplicity, we consider a simple scalar equation,

$$\frac{\partial u}{\partial t} + \frac{\partial F}{\partial x} = 0 \tag{4.142}$$

For the general case, both the unknown u and the variable $F(u)$ are vectors. We may write Eq. (4.142) as

$$\frac{\partial u}{\partial t} + A\frac{\partial u}{\partial x} = 0 \tag{4.143}$$

where $A = A(u)$ is the Jacobian matrix $\partial F_i/\partial u_j$ for the general case and is dF/du for our simple equation. Our equation or system of equations is hyperbolic, which means that the eigenvalues of the matrix A are all real. A *genuine solution* of Eq. (4.143) is one in which u is continuous but bounded discontinuities in the derivatives of u may occur (Lipschitz continuous). A *weak solution* of Eq. (4.143) is a solution that is genuine except along a surface in (x, t) space, across which the function u may be discontinuous. A constraint is placed upon the jump in u across the discontinuity in the domain of interest. If w is a test vector that is continuous and has continuous first derivatives but vanishes outside some bounded set, then u is termed a weak solution of Eq. (4.142) if

$$\iint_D (w_t u + w_x F)\, dx\, dt + \int w(x,0)\phi(x)\, dx = 0 \tag{4.144}$$

where $\phi(x) = u(x, 0)$. A genuine solution is a weak solution, and a weak solution that is continuous is a genuine solution. A complete discussion of the weak solution concept may be found in the excellent texts by Whitham (1974) and Jeffrey and Taniuti (1964). The mathematical theory of weak solutions for hyperbolic equations is a relatively recent development. Clearly, the existence of shock waves in inviscid supersonic flow is an example of a weak solution. It is interesting to recognize that the shock solutions in inviscid supersonic flow were known 50–100 years before the theory of weak solutions for hyperbolic systems was developed.

Let us return to the study of the inviscid Burgers equation and develop the requirements for a weak solution, i.e., the requirements necessary for the existence of a solution with a discontinuity such as that shown in Fig. 4.30.

Let $w(x, t)$ be an arbitrary test function that is continuous and has continuous first derivatives. Let $w(x, t)$ vanish on the boundary B of the domain

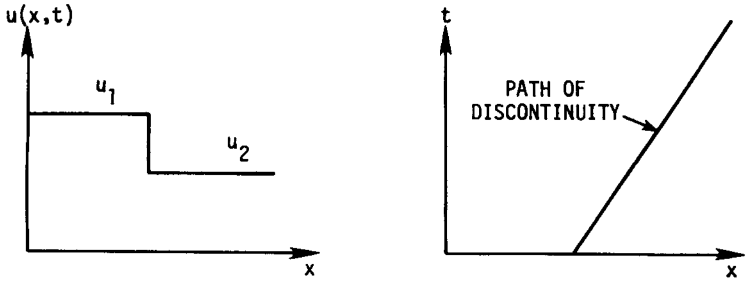

Figure 4.30 Typical traveling discontinuity problem for Burgers' equation.

D and everywhere outside D (complement of D). D is an arbitrary rectangular domain in the (x, t) plane. We may write

$$\iint_{\mathrm{D}} \left(\frac{\partial u}{\partial t} + \frac{\partial F}{\partial x} \right) w(x,t)\, dx\, dt = 0 \tag{4.145}$$

or

$$\iint_{\mathrm{D}} (u w_t + F w_x)\, dx\, dt = 0 \tag{4.146}$$

Equations (4.145) and (4.146) are equivalent when both u and F are continuous and have continuous first derivatives. The second integral of Eq. (4.144) does not appear, since the function w vanishes on the boundary. Functions $u(x, t)$, which satisfy Eq. (4.146) for all test functions w, are called weak solutions of the inviscid Burgers equation. We do not require that u be differentiable in order to satisfy Eq. (4.146).

Suppose our domain D is now a rectangular region in the (x, t) plane, which is separated by a curve $\tau(x, t) = 0$, across which u is discontinuous. We assume that u is continuous and has continuous derivatives to the left of $\tau(\mathrm{D}_1)$ and to the right of $\tau(\mathrm{D}_2)$. Let the test function vanish on the boundary of D and outside of D. With these restrictions, Eq. (4.146) can be integrated by parts to yield

$$\iint_{\mathrm{D}_1} \left(\frac{\partial u}{\partial t} + \frac{\partial F}{\partial x} \right) w\, dx\, dt + \iint_{\mathrm{D}_2} \left(\frac{\partial u}{\partial t} + \frac{\partial F}{\partial x} \right) w\, dx\, dt$$

$$+ \int_{\tau} ([u] \cos \alpha_1 + [F] \cos \alpha_2)\, ds = 0 \tag{4.147}$$

The last integrand is evaluated along the curve $\tau(x, t) = 0$ separating the two regions D_1 and D_2. This integral occurs through the limits of the integration by parts on the discontinuity surface $\tau(x, t) = 0$. The square brackets denote the jump in the quantity across the discontinuity, and $\cos \alpha_1, \cos \alpha_2$ are the cosines of the angles between the normal to $\tau(x, t) = 0$ and the t and x directions, respectively. The problem is illustrated in Fig. 4.31.

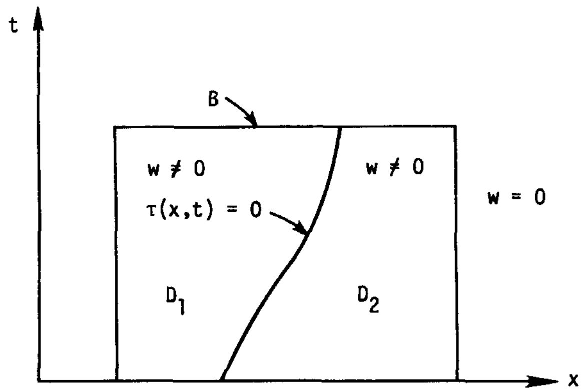

Figure 4.31 Schematic representation of an arbitrary domain with a discontinuity.

The integrals in Eq. (4.147) over D_1 and D_2 are zero by Eq. (4.145). We conclude that since the last integral vanishes for all test functions w with the required properties, we must have

$$[u]\cos\alpha_1 + [F]\cos\alpha_2 = 0 \tag{4.148}$$

This is the condition that u be a weak solution for Burgers' equation. Let us apply this condition to a moving discontinuity. Suppose initial data are prescribed for $u(x,0)$ as shown in Fig. 4.30, where u_1 and u_2 denote the values to the left and to the right of the discontinuity. In one dimension, we may write the equation of the surface $\tau(x,t) = 0$ as $t - t_1(x) = 0$. The direction cosines as required in Eq. (4.148) become

$$\cos\alpha_1 = \frac{1}{\left[1 + t_1'^2\right]^{1/2}} \qquad \cos\alpha_2 = -\frac{t_1'}{\left[1 + t_1'^2\right]^{1/2}}$$

where the prime denotes differentiation with respect to x. Thus

$$\frac{[u]}{[1 + t'^2]^{1/2}} - \frac{[F]t'}{[1 + t'^2]^{1/2}} = 0$$

or

$$u_2 - u_1 = \frac{u_2^2 - u_1^2}{2}\frac{dt}{dx}$$

Therefore

$$\frac{dx}{dt} = \frac{u_1 + u_2}{2} \tag{4.149}$$

which shows that the discontinuity travels at the average value of the u function across the wave front. Since we now see that a discontinuity in u simply propagates at constant speed $(u_1 + u_2)/2$ with uniform states on each side, a

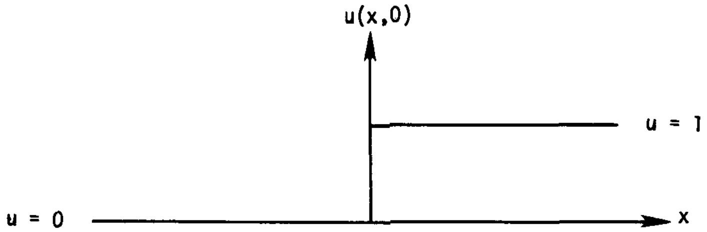

Figure 4.32 Initial data for rarefaction wave.

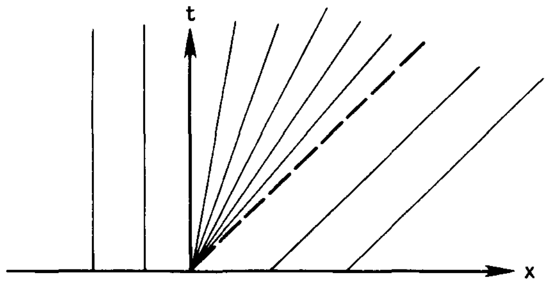

Figure 4.33 Characteristics for centered expansion.

numerical solution of a similar problem for a discontinuity can be compared with the exact solution. These comparisons are presented for a number of finite-difference/finite-volume methods in this section.

Rarefactions are as prevalent in high-speed flows as shock waves, and the exact solution of Burgers' equation for a rarefaction is known. Consider initial data $u(x, 0)$ as shown in Fig. 4.32. The characteristic for Burgers' equation is given by

$$\frac{dt}{dx} = \frac{1}{u} \tag{4.150}$$

Figure 4.33 shows the characteristic diagram plotted in the (x, t) plane. In the left half-plane, the characteristics are simply vertical lines, while they are lines at an angle of $\pi/4$ radians to the right of the characteristic that bounds the expansion. This particular problem is similar to a centered expansion wave in compressible flow. Here the expansion is bounded by the $x = 0$ axis and the characteristic originating at the origin denoted by the dashed line. The solution for this problem may be written

$$\begin{aligned} u &= 0 & x &\leqslant 0 \\ u &= \frac{x}{t} & 0 &< x < t \\ u &= 1 & x &\geqslant t \end{aligned}$$

The initial distribution of u forms a centered expansion, where the width of the expansion grows linearly with time.

We have examined two problems, shocks and rarefactions, which are frequently encountered in high-speed flows by using the simple analog provided by Burgers' equation. Clearly, these types of solutions can occur in systems of nonlinear equations of the hyperbolic type. Armed with simple analytic solutions for these two important cases, let us examine the application of some numerical algorithms to the nonlinear, inviscid Burgers equation.

4.4.1 Lax Method

First-order methods for solving hyperbolic equations are infrequently used. The Lax (1954) method is presented as a typical first-order method to demonstrate the application to a nonlinear equation and the dissipative character of the result.

The conservation form of the basic PDE,

$$\frac{\partial u}{\partial t} + \frac{\partial F}{\partial x} = 0$$

is used for all examples that follow. For the Lax method, we expand in a Taylor series about the point (x, t), retaining only the first two terms

$$u(x, t + \Delta t) = u(x, t) + \Delta t\left(\frac{\partial u}{\partial t}\right)_{x,t} + \cdots$$

and substitute for the time derivative

$$u(x, t + \Delta t) = u(x, t) - \Delta t\left(\frac{\partial F}{\partial x}\right)_{x,t} + \cdots$$

Using centered differences and averaging the first term yields the Lax method (see Section 4.1.3):

$$u_j^{n+1} = \frac{u_{j+1}^n + u_{j-1}^n}{2} - \frac{\Delta t}{\Delta x}\frac{F_{j+1}^n - F_{j-1}^n}{2} \tag{4.151}$$

In Burgers' equation, $F = u^2/2$. The amplification factor in this case is

$$G = \cos\beta - i\frac{\Delta t}{\Delta x}A\sin\beta \tag{4.152}$$

where A is the Jacobian dF/du, which is just the single element u for Burgers' equation. The stability requirement for this method is

$$\left|\frac{\Delta t}{\Delta x}u_{\max}\right| \leqslant 1 \tag{4.153}$$

because $u_{\max}$ is the maximum eigenvalue of the A matrix with the single element u.

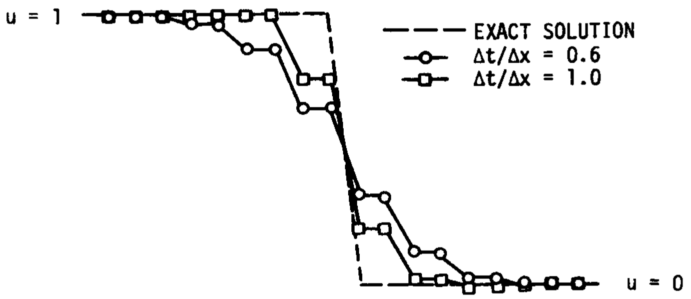

Figure 4.34 Numerical solution of Burgers' equation using Lax method.

The Lax method applied to a 1-0 right-moving discontinuity produces the solutions shown in Fig. 4.34. The location of the moving discontinuity is correctly predicted, but the dissipative nature of the method is evident in the smearing of the discontinuity over several mesh intervals. As previously noted, this smearing becomes worse as the Courant number decreases. It is of interest to note that the application of the Lax method to Burgers' equation with a discontinuity produces the double-point solutions as shown. A further comment on these results is in order. Notice that the computed solutions are monotone, i.e., the solution does not oscillate. Godunov (1959) has shown that monotone behavior of a solution cannot be assured for finite-difference methods with more than first-order accuracy. This monotone property is very desirable when discontinuities are computed as part of the solution. Unfortunately, the desirability of monotone behavior must be reconciled with the highly dissipative character of the results. The relative importance of these properties must be carefully evaluated for each case.

The finite-volume equivalent of the Lax method can be readily developed by noting that first-order integration (in time) over a control volume (see Fig. 4.35) provides the expression

$$u_j^{n+1} = u_j^n - \frac{\Delta t}{\Delta x}\left[f_{j+\frac{1}{2}}^n - f_{j-\frac{1}{2}}^n\right] \tag{4.154}$$

In this expression the control volume may be considered to be centered at the point $(j, n + \frac{1}{2})$. The flux terms, $f_{j\pm\frac{1}{2}}^n$ are referred to as the numerical fluxes, since they represent the flux at the surface of the control volume in the finite-volume formulation. Functionally, the numerical flux is written

$$f_{j+\frac{1}{2}} = f(u_j, u_{j+1})$$

The numerical flux must be consistent with the analytical flux F, so that

$$f(u_j, u_{j+1}) = F(u_j) \tag{4.155}$$

when

$$u_j = u_{j+1} = u$$

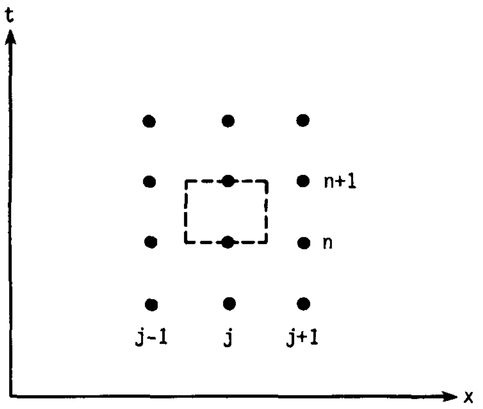

Figure 4.35 Control volume for Lax method.

The problem of finding the flux function is very important because it represents the control-volume boundary flux. This is needed in constructing methods for solving the conservative form of the equations of fluid dynamics.

For the Lax method the numerical flux becomes

$$f_{j+\frac{1}{2}} = \frac{1}{2}\left[F_j + F_{j+1} - \frac{\Delta x}{\Delta t}(u_{j+1} - u_j)\right] \tag{4.156}$$

The last term in this expression can be viewed as a dissipation term. The order of the numerical scheme can be altered by using a different form for this term. It will be of value to compare Eq. (4.156) with the numerical flux terms of other methods in this chapter.

Another observation can be made regarding the Lax method. Notice that the computer solutions in Fig. 4.34 are monotone, i.e., the solution does not oscillate. Godunov (1959) studied numerical methods applied to the simple 1-D wave equation (Eq. 4.2) having the linear form

$$u_j^{n+1} = \sum_k a_k u_{j+k}^n \tag{4.157}$$

If the right-hand side is expanded in a Taylor series, second-order accuracy for such a method is established if

$$\sum_k a_k = 1 \tag{4.158}$$

$$\sum_k k a_k = -\nu \tag{4.159}$$

$$\sum_k k^2 a_k = \nu^2 \tag{4.160}$$

This may be verified by examining the second-order schemes previously considered in Section 4.1.

If solutions produced by Eq. (4.157) do not oscillate, what are the conditions that guarantee monotone behavior? A necessary and sufficient condition is that all of the coefficients a_k be positive. Consider the change in the solution between points j and $j + 1$:

$$u_{j+1}^{n+1} - u_j^{n+1} = \sum_k a_k (u_{j+k+1}^n - u_{j+k}^n) \tag{4.161}$$

If the solution at level n is monotone, then every difference on the right-hand side is of the same sign. Thus, if the a_k are all positive, then the differences on the right- and left-hand sides carry the same sign. This is also a necessary condition because for at least one value of a_k of opposite sign, monotone initial data may be constructed that produce an oscillation at the next level.

Godunov's (1959) theorem states that second-order schemes are not monotone. This may be proved by letting $a_k = (e_k)^2$ and substituting into Eqs. (4.158)–(4.160) to obtain

$$\sum_k (e_k)^2 \left(\sum k e_k^2 \right)^2 = \sum_k k^2 e_k^2$$

This equation violates the Cauchy inequality (Taylor, 1955) and shows that no monotone second-order schemes exist. This result provided a major difficulty that needed to be overcome in the development of methods for solving hyperbolic PDEs. In regions where discontinuities develop, measures must be taken in order to avoid oscillations.

4.4.2 Lax-Wendroff Method

The Lax-Wendroff method (Lax and Wendroff, 1960) was one of the first second-order finite-difference methods for hyperbolic PDEs. The development of the Lax-Wendroff scheme for nonlinear equations again follows from a Taylor series:

$$u(x, t + \Delta t) = u(x, t) + \Delta t \left(\frac{\partial u}{\partial t} \right)_{x,t} + \frac{(\Delta t)^2}{2} \left(\frac{\partial^2 u}{\partial t^2} \right)_{x,t} + \cdots$$

The first time derivative can be directly replaced using the differential equation, but we need to examine the second-derivative term in more detail. We consider the original equation in the form

$$\frac{\partial u}{\partial t} = -\frac{\partial F}{\partial x}$$

Taking the time derivative of this expression yields

$$\frac{\partial^2 u}{\partial t^2} = -\frac{\partial^2 F}{\partial t\, \partial x} = -\frac{\partial^2 F}{\partial x\, \partial t} = -\frac{\partial}{\partial x} \left(\frac{\partial F}{\partial t} \right)$$

where the order of differentiation on F has been interchanged. Now $F = F(u)$, which permits us to write

$$\frac{\partial u}{\partial t} = -\frac{\partial F}{\partial x} = -\frac{\partial F}{\partial u}\frac{\partial u}{\partial x} = -A\frac{\partial u}{\partial x}$$

and

$$\frac{\partial F}{\partial t} = \frac{\partial F}{\partial u}\frac{\partial u}{\partial t} = A\frac{\partial u}{\partial t}$$

Hence we may replace $\partial F/\partial t$ with

$$\frac{\partial F}{\partial t} = -A\frac{\partial F}{\partial x}$$

so that

$$\frac{\partial^2 u}{\partial t^2} = \frac{\partial}{\partial x}\left(A\frac{\partial F}{\partial x}\right)$$

The Jacobian A contains a single element for Burgers' equation. It is clear that A is a matrix when u and F are vectors in treating a system of equations. Making the appropriate substitution in the Taylor-series expansion for u, we obtain

$$u(x,t+\Delta t) = u(x,t) - \Delta t\frac{\partial F}{\partial x} + \frac{(\Delta t)^2}{2}\frac{\partial}{\partial x}\left(A\frac{\partial F}{\partial x}\right) + \cdots$$

After using central differencing, the Lax-Wendroff method is obtained:

$$u_j^{n+1} = u_j^n - \frac{\Delta t}{\Delta x}\frac{F_{j+1}^n - F_{j-1}^n}{2} + \frac{1}{2}\left(\frac{\Delta t}{\Delta x}\right)^2 \times\left[A_{j+1/2}^n(F_{j+1}^n - F_j^n) - A_{j-1/2}^n(F_j^n - F_{j-1}^n)\right] \tag{4.162}$$

The Jacobian matrix is evaluated at the half interval, i.e.,

$$A_{j+1/2} = A\left(\frac{u_j + u_{j+1}}{2}\right)$$

In Burgers' equation, $F = u^2/2$ and $A = u$. In this case $A_{j+1/2} = (u_j + u_{j+1})/2$ and $A_{j-1/2} = (u_j + u_{j-1})/2$. The amplification factor for this method is

$$G = 1 - 2\left(\frac{\Delta t}{\Delta x}A\right)^2(1 - \cos\beta) - 2i\frac{\Delta t}{\Delta x}A\sin\beta \tag{4.163}$$

and the stability requirement reduces to $|(\Delta t/\Delta x)u_{\max}| \leqslant 1$.

The results obtained when the Lax-Wendroff method is applied to our example problem are shown in Fig. 4.36. The right-moving discontinuity is correctly positioned and is sharply defined. The dispersive nature of this method is evidenced through the presence of oscillations near the discontinuity. Even though the method uses central differences, some asymmetry will occur, since the wave is moving. The solution shows more oscillations when a Courant

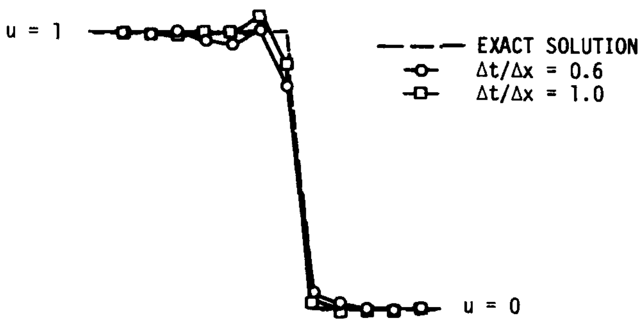

Figure 4.36 Application of the Lax-Wendroff method to the inviscid Burgers equation.

number of 0.6 is used than for a Courant number of 1.0. In general, as the Courant number is reduced, the quality of the solution will be degraded (see Section 4.1.6).

The numerical flux for the Lax-Wendroff scheme consistent with Eq. (4.156) may be written as

$$f_{j+\frac{1}{2}} = \frac{1}{2}(F_j + F_{j+1}) - \frac{1}{2}\frac{\Delta t}{\Delta x}(\lambda_{j+\frac{1}{2}})^2(u_{j+1} - u_j) \tag{4.164}$$

where $\lambda_{j+\frac{1}{2}}$ is defined as the eigenvalue of the Jacobian $A_{j+\frac{1}{2}}$, which is simply $u_{j+\frac{1}{2}}$ for Burgers' equation.

A comparison of Eq. (4.164) with Eq. (4.156) can yield valuable insight into the order and the behavior of the numerical methods. As noted in the previous section, the Lax scheme is monotone, while Fig. 4.36 shows that the second-order Lax-Wendroff scheme is not. If the numerical flux terms are compared, we see that the difference in these terms is

$$f^{LW}_{j+\frac{1}{2}} - f^{L}_{j+\frac{1}{2}} = \frac{\Delta x}{\Delta t}\left(\frac{u_{j+1} - u_j}{2}\right)\left[1 - \left(\lambda_{j+\frac{1}{2}}\frac{\Delta t}{\Delta x}\right)^2\right] \tag{4.165}$$

This difference is a correction that may be added to the Lax method to provide second-order accuracy and, in fact, modify the solution to the form of the Lax-Wendroff scheme.

However, if oscillations at discontinuities occur as seen in Fig. 4.36, the correction term should be suppressed in the region where discontinuities appear. This will have the effect of reducing the order of the method and have the potential of providing monotone or near-monotone profiles through discontinuities. The control of the correction term is usually accomplished by the use of *limiters*. With this idea in mind, the flux may be written

$$f_{j+\frac{1}{2}} = \frac{1}{2}\left[F_j + F_{j+1} - \frac{\Delta x}{\Delta t}(u_{j+1} - u_j)\right] + \phi\frac{\Delta x}{\Delta t}\left(\frac{u_{j+1} - u_j}{2}\right)\left[1 - \left(\lambda_{j+\frac{1}{2}}\frac{\Delta t}{\Delta x}\right)^2\right] \tag{4.166}$$

where ϕ is a function that can be adjusted to limit the addition of the second-order terms. The hybrid method presented by Harten and Zwas (1972) used this idea. The concept of using limiters to improve the effectiveness of solution techniques is discussed in detail in Section 4.4.12.

4.4.3 MacCormack Method

MacCormack's (1969) method is a predictor-corrector version of the Lax-Wendroff scheme, as has been discussed in Section 4.1.8. This method is much easier to apply than the Lax-Wendroff scheme because the Jacobian does not appear. When applied to the inviscid Burgers equation, the MacCormack method becomes

$$\begin{aligned} \overline{u_j^{n+1}} &= u_j^n - \frac{\Delta t}{\Delta x}(F_{j+1}^n - F_j^n) \\ u_j^{n+1} &= \frac{1}{2}\left[u_j^n + \overline{u_j^{n+1}} - \frac{\Delta t}{\Delta x}\left(\overline{F_j^{n+1}} - \overline{F_{j-1}^{n+1}}\right)\right] \end{aligned} \tag{4.167}$$

The amplification factor and stability requirement are the same as presented for the Lax-Wendroff method. The results of applying this method are shown in Fig. 4.37. Again the right-moving wave is well defined. We note that the solutions obtained for the same problem at the same Courant number are different from those obtained using the Lax-Wendroff scheme. This is due both to the switched differencing in the predictor and the corrector and the nonlinear nature of the governing PDE. One should expect results that show some differences, even though both methods are equivalent for linear problems.

In general, the MacCormack method provides good resolution at discontinuities. It should be noted in passing that reversing the differencing in the predictor and corrector steps leads to quite different results. The best resolution of discontinuities occurs when the difference in the predictor is in the

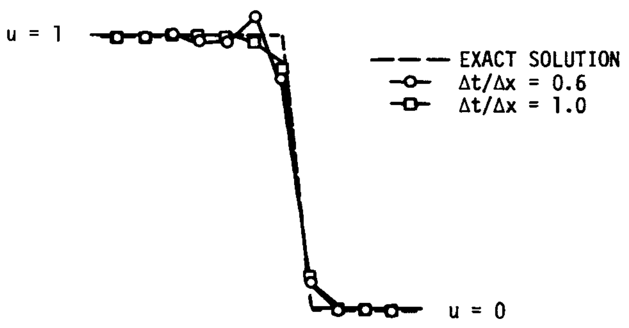

Figure 4.37 Solution of Burgers' equation using MacCormack's method.

direction of propagation of the discontinuity. This will be apparent when problems at the end of the chapter are completed.

4.4.4 Rusanov (Burstein-Mirin) Method

The third-order Rusanov or Burstein-Mirin method was discussed in Section 4.1.11. This method uses central differencing and, when applied to Eq. (4.142), becomes

$$u_{j+1/2}^{(1)} = \frac{1}{2}(u_{j+1}^n + u_j^n) - \frac{1}{3}\frac{\Delta t}{\Delta x}(F_{j+1}^n - F_j^n)$$

$$u_j^{(2)} = u_j^n - \frac{2}{3}\frac{\Delta t}{\Delta x}\left(F_{j+1/2}^{(1)} - F_{j-1/2}^{(1)}\right)$$

$$u_j^{n+1} = u_j^n - \frac{1}{24}\frac{\Delta t}{\Delta x}(-2F_{j+2}^n + 7F_{j+1}^n - 7F_{j-1}^n + 2F_{j-2}^n) - \frac{3}{8}\frac{\Delta t}{\Delta x}\left(F_{j+1}^{(2)} - F_{j-1}^{(2)}\right)$$
$$- \frac{\omega}{24}(u_{j+2}^n - 4u_{j+1}^n + 6u_j^n - 4u_{j-1}^n + u_{j-2}^n) \qquad (4.168)$$

The last term in the third step represents a fourth-derivative term,

$$(\Delta x)^4 \frac{\partial^4 u}{\partial x^4}$$

and is added for stability. The third-order accuracy of the method is unaffected, since this added term is $O[(\Delta x)^4]$. A stability analysis of this method shows that the amplification factor is

$$G = 1 - \left(\frac{\Delta t}{\Delta x}u\right)^2 \frac{\sin^2\beta}{2} - \frac{\omega}{6}(1 - \cos\beta) + i\frac{\Delta t}{\Delta x}u\sin\beta$$
$$\times\left\{1 + \frac{1}{3}(1 - \cos\beta)\left[1 - \left(\frac{\Delta t}{\Delta x}u\right)^2\right]\right\} \qquad (4.169)$$

It follows that stability is assured for Burgers' equation if

$$|\nu| \leqslant 1 \qquad \text{or} \qquad \left|\frac{\Delta t}{\Delta x}u_{\max}\right| \leqslant 1$$

and

$$4\nu^2 - \nu^4 \leqslant \omega \leqslant 3 \qquad (4.170)$$

Application of this method to Burgers' equation for a right-moving shock produces the results shown in Fig. 4.38. The magnitude and position of the discontinuity are correctly produced, but the results show an overshoot on both sides of the shock front. A schematic showing the numerical solution as it is computed from the base points is shown in Fig. 4.39.

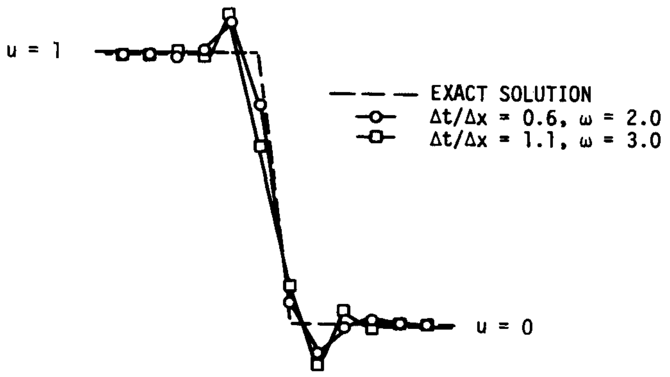

Figure 4.38 Rusanov method applied to Burgers' equation.

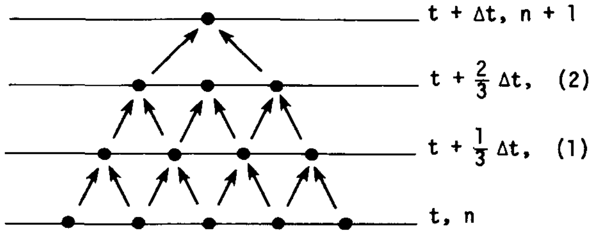

Figure 4.39 Point pyramid for the Rusanov method.

4.4.5 Warming-Kutler-Lomax Method

Warming et al. (1973) developed a third-order scheme using noncentered differences. This technique uses the MacCormack method for the first two levels evaluated at $\frac{2}{3}\,\Delta t$. The advantage of this method over the Rusanov technique is that only values at integral mesh points are required in the calculation.

The Warming-Kutler-Lomax (WKL) method applied to Eq. (4.142) becomes

$$u_j^{(1)} = u_j^n - \frac{2}{3}\frac{\Delta t}{\Delta x}(F_{j+1}^n - F_j^n)$$

$$u_j^{(2)} = \frac{1}{2}\left[u_j^n + u_j^{(1)} - \frac{2}{3}\frac{\Delta t}{\Delta x}\left(F_j^{(1)} - F_{j-1}^{(1)}\right)\right]$$

$$u_j^{n+1} = u_j^n - \frac{1}{24}\frac{\Delta t}{\Delta x}(-2F_{j+2}^n + 7F_{j+1}^n - 7F_{j-1}^n + 2F_{j-2}^n) - \frac{3}{8}\frac{\Delta t}{\Delta x}\left(F_{j+1}^{(2)} - F_{j-1}^{(2)}\right)$$
$$- \frac{\omega}{24}(u_{j+2}^n - 4u_{j+1}^n + 6u_j^n - 4u_{j-1}^n + u_{j-2}^n) \qquad (4.171)$$

The third level for the WKL scheme is exactly the same as that used in the

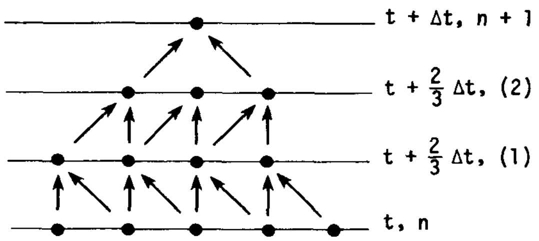

Figure 4.40 Point schematic for WKL method.

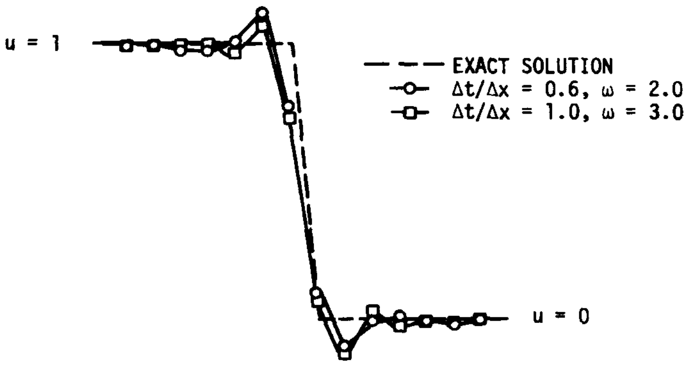

Figure 4.41 Burgers' equation solution using the WKL method.

Rusanov technique. We should note that different third-order schemes can be generated by altering the first two steps. Burstein and Mirin have shown that any second-order method may be used to generate $u_j^{(2)}$. The linear stability bound for the WKL and the Rusanov methods is the same as given in Eq. (4.170). The schematic illustrating the grid points used in the WKL method is presented in Fig. 4.40. Notice that the preferential treatment of the first two levels is readily apparent in this diagram. The differencing in the first two levels can be reversed or even cycled from time step to time step.

The results of using the WKL method to solve Burgers' equation for a right-moving discontinuity are shown in Fig. 4.41. The solution is nearly the same as that obtained in the previous section. Based upon the calculated results, either of the third-order methods may be used with approximately equal accuracy.

4.4.6 Tuned Third-Order Methods

The parameter ω, which appears in the third level of the methods of the previous two sections, can be chosen arbitrarily, as long as the stability bound is not violated. Once ω is selected at the beginning of a calculation, it retains the same value throughout the mesh. However, if the numerical damping term is written in conservation-law form for the third level, i.e.,

$$\frac{\partial}{\partial x}\left(\omega\frac{\partial^3 u}{\partial x^3}\right)$$

then ω may be altered from point to point in the calculation, and correct flux conservation in the mesh is assured. Using this approach, the ω term in the last level of either the Rusanov or WKL method can be written as

$$\begin{aligned}&-\frac{\omega_{j+1/2}^n}{24}\left(u_{j+2}^n - 3u_{j+1}^n + 3u_j^n - u_{j-1}^n\right)\\&+\frac{\omega_{j-1/2}^n}{24}\left(u_{j+1}^n - 3u_j^n + 3u_{j-1}^n - u_{j-2}^n\right)\end{aligned} \tag{4.172}$$

The $\omega_{j\pm 1/2}^n$ values are now varied according to the effective Courant number in the mesh. Warming et al. (1973) suggest that these parameters be calculated at each point in the mesh to minimize either the dispersive error or the dissipative error.

A discussion of the modified equation for third-order methods is presented in Section 4.1.11. If the minimum dispersive error is desired, then according to Eq. (4.68), we should choose

$$\omega_{j\pm 1/2}^n = \frac{\left(4\nu_{j\pm 1/2}^2 + 1\right)\left(4 - \nu_{j\pm 1/2}^2\right)}{5} \tag{4.173}$$

It remains to arrive at a rational method to determine the effective Courant numbers, $\nu_{j\pm 1/2}$. Warming et al. (1973) suggest that the effective Courant numbers, used to determine the $\omega_{j\pm 1/2}$ parameters, be the average value at the mesh points used in the difference formula. Since the term containing $\omega_{j+1/2}$ involves points $j + 2$, $j + 1$, j, and $j - 1$, we can write

$$\nu_{j+1/2} = \frac{1}{4}\left(\lambda_{j+2} + \lambda_{j+1} + \lambda_j + \lambda_{j-1}\right)\frac{\Delta t}{\Delta x} \tag{4.174}$$

and similarly,

$$\nu_{j-1/2} = \frac{1}{4}\left(\lambda_{j+1} + \lambda_j + \lambda_{j-1} + \lambda_{j-2}\right)\frac{\Delta t}{\Delta x}$$

where λ is the local eigenvalue. For Burgers' equation, λ is just the unknown u. Results obtained using this variable ω or tuned approach are shown in Fig. 4.42. This shows that both third-order methods provide satisfactory solutions for the minimum dispersion case. A slightly larger overshoot occurs at the left of the discontinuity, but a nearly exact solution is obtained on the right. The minimum dissipative method of computing $\omega_{j\pm 1/2}$ is not recommended. The ω parameter was added to provide stability, and when the dissipation is minimized, stability problems can occur. Even for stable solutions, large oscillations may be present. It should be noted that the parameters $\omega_{j\pm 1/2}$ may be computed using any

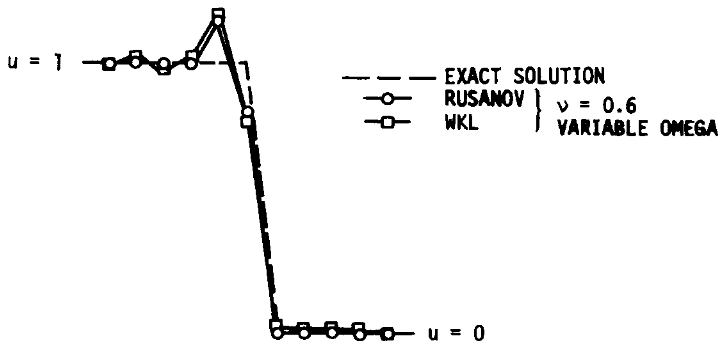

Figure 4.42 Tuned or variable ω method applied to Burgers' equation.

technique that does not violate the stability bound. Clearly, a different computed solution will be obtained for each way of computing these parameters.

4.4.7 Implicit Methods

The time-centered implicit method (trapezoidal method) is presented in Section 4.1.10. This scheme is based upon Eq. (4.58). If we substitute into Eq. (4.58) for the time derivatives using our model equation, we obtain

$$u_j^{n+1} = u_j^n - \frac{\Delta t}{2}\left[\left(\frac{\partial F}{\partial x}\right)^n + \left(\frac{\partial F}{\partial x}\right)^{n+1}\right] \tag{4.175}$$

It is immediately apparent that we now have a nonlinear problem, and some sort of linearization or iteration technique must be used. Beam and Warming (1976) have suggested that we write

$$F^{n+1} \approx F^n + \left(\frac{\partial F}{\partial u}\right)^n (u^{n+1} - u^n) = F^n + A^n(u^{n+1} - u^n)$$

Thus

$$u_j^{n+1} = u_j^n - \frac{\Delta t}{2}\left\{2\left(\frac{\partial F}{\partial x}\right)^n + \frac{\partial}{\partial x}\left[A\left(u_j^{n+1} - u_j^n\right)\right]\right\}$$

If the x derivatives are replaced by second-order central differences, then

$$-\frac{\Delta t A_{j-1}^n}{4\,\Delta x}u_{j-1}^{n+1} + u_j^{n+1} + \frac{\Delta t A_{j+1}^n}{4\,\Delta x}u_{j+1}^{n+1}$$
$$= -\frac{\Delta t}{\Delta x}\frac{F_{j+1}^n - F_{j-1}^n}{2} - \frac{\Delta t A_{j-1}^n u_{j-1}^n}{4\,\Delta x} + u_j^n + \frac{\Delta t A_{j+1}^n}{4\,\Delta x}u_{j+1}^n \tag{4.176}$$

The Jacobian A has the single element u for Burgers' equation, and a further simplification of the right side is possible. We see that the linearization applied

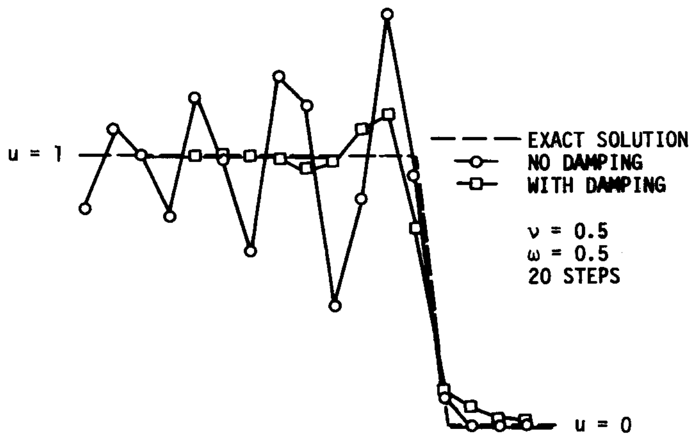

Figure 4.43 Solution of Burgers' equation using Beam-Warming (trapezoidal) method.

by Beam and Warming leads to a linear system of algebraic equations at the next time level. This is a tridiagonal system and may be solved using the Thomas algorithm.

As pointed out in Section 4.1.10, this method is stable for any time step. It should be noted that the roots of the characteristic equation always lie on the unit circle. This is consistent with the fact that the modified equation contains no even derivative terms. Consequently, artificial smoothing is added to the scheme. The usual fourth difference,

$$-\frac{\omega}{8}(u_{j+2}^n - 4u_{j+1}^n + 6u_j^n - 4u_{j-1}^n + u_{j-2}^n) \tag{4.177}$$

may be added to Eq. (4.176), and the formal accuracy of the method is unaltered. According to Beam and Warming, the implicit formula Eq. (4.176) with explicit damping added is stable if

$$0 < \omega \leqslant 1 \tag{4.178}$$

Figure 4.43 shows the results of applying the time-centered implicit formula to a right-moving discontinuity. The solution with no damping is clearly unacceptable. When explicit damping given by Eq. (4.177) is added, better results are obtained.

In addition to the trapezoidal formula just presented, Beam and Warming (1976) developed a three-point-backward implicit and an Euler implicit method as part of a family of techniques. The Beam and Warming version of the Euler implicit scheme follows from the backward Euler formula:

$$u^{n+1} = u^n + \Delta t\left(\frac{\partial u}{\partial t}\right)^{n+1}$$

which for our nonlinear equation becomes

$$u^{n+1} = u^n - \Delta t\left(\frac{\partial F}{\partial x}\right)^{n+1}$$

If the same linearization is applied, we obtain

$$-\frac{\Delta t A_{j-1}^n}{2\,\Delta x}u_{j-1}^{n+1} + u_j^{n+1} + \frac{\Delta t A_{j+1}^n}{2\,\Delta x}u_{j+1}^{n+1}$$
$$= -\frac{\Delta t}{\Delta x}\frac{F_{j+1}^n - F_{j-1}^n}{2} - \frac{\Delta t A_{j-1}^n}{2\,\Delta x}u_{j-1}^n + u_j^n + \frac{\Delta t A_{j+1}^n u_{j+1}^n}{2\,\Delta x} \qquad (4.179)$$

This is again a tridiagonal system and is easily solved. We note that this scheme is unconditionally stable, but damping must be added such as that given in Eq. (4.177), to ensure a usable numerical result.

A simpler form of the implicit algorithms presented in this section can be obtained if they are written in "delta" form. This form uses the increments in the conserved variables and fluxes. In multidimensional problems it has the advantage of providing a steady-state solution that is independent of the time step in problems that possess a steady-state solution. Let us develop the time-centered implicit method using the delta form. Let $\Delta u_j = u_j^{n+1} - u_j^n$. The trapezoidal formula Eq. (4.175) may be written

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

Again a local linearization is used to obtain

$$F_j^{n+1} = F_j^n + A_j^n\,\Delta u_j$$

The final form of the difference equation becomes

$$-\frac{\Delta t A_{j-1}^n}{4\,\Delta x}\Delta u_{j-1} + \Delta u_j + \frac{\Delta t A_{j+1}^n}{4\,\Delta x}\Delta u_{j+1} = -\frac{\Delta t}{2\,\Delta x}(F_{j+1}^n - F_{j-1}^n) \qquad (4.180)$$

This is much simpler than Eq. (4.176). The tridiagonal form is still retained, but the right side does not require the multiplications of the original algorithm. This can be important for systems of equations where the operation count is large. The solution of Eq. (4.180) provides the incremental changes in the unknowns between two time levels. As noted previously, the stability of the delta form is unrestricted, but the usual higher-order damping terms must be added. Results obtained using the delta form for our simple right-moving shock are shown in Fig. 4.44. The solutions with and without damping are essentially identical to those obtained using the expanded form, as should be expected. The delta form of the time-centered implicit scheme is recommended over the expanded version. In problems with time asymptotic solutions, the Δu terms approach zero, and in all cases, matrix multiplications are reduced.

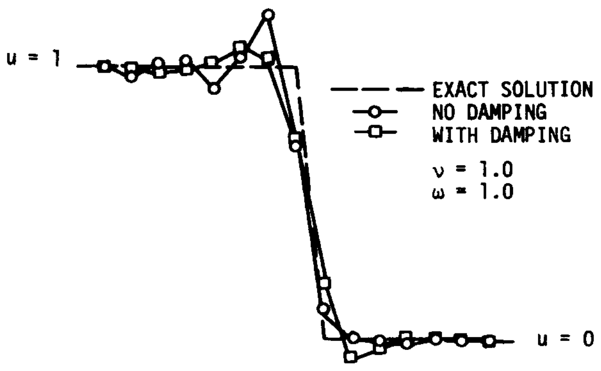

Figure 4.44 Solution for right-moving discontinuity time-centered implicit method, delta form.

Solutions of the inviscid Burgers equation computed with an implicit scheme are generally inferior to those calculated with explicit techniques, and more computational effort per integration step is required. In addition, transient results are usually desired, and the larger step sizes permitted by implicit schemes are not of major significance. When discontinuities are present, results produced with explicit methods are superior to those produced with implicit techniques using central differences. For these reasons, explicit numerical methods are recommended for solving the inviscid Burgers equation.

4.4.8 Godunov Scheme

The numerical methods applied to Burgers' equation in this chapter have used Taylor series to establish appropriate expressions for the values of the dependent variables at the next time level. Differences in the spatial directions were also based upon the requirement of having a certain accuracy using a series approximation. Taylor-series expansions work very well when conditions for convergence of the series are met. In fact, the series will converge everywhere, provided the function that is approximated is sufficiently smooth. In the case of a finite-difference method, we assume that a series expansion is an appropriate means of obtaining a difference approximation and the functions are continuous and have continuous derivatives at least through the order of the difference approximation. This is certainly not true when shock waves or other discontinuities are present. Godunov (1959) recognized this basic problem and proposed to avoid the requirement of differentiability by using a finite-volume approximation in solving the conservation equations and evaluating the flux terms at the cell interfaces by the solution of a Riemann problem. In this section we will describe the Godunov scheme specifically applied to Burgers' equation and see how this method leads to a numerical technique that treats the problem of discontinuous solutions in a very specific way.

Consider the inviscid Burgers equation, Eq. (4.142), and a finite-volume approximation with a control volume as shown in Fig. 4.35. For an explicit method the control volume extends from t to $t + \Delta t$ and from $x - \Delta x/2$ to $x + \Delta x/2$. If a control volume centered at $(j, n + \frac{1}{2})$ is selected, the resulting numerical approximation for the dependent variable may be written

$$\bar{u}_j^{n+1} = \bar{u}_j^n - \frac{\Delta t}{\Delta x}\left[f(u_{j+\frac{1}{2}}) - f(u_{j-\frac{1}{2}})\right] \tag{4.181}$$

In this equation the value of u is averaged over the volume element, i.e.,

$$\bar{u}_j = \frac{1}{\Delta x}\int_{x-\Delta x/2}^{x+\Delta x/2} u(x,t)\,dx$$

and the flux term is the time-averaged value of the flux at the control-volume interface:

$$f = \frac{1}{\Delta t}\int_t^{t+\Delta t} f\,dt$$

The Godunov method solves a local Riemann problem at each cell interface in order to obtain a value of the flux necessary to advance the solution. The Riemann problem specifically for Burgers' equation is

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x}\left(\frac{u^2}{2}\right) = 0 \tag{4.182}$$

with initial conditions

$$u(x,0) = \begin{cases} u_j & x \leqslant 0 \\ u_{j+1} & x > 0 \end{cases}$$

The geometry of the problem is shown in Fig. 4.45. The averaged values of the dependent variable give the appearance of a slab-like variation in distribution,

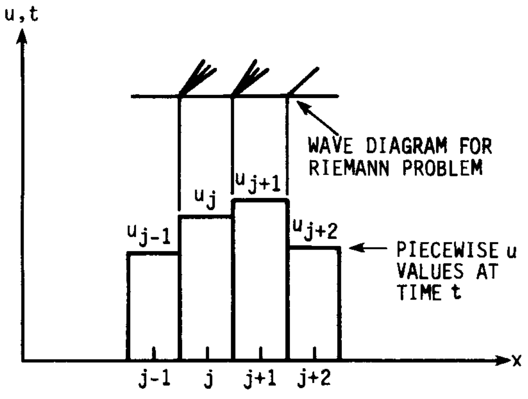
u,t
WAVE DIAGRAM FOR RIEMANN PROBLEM
u j-1
u j
u j+1
u j+2
PIECEWISE u VALUES AT TIME t
x
j-1
j
j+1
j+2

Figure 4.45 Wave diagram for Godunov's method.

leading to a discontinuity at each cell interface. At each cell interface, either a shock or an expansion is initiated and propagates in time. As noted in earlier discussions of solutions for the Riemann problem for Burgers' equation, the solution is self-similar where the similarity variable is x/t. In this sense, the value of x is assumed to be zero at the interface. There are several cases to consider in solving this problem.

With the notation

$$\begin{aligned} c_{j+\frac{1}{2}} &= \left(\frac{dx}{dt}\right)_{j+\frac{1}{2}} \\ &= \frac{u_j + u_{j+1}}{2} \end{aligned}$$

the solution of the Riemann problem for this equation may be written for the following cases:

Case 1: Shock waves

$$u_j > u_{j+1}$$

$$\begin{aligned} u &= \begin{cases} u_j & x/t < c_{j+\frac{1}{2}} \\ u_{j+1} & x/t > c_{j+\frac{1}{2}} \end{cases} \\ f_{j+\frac{1}{2}} &= \begin{cases} \frac{1}{2}u_j^2 & c_{j+\frac{1}{2}} > 0 \\ \frac{1}{2}u_{j+1}^2 & c_{j+\frac{1}{2}} < 0 \end{cases} \end{aligned} \tag{4.183}$$

Case 2: Expansion waves

$$u_j < u_{j+1}$$

$$\begin{aligned} u &= \begin{cases} u_j & x/t < u_j \\ x/t & u_j < x/t < u_{j+1} \\ u_{j+1} & x/t > u_{j+1} \end{cases} \\ f_{j+\frac{1}{2}} &= \begin{cases} 0 & u_j < 0 < u_{j+1} & \\ \frac{1}{2}u_j^2 & c_{j+\frac{1}{2}} > 0 & u_{j+1} > u_j > 0 \\ \frac{1}{2}u_{j+1}^2 & c_{j+\frac{1}{2}} < 0 & u_j < u_{j+1} < 0 \end{cases} \end{aligned} \tag{4.184}$$

An assumption implicit in the above discussion is that the waves from adjacent cells do not interact. This is required in order to write a simple solution for the state variables at the cell boundaries and is assured only if the wave can travel, at most, half of one cell in distance. As a consequence, the stability restriction placed on the Godunov scheme is that

$$\left| u_{\max}\frac{\Delta t}{\Delta x} \right| \leqslant \frac{1}{2}$$

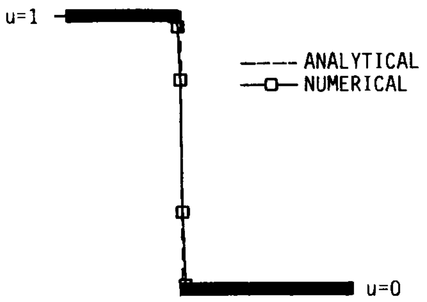

Figure 4.46 Godunov method applied to shock problem.

Using Eq. (4.181), we can now integrate to obtain a solution of Burgers' equation where the flux is evaluated using the solution of the Riemann problem. A typical result is shown in Fig. 4.46. The results of this calculation show that the solution is superior for shock propagation. The results for the case of an expansion wave are nearly as good. It should be remembered that these results are only for a 1-D calculation and that higher-dimensional applications lead to a much more rigorous test of the Godunov idea. In multiple dimensions the flux at the control-volume boundaries is still determined by solving the 1-D Riemann problem and the influence of this 1-D view is shown in Chapter 6.

4.4.9 Roe Scheme

The solution of Burgers' equation using the Godunov method is easily accomplished, and the results are excellent. However, when this method is applied to the solution of the equations that govern fluid flow, it is necessary to employ a computationally inefficient iterative technique. One idea that has been used with good success in computing the solution to nonlinear systems is to solve an approximate Riemann problem rather than having to deal with the exact nonlinear iterative scheme. One of the most popular approximate Riemann solvers was proposed by Roe (1980, 1981). Roe suggested solving the linear problem

$$\frac{\partial u}{\partial t} + \bar{A}\frac{\partial u}{\partial x} = 0 \tag{4.185}$$

where $\bar{A}$ is a constant matrix that is dependent on local conditions. In the case of Burgers' equation, the $\bar{A}$ matrix is, of course, a single scalar element. The $\bar{A}$ matrix is constructed to satisfy what Roe termed property U, which includes the following conditions:

1. For any u_j, u_{j+1},

$$F_{j+1} - F_j = \bar{A}(u_{j+1} - u_j) \tag{4.186}$$

2. When $u = u_j = u_{j+1}$, then

$$\bar{A}(u_j, u_{j+1}) = \bar{A}(u, u) = \frac{\partial F}{\partial u} = u \tag{4.187}$$

The first of these conditions ensures that the correct jump is recovered when a discontinuity is encountered. The second condition provides that in smooth regions, the Jacobian, in this case the nonlinear wave speed, reduces to the correct value.

In applying Roe's scheme to Burgers' equation, $\bar{u}$ is a constant and denotes the averaged value of $\bar{A}$. We then consider the linear problem

$$\frac{\partial u}{\partial t} + \bar{u}\frac{\partial u}{\partial x} = 0 \tag{4.188}$$

where the appropriate value of $\bar{u}$ for cells j and $j + 1$ is determined from the first of the requirements noted above. This gives

$$\bar{u} = \bar{u}_{j+\frac{1}{2}} = \frac{F_{j+1} - F_j}{u_{j+1} - u_j} \tag{4.189}$$

which for Burgers' equation reduces to

$$\bar{u}_{j+\frac{1}{2}} = \begin{cases} \dfrac{u_j + u_{j+1}}{2} & u_j \neq u_{j+1} \\ u_j & u_j = u_{j+1} \end{cases} \tag{4.190}$$

With this information, the numerical flux can be developed. Since the original Riemann problem has been reduced to a linearized form, the Rankine-Hugoniot relations directly provide the relationship between the jump in the flux and the jump in the dependent variable u across a wave, i.e.,

$$F_{j+1} - F_j = \bar{u}_{j+\frac{1}{2}}(u_{j+1} - u_j) \tag{4.191}$$

As a consequence, this approximate Riemann solver only recognizes discontinuities and cannot distinguish between expansions that are discontinuous and shock waves. An alteration will be added below to correct for this fact.

Consider the approximate Riemann problem for Burgers' equation as depicted in Fig. 4.47. For this case, we see that a single wave emanates from the cell interface. This single wave travels either in the positive or negative direction, depending upon $\bar{u}_{j+\frac{1}{2}} = dx/dt$. Utilizing the definition of the jump across this wave, we may write either

$$f_{j+\frac{1}{2}} - F_j = \bar{u}^-_{j+\frac{1}{2}}(u_{j+1} - u_j)$$

or

$$F_{j+1} - f_{j+\frac{1}{2}} = \bar{u}^+_{j+\frac{1}{2}}(u_{j+1} - u_j)$$

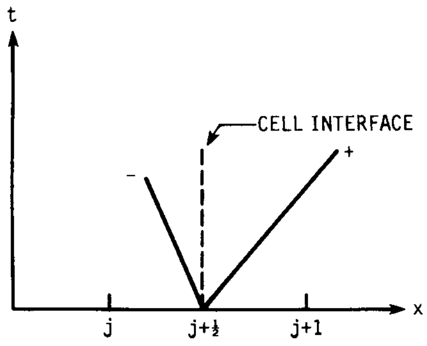

Figure 4.47 Wave diagram for Roe scheme applied to Burgers' equation.

The numerical flux may then be written in the symmetric form

$$f_{j+\frac{1}{2}} = \frac{F_j + F_{j+1}}{2} + \frac{1}{2}\left(\bar{u}^-_{j+\frac{1}{2}} - \bar{u}^+_{j+\frac{1}{2}}\right)(u_{j+1} - u_j)$$

Consider the individual contributions that occur when the wave travels either from the right or left. If each case is evaluated separately, the numerical flux can be represented in a single equation of the form

$$f_{j+\frac{1}{2}} = \frac{F_j + F_{j+1}}{2} - \frac{1}{2}|\bar{u}_{j+\frac{1}{2}}|(u_{j+1} - u_j) \tag{4.192}$$

Figure 4.48 illustrates the calculation of a propagating discontinuity using Roe's scheme, and the results show excellent agreement with the known analytical solution.

Roe's scheme has the stability bound common to explicit methods, i.e., the Courant number must be less than 1. The Roe formulation propagates the difference in dependent variables between two points as if a discontinuity existed between these points. A consequence of this is that expansion shocks that are nonphysical may appear. In Burgers' equation this is a problem when $\bar{u}_{j+\frac{1}{2}}$ vanishes. This is referred to as a sonic transition. The classical example is a solution computed using Roe's scheme with initial data

$$u = \begin{cases} -1 & 0 \leqslant x \leqslant x_0 \\ +1 & x_0 < x \leqslant L \end{cases}$$

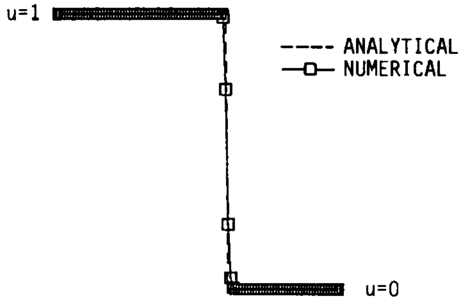

Figure 4.48 Discontinuity propagated using Roe's scheme.

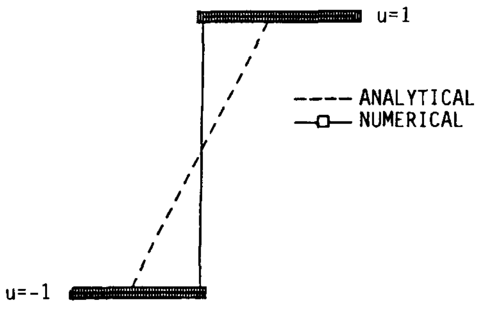

Figure 4.49 Expansion shock using Roe's scheme.

The analytic solution is a centered expansion about $x = x_0$. The solution computed using the Roe scheme is shown in Fig. 4.49. The initial data are faithfully reproduced, showing an incorrect stationary expansion shock.

This nonphysical behavior is due to the fact that the scheme cannot distinguish between an expansion shock and a compression shock. Each is a valid solution for this formulation. The existence of the expansion shock is said to violate the entropy condition, allowing incorrect physical behavior. Oleinik (1957) and, later, Lax (1973) developed conditions that must be satisfied by discontinuous solutions of hyperbolic equations. The simplest statement of this condition applied to Burgers' equation may be written

$$u_R < \left(\frac{dx}{dt}\right)_{x_0} < u_L \tag{4.193}$$

where u_R and u_L are the values to the right and the left of the discontinuity. The initial data for the expansion violate this condition, eliminating expansion shocks. In the example just cited, no information is present regarding the sonic transition ($u = 0$), and Roe's scheme interprets the two-point expansion in the initial data as a discontinuity. Unlike the Godunov scheme, where admissible solutions incorporate the correct physical behavior, a modification of the numerics is necessary. A number of techniques to accomplish this have been proposed. Harten and Hyman (1983) proposed the following modification of $\bar{u}_{j+\frac{1}{2}}$. Let

$$\epsilon = \max\left(0, \frac{u_{j+1} - u_j}{2}\right)$$

then

$$\bar{u}_{j+\frac{1}{2}} = \begin{cases} \bar{u}_{j+\frac{1}{2}} & \bar{u}_{j+\frac{1}{2}} \geqslant \epsilon \\ \epsilon & \bar{u}_{j+\frac{1}{2}} < \epsilon \end{cases} \tag{4.194}$$

The compression case ($\epsilon = 0$) uses the unaltered scheme to propagate discontinuities, while the case of an expansion fan [$\epsilon = (u_{j+1} - u_j)/2$] requires the modification. Figure 4.50 shows the results computed for an expansion $(-1, 1)$ using Roe's scheme with the entropy fix included. The agreement between the numerical and analytical solutions is good and is approximately

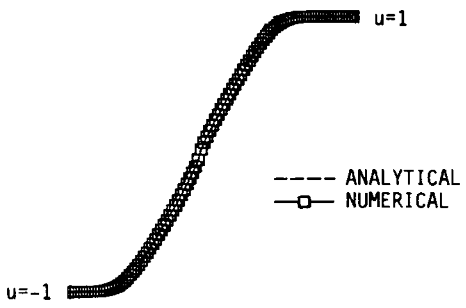

Figure 4.50 Expansion using Roe's scheme with entropy correction.

what one would expect for a first-order method. The modification of Roe's scheme may be viewed as a way of adding an appropriate amount of dissipation to the basic method when the dissipation at the sonic transition goes to zero. The alteration given here is equivalent to the introduction of a small expansion in the approximate Riemann solution when an expansion appears through a sonic point.

4.4.10 Enquist-Osher Scheme

In the previous section, Roe's scheme was seen to replace both shock and expansion waves by discontinuities. In order to correctly reproduce expansions and not produce expansion shocks, a modification of the numerical flux was introduced to correct the physical behavior. The scheme introduced by Enquist and Osher (1980, 1981) treats the change in the dependent variables across waves as a continuous transition in state space and produces a method that is monotone and conservative and that correctly treats both shock and expansion waves. In this scheme, the shock discontinuities in the exact Riemann solution are replaced by smooth compression waves. The discontinuities are resolved with at most two interior points. The stability bound of this approximate Riemann solver is the usual Courant-Friedrichs-Lewy (CFL) condition for explicit methods.

The flux at any location is written as the sum of contributions from crossing waves with slopes in (x, t) space that are positive and negative. Thus

$$F = F^+ + F^- \tag{4.195}$$

where the individual components are obtained by an integration in phase space defined by

$$F^+(u) = \int_0^u \mu(\tau)F'(\tau)\,d\tau \tag{4.196a}$$

$$F^-(u) = \int_0^u [1 - \mu(\tau)]F'(\tau)\,d\tau \tag{4.196b}$$

and the switch is defined as

$$\mu(\tau) = \begin{cases} 1 & A = \partial F/\partial u \geqslant 0 \\ 0 & A < 0 \end{cases}$$

In this evaluation of the flux, the sonic transition is specifically identified through the switch in the integration. With this notation, we may write the cell interface flux by starting at either side, as we did in Roe's scheme:

$$f_{j+\frac{1}{2}} = F_j + \int_{u_j}^{u_{j+1}} [1 - \mu(\tau)] F'(\tau)\, d\tau \tag{4.197a}$$

$$f_{j+\frac{1}{2}} = F_{j+1} - \int_{u_j}^{u_{j+1}} \mu(\tau) F'(\tau)\, d\tau \tag{4.197b}$$

The numerical flux is usually written in the symmetric form

$$f_{j+\frac{1}{2}} = \frac{F_j + F_{j+1}}{2} - \frac{1}{2}\int_{u_j}^{u_{j+1}} |u|\, du \tag{4.198}$$

where the flux derivative representing the Jacobian and the switch have been replaced by $|u|$ in the integral. With this definition of the interface flux, the correct form for Burgers' equation may now be written

$$f_{j+\frac{1}{2}} = \begin{cases} u_{j+1}^2/2 & u_j, u_{j+1} < 0 \\ u_j^2/2, & u_j, u_{j+1} > 0 \end{cases} \tag{4.199}$$

If u_j and u_{j+1} are of opposite sign, then

$$f_{j+\frac{1}{2}} = \begin{cases} 0 & u_j < 0 < u_{j+1} \\ \left(u_j^2 + u_{j+1}^2\right)/2 & u_j > 0 > u_{j+1} \end{cases} \tag{4.200}$$

While the Godunov scheme treats shocks as discontinuities, the Enquist-Osher scheme replaces shocks by what van Leer (1984) describes as overturned centered compression waves. Discontinuities are excluded owing to the smooth transitions in the phase space integrals. The treatment of sonic transitions leads to a small deviation in the slope of expansions when such points are present.

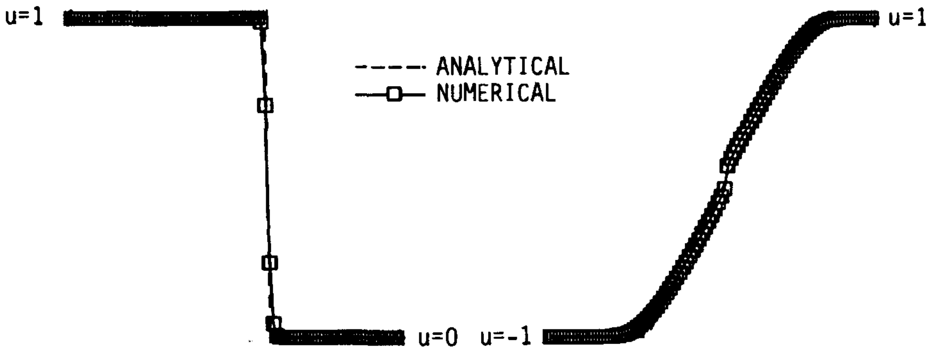

Figure 4.51 Results using the Enquist-Osher scheme. (a) Shock wave. (b) Expansion wave.

Such a case is shown in Fig. 4.51, where the results produced by applying the Enquist-Osher scheme to both a propagating discontinuity and expansion are depicted. According to Chakravarthy and Osher (1985), this small deviation at the sonic location in the expansion is bounded and does not lead to erroneous results. The results shown in Fig. 4.51 were computed using the explicit scheme given in Eq. (4.154). For either the Roe or Enquist-Osher scheme, the stability bound is the usual requirement of Courant number less than 1. Since only an approximate Riemann problem is considered, the smaller limit imposed by the Godunov method is avoided.

4.4.11 Higher-Order Upwind Schemes

In applying finite-difference methods for the solution of PDEs, a higher-order approximation is obtained by introducing more points in the stencil. For upwind schemes a first derivative can be approximated to first order using two points, while three points are necessary for a second-order expression. In the Riemann or approximate Riemann solvers, a higher-order approximation must be interpreted in terms of flux values at control-volume boundaries.

In the original Godunov approach, state variables were assumed to be constant in control volumes. For the first-order schemes, this assumption was sufficient. Van Leer (1979) extended this idea by assuming that the state variables, as projected on each cell, can have a variation. The cell-averaged values from the Riemann solution are used to reconstruct the assumed variation in each control volume. This idea essentially leads to a higher-order extrapolation of the flux or state variables at the cell boundaries. For the variable extrapolation approach, van Leer coined the term "monotone upstream-centered schemes for conservation laws." This is referred to as the MUSCL approach, or sometimes MUSCL differencing. It should be noted that the flux can be used in the formulas for extrapolation to the cell boundaries. In this case, the flux terms do not need to be recalculated, since the extrapolation directly provides the needed information.

For the first-order Godunov scheme, the average value of the dependent variable is set equal to the constant value of that variable in the cell. For higher-order schemes, the values assigned to each cell are the averages over each cell. Depending on the accuracy desired, the dependent variables may be curve fit with a polynomial of arbitrary order in each cell. Consider the piecewise linear representation shown in Fig. 4.52. For cell j, the expression for the variation of u is a function of position in the cell, and the average value of u is assigned to the cell. In order to arrive at an extrapolation for the control-volume boundary value, a Taylor-series representation is employed:

$$u_{j+\frac{1}{2}} = u_j + \frac{\Delta x}{2}(u_x)_j + \frac{(\Delta x)^2}{8}(u_{xx})_j + \cdots \qquad (4.201)$$

where the derivatives are evaluated by differences using cell averages.

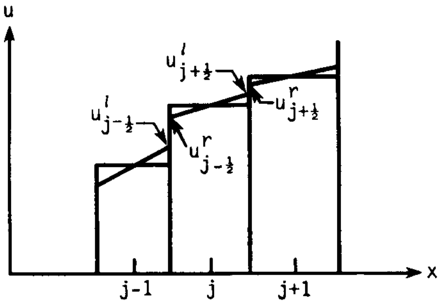

Figure 4.52 Assumed linear variation in each cell.

For cell j the expressions for the independent variables will have a right and left side. The usual expressions for the right and left extrapolations are

$$u^l_{j+\frac{1}{2}} = u_j + \frac{1-\kappa}{4}\,\delta u_{j-\frac{1}{2}} + \frac{1+\kappa}{4}\,\delta u_{j+\frac{1}{2}} \tag{4.202}$$

$$u^r_{j+\frac{1}{2}} = u_{j+1} - \frac{1+\kappa}{4}\,\delta u_{j+\frac{1}{2}} - \frac{1-\kappa}{4}\,\delta u_{j+\frac{3}{2}} \tag{4.203}$$

These expressions are Taylor-series representations for the state variables, with difference approximations for the derivatives. When fluxes are used instead of the independent variables, the fluxes are introduced in the extrapolation to the cell boundaries. This has the advantage of providing the flux directly, and the flux does not require reformulation from the independent variable extrapolation. Care must be exercised using this approach because it may not provide a solution that is as good as that using the independent variable extrapolation and subsequent reformulation of the fluxes at the cell boundaries (Anderson et al., 1987). In the extrapolation the value of κ determines the type of method (van Leer, 1979; Yee, 1989). For example,

$$\kappa = \begin{cases} -1 & \text{upwind scheme} \\ 0 & \text{Fromm's (1968) method} \\ 1 & \text{central difference} \end{cases} \tag{4.204}$$

It is of value to consider the case of $\kappa = -1$ and the relationship of this upwind extrapolation to an upwind finite-difference approximation. The second-order upwind scheme studied in Section 4.1.9 can be recovered using the one-sided extrapolation with $\kappa = -1$ (see Prob. 4.60). The values for the extrapolated variables on both sides of the interface become

$$u^l_{j+\frac{1}{2}} = u_j + \frac{1}{2}\,\delta u_{j-\frac{1}{2}}$$

$$u^r_{j+\frac{1}{2}} = u_{j+1} - \frac{1}{2}\,\delta u_{j+\frac{3}{2}}$$

while for $\kappa = 1$, a central approximation becomes

$$u^l_{j+\frac{1}{2}} = u_j + \frac{1}{2}\,\delta u_{j+\frac{1}{2}}$$

$$u^r_{j-\frac{1}{2}} = u_j - \frac{1}{2}\,\delta u_{j-\frac{1}{2}}$$

Both the upwind and central differences are linear approximations to the cell boundary values of u. The difference between them is the different approximation to the slope. The interface values for the $\kappa = 1$ case reduce to the same average between adjacent cells, indicating the central-difference scheme.

Second-order flux values are formed from the dependent variables by replacement. Consider Roe's scheme, where the first-order flux was given by Eq. (4.192). Functionally, we may write

$$f^{(1)}_{j+\frac{1}{2},Roe} = f(u_j, u_{j+1}) \tag{4.205}$$

The second-order flux is obtained by replacing u_j, u_{j+1} in the flux equation with right and left values:

$$f^{(2)}_{j+\frac{1}{2},Roe} = f\left(u^l_{j+\frac{1}{2}}, u^r_{j+\frac{1}{2}}\right) \tag{4.206}$$

The second-order Roe flux may then be written

$$f^{(2)}_{j+\frac{1}{2},Roe} = \frac{1}{2}\left[F\left(u^l_{j+\frac{1}{2}}\right) + F\left(u^r_{j+\frac{1}{2}}\right) - \left|\overline{u_{j+\frac{1}{2}}}\right|\left(u^r_{j+\frac{1}{2}} - u^l_{j+\frac{1}{2}}\right)\right] \tag{4.207}$$

where

$$\overline{u_{j+\frac{1}{2}}} = \frac{F\left(u^r_{j+\frac{1}{2}}\right) - F\left(u^l_{j+\frac{1}{2}}\right)}{u^r_{j+\frac{1}{2}} - u^l_{j+\frac{1}{2}}} \tag{4.208}$$

and is defined in the same manner as the equivalent term in the first-order flux.

An explicit predictor-corrector method similar to the two-step Lax-Wendroff method, producing a second-order solution in space and time, may now be constructed. The predictor step is

$$u^{n+\frac{1}{2}}_j = u^n_j - \frac{\Delta t}{2\,\Delta x}\left(f^n_{j+\frac{1}{2}} - f^n_{j-\frac{1}{2}}\right) \tag{4.209}$$

where f represents a first-order flux. Variable extrapolations are then employed to obtain the left and right interface values of the dependent variables, yielding

$$u^{l,n+\frac{1}{2}}_{j+\frac{1}{2}} = u^{n+\frac{1}{2}}_j + \frac{1-\kappa}{4}\,\delta u^{n+\frac{1}{2}}_{j-\frac{1}{2}} + \frac{1+\kappa}{4}\,\delta u^{n+\frac{1}{2}}_{j+\frac{1}{2}} \tag{4.210}$$

$$u^{r,n+\frac{1}{2}}_{j+\frac{1}{2}} = u^{n+\frac{1}{2}}_j - \frac{1+\kappa}{4}\,\delta u^{n+\frac{1}{2}}_{j+\frac{1}{2}} - \frac{1-\kappa}{4}\,\delta u^{n+\frac{1}{2}}_{j+\frac{3}{2}} \tag{4.211}$$

The corrector step is written

$$u_j^{n+1} = u_j^n - \frac{\Delta t}{\Delta x}\left(f_{j+\frac{1}{2}}^{n+\frac{1}{2}} - f_{j-\frac{1}{2}}^{n+\frac{1}{2}}\right) \tag{4.212}$$

where

$$f_{j+\frac{1}{2}}^{n+\frac{1}{2}} = f\left(u_{j+\frac{1}{2}}^{l,n+\frac{1}{2}}, u_{j+\frac{1}{2}}^{r,n+\frac{1}{2}}\right) \tag{4.213}$$

The stability of this scheme depends on the extrapolation used to construct the flux terms. Upwind schemes of higher order have a less restrictive stability constraint than central schemes, as we have seen in previous sections. A Courant number less than 1 is necessary for the central scheme, while the less restrictive value of 2 may be used for the upwind case. Figure 4.53 shows the results of applying this scheme to Burgers' equation for a shock propagating to the right. As is characteristic of second-order schemes, oscillations are present in the calculation. It is desirable to eliminate these wiggles and maintain the monotone character of the initial profile. In Section 4.4.1 the concept of a monotone solution was introduced when the Godunov theorem was presented. It is apparent that the nonmonotone behavior of the present solutions must be modified in some way if a better result is to be obtained. It should also be apparent that the behavior of the present solution obtained with the second-order upwind scheme does little to suggest an improvement over a central-difference scheme. In both cases the solution oscillates, and some way of controlling this behavior is desired. In the case of the upwind scheme, one can argue that the physics of the problem is more closely represented, even though the solution does not show any marked improvement over the central schemes. Improvements in the results can be achieved with the introduction of limiters that avoid the overshoots and undershoots shown in typical solutions.

4.4.12 TVD Schemes

The results of the previous section showed that even though upwind schemes appear to account for physics in a more appropriate way, as compared to central-difference schemes, higher-order methods share the same deficiencies

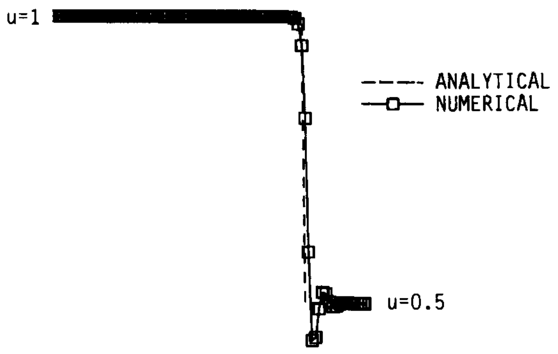

Figure 4.53 Second-order Roe scheme applied to Burgers' equation without limiting.

when discontinuities are encountered. The problem can be demonstrated computationally by starting with a monotone profile for a compression wave. As the compression front steepens, higher-order methods produce numerical solutions with new extrema. Consequently, oscillations are introduced that are undesirable. This numerical experiment (see Prob. 4.64) is a computational verification of Godunov's theorem given in Section 4.4.1.

Lax (1973) showed that for scalar conservation laws of the form given by Burgers' equation, Eq. (4.142), the total variation of physically possible solutions does not increase in time. The total variation (TV) is given by

$$\mathrm{TV} = \int \left| \frac{\partial u}{\partial x} \right| dx \tag{4.214}$$

and the total variation for the discrete case is

$$\mathrm{TV}(u) = \sum_j |u_{j+1} - u_j| \tag{4.215}$$

A numerical method is said to be total variation diminishing, or TVD, if

$$\mathrm{TV}(u^{n+1}) \leqslant \mathrm{TV}(u^n) \tag{4.216}$$

Harten (1983) proved that

1. a monotone scheme is TVD, and
2. a TVD scheme is monotonicity preserving.

Thus, if higher-order TVD schemes can be constructed, these schemes will be monotonicity preserving.

The central idea in constructing a TVD scheme is to attempt to develop a higher-order method that will avoid oscillations and exhibit properties similar to those of a monotone scheme. For such schemes the solution is first order near discontinuities and higher order in smooth regions. The transition to higher order is accomplished by the use of slope limiters on the dependent variables or flux limiters. Boris and Book (1973) developed the first nonlinear limiting by adding a limited difference between the first- and second-order fluxes to prevent oscillations associated with second-order schemes. The work of van Leer (1974) was based upon limiting the extrapolation of the dependent variables to the cell boundaries in order to prevent overshoots in the solution. Sweby (1984) and Roe (1985) developed limiters based on the TVD models of the Lax-Wendroff scheme, while Harten (1978) applied the concept of a modified flux to achieve the same effect. It is important to understand that methods that are either central differenced or upwind can be made TVD by suitable modification with limiters. When such higher-order schemes are constructed, a central-difference scheme may appear to be upwind in many regions, and upwind schemes may sometimes appear to be similar to a central-difference scheme.

The problem of constructing a solution of Burgers' equation that does not include unwanted oscillations when MUSCL differencing is employed can now be considered. In this case, unwanted oscillations can be avoided if the slopes of the variables used in the extrapolations are limited in such a way that the end point values do not create a new maximum or minimum. For the case of $u_{j-1} < u_{j+1}$, it is desired that

$$u^{r}_{j-\frac{1}{2}} \geqslant u_{j-1} \tag{4.217a}$$

$$u^{l}_{j+\frac{1}{2}} \leqslant u_{j+1} \tag{4.217b}$$

For this specification of the extrapolated values of the dependent variables, an easy control is devised by the introduction of slope limiters in the following way:

$$u^{l}_{j+\frac{1}{2}} = u_j + \frac{1-\kappa}{4}\overline{\delta^{+}u}_{j-\frac{1}{2}} + \frac{1+\kappa}{4}\overline{\delta^{-}u}_{j+\frac{1}{2}} \tag{4.218a}$$

$$u^{r}_{j+\frac{1}{2}} = u_{j+1} - \frac{1+\kappa}{4}\overline{\delta^{+}u}_{j+\frac{1}{2}} - \frac{1-\kappa}{4}\overline{\delta^{-}u}_{j+\frac{3}{2}} \tag{4.218b}$$

In these equations, the quantities $\overline{\delta^{\pm}u_j}$ are limited slopes. A number of different limiters may be used. One choice is the minmod limiter defined by

$$\overline{\delta^{-}u}_{j+\frac{1}{2}} = \text{minmod}(\delta u_{j+\frac{1}{2}}, \omega\, \delta u_{j-\frac{1}{2}}) \tag{4.219a}$$

$$\overline{\delta^{+}u}_{j+\frac{1}{2}} = \text{minmod}(\delta u_{j+\frac{1}{2}}, \omega\, \delta u_{j+\frac{3}{2}}) \tag{4.219b}$$

The minmod limiter is used extensively in TVD numerical methods. This is a function that selects the smallest number from a set when all have the same sign but is zero if they have different signs. For example,

$$\text{minmod}(x,y) = \begin{cases} x & \text{if} \quad |x| < |y| \quad \text{and} \quad xy > 0 \\ y & \text{if} \quad |x| > |y| \quad \text{and} \quad xy > 0 \\ 0 & \text{if} \quad \quad \quad \quad \quad \quad \quad \quad xy < 0 \end{cases} \tag{4.220}$$

or written in another form,

$$\text{minmod}(x, \omega y) = \text{sgn}(x)\max\{0, \min[|x|, \omega y\, \text{sgn}(x)]\} \tag{4.221}$$

with the limits on ω given as

$$1 \leqslant \omega \leqslant \frac{3-\kappa}{1-\kappa} \tag{4.222}$$

and the values of κ not equal to 1. The usual notation of sgn represents the sign of the argument indicated. The values of the limited slopes in these expressions are selected to satisfy the end point conditions given by Eqs. (4.217). The higher-order fluxes are computed as indicated in the previous section, and the slopes are limited to prevent the occurrence of nonphysical oscillations. The same calculation presented in Fig. 4.53 without the use of limiters is repeated in Fig. 4.54 including limiting. The improvement in the solution is apparent, and the MUSCL approach does provide a good way of calculating a higher-order solution that is dramatically better when using limiters.

In general, higher-order schemes that are either upwind or central differenced can be modified to have the TVD property, thus avoiding the

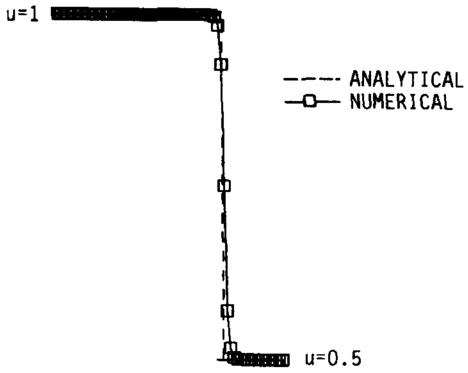

Figure 4.54 Second-order Roe scheme applied to Burgers' equation using the minmod limiter.

deficiencies of the classical shock-capturing numerical methods. Insight into the process of designing limiters is gained by utilizing the general class of numerical schemes of Yee (1987) and following the analysis of Harten (1984). Consider the one-parameter family of difference schemes written in conservative form:

$$u_j^{n+1} + \frac{\Delta t}{\Delta x}\theta\left(f_{j+\frac{1}{2}}^{n+1} - f_{j-\frac{1}{2}}^{n+1}\right) = u_j^n - \frac{\Delta t}{\Delta x}(1-\theta)\left(f_{j+\frac{1}{2}}^{n} - f_{j-\frac{1}{2}}^{n}\right) \tag{4.223}$$

where θ varies between 0 and 1 and

$$f_{j+\frac{1}{2}}^n = f(u_{j-1}^n, u_j^n, u_{j+1}^n, u_{j+2}^n) \tag{4.224}$$

This scheme represents several variations. If $\theta = 0$, this is an explicit scheme, while it is implicit if $\theta \neq 0$. If $\theta = \frac{1}{2}$, the differencing is trapezoidal and the technique is second order in time. This one-parameter family of methods can be more easily recognized if the average flux is defined as

$$\bar{f}_{j+\frac{1}{2}} = (1-\theta)f_{j+\frac{1}{2}}^n + \theta f_{j+\frac{1}{2}}^{n+1}$$

With this notation, Eq. (4.223) may be written in the familiar form

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

The average flux is consistent with the conservation statement, in that

$$\bar{f}(u,u,u,u) = f(u)$$

The general scheme may be written in terms of implicit and explicit operators as

$$L(u^{n+1}) = R(u^n) \tag{4.225}$$

where the L and R represent the operators defined by

$$L(u_j) = u_j + \theta\frac{\Delta t}{\Delta x}(f_{j+\frac{1}{2}} - f_{j-\frac{1}{2}}) \tag{4.226}$$

$$R(u_j) = u_j - \frac{\Delta t}{\Delta x}(1 - \theta)(f_{j+\frac{1}{2}} - f_{j-\frac{1}{2}}) \tag{4.227}$$

In order to ensure that the numerical schemes represented by Eq. (4.223) are TVD, it is required that

$$\text{TV}(u^{n+1}) \leqslant \text{TV}(u^n)$$

Harten (1984) showed that the sufficient conditions for this are

$$\text{TV}[R(u^n)] \leqslant \text{TV}(u^n) \tag{4.228a}$$

$$\text{TV}[L(u^{n+1})] \geqslant \text{TV}(u^{n+1}) \tag{4.228b}$$

Rewriting Eq. (4.223) in the form

$$u_j^{n+1} - \frac{\Delta t}{\Delta x}\theta\left(\overline{C}_{j+\frac{1}{2}}^{-}\,\delta u_{j+\frac{1}{2}} - \overline{C}_{j-\frac{1}{2}}^{+}\,\delta u_{j-\frac{1}{2}}\right)^{n+1}$$
$$= u_j^n + \frac{\Delta t}{\Delta x}(1 - \theta)\left(\overline{C}_{j+\frac{1}{2}}^{-}\,\delta u_{j+\frac{1}{2}} - \overline{C}_{j-\frac{1}{2}}^{+}\,\delta u_{j-\frac{1}{2}}\right)^{n} \tag{4.229}$$

Harten proved that the sufficient conditions are

$$C_{j+\frac{1}{2}}^{\pm} = \frac{\Delta t}{\Delta x}(1 - \theta)\overline{C}_{j+\frac{1}{2}}^{\pm} \geqslant 0 \tag{4.230}$$

and

$$C_{j+\frac{1}{2}}^{+} + C_{j+\frac{1}{2}}^{-} = \frac{\Delta t}{\Delta x}(1 - \theta)\left(\overline{C}_{j+\frac{1}{2}}^{+} + \overline{C}_{j+\frac{1}{2}}^{-}\right) \leqslant 1 \tag{4.231}$$

with

$$-\infty < C \leqslant -\frac{\Delta t}{\Delta x}\theta\overline{C}_{j+\frac{1}{2}}^{\pm} \leqslant 0 \tag{4.232}$$

where C is some positive constant. For values of $\theta = 0$ and 1, the method is first order and the TVD conditions are satisfied. In fact, the Lax scheme and all first-order upwind schemes can be shown to have the TVD property (see Prob. 4.65). If the value of θ is taken to be 0.5, the method is a trapezoidal scheme, and the TVD requirements are not satisfied. Consequently, the scheme must be modified with the limiter concept to avoid oscillations. The construction of appropriate limiters is possible using the requirements given in the preceding equations for guidance.

Jameson and Lax (1984) generalized the idea of TVD schemes for multipoint methods. In the case of higher-order schemes, the terms $C^{\pm}$ are written as

$$C_{j+\frac{1}{2}}^{-} = C^{-}(u_{j+2}, u_{j+1}, u_j, u_{j-1}) \tag{4.233a}$$

$$C_{j-\frac{1}{2}}^{+} = C^{+}(u_{j+1}, u_j, u_{j-1}, u_{j-2}) \tag{4.233b}$$

The general explicit scheme

$$u_j^{n+1} = u_j^n + \frac{\Delta t}{\Delta x} \sum_{k=1}^{J} \left(C_{j+k-\frac{1}{2}}^{-(k)} \, \delta u_{j+k-\frac{1}{2}} - C_{j-k+\frac{1}{2}}^{+(k)} \, \delta u_{j-k+\frac{1}{2}} \right)^n \tag{4.234}$$

satisfies the TVD conditions when

$$C_{j+\frac{1}{2}}^{+(1)} \geqslant C_{j+\frac{1}{2}}^{+(2)} \geqslant \cdots \geqslant C_{j+\frac{1}{2}}^{+(k)} \geqslant 0 \tag{4.235a}$$

$$C_{j+\frac{1}{2}}^{-(1)} \geqslant C_{j+\frac{1}{2}}^{-(2)} \geqslant \cdots \geqslant C_{j+\frac{1}{2}}^{-(k)} \geqslant 0 \tag{4.235b}$$

$$\frac{\Delta t}{\Delta x}\left(C_{j+\frac{1}{2}}^{+(1)} + C_{j+\frac{1}{2}}^{-(1)} \right) \leqslant 1 \tag{4.235c}$$

which is the same condition derived by Harten for the three-point schemes. The general implicit scheme

$$\begin{aligned} u_j^{n+1} &- \frac{\Delta t}{\Delta x} \sum_{k=1}^{J} B_{j+k-\frac{1}{2}}^{-(k)} \delta u_{j+k-\frac{1}{2}} - B_{j-k+\frac{1}{2}}^{+(k)} \, \delta u_{j-k+\frac{1}{2}} \Bigg)^{n+1} \\ &= u_j^n + \frac{\Delta t}{\Delta x} \sum_{k=1}^{J} \left(C_{j+k-\frac{1}{2}}^{-(k)} \, \delta u_{j+k-\frac{1}{2}} - C_{j-k+\frac{1}{2}}^{+(k)} \, \delta u_{j-k+\frac{1}{2}} \right)^n \end{aligned} \tag{4.236}$$

satisfies the TVD conditions if and only if the coefficients of the implicit and explicit operators obey the expressions

$$\begin{aligned} B_{j+\frac{1}{2}}^{+(1)} \geqslant B_{j+\frac{1}{2}}^{+(2)} \geqslant \cdots \geqslant B_{j+\frac{1}{2}}^{+(k)} \geqslant 0 \\ B_{j+\frac{1}{2}}^{-(1)} \geqslant B_{j+\frac{1}{2}}^{-(2)} \geqslant \cdots \geqslant B_{j+\frac{1}{2}}^{-(k)} \geqslant 0 \end{aligned} \tag{4.237}$$

and

$$\begin{aligned} C_{j+\frac{1}{2}}^{+(1)} \geqslant C_{j+\frac{1}{2}}^{+(2)} \geqslant \cdots \geqslant C_{j+\frac{1}{2}}^{+(k)} \geqslant 0 \\ C_{j+\frac{1}{2}}^{-(1)} \geqslant C_{j+\frac{1}{2}}^{-(2)} \geqslant \cdots \geqslant C_{j+\frac{1}{2}}^{-(k)} \geqslant 0 \end{aligned} \tag{4.238}$$

This condition is also consistent with the earlier development for the general three-point scheme.

Armed with the information gleaned from the TVD conditions, we can now test the second-order upwind scheme (Section 4.1.9) to determine if it is a TVD method. The second-order upwind method applied to the first-order wave equation yields

$$u_j^{n+1} = u_j^n - \nu(u_j^n - u_{j-1}^n) + \frac{1}{2}\nu(\nu - 1)(u_j^n - 2u_{j-1}^n + u_{j-2}^n) \tag{4.239}$$

This may be written as

$$u_j^{n+1} = u_j^n - \frac{\nu}{2}(3 - \nu)\, \delta u_{j-\frac{1}{2}}^n - \frac{\nu}{2}(\nu - 1)\, \delta u_{j-\frac{3}{2}}^n \tag{4.240}$$

showing that

$$C_{j-\frac{1}{2}}^{+(1)} = \frac{c}{2}(3 - \nu) \qquad C_{j-\frac{3}{2}}^{+(2)} = \frac{c}{2}(\nu - 1) \tag{4.241}$$

In this case the coefficient $C^{+(2)}_{j-3/2}$ has the wrong sign when the CFL condition is satisfied, and oscillations will occur at discontinuities.

As another example, let us test the Lax-Wendroff scheme, Eq. (4.44), to see if it satisfies the TVD condition. This scheme is written

$$u_j^{n+1} = u_j^n - \frac{\nu}{2}(u_{j+1}^n - u_{j-1}^n) + \frac{\nu^2}{2}(u_{j+1}^n - 2u_j^n + u_{j-1}^n) \tag{4.242}$$

or equivalently,

$$u_j^{n+1} = u_j^n - \frac{\nu}{2}(1-\nu)\,\delta u_{j+\frac{1}{2}}^n - \frac{\nu}{2}(1+\nu)\,\delta u_{j-\frac{1}{2}}^n \tag{4.243}$$

For this example,

$$C^+_{j-\frac{1}{2}} = \frac{c}{2}(1+\nu) \qquad C^-_{j+\frac{1}{2}} = -\frac{c}{2}(1-\nu) \tag{4.244}$$

which indicates that the TVD conditions are not satisfied in the range of Courant numbers where this scheme is stable.

The idea of limiting to control oscillations is demonstrated effectively by considering the second-order upwind scheme in the form of Eq. (4.239). This may be rewritten as

$$u_j^{n+1} = u_j^n - \nu\,\delta u_{j-\frac{1}{2}}^n + \frac{\nu}{2}(\nu-1)\left(\delta u_{j-\frac{1}{2}}^n - \delta u_{j-\frac{3}{2}}^n\right) \tag{4.245}$$

The terms following the positive sign represent contributions from the second-order corrections to the first-order difference. The idea is to restrict the corrections in regions of rapid change to avoid undesirable behavior by limiting the magnitude of the difference in u or, more generally, the flux or variable gradients. In this sense, the scheme may be written

$$u_j^{n+1} = u_j^n - \nu\,\delta u_{j-\frac{1}{2}}^n + \frac{\nu}{2}(\nu-1)\left(\bar{\delta} u_{j-\frac{1}{2}}^n - \bar{\delta} u_{j-\frac{3}{2}}^n\right) \tag{4.246}$$

In this expression the quantity $\bar{\delta}$ is defined as

$$\bar{\delta} u_j = \psi\,\delta u_j$$

where ψ is a limiter function. Equation (4.246) may be written in a form showing the dependence of the limited portion on successive changes in the dependent variable in the following way:

$$u_j^{n+1} = u_j^n - \nu\,\delta u_{j-\frac{1}{2}}^n + \frac{\nu}{2}(\nu-1)\left(\psi^+_{j-\frac{1}{2}}\,\delta u_{j-\frac{1}{2}}^n - \psi^+_{j-\frac{3}{2}}\,\delta u_{j-\frac{3}{2}}^n\right) \tag{4.247}$$

In order to arrive at the conditions to ensure the TVD property is contained in the difference formulation, this expression is written as

$$u_j^{n+1} = u_j^n - \nu\,\delta u_{j-\frac{1}{2}}^n + \frac{\nu}{2}(\nu-1)\left(\psi^+_{j-\frac{1}{2}} - \frac{\psi^+_{j-\frac{3}{2}}}{r^+_{j-\frac{3}{2}}}\right)\delta u_{j-\frac{1}{2}}^n \tag{4.248}$$

where the ratio $r_{j+1/2}^{+}$ is defined as

$$r_{j+\frac{1}{2}}^{+} = \frac{u_{j+2} - u_{j+1}}{u_{j+1} - u_j} \tag{4.249}$$

Limiters are customarily written in terms of the successive variations in the dependent variables or the flux. In addition to the r^{+} definition, one may also define a quantity r^{-} of the form

$$r_{j+\frac{1}{2}}^{-} = \frac{u_j - u_{j-1}}{u_{j+1} - u_j} \tag{4.250}$$

Limiters are then written in terms of these ratios. While they may be written in general as functions of any number of successive variations, for simplicity, they are usually written showing a dependence on only the single ratio at the local point in question:

$$\psi_{j+\frac{1}{2}}^{+} = \psi\left(r_{j+\frac{1}{2}}^{+}\right) \tag{4.251}$$

If a wave propagating in the negative direction is present, limiting on the opposite family can be achieved using

$$\psi_{j+\frac{1}{2}}^{-} = \psi\left(r_{j+\frac{1}{2}}^{-}\right) \tag{4.252}$$

The TVD conditions that must be satisfied by the upwind method are

$$0 \leqslant \nu\left[1 + \frac{1}{2}(1 - \nu)\left(\psi_{j-\frac{1}{2}}^{+} - \frac{\psi_{j-\frac{3}{2}}^{+}}{r_{j-\frac{3}{2}}^{+}}\right)\right] \leqslant 1 \tag{4.253}$$

where

$$C_{j+\frac{1}{2}}^{-} = 0$$

The TVD condition shows that the limiter must satisfy an equation of the form

$$\frac{\psi(r_1)}{r_1} - \psi(r_2) \leqslant \frac{2}{\nu} \tag{4.254}$$

The stability bounds for this formulation are normally selected to satisfy the standard CFL condition. This provides a bound on the allowable values of ν, and the TVD condition can be simplified to

$$\frac{\psi(r_1)}{r_1} - \psi(r_2) \leqslant 2 \tag{4.255}$$

Many different formulations for ψ can be written to satisfy this inequality. Several constraints are imposed on the construction of the different forms. These include the fact that ψ must be a positive function, leading to the condition

$$\psi(r) \geqslant 0 \quad \text{for} \quad r \geqslant 0 \tag{4.256}$$

and when r is negative, ψ is set equal to zero, i.e.,

$$\psi(r) = 0 \quad \text{for} \quad r = 0 \tag{4.257}$$

This results in the following general constraints:

$$0 \leqslant \psi(r) \leqslant 2r \tag{4.258}$$

If an additional constraint on the magnitude of the limiter is imposed, i.e.,

$$\psi(r) \leqslant 2 \tag{4.259}$$

the TVD requirement may be written as

$$0 \leqslant \psi(r) \leqslant \min(2r, 2) \tag{4.260}$$

The second-order upwind scheme will then satisfy the TVD condition at any point in the shaded area of Fig. 4.55. If the limiters are set equal to 1, the original unlimited upwind, or Beam-Warming (1976) explicit upwind scheme, is recovered. Roe (1985) has developed weaker conditions for the limiters of the form

$$\frac{\psi(r)}{r} \leqslant \frac{2}{1-\nu} \tag{4.261}$$

and

$$\psi(r) \leqslant \frac{2}{\nu} \tag{4.262}$$

These conditions are the TVD conditions for the basic Lax-Wendroff method (see Prob. 4.65). We have used a linear equation to develop the TVD conditions. The corrections necessary to avoid the oscillations associated with the method are alterations to the second-order terms that appear as nonlinear terms in the difference formulas. It should be clear that the corrections that make a method TVD are always associated with nonlinear limiting even for linear convection problems.

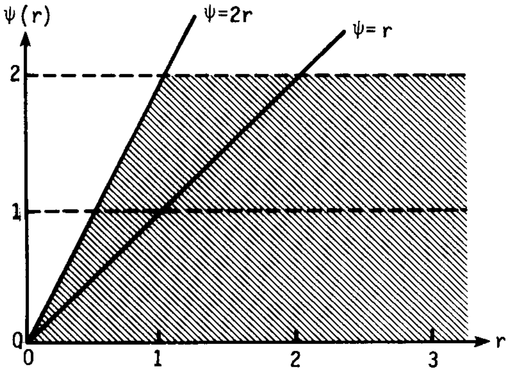

Figure 4.55 Limiter range for second-order TVD schemes.

Other limiters that are useful in computations have been developed and published in the open literature. One of the early examples is the van Leer (1979) limiter, which is of the form

$$\psi(r) = \frac{r + |r|}{1 + r^2} \tag{4.263}$$

The van Albada (1982) limiter is of the form

$$\psi(r) = \frac{r + r^2}{1 + r^2} \tag{4.264}$$

The minmod limiter is also used extensively in TVD numerical methods. As previously noted, this is a function that selects the smallest number from a set when all have the same sign but is zero if they have different signs. This limiter may be written

$$\psi(r) = \mathrm{minmod}(1, r) \tag{4.265}$$

Another limiter that has received use, particularly when contact surfaces are of importance, is the "Superbee" limiter of Roe (1985), which is written

$$\psi(r) = \max[0, \min(2r, 1), \min(r, 2)] \tag{4.266}$$

All of these limiters satisfy a symmetry condition written in the form

$$\frac{\psi(r)}{r} = \psi\left(\frac{1}{r}\right) \tag{4.267}$$

which ensures that the limits for forward and backward gradients are treated the same way. If this condition is not satisfied, the treatment afforded gradients by the limiters is not symmetric. Other TVD schemes may be constructed and are available for both upwind and central-difference methods. The Roe-Sweby scheme (Roe, 1984; Sweby, 1984) begins with a first-order upwind scheme with a numerical flux from Eq. (4.192) written in the form

$$f_{j+\frac{1}{2}} = \frac{1}{2}\left[F_{j+1} + F_j - \mathrm{sgn}(\bar{u}_{j+\frac{1}{2}})(F_{j+1} - F_j)\right] \tag{4.268}$$

A second-order correction is added to this flux, which is a limiter multiplied by the difference in the first-order upwind flux and the flux for the Lax-Wendroff scheme. Thus

$$f^{\mathrm{RS}}_{j+\frac{1}{2}} = f_{j+\frac{1}{2}} + \psi(r)\left(f^{\mathrm{LW}}_{j+\frac{1}{2}} - f_{j+\frac{1}{2}}\right) \tag{4.269}$$

This results in an expression for the flux

$$f^{\mathrm{RS}}_{j+\frac{1}{2}} = f_{j+\frac{1}{2}} + \frac{\psi(r)}{2}\left[\mathrm{sgn}(\bar{u}_{j+\frac{1}{2}}) - \frac{\Delta t}{\Delta x}\bar{u}_{j+\frac{1}{2}}\right](F_{j+1} - F_j) \tag{4.270}$$

where the notation for r is

$$r = \frac{u_{j+1+\sigma} - u_{j+\sigma}}{\delta u_{j+\frac{1}{2}}} \tag{4.271}$$

and σ may take on integer values defining the r values as previously used for $r^{\pm}$. The limiters that may be applied with success in this expression are the same as previously given in Eqs. (4.261)–(4.266). The Roe-Sweby scheme produces results for the 1-D case that are similar to the Burgers equation solution shown previously for the MUSCL scheme (see Prob. 4.66).

Another technique for creating a TVD scheme is Harten's modified flux method (Harten, 1984). In this approach, the T.E. of a first-order upwind scheme is developed, and the idea is to subtract this away like an antidiffusive flux term. This is accomplished by modifying the flux used in the original approximation. The modifications to the flux are due to the T.E., and as a result must necessarily be limited to obtain a TVD scheme. This technique is discussed in more detail in Chapter 6.

4.5 BURGERS' EQUATION (VISCOUS)

The complete nonlinear Burgers equation

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \mu\frac{\partial^2 u}{\partial x^2} \tag{4.272}$$

is a parabolic PDE, which can serve as a model equation for the boundary-layer equations, the "parabolized" Navier-Stokes equations, and the complete Navier-Stokes equations. In order to better model the steady boundary-layer and "parabolized" Navier-Stokes equations, the independent variables t and x can be replaced by x and y to give

$$\frac{\partial u}{\partial x} + u\frac{\partial u}{\partial y} = \mu\frac{\partial^2 u}{\partial y^2} \tag{4.273}$$

where x is the marching direction.

As with previous model equations, Burgers' equation has exact analytical solutions for certain boundary and initial conditions. These exact solutions are useful when comparing different numerical algorithms. The exact steady-state solution [i.e., $\lim_{t\to\infty} u(x,t)$] of Eq. (4.272) for the boundary conditions

$$u(0,t) = u_0 \tag{4.274}$$

$$u(L,t) = 0 \tag{4.275}$$

is given by

$$u = u_0\bar{u}\left\{\frac{1 - \exp[\bar{u}\,\mathrm{Re}_L(x/L - 1)]}{1 + \exp[\bar{u}\,\mathrm{Re}_L(x/L - 1)]}\right\} \tag{4.276}$$

where

$$\mathrm{Re}_L = \frac{u_0 L}{\mu} \tag{4.277}$$

and $\bar{u}$ is a solution of the equation

$$\frac{\bar{u} - 1}{\bar{u} + 1} = \exp(-\bar{u}\,\mathrm{Re}_L) \tag{4.278}$$

For simplicity, the linearized Burgers equation

$$\frac{\partial u}{\partial t} + c\frac{\partial u}{\partial x} = \mu\frac{\partial^2 u}{\partial x^2} \tag{4.279}$$

is often used in place of Eq. (4.272). Note that if $\mu = 0$, the wave equation is obtained. If $c = 0$, the heat equation is obtained. The exact steady-state solution of Eq. (4.279) for the boundary conditions given by Eqs. (4.274) and (4.275) is

$$u = u_0\left\{\frac{1 - \exp\left[R_L(x/L - 1)\right]}{1 - \exp(-R_L)}\right\} \tag{4.280}$$

where

$$R_L = \frac{cL}{\mu}$$

The exact unsteady solution of Eq. (4.279) for the initial condition

$$u(x,0) = \sin(kx)$$

and periodic boundary conditions is

$$u(x,t) = \exp(-k^2\mu t)\sin k(x - ct) \tag{4.281}$$

This latter exact solution is useful in evaluating the temporal accuracy of a method.

Equations (4.272) and (4.279) can be combined into a generalized equation (Rakich, 1978):

$$u_t + (c + bu)u_x = \mu u_{xx} \tag{4.282}$$

where c and b are free parameters. If $b = 0$, the linearized Burgers equation is obtained and if $c = 0$ and $b = 1$, the nonlinear Burgers equation is obtained. If $c = \frac{1}{2}$ and $b = -1$, the generalized Burgers equation has the stationary solution

$$u = -\frac{c}{b}\left[1 + \tanh\frac{c(x - x_0)}{2\mu}\right] \tag{4.283}$$

which is shown in Fig. 4.56 for $\mu = \frac{1}{4}$. Hence, if the initial distribution of u is given by Eq. (4.283), the exact solution does not vary with time but remains fixed at the initial distribution. Additional exact solutions of Burgers' equation can be found in the paper by Benton and Platzman (1972), which describes 35 different exact solutions.

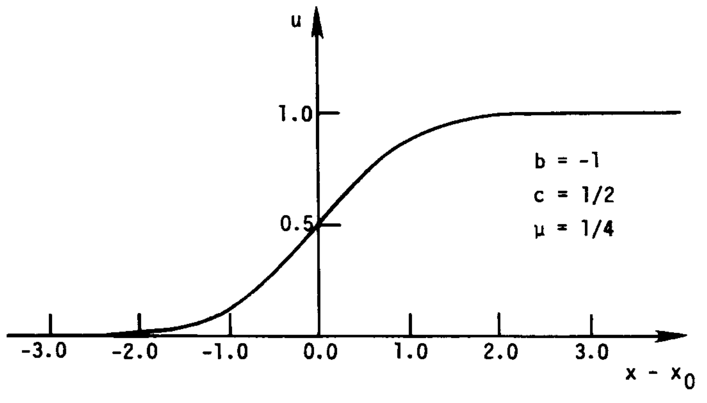

Figure 4.56 Exact solution of Eq. (4.283).

Equation (4.282) can be put into conservation form:

$$u_t + \bar{F}_x = 0 \tag{4.284}$$

where $\bar{F}$ is defined by

$$\bar{F} = cu + \frac{bu^2}{2} - \mu u_x \tag{4.285}$$

Alternatively, Eq. (4.282) can be rewritten as

$$u_t + F_x = \mu u_{xx} \tag{4.286}$$

where F is defined by

$$F = cu + \frac{bu^2}{2} \tag{4.287}$$

For the linearized case ($b = 0$), F reduces to

$$F = cu$$

If we let $A = \partial F/\partial u$, then Eq. (4.286) becomes

$$u_t + Au_x = \mu u_{xx} \tag{4.288}$$

where $A = u$ for the nonlinear Burgers equation ($c = 0$, $b = 1$) and $A = c$ for the linear Burgers equation ($b = 0$). We will use either Eq. (4.286) or Eq. (4.288) to represent Burgers' equation in the following discussion of applicable finite-difference/finite-volume schemes.

The various schemes described previously for the inviscid Burgers equation can also be applied to the complete Burgers equation. This is accomplished by simply adding a second-order (or higher-order) central-difference expression for the viscous term u_{xx}. We will describe these methods as well as other methods for solving the complete Burgers equation in the following sections.

4.5.1 FTCS Method

Roache (1972) has given the name FTCS method to the scheme obtained by applying forward-time and centered-space differences to the linearized Burgers equation [i.e., Eq. (4.288) with $A = c$]. The resulting algorithm is

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} + c\frac{u_{j+1}^n - u_{j-1}^n}{2\,\Delta x} = \mu\frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{(\Delta x)^2} \tag{4.289}$$

This is a first-order, explicit, one-step scheme with a T.E. of $O[\Delta t, (\Delta x)^2]$. The modified equation can be written as

$$u_t + cu_x = \left(\mu - \frac{c^2\,\Delta t}{2}\right)u_{xx} + \frac{c(\Delta x)^2}{3}\left(3r - \nu^2 - \frac{1}{2}\right)u_{xxx}$$
$$+ \frac{c(\Delta x)^3}{12}\left(\frac{r}{\nu} - \frac{3r^2}{\nu} - 2\nu + 10\nu r - 3\nu^3\right)u_{xxxx} + \cdots \tag{4.290}$$

where r is defined as $\mu\,\Delta t/(\Delta x)^2$ for the viscous Burgers equation and $\nu = c\,\Delta t/\Delta x$. Note that if $r = \frac{1}{2}$ and $\nu = 1$, the coefficients of the first two terms on the right-hand side of the modified equation become zero. Unfortunately, this eliminates the viscous term (μu_{xx}) that appears in the PDE we wish to solve. Thus the FTCS method with $r = \frac{1}{2}$ and $\nu = 1$, which incidentally reduces to $u_j^{n+1} = u_{j-1}^u$, is an unacceptable difference representation for Burgers' equation.

The "heuristic" stability analysis, described in Section 4.1.2, requires that the coefficient on u_{xx} be greater than zero. Hence

$$\frac{c^2\,\Delta t}{2} \leqslant \mu$$

or

$$\frac{c^2(\Delta t)^2}{(\Delta x)^2} \leqslant 2\mu\frac{\Delta t}{(\Delta x)^2}$$

which can be rewritten as

$$\nu^2 \leqslant 2r \tag{4.291}$$

A very useful parameter that arises naturally when solving Burgers' equation is the *mesh Reynolds number*, which is defined by

$$\mathrm{Re}_{\Delta x} = \frac{c\,\Delta x}{\mu} \tag{4.292}$$

This nondimensional parameter gives the ratio of convection to diffusion and plays an important role in determining the character of the solution for Burgers' equation. The mesh (cell) Reynolds number (also called Peclet number) can be expressed in terms of ν and r in the following manner:

$$\mathrm{Re}_{\Delta x} = \frac{c\,\Delta x}{\mu} = \frac{c\,\Delta t}{\Delta x}\frac{(\Delta x)^2}{\mu\,\Delta t} = \frac{\nu}{r}$$

Thus the stability condition given by Eq. (4.291) becomes

$$\mathrm{Re}_{\Delta x} \leqslant \frac{2}{\nu} \tag{4.293}$$

As pointed out earlier, the "heuristic" stability analysis does not always give the complete stability restrictions for a given numerical scheme, and this happens in the present case. In order to obtain all of the stability conditions it is necessary to use the Fourier stability analysis. For the FTCS method, the amplification factor is

$$G = 1 + 2r(\cos\beta - 1) - i\nu(\sin\beta) \tag{4.294}$$

which is plotted in Fig. 4.57(a) for a given ν and r. The equation for G describes an ellipse that is centered on the positive real axis at $(1 - 2r)$ and has semimajor and semiminor axes given by $2r$ and ν, respectively. In addition, the ellipse is tangent to the unit circle at the point where the positive real axis intersects the unit circle. For stability, it is necessary that $|G| \leqslant 1$, which requires that the ellipse be entirely within the unit circle. This leads to the following necessary stability restrictions, which are based on the lengths of the semimajor and semiminor axes:

$$\nu \leqslant 1 \qquad 2r \leqslant 1 \tag{4.295}$$

It is possible, however, for these restrictions to be satisfied and the solution to still be unstable, as can be seen in Fig. 4.57(b). Of course, the complete stability

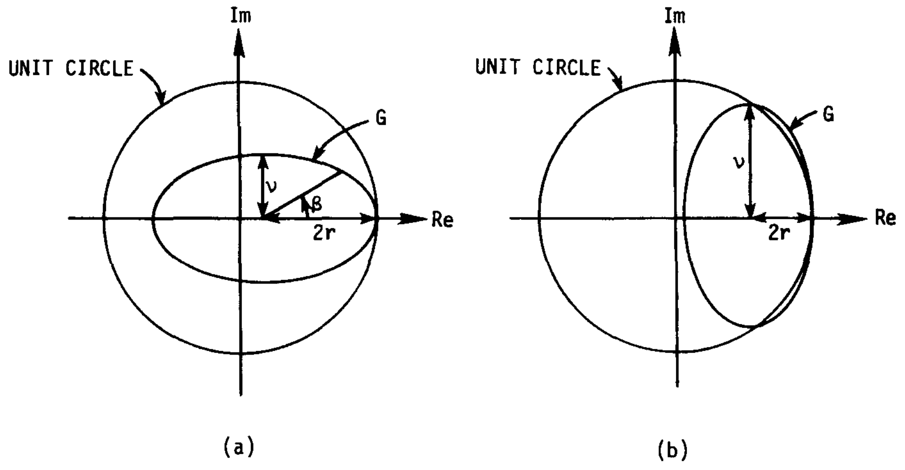

Figure 4.57 Stability of FTCS method: (a) $\nu < 1$, $r < \frac{1}{2}$, $\nu^2 < 2r$; (b) $\nu < 1$, $r < \frac{1}{2}$, $\nu^2 > 2r$.

limitations can be obtained by examining the modulus of the amplification factor in the usual manner. This analysis yields

$$\nu^2 \leqslant 2r \qquad r \leqslant \tfrac{1}{2} \tag{4.296}$$

Note that the first restriction was obtained previously by the heuristic stability analysis and that the two inequalities can be combined to yield a third inequality,

$$\nu \leqslant 1$$

which was obtained by graphical considerations. In terms of the mesh Reynolds number, the stability restrictions become

$$2\nu \leqslant \mathrm{Re}_{\Delta x} \leqslant \frac{2}{\nu} \tag{4.297}$$

It should be mentioned that the right-hand inequality is incorrectly given as $\mathrm{Re}_{\Delta x} \leqslant 2$ in some references.

An important characteristic of finite-difference schemes that are used to solve Burgers' equation is whether they produce oscillations (wiggles) in the solution. Obviously, we do not want these oscillations to occur in our solutions of fluid flow problems. The FTCS method will produce oscillations in the solution of Burgers' equation for mesh Reynolds numbers in the range

$$2 \leqslant \mathrm{Re}_{\Delta x} \leqslant \frac{2}{\nu}$$

For mesh Reynolds numbers slightly above $2/\nu$, the oscillations will eventually cause the solution to "blow up," as expected from our previous stability analysis. In order to explain the origin of the wiggles, let us rewrite Eq. (4.289) in the following form:

$$u_j^{n+1} = \left(r - \frac{\nu}{2}\right)u_{j+1}^n + (1 - 2r)u_j^n + \left(r + \frac{\nu}{2}\right)u_{j-1}^n \tag{4.298}$$

which is equivalent to

$$u_j^{n+1} = \frac{r}{2}(2 - \mathrm{Re}_{\Delta x})u_{j+1}^n + (1 - 2r)u_j^n + \frac{r}{2}(2 + \mathrm{Re}_{\Delta x})u_{j-1}^n \tag{4.299}$$

Furthermore, assume we are trying to find the steady-state solution of Burgers' equation for the initial condition,

$$u(x,0) = 0 \qquad 0 \leqslant x < 1$$

and boundary conditions,

$$u(0,t) = 0$$
$$u(1,t) = 1$$

using an 11-point mesh. For the first time step the values of u at time level $n + 1$ are all zero except at $j = 10$, where

$$u_{10}^{n+1} = \frac{r}{2}(1 - \mathrm{Re}_{\Delta x})(1) + (1 - 2r)(0) + \frac{r}{2}(2 + \mathrm{Re}_{\Delta x})(0) = \frac{r}{2}(2 - \mathrm{Re}_{\Delta x})$$

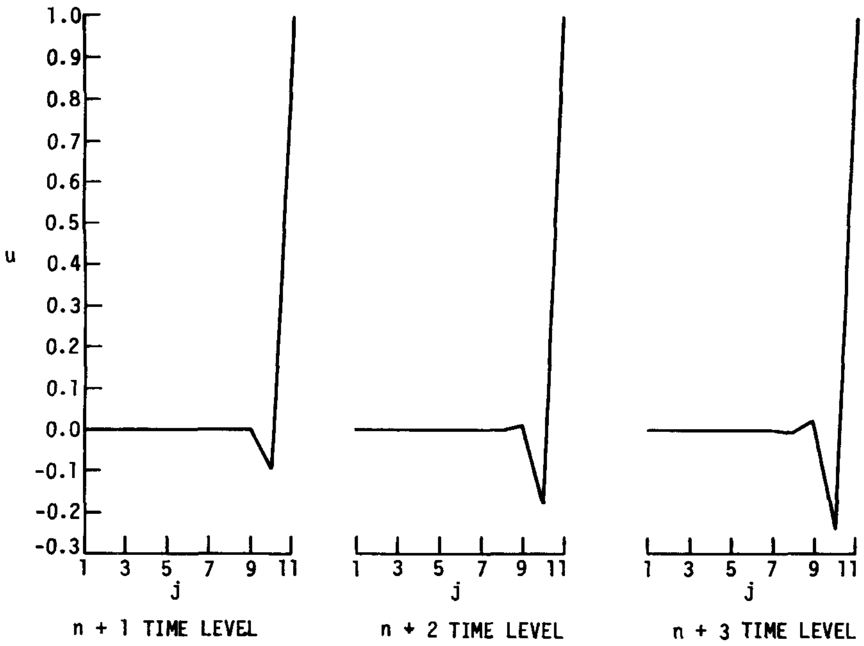

Figure 4.58 Oscillations in numerical solution of Burgers' equation: (a) $n + 1$ time level; (b) $n + 2$ time level; (c) $n + 3$ time level.

and at the boundary ($j = 11$), where u_{11} is fixed at 1. If $\text{Re}_{\Delta x} > 2$, the value of u_{10}^{n+1} will be negative, which will initiate an oscillation as shown in Fig. 4.58a. This figure is drawn for the conditions

$$\nu = 0.4$$
$$r = 0.1$$
$$\text{Re}_{\Delta x} = 4 < \frac{2}{\nu}$$

which make $u_{10}^{n+1} = -0.1$. During the next time step, the oscillation propagates one grid point further from the right-hand boundary. The values of u at $j = 9$ and $j = 10$ become

$$u_9^{n+2} = +0.01$$
$$u_{10}^{n+2} = -0.18$$

and the resulting solution is shown in Fig. 4.58b. The wiggles will eventually propagate to the other boundary but will remain bounded throughout the iteration to steady state. The oscillations that occur in this case are similar to the oscillations that appear when a second-order (or higher) scheme is used to solve the inviscid Burgers equation for a propagating discontinuity.

Additional insight into the origin of the wiggles can be obtained by examining the coefficients in Eq. (4.299) from a physical standpoint. We observe that when $\mathrm{Re}_{\Delta x} > 2$, the coefficient in front of u_{j+1}^n becomes negative. Hence, the larger the value for u_{j+1}^n, the smaller the value for u_j^{n+1}. This represents a nonphysical behavior for a viscous problem, since we would expect a greater "pull" on u_j^{n+1} (i.e., increased value) because of viscosity as u_{j+1}^n is increased. As a consequence of this nonphysical behavior, oscillations are produced in the solution.

The oscillations of the FTCS method can be eliminated if the second-order central difference used for the convective term cu_x is replaced by a first-order upwind difference. The resulting algorithm for $c > 0$ becomes

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} + c\frac{u_j^n - u_{j-1}^n}{\Delta x} = \mu\frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{(\Delta x)^2} \tag{4.300}$$

This first-order scheme eliminates the oscillations by adding additional dissipation to the solution. Unfortunately, the amount of dissipation causes the resulting solution to be sufficiently inaccurate to exclude Eq. (4.300) as a viable difference scheme for Burgers' equation. The large amount of dissipation is evident when we examine the modified equation for the scheme

$$u_t + cu_x = \left[\mu\left(1 + \frac{\mathrm{Re}_{\Delta x}}{2}\right) - \frac{c^2\,\Delta t}{2}\right]u_{xx} + \cdots \tag{4.301}$$

and compare it to the modified equation of the FTCS method. Equation (4.301) has the additional term $\mu\,\mathrm{Re}_{\Delta x}/2$ appearing in the coefficient of u_{xx}. Hence, if $\mathrm{Re}_{\Delta x} > 2$, this additional term produces more dissipation (diffusion) than is present in the original problem governed by Burgers' equation. In order to reduce dispersive errors without adding a large amount of artificial viscosity, Leonard (1979a, 1979b) has suggested using a third-order upstream (upwind) difference for the convective term. The resulting algorithm for $c > 0$ is

$$\begin{aligned}\frac{u_j^{n+1} - u_j^n}{\Delta t} &+ c\left(\frac{u_{j+1}^n - u_{j-1}^n}{2\,\Delta x} - \frac{u_{j+1}^n - 3u_j^n + 3u_{j-1}^n - u_{j-2}^n}{6\,\Delta x}\right)\\ &= \mu\frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{(\Delta x)^2}\end{aligned} \tag{4.302}$$

and for $c < 0$ the algorithm becomes

$$\begin{aligned}\frac{u_j^{n+1} - u_j^n}{\Delta t} &+ c\left(\frac{u_{j+1}^n - u_{j-1}^n}{2\,\Delta x} - \frac{u_{j+2}^n - 3u_{j+1}^n + 3u_j^n - u_{j-1}^n}{6\,\Delta x}\right)\\ &= \mu\frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{(\Delta x)^2}\end{aligned} \tag{4.303}$$

4.5.2 Leap Frog/DuFort-Frankel Method

We have noted earlier that the linearized Burgers equation is a combination of the first-order wave equation and the heat equation. This suggests that we might be able to combine some of the algorithms given previously for the wave equation and the heat equation. The leap frog/DuFort-Frankel method is one such example. When applied to Eq. (4.288), this method becomes

$$\frac{u_j^{n+1} - u_j^{n-1}}{2\,\Delta t} + A_j^n \frac{u_{j+1}^n - u_{j-1}^n}{2\,\Delta x} = \mu \frac{u_{j+1}^n - u_j^{n+1} - u_j^{n-1} + u_{j-1}^n}{(\Delta x)^2} \tag{4.304}$$

This explicit, one-step scheme is first-order accurate with a T.E. of $O[(\Delta t/\Delta x)^2, (\Delta t)^2, (\Delta x)^2]$. The modified equation for the linear case ($A = c$) can be written as

$$u_t + cu_x = \mu(1 - \nu^2)u_{xx} + \left[\frac{2\mu^2 c(\Delta t)^2}{(\Delta x)^2} - \frac{1}{6}c(\Delta x)^2 + \frac{1}{6}c^3(\Delta t)^2 - \frac{2\mu^2 c^3(\Delta t)^4}{(\Delta x)^4}\right]u_{xxx} + \cdots \tag{4.305}$$

Also for the linear case, a Fourier stability analysis can be performed, which gives the stability condition

$$\nu \leqslant 1$$

Note that this stability condition is independent of the viscosity coefficient μ because of the DuFort-Frankel type of differencing used for the viscous term. However, because consistency requires that $(\Delta t/\Delta x)^2$ approach zero as Δt and Δx approach zero, a much smaller time step than allowed by $\nu \leqslant 1$ is implied. For this reason, the leap frog/DuFort-Frankel scheme seems better suited for the calculation of steady solutions (where time accuracy is unimportant) than for unsteady solutions according to Peyret and Viviand (1975). In the nonlinear case, this scheme is unstable if $\mu = 0$.

4.5.3 Brailovskaya Method

The following two-step explicit method for Eq. (4.286) was proposed by Brailovskaya (1965):

$$\begin{aligned}
&\text{Predictor:} \quad && \overline{u_j^{n+1}} = u_j^n - \frac{\Delta t}{2\,\Delta x}(F_{j+1}^n - F_{j-1}^n) + r(u_{j+1}^n - 2u_j^n + u_{j-1}^n) \\
&\text{Corrector:} \quad && u_j^{n+1} = u_j^n - \frac{\Delta t}{2\,\Delta x}\left(\overline{F_{j+1}^{n+1}} - \overline{F_{j-1}^{n+1}}\right) + r(u_{j+1}^n - 2u_j^n + u_{j-1}^n)
\end{aligned} \tag{4.306}$$

This scheme is formally first-order accurate with a T.E. of $O[\Delta t, (\Delta x)^2]$. If only a steady-state solution is desired, the first-order temporal accuracy is not important. For the linear Burgers equation, the von Neumann necessary condition for stability is

$$|G|^2 = 1 - \{\nu^2 \sin^2 \beta(1 - \nu^2 \sin^2 \beta) + 4r(1 - \cos \beta) \times [1 - r(1 - \cos \beta)(1 + \nu^2 \sin^2 \beta)]\} \leqslant 1 \quad (4.307)$$

If we ignore viscous effects (i.e., set $r = 0$), the stability condition becomes

$$\nu \leqslant 1$$

On the other hand, if we ignore the convection term, i.e., set $\nu = 0$, the stability condition becomes

$$r \leqslant \tfrac{1}{2}$$

Based on these observations, Carter (1971) has suggested the following stability criterion for the Brailovskaya scheme:

$$\Delta t \leqslant \min \left[\frac{(\Delta x)^2}{2\mu}, \frac{(\Delta x)}{|A|} \right] \quad (4.308)$$

An attractive feature of this scheme is that the viscous term remains the same in both predictor and corrector steps and needs to be computed only once.

4.5.4 Allen-Cheng Method

Allen and Cheng (1970) modified the Brailovskaya scheme to eliminate the stability restriction on r. Their scheme is given by

$$\begin{aligned} \text{Predictor:} \quad & \overline{u_j^{n+1}} = u_j^n - \frac{\Delta t}{2\,\Delta x}(F_{j+1}^n - F_{j-1}^n) \\ & \qquad + r\left(u_{j+1}^n - 2\overline{u_j^{n+1}} + u_{j-1}^n\right) \\ \text{Corrector:} \quad & u_j^{n+1} = u_j^n - \frac{\Delta t}{2\,\Delta x}\left(\overline{F_{j+1}^{n+1}} - \overline{F_{j-1}^{n+1}}\right) \\ & \qquad + r\left(\overline{u_{j+1}^{n+1}} - 2u_j^{n+1} + \overline{u_{j-1}^{n+1}}\right) \end{aligned} \quad (4.309)$$

The unconventional differencing of the viscous term eliminates the stability restriction on r, so that the stability condition becomes

$$\nu \leqslant 1$$

for the linear Burgers equation. As a result, when μ is large, this method permits a much larger time step to be taken than does the Brailovskaya scheme. The Allen-Cheng method is formally first-order accurate with a T.E. of $O[\Delta t, (\Delta x)^2]$.

4.5.5 Lax-Wendroff Method

We have previously applied the two-step Lax-Wendroff method to the wave equation. When applied to the complete Burgers equation, several different variations of the method are possible, including the following:

$$\begin{aligned}
\text{Step 1:} \quad u_j^{n+1/2} &= \frac{1}{2}(u_{j+1/2}^n - u_{j-1/2}^n) - \frac{\Delta t}{\Delta x}(F_{j+1/2}^n - F_{j-1/2}^n) \\
&\quad + r\left[(u_{j-3/2}^n - 2u_{j-1/2}^n + u_{j+1/2}^n)\right. \\
&\quad \left. + (u_{j+3/2}^n - 2u_{j+1/2}^n + u_{j-1/2}^n)\right] \qquad (4.310) \\
\text{Step 2:} \quad u_j^{n+1} &= u_j^n - \frac{\Delta t}{\Delta x}\left(F_{j+1/2}^{n+1/2} - F_{j-1/2}^{n+1/2}\right) \\
&\quad + r(u_{j+1}^n - 2u_j^n + u_{j-1}^n)
\end{aligned}$$

This version is based on the Lax-Wendroff scheme used by Thommen (1966) to solve the Navier-Stokes equations. An alternate version has been proposed by Palumbo and Rubin (1972), which computes provisional values at time level $n+1$ instead of $n + \frac{1}{2}$. The present version is formally first-order accurate with a T.E. of $O[\Delta t, (\Delta x)^2]$. The exact linear stability condition is

$$\frac{\Delta t}{(\Delta x)^2}(A^2 \Delta t + 2\mu) \leqslant 1 \qquad (4.311)$$

4.5.6 MacCormack Method

The original MacCormack method (1969) applied to the complete Burgers equation (4.286) is

$$\begin{aligned}
\text{Predictor:} \quad \overline{u_j^{n+1}} &= u_j^n - \frac{\Delta t}{\Delta x}(F_{j+1}^n - F_j^n) + r(u_{j+1}^n - 2u_j^n + u_{j-1}^n) \\
\text{Corrector:} \quad u_j^{n+1} &= \frac{1}{2}\left[u_j^n + \overline{u_j^{n+1}} - \frac{\Delta t}{\Delta x}\left(\overline{F_j^{n+1}} - \overline{F_{j-1}^{n+1}}\right)\right. \qquad (4.312) \\
&\quad \left. + r\left(\overline{u_{j+1}^{n+1}} - 2\overline{u_j^{n+1}} + \overline{u_{j-1}^{n+1}}\right)\right]
\end{aligned}$$

which is second-order accurate in both time and space. In this version of the MacCormack scheme, a forward difference is employed in the predictor step for $\partial F/\partial x$ and a backward difference is used in the corrector step. The alternate version of the MacCormack scheme employs a backward difference in the predictor step and a forward difference in the corrector step. Both variants of the MacCormack scheme are second-order accurate. It is not possible to obtain a simple stability criterion for the MacCormack scheme applied to the Burgers equation. However, either the condition given by Eq. (4.308) or the empirical formula (Tannehill et al., 1975)

$$\Delta t \leqslant \frac{(\Delta x)^2}{|A|\,\Delta x + 2\mu} \tag{4.313}$$

can be used with an appropriate safety factor. The latter formula reduces to the usual viscous condition $r \leqslant \frac{1}{2}$ when $|A|$ is set equal to zero, and reduces to the usual inviscid condition $|A|\,\Delta t/\Delta x \leqslant 1$ when μ is set equal to zero. The MacCormack method has been widely used to solve not only the Euler equations but also the Navier-Stokes equations for laminar flow. For multidimensional problems, a time-split version of the MacCormack scheme has been developed and will be described in Section 4.5.8. For high-Reynolds-number problems, MacCormack has devised several newer methods, which will be discussed in Chapter 9.

An interesting variation of the original MacCormack scheme is obtained when overrelaxation is applied to both predicted and corrected values (Désidéri and Tannehill, 1977a) in the following manner:

Predictor: $$\overline{v_j^{n+1}} = u_j^n - \frac{\Delta t}{\Delta x}(F_{j+1}^n - F_j^n) + r(u_{j+1}^n - 2u_j^n + u_{j-1}^n)$$

$$\overline{u_j^{n+1}} = u_j^{\overline{n}} + \overline{\omega}\left(\overline{v_j^{n+1}} - u_j^{\overline{n}}\right) \tag{4.314}$$

Corrector: $$v_j^{n+1} = \overline{u_j^{n+1}} - \frac{\Delta t}{\Delta x}\left(\overline{F_j^{n+1}} - \overline{F_{j-1}^{n+1}}\right)$$

$$+ r\left(\overline{u_{j+1}^{n+1}} - 2\overline{u_j^{n+1}} + \overline{u_{j-1}^{n+1}}\right)$$

$$u_j^{n+1} = u_j^n + \omega\left(v_j^{n+1} - u_j^n\right) \tag{4.315}$$

In these equations, the v's are intermediate quantities, the u's denote final predictions, $\overline{\omega}$ and ω are overrelaxation parameters, and $u_j^{\overline{n}}$ represents the predicted value for u_j from the previous step. The original MacCormack scheme is obtained by setting $\overline{\omega} = 1$ and $\omega = \frac{1}{2}$. In general, the overrelaxed MacCormack method is first-order accurate with a T.E. of $O[\Delta t, (\Delta x)^2]$. However, it can be shown (Désidéri and Tannehill, 1977b) that if

$$\omega\overline{\omega} = |\overline{\omega} - \omega| \tag{4.316}$$

the method is second-order accurate in time when applied to the linearized Burgers equation. The overrelaxation scheme accelerates the convergence over that of the original MacCormack scheme by an approximate factor Ω, given by

$$\Omega = \frac{2\overline{\omega}\omega}{1 - (\overline{\omega} - 1)(\omega - 1)} \tag{4.317}$$

A Fourier stability analysis applied to the linearized Burgers equation does not yield a necessary and sufficient stability condition in the form of an algebraic relation between the parameters ν, r, $\overline{\omega}$, and ω. However, a necessary condition of stability is

$$|(\overline{\omega} - 1)(\omega - 1)| \leqslant 1 \tag{4.318}$$

In general, the stability limitation must be computed numerically, and it is usually more restrictive than the conditions $\overline{\omega} \leqslant 2$ and $\omega \leqslant 2$.

4.5.7 Briley-McDonald Method

The Briley-McDonald (1974) method is an implicit scheme which is often based on the following time differencing (Euler implicit) of Eq. (4.286):

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} + \left(\frac{\partial F}{\partial x}\right)_j^{n+1} = \mu\left(\frac{\partial^2 u}{\partial x^2}\right)_j^{n+1} \tag{4.319}$$

The term $(\partial F/\partial x)_j^{n+1}$ is expanded as

$$\left(\frac{\partial F}{\partial x}\right)_j^{n+1} = \left(\frac{\partial F}{\partial x}\right)_j^{n} + \Delta t\left[\frac{\partial}{\partial t}\left(\frac{\partial F}{\partial x}\right)\right]_j^n + O[(\Delta t)^2] \tag{4.320}$$

thereby introducing $\partial/\partial t(\partial F/\partial x)$, which can be replaced by

$$\frac{\partial}{\partial t}\left(\frac{\partial F}{\partial x}\right) = \frac{\partial}{\partial x}\left(\frac{\partial F}{\partial t}\right) = \frac{\partial}{\partial x}\left(\frac{\partial F}{\partial u}\frac{\partial u}{\partial t}\right) = \frac{\partial}{\partial x}\left(A\frac{\partial u}{\partial t}\right) \tag{4.321}$$

Finally, if we combine Eqs. (4.319), (4.320), and (4.321) and employ forward-time differences and centered-spatial differences, the Briley-McDonald method is obtained:

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} + \frac{F_{j+1}^n - F_{j-1}^n}{2\,\Delta x} + \frac{A_{j+1}^n\left(u_{j+1}^{n+1} - u_{j+1}^n\right) - A_{j-1}^n\left(u_{j-1}^{n+1} - u_{j-1}^n\right)}{2\,\Delta x}$$
$$= \mu\,\hat{\delta}_x^2 u_j^{n+1} \tag{4.322}$$

This scheme is formally first-order accurate with a T.E. of $O[\Delta t, (\Delta x)^2]$. However, at steady-state the accuracy is $O[(\Delta x)^2]$. The temporal accuracy can be increased by using trapezoidal differencing or by using additional time levels in the same manner as discussed earlier for the Beam-Warming scheme. For example, if we apply trapezoidal time differencing to Eq. (4.286), the following equation is obtained:

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} + \frac{1}{2}\left[\left(\frac{\partial F}{\partial x}\right)_j^n + \left(\frac{\partial F}{\partial x}\right)_j^{n+1}\right] = \frac{1}{2}\mu\left[\left(\frac{\partial^2 u}{\partial x^2}\right)_j^n + \left(\frac{\partial^2 u}{\partial x^2}\right)_j^{n+1}\right] \tag{4.323}$$

Proceeding as before, we find the resulting second-order accurate scheme to be

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} + \frac{F_{j+1}^n - F_{j-1}^n}{2\,\Delta x} + \frac{A_{j+1}^n\left(u_{j+1}^{n+1} - u_{j+1}^n\right) - A_{j-1}^n\left(u_{j-1}^{n+1} - u_{j-1}^n\right)}{4\,\Delta x}$$
$$= \frac{\mu}{2(\Delta x)^2}\left[(\delta_x^2 u)_j^n + (\delta_x^2 u)_j^{n+1}\right] \tag{4.324}$$

Both of these schemes, Eq. (4.322) and Eq. (4.324), are unconditionally stable and produce tridiagonal systems of linear algebraic equations that can be solved using the Thomas algorithm.

The Briley-McDonald method is directly related to the method developed by Beam and Warming (1978) to solve the Navier-Stokes equations. In fact, when the two methods are applied to Burgers' equation, they can be reduced to the same form. In order to do this, the delta terms in the Beam-Warming method must be replaced by their equivalent expressions [i.e., Δu_j^n is replaced by $(u_j^{n+1} - u_j^n)$]. The Beam-Warming method for the Navier-Stokes equations is discussed in Chapter 9.

4.5.8 Time-Split MacCormack Method

In order to illustrate methods that are designed specifically for multidimensional problems, we introduce the 2-D Burgers equation

$$\frac{\partial u}{\partial t} + \frac{\partial F}{\partial x} + \frac{\partial G}{\partial y} = \mu\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) \tag{4.325}$$

If we let $A = \partial F/\partial u$ and $B = \partial G/\partial u$, Eq. (4.325) can be rewritten as

$$u_t = Au_x + Bu_y = \mu(u_{xx} + u_{yy}) \tag{4.326}$$

The exact steady-state solution (derived by Rai, 1982) of the 2-D linearized Burgers equation,

$$u_t + cu_x + du_y = \mu(u_{xx} + u_{yy}) \tag{4.327}$$

for the boundary conditions $(0 \leqslant t \leqslant \infty)$,

$$\begin{aligned} u(x,0,t) &= \frac{1 - \exp[(x-1)c/\mu]}{1 - \exp(-c/\mu)} \qquad & u(x,1,t) &= 0 \\ u(0,y,t) &= \frac{1 - \exp[(y-1)d/\mu]}{1 - \exp(-d/\mu)} \qquad & u(1,y,t) &= 0 \end{aligned} \tag{4.328}$$

and the initial condition $(0 < x \leqslant 1,\ 0 < y \leqslant 1)$,

$$u(x,y,0) = 0$$

is given by

$$u(x,y) = \left\{\frac{1 - \exp[(x-1)c/\mu]}{1 - \exp(-c/\mu)}\right\}\left\{\frac{1 - \exp[(y-1)d/\mu]}{1 - \exp(-d/\mu)}\right\} \tag{4.329}$$

Note that the extension of this form of solution to the 3-D linearized Burgers equation is straightforward. All of the methods we have discussed for the 1-D Burgers equation can be readily extended to the 2-D Burgers equation. However, because of the more restrictive stability conditions of the explicit methods and the desire to maintain tridiagonal matrices in the implicit schemes, it is usually necessary to modify the previous algorithms for multidimensional problems. As an example, let us first consider the explicit time-split MacCormack method.

The time-split MacCormack method (MacCormack, 1971; MacCormack and Baldwin, 1975) "splits" the original MacCormack scheme into a sequence of 1-D operations, thereby achieving a less restrictive stability condition. In other words, the splitting makes it possible to advance the solution in each direction with the maximum allowable time step. This is particularly advantageous if the allowable time steps ($\Delta t_x, \Delta t_y$) are much different because of differences in the mesh spacings ($\Delta x, \Delta y$). In order to explain this method, we will make use of the 1-D difference operators $L_x(\Delta t_x)$ and $L_y(\Delta t_y)$. The $L_x(\Delta t_x)$ operator applied to $u^n_{i,j}$,

$$u^*_{i,j} = L_x(\Delta t_x)u^n_{i,j} \tag{4.330}$$

is by definition equivalent to the two-step formula:

$$\begin{aligned} \overline{u^*_{i,j}} &= u^n_{i,j} - \frac{\Delta t_x}{\Delta x}(F^n_{i+1,j} - F^n_{i,j}) + \mu\,\Delta t_x \hat{\delta}^2_x u^n_{i,j} \\ u^*_{i,j} &= \frac{1}{2}\left[u^n_{i,j} + \overline{u^*_{i,j}} - \frac{\Delta t_x}{\Delta x}\left(\overline{F^*_{i,j}} - \overline{F^*_{i-1,j}}\right) + \mu\,\Delta t_x \hat{\delta}^2_x \overline{u^*_{i,j}}\right] \end{aligned} \tag{4.331}$$

These expressions make use of a dummy time index, which is denoted by the asterisk. The $L_y(\Delta t_y)$ operator is defined in a similar manner, that is,

$$u^*_{i,j} = L_y(\Delta t_y)u^n_{i,j} \tag{4.332}$$

is equivalent to

$$\begin{aligned} \overline{u^*_{i,j}} &= u^n_{i,j} - \frac{\Delta t_y}{\Delta y}(G^n_{i,j+1} - G^n_{i,j}) + \mu\,\Delta t_y \hat{\delta}^2_j u^n_{i,j} \\ u^*_{i,j} &= \frac{1}{2}\left[u^n_{i,j} + \overline{u^*_{i,j}} - \frac{\Delta t_y}{\Delta y}\left(\overline{G^*_{i,j}} - \overline{G^*_{i,j-1}}\right) + \mu\,\Delta t_y \hat{\delta}^2_y \overline{u^*_{i,j}}\right] \end{aligned} \tag{4.333}$$

A second-order accurate scheme can be constructed by applying the L_x and L_y operators to $u^n_{i,j}$ in the following manner:

$$u^{n+1}_{i,j} = L_y\left(\frac{\Delta t}{2}\right)L_x(\Delta t)L_y\left(\frac{\Delta t}{2}\right)u^n_{i,j} \tag{4.334}$$

This scheme has a T.E. of $O[(\Delta t)^2, (\Delta x)^2, (\Delta y)^2]$. In general, a scheme formed by a sequence of these operators is (1) stable, if the time step of each operator does not exceed the allowable step size for that operator; (2) consistent, if the sums of the time steps for each of the operators are equal; and (3) second-order accurate, if the sequence is symmetric. Other sequences that satisfy these criteria are given by

$$\begin{aligned} u^{n+1}_{i,j} &= L_y\left(\frac{\Delta t}{2}\right)L_x\left(\frac{\Delta t}{2}\right)L_x\left(\frac{\Delta t}{2}\right)L_y\left(\frac{\Delta t}{2}\right)u^n_{i,j} \\ u^{n+1}_{i,j} &= \left[L_y\left(\frac{\Delta t}{2m}\right)\right]^m L_x(\Delta t)\left[L_y\left(\frac{\Delta t}{2m}\right)\right]^m u^n_{i,j} \qquad m = \text{integer} \end{aligned} \tag{4.335}$$

The last sequence is quite useful for the case where $\Delta y \ll \Delta x$.

4.5.9 ADI Methods

Polezhaev (1967) used an adaptation of the Peaceman-Rachford ADI scheme to solve the compressible Navier-Stokes equations. When applied to the 2-D Burgers equation, Eq. (4.326), this scheme becomes

$$\begin{aligned}\left[1+\frac{\Delta t}{2}\left(A_{i,j}^{n}\frac{\bar{\delta}_x}{2\,\Delta x}-\mu\hat{\delta}_x^2\right)\right]u_{i,j}^{*}&=\left[1-\frac{\Delta t}{2}\left(B_{i,j}^{n}\frac{\bar{\delta}_y}{2\,\Delta y}-\mu\hat{\delta}_y^2\right)\right]u_{i,j}^{n}\\ \left[1+\frac{\Delta t}{2}\left(B_{i,j}^{*}\frac{\bar{\delta}_y}{2\,\Delta y}-\mu\hat{\delta}_y^2\right)\right]u_{i,j}^{n+1}&=\left[1-\frac{\Delta t}{2}\left(A_{i,j}^{n}\frac{\bar{\delta}_x}{2\,\Delta x}-\mu\hat{\delta}_x^2\right)\right]u_{i,j}^{*}\end{aligned}\tag{4.336}$$

This method is first-order accurate with a T.E. of $O[\Delta t,(\Delta x)^2,(\Delta y)^2]$ and is unconditionally stable for the linear case. Obviously, a tridiagonal system of algebraic equations must be solved during each step.

When the Briley-McDonald scheme, Eq. (4.322), is applied directly to the 2-D Burgers equation, a tridiagonal system of algebraic equations is no longer obtained. This difficulty can be avoided by applying the two-step ADI procedure of Douglas and Gunn (1964):

$$\left[1+\Delta t\left(\frac{\bar{\delta}_x}{2\,\Delta x}A_{i,j}^{n}-\mu\hat{\delta}_x^2\right)\right]u_{i,j}^{*}=\left[1-\Delta t\left(\frac{\bar{\delta}_y}{2\,\Delta y}B_{i,j}^{n}-\mu\hat{\delta}_y^2\right)\right]u_{i,j}^{n}+(\Delta t)S_{i,j}^{n}\tag{4.337}$$

$$\left[1+\Delta t\left(\frac{\bar{\delta}_y}{2\,\Delta y}B_{i,j}^{n}-\mu\hat{\delta}_x^2\right)\right]u_{i,j}^{n+1}=u_{i,j}^{n}-\Delta t\left(\frac{\bar{\delta}_x}{2\,\Delta x}A_{i,j}^{n}-\mu\hat{\delta}_x^2\right)u_{i,j}^{*}+\Delta t\,S_{i,j}^{n}\tag{4.338}$$

where

$$S_{i,j}^{n}=-\frac{\bar{\delta}_x}{2\,\Delta x}F_{i,j}^{n}-\frac{\bar{\delta}_y}{2\,\Delta y}G_{i,j}^{n}+\frac{\bar{\delta}_x}{2\,\Delta x}(A_{i,j}^{n}u_{i,j}^{n})+\frac{\bar{\delta}_y}{2\,\Delta y}(B_{i,j}^{n}u_{i,j}^{n})$$

4.5.10 Predictor-Corrector, Multiple-Iteration Method

Rubin and Lin (1972) devised a predictor-corrector, multiple-iteration method to solve the 3-D "parabolized" Navier-Stokes equations. Their scheme eliminates cross coupling of grid points in the normal (y) and lateral (z) directions and uses an iterative procedure to recover acceptable accuracy. In order to illustrate this method, let us use the following 3-D linear Burgers equation,

$$u_x+cu_y+du_z=\mu(u_{yy}+u_{zz})\tag{4.339}$$

as a model for the "parabolized" Navier-Stokes equations. The predictor-corrector, multiple-iteration method applied to this model equation is

$$\begin{aligned} u_{i+1,j,k}^{m+1} = u_{i,j,k} &- \frac{c\,\Delta x}{2\,\Delta y}\left(u_{i+1,j+1,k}^{m+1} - u_{i+1,j-1,k}^{m+1}\right) \\ &- \frac{d\,\Delta x}{2\,\Delta z}\left(u_{i+1,j,k+1}^{m} - u_{i+1,j,k-1}^{m}\right) \\ &+ \frac{\mu\,\Delta x}{(\Delta y)^2}\left(u_{i+1,j+1,k}^{m+1} - 2u_{i+1,j,k}^{m+1} + u_{i+1,j-1,k}^{m+1}\right) \\ &+ \frac{\mu\,\Delta x}{(\Delta z)^2}\left(u_{i+1,j,k+1}^{m} - 2u_{i+1,j,k}^{m+1} + u_{i+1,j,k-1}^{m}\right) \end{aligned} \tag{4.340}$$

where the superscript m indicates the iteration level and $x = i\,\Delta x$, $y = j\,\Delta y$, and $z = k\,\Delta z$. For the first iteration, m is set equal to zero and the corresponding terms are approximated by either linear replacement,

$$u_{i+1,j,k}^{0} = u_{i,j,k}$$

or by Taylor-series expansions such as

$$u_{i+1,j,k}^{0} = 2u_{i,j,k} - u_{i-1,j,k} + O[(\Delta x)^2]$$

As a result, Eq. (4.340) has three unknowns

$$m = 0 \quad \begin{cases} u_{i+1,j+1,k}^{1} \\ u_{i+1,j,k}^{1} \\ u_{i+1,j-1,k}^{1} \end{cases} \tag{4.341}$$

which produces a tridiagonal system of algebraic equations. The computation in the $i+1$ plane proceeds outward from the known boundary conditions at $k = 1$ to the last k column of grid points. This completes the first iteration. For the next iteration ($m = 1$) the three unknowns in Eq. (4.340) are

$$m = 1 \quad \begin{cases} u_{i+1,j+1,k}^{2} \\ u_{i+1,j,k}^{2} \\ u_{i+1,j-1,k}^{2} \end{cases} \tag{4.342}$$

This iteration procedure is continued until the solution is converged in the $i+1$ plane. Usually, only two iterations ($m = 0$, $m = 1$) are required to recover acceptable accuracy. The computation then advances to the $i+2$ plane.

4.5.11 Roe Method

The Roe (1981) scheme was previously applied to the inviscid Burgers equation in Section 4.4.9. When applied to the complete Burgers' equation (Eq. 4.286), the algorithm becomes

$$u_j^{n+1} = u_j^n - \frac{\Delta t}{2\,\Delta x}\left[(F_{j+1}^n - F_{j-1}^n) - |\bar{u}_{j+\frac{1}{2}}^n|(u_{j+1}^n - u_j^n) + |\bar{u}_{j-\frac{1}{2}}^n|(u_j^n - u_{j-1}^n)\right]$$
$$+ r(u_{j+1}^n - 2u_j^n + u_{j-1}^n) \qquad (4.343)$$

where $F = u^2/2$ and

$$\bar{u}_{j+\frac{1}{2}}^n = \frac{u_j^n + u_{j+1}^n}{2}$$

This explicit one-step method is first-order accurate with a T.E. of $O[\Delta t, (\Delta x)^2]$. Higher-order versions of Roe's scheme can be obtained using the techniques described in Sections 4.4.11 and 4.4.12.

4.6 CONCLUDING REMARKS

In this chapter an attempt has been made to introduce basic numerical methods for solving simple model PDEs. It has not been the intent to include all numerical techniques that have been proposed for these equations. Some very useful methods have not been included. However, those that have been presented should provide a reasonable background for the more complex applications that follow in Chapters 6–9.

Based on the information presented on the various techniques, it is clear that many different numerical methods can be used to solve the same problem. The differences in the quality of the solutions produced using the applicable methods are frequently small, and the selection of an optimum technique becomes difficult. However, the selection process can be aided by the experience gained in programming the various methods to solve the model equations presented in this chapter.

PROBLEMS

4.1 Derive Eq. (4.19).

4.2 Derive the modified equation for the Lax method applied to the wave equation. Retain terms up to and including u_{xxxx}.

4.3 Repeat Prob. 4.2 for the Euler implicit scheme.

4.4 Derive the modified equation for the leap frog method. Retain terms up to and including u_{xxxxx}.

4.5 Repeat Prob. 4.4 for the Lax-Wendroff method.

4.6 Determine the errors in amplitude and phase for $\beta = 90°$ if the Lax method is applied to the wave equation for 10 time steps with $\nu = 0.5$.

4.7 Repeat Prob. 4.6 for the MacCormack scheme.

4.8 Suppose the Lax scheme is used to solve the wave equation ($c = 0.75$) for the initial condition

$$u(x,0) = 2\sin(2\pi x - 0.4\pi) \qquad 0 \leqslant x \leqslant 2$$

and periodic boundary conditions with $\Delta x = 0.02$ and $\Delta t = 0.02$.

(*a*) Use the amplification factor to find the amplitude and phase errors after 10 steps.

(*b*) Use a truncated version of the modified equation to determine (approximately) the amplitude and phase errors after 10 steps.
Hint: The exact solution for the PDE

$$u_t + cu_x = \mu\mu_{xx} + du_{xxx}$$

with initial condition

$$u(x,0) = A_0 \sin(kx)$$

and periodic boundary conditions is

$$u(x,t) = A_0 \exp(-k^2\mu t) \sin\{k[x - (c + k^2 d)t]\}$$

4.9 Suppose the Lax-Wendroff scheme is used to solve the wave equation ($c = 0.75$) for the initial condition

$$u(x,0) = 2\sin(\pi x) \qquad 0 \leqslant x \leqslant 2$$

and periodic boundary conditions with $\Delta x = 0.1$ and $\Delta t = 0.1$.

(*a*) Use the amplification factor to find the amplitude and phase errors after 10 steps.

(*b*) Use a truncated version of the modified equation to determine (approximately) the amplitude and phase errors after 10 steps.
Hint: The exact solution for the PDE

$$u_t + cu_x = du_{xxx} + \mu u_{xxxx}$$

with initial condition

$$u(x,0) = A_0 \sin(kx)$$

and periodic boundary conditions is given by

$$u(x,t) = A_0 \exp(k^4\mu t) \sin\{k[x - (c + k^2 d)t]\}$$

4.10 Repeat Prob. 4.9 for the leap-frog method.

4.11 Suppose the Rusanov scheme is used to solve the wave equation ($c = 0.5$) for the initial condition

$$u(x,0) = 2\sin(\pi x - 0.3\pi) \qquad 0 \leqslant x \leqslant 2$$

and periodic boundary conditions with $\Delta x = 0.1$, $\Delta t = 0.1$, and $\omega = 1.0$.

(*a*) Use the amplification factor to find the amplitude and phase errors after 10 steps.

(*b*) Use a truncated version of the modified equation to determine (approximately) the amplitude and phase errors after 10 steps, if the exact solution for the PDE

$$u_t + cu_x = \mu u_{xxxx} + du_{xxxxx}$$

with initial condition

$$u(x,0) = A_0 \sin(kx)$$

and periodic boundary conditions is given by

$$u(x,t) = A_0 \exp(k^4\mu t) \sin\{k[x - (c - k^4 d)t]\}$$

4.12 Derive the amplification factor for the leap frog method applied to the wave equation and determine the stability restriction for this scheme.

4.13 Repeat Prob. 4.12 for the second-order upwind method.

4.14 Show that the Rusanov method applied to the wave equation is equivalent to the following one-step scheme:

$$u_j^{n+1} = u_j^n - \nu(\mu_x\delta_x)\left(1 - \frac{\delta_x^2}{6}\right)u_j^n + \nu^2\delta_x^2\left(\frac{1}{2} + \frac{\delta_x^2}{8}\right)u_j^n - \frac{\nu^3}{6}(\mu_x\delta_x^3)u_j^n - \frac{\omega}{24}\delta_x^4 u_j^n$$

4.15 Evaluate the stability of the Rusanov method applied to the wave equation using the Fourier stability analysis. Hint: See Prob. 4.14.

4.16 The following second-order accurate explicit scheme for the wave equation was proposed by Crowley (1967):

$$u_j^{n+1} = u_j^n - \nu(\mu_x \delta_x)u_j^n + \frac{\nu^2}{2}(\mu_x^2 \delta_x^2)u_j^n - \frac{1}{8}\nu^3(\mu_x \delta_x^3)u_j^n$$

(*a*) Derive the modified equation for this scheme. Retain terms up to and including u_{xxxxx}.

(*b*) Evaluate the necessary condition for stability.

(*c*) Determine the errors in amplitude and phase for $\beta = 90°$ if this scheme is applied to the wave equation for 10 time steps with $\nu = 1$.

4.17 Solve the wave equation $u_t + u_x = 0$ on a digital computer using

(*a*) Lax scheme

(*b*) Lax-Wendroff scheme

for the initial condition

$$u(x,0) = \sin 2n\pi\left(\frac{x}{40}\right) \qquad 0 \leqslant x \leqslant 40$$

and periodic boundary conditions. Choose a 41 grid point mesh with $\Delta x = 1$ and compute to $t = 18$. Solve this problem for $n = 1, 3$ and $\nu = 1.0, 0.6, 0.3$ and compare graphically with the exact solution. Determine β for $n = 1$ and $n = 3$, and calculate the errors in amplitude and phase for each scheme with $\nu = 0.6$. Compare these errors with the errors appearing on the graphs.

4.18 Repeat Prob. 4.17 using the following schemes:

(*a*) Windward differencing scheme

(*b*) MacCormack scheme

4.19 Repeat Prob. 4.17 using the following schemes:

(*a*) MacCormack scheme

(*b*) Rusanov scheme ($\omega = 3$)

4.20 Solve the wave equation $u_t + u_x = 0$ on a digital computer using

(*a*) Windward differencing scheme

(*b*) MacCormack scheme

for the initial conditions

$$u(x,0) = 1 \qquad x \leqslant 10$$
$$u(x,0) = 0 \qquad x > 10$$

and Dirichlet boundary conditions. Choose a 41 grid point mesh with $\Delta x = 1$ and compute to $t = 18$. Solve this problem for $\nu = 1.0, 0.6$, and 0.3 and compare graphically with the exact solution.

4.21 Apply the windward differencing scheme to the two-dimensional wave equation

$$u_t + c(u_x + u_y) = 0$$

and determine the stability of the resulting scheme.

4.22 Derive the modified equation for the simple implicit method applied to the 1-D heat equation. Retain terms up to and including u_{xxxxxx}.

4.23 Evaluate the stability of the combined method B applied to the 1-D heat equation.

4.24 Determine the amplification factor of the ADE method of Saul'yev and examine the stability.

4.25 For the grid points $(i + j + n)$ even, show that the hopscotch method reduces to

$$u_{i,j}^{n+2} = 2u_{i,j}^{n+1} - u_{i,j}^n$$

4.26 Use the simple explicit method to solve the 1-D heat equation on the computational grid (Fig. P4.1) with boundary conditions

$$u_1^n = 2 = u_3^n$$

and initial conditions

$$u_1^1 = 2 = u_3^1 \qquad u_2^1 = 1$$

Show that if $r = \frac{1}{4}$, the steady-state value of u along $j = 2$ becomes

$$u_2^\infty = \lim_{n \to \infty} \sum_{k=1}^{n} \frac{1}{2^{k-1}}$$

Note that this infinite series is a geometric series that has a known sum.

4.27 Apply the ADI scheme to the 2-D heat equation and find u^{n+1} at the internal grid points in the mesh shown in Fig. P4.2 for $r_x = r_y = 2$. The initial conditions are

$$u^n = 1 - \frac{x}{3\,\Delta x} \qquad \text{along } y = 0$$

$$u^n = 1 - \frac{y}{2\,\Delta y} \qquad \text{along } x = 0$$

$$u^n = 0 \qquad \text{everywhere else}$$

and the boundary conditions remain fixed at their initial values.

4.28 Solve the heat equation $u_t = 0.2u_{xx}$ on a digital computer using

(*a*) Simple explicit method

(*b*) Barakat and Clark ADE method

for the initial condition

$$u(x,0) = 100 \sin \frac{\pi x}{L} \qquad L = 1$$

and boundary conditions

$$u(0,t) = u(L,t) = 0$$

Compute to $t = 0.5$ using the parameters in Table P4.1 (if possible) and compare graphically with the exact solution.

4.29 Repeat Prob. 4.28 using the Crank-Nicolson scheme.

4.30 Repeat Prob. 4.28 using the DuFort-Frankel scheme.

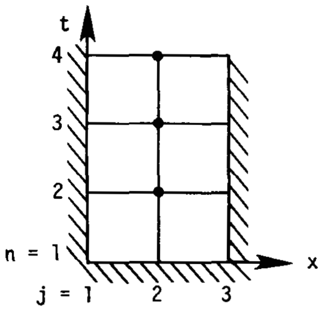

Figure P4.1

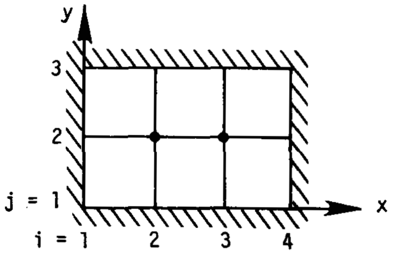

Figure P4.2

Table P4.1

Case	Number of grid points	r
1	11	0.25
2	11	0.50
3	16	0.50
4	11	1.00
5	11	2.00

4.31 The heat equation

$$\frac{\partial T}{\partial t} = \alpha \frac{\partial^2 T}{\partial x^2}$$

governs the time-dependent temperature distribution in a homogeneous constant property solid under conditions where the temperature varies only in one space dimension. Physically, this may be nearly realized in a long thin rod or very large (infinite) wall of finite thickness.

Consider a large wall of thickness L whose initial temperature is given by $T(t, x) = c \sin \pi x/L$. If the faces of the wall continue to be held at 0°, then a solution for the temperature at $t > 0$, $0 \leqslant x \leqslant L$ is

$$T(t, x) = c \exp\left(\frac{-\alpha\pi^2 t}{L^2}\right) \sin \frac{\pi x}{L}$$

For this problem let $c = 100°\text{C}$, $L = 1$ m, $\alpha = 0.02\ \text{m}^2/\text{h}$. We will consider two explicit methods of solution, A. Simple explicit method, Eq. (4.73). Stability requires that $\alpha\,\Delta t/(\Delta x)^2 \leqslant \frac{1}{2}$ for this method, B. Alternating direction explicit (ADE) method, Eq. (4.107). This particular version of the ADE method was suggested by Barakat and Clark (1966). In this algorithm, the equation for p_j^{n+1} can be solved explicitly starting from the boundary at $x = 0$, whereas the equation for q_j^{n+1} should be solved starting at the boundary at $x = L$. There is no stability constraint on the size of the time step for this method. Develop computer programs to solve the problem described above by methods A and B. Also, you will want to provide a capability for evaluating the exact solution for purposes of comparison. Make at least the following comparisons:

1. For $\Delta x = 0.1$, $\Delta t = 0.1$ [resulting in $\alpha\,\Delta t/(\Delta x)^2 = 0.2$], compare the results from methods A and B and the exact solution for $t = 10$ h. A graphical comparison is suggested.
2. Repeat the above comparison after refining the space grid, i.e., let $\Delta x = 0.066667$ (15 increments). Is the reduction in error as suggested by $O[(\Delta x)^2]$?
3. For $\Delta x = 0.1$ choose Δt such that $\alpha\,\Delta t/(\Delta x)^2 = 0.5$ and compare the predictions of methods A and B and the exact solution for $t \approx 10$ h.
4. Demonstrate that method A does become unstable as $\alpha\,\Delta t/(\Delta x)^2$ exceeds 0.5. One suggestion is to plot the centerline temperature vs. time for $\alpha\,\Delta t/(\Delta x)^2 = 0.6$ for 10–20 hours of problem time.
5. For $\Delta x = 0.1$, choose Δt such that $\alpha\,\Delta t/(\Delta x)^2 = 1.0$ and compare the results of method B and the exact solution for $t \approx 10$ h.
6. Increment $\alpha\,\Delta t/(\Delta x)^2$ to 2, then 3, etc., and repeat comparison 5 above until the agreement with the exact solution becomes noticeably poor.

4.32 Work Prob. 4.31 letting method B be the Crank-Nicolson scheme.

4.33 Work Prob. 4.31 letting method B be the simple implicit scheme.

4.34 Derive a way to solve the problem described in Prob. 4.31 utilizing the fourth-order accurate representation of the second derivative given by Eq. (3.35).

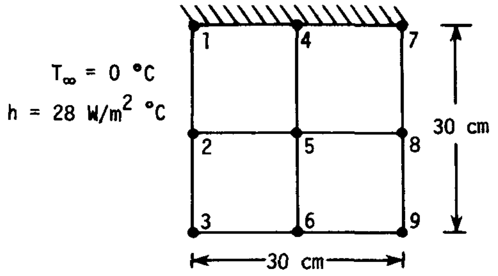

Figure P4.3

4.35 Use the difference scheme of Eq. (3.35) for second derivatives

$$\frac{\partial^2 u}{\partial x^2} \approx \frac{\delta_x^2 u_{i,j}}{h^2(1 + \delta_x^2/12)}$$

to develop a finite-difference representation for Laplace's equation where $\Delta x = \Delta y$. Write out the scheme explicitly in terms of u on the finite-difference mesh. What is the T.E. of this representation?

4.36 Evaluate the T.E. of the difference scheme of Eq. (4.114) for Laplace's equation (*a*) when $\Delta x = \Delta y$, (*b*) when $\Delta x \neq \Delta y$.

4.37 What is the T.E. for the difference equation employing the nine-point scheme of Eq. (4.114) with $\Delta x = \Delta y$ for the Poisson equation $u_{xx} + u_{yy} = x + y$?

4.38 In the cross section illustrated in Fig. P4.3, the surface 1-4-7 is insulated (adiabatic). The convective heat transfer coefficient at surface 1-2-3 is 28 W/m^2°C. The thermal conductivity of the solid material is 3.5 W/m°C. The temperature at nodes 3, 6, 7, 8, 9 is held constant at 100°C. Using Gauss-Seidel iteration, compute the temperature at nodes 1, 2, 4, and 5.

4.39 A cylindrical pin fin (Fig. P4.4) is attached to a 200°C wall while its surface is exposed to a gas at 30°C. The convection heat transfer coefficient is 300 W/m^2°C. The fin is made of stainless steel with a thermal conductivity of 18 W/m°C. Use five subdivisions and find the steady-state nodal temperatures by Gauss-Seidel iteration. Compute the total rate at which heat is transferred from the fin. You may neglect the heat loss from the outer end of the fin (i.e., assume end is adiabatic).

4.40 Determine the inviscid (ideal) flow in a 2-D channel containing a cylinder. Use a stream function formulation in which

$$u = \frac{\partial \psi}{\partial y} \qquad v = -\frac{\partial \psi}{\partial x}$$

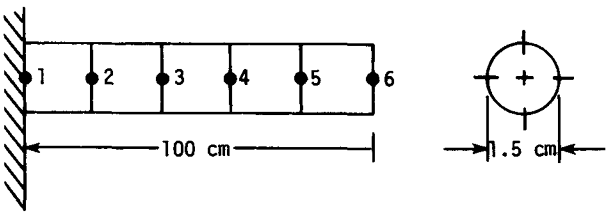

Figure P4.4

Without viscous effects, the rotation of fluid particles cannot be changed, leading to

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

The flow domain is sketched in Fig. P4.5. Use a grid of $\Delta x = \Delta y = 0.5$ cm wherever possible. Unequal grid spacing will be needed near the cylinder. At the inlet (along A-B) the flow is uniform at 1 m/s. Along B-C the stream function remains constant. Along C-D, $\partial\psi/\partial x = 0$. The stream function is zero along D-E-A.

(*a*) Determine the values of ψ throughout the flow.

(*b*) Determine the velocity distribution along C-D.

(*c*) Sketch as best you can the streamline pattern for the flow.

(*d*) From the computed results, estimate the pressure coefficient at the top of the cylinder (point D) and compare this with the pressure coefficient from the "exact" analytical solution for inviscid flow around a cylinder in a stream of infinite extent.

4.41 Solve the steady-state, 2-D heat conduction equation in the unit square, $0 < x < 1$, $0 < y < 1$, by finite-differences using mesh increments $\Delta x = \Delta y = 0.1$ and 0.05. Compare the center temperatures with the exact solution. Use boundary conditions

$$T = 0 \quad \text{at } x = 0,\, x = 1$$

$$\frac{\partial T}{\partial y} = 0 \quad \text{at } y = 0$$

$$T = \sin(\pi x) \quad \text{at } y = 1$$

4.42 For the conditions of Prob. 4.41,

(*a*) Use the Gauss-Seidel iterative procedure with SOR. Establish a convergence criteria. For each mesh size, use $\omega = 1$ and at least three other values of ω between 1 and 2 in an attempt to determine an appropriate ω_{opt}. Compare this with the value predicted by the Young-Frankel theory. For each calculation, use the same initial guess. Make a plot of the number of iterations required for convergence to the same specified tolerance vs. ω. Also compare the computer solution at the center with the exact solution.

(*b*) Devise and explain an SOR point iterative algorithm based on the red-black (checkerboard) strategy. Use this algorithm for the same series of computations and comparisons as indicated for Prob. 4.42(*a*). Compare the number of iterations required with the results in Prob. 4.42(*a*).

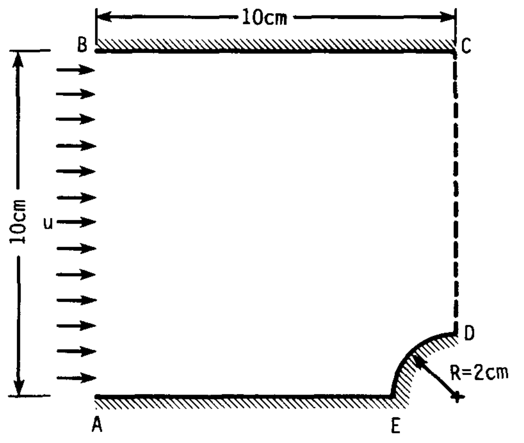

Figure P4.5

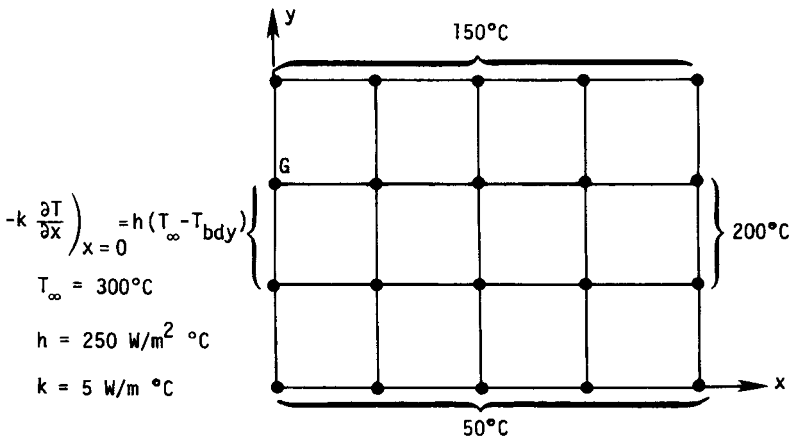
y
150°C
G
$-k \frac{\partial T}{\partial x}\Big)_{x=0} = h(T_\infty - T_{bdy})$
T_∞ = 300°C
h = 250 W/m² °C
k = 5 W/m °C
200°C
x
50°C

Figure P4.6

(*c*) Using the same initial guess and convergence criterion as in Prob. 4.42(*a*), solve the problem for $\Delta x = \Delta y = 0.1$ using the line iterative procedure with SOR (SLOR). Solve the problem for $\omega = 1$ and at least three other values of ω. Does this scheme appear to have the same ω_{opt} for this problem? Compare the number of iterations required with the results in Prob. 4.42(*a*) and 4.42(*b*).

4.43 Consider steady-state conduction governed by Laplace's equation in the 2-D domain shown in Fig. P4.6. The boundary conditions are shown in the figure. The mesh is square, i.e., $\Delta x = \Delta y = 0.02$ m.

(*a*) Develop an approximate difference equation for the boundary temperature at point *G* using the control-volume approach.

(*b*) After obtaining a suitable finite-difference representation for Laplace's equation, use Gauss-Seidel iteration to obtain the steady-state temperature distribution.

4.44 Solve Prob. 4.43 using the line iterative method.

4.45 It is required to estimate the temperature distribution in the two-dimensional wall of a combustion chamber at steady state. The geometry has been simplified for this preliminary analysis and is given in Fig. P4.7. Write a computer program using Gauss-Seidel iteration with SOR to solve this problem. Give careful attention to the equations at the boundaries. Use grid spacing of 2 cm ($\Delta x = \Delta y$), resulting in a 6×11 mesh, and use a thermal conductivity of 20 W/m^2°C.

(*a*) Compute the steady-state temperature distribution.

(*b*) Compute the rate of heat transfer to the top, and check to see how closely it matches the heat removed by the coolant.

(*c*) For the same convergence criteria, repeat the calculation for at least three values of the relaxation parameter ω. If sufficient computer time is available, make a more detailed search for ω_{opt}.

4.46 Solve Prob. 4.45 using the line iterative method.

4.47 Solve Prob. 4.45 using the ADI method.

4.48 Write a computer program to solve Laplace's equation with Dirichlet boundary conditions on a unit square using the Gauss-Seidel procedure with

(*a*) SOR (using both $\omega = 1$ and $\omega = \omega_{\text{opt}}$)

(*b*) a two-level multigrid scheme

Compare the relative efficiencies of the three procedures for side temperatures (going counterclockwise around the square) of 20, 40, 80, and 100 using three different grids, 9×9, 17×17, and 33×33. Start all calculations with initial guesses of zero. Indicate the solution

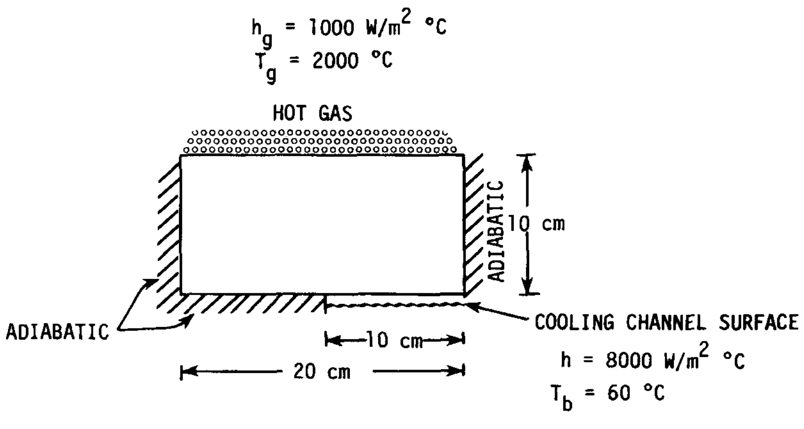

Figure P4.7

obtained at the center. Determine the multigrid effort in terms of "work units" of equivalent fine-grid iterations. Four coarse-grid sweeps approximately equal one fine-grid sweep. Three or four fine-grid sweeps per cycle should work well. Converge the coarse-grid calculation. A number of convergence criteria are workable. Monitoring the magnitude of the maximum change from one iterative sweep to the next, normalized with the maximum or average boundary value, is easy to implement:

$$\frac{|f^{k+1} - f^k|}{|f_{\text{ref}}|}$$

where f denotes either u or Δu and f_{ref} is the maximum or average value of u on the boundary. A convergence level of 10^{-5} on that basis is suggested.

4.49 Use the Lax method to solve the inviscid Burgers equation using a mesh with 51 points in the x direction. Solve this equation for a right propagating discontinuity with initial data $u = 1$ on the first 11 mesh points and $u = 0$ at all other points. Repeat your calculations for Courant numbers of 1.0, 0.6, and 0.3 and compare your numerical solutions with the analytical solution at the same time.

4.50 Repeat Prob. 4.49 using MacCormack's method. Use both a forward-backward and a backward-forward predictor-corrector sequence.

4.51 Repeat Prob. 4.49 using the WKL method.

4.52 Repeat Prob. 4.49 using the Beam-Warming method.

4.53 Solve the inviscid Burgers equation for an expansion with initial data $u = 0$ for the first 21 mesh points and $u = 1$ elsewhere. Use MacCormack's method with both forward-backward and backward-forward predictor-corrector sequences. Compare your results at two different Courant numbers with the analytic solution.

4.54 Repeat Prob. 4.53 using the Beam-Warming method (trapezoidal) and the Euler implicit scheme.

4.55 Solve the inviscid Burgers equation for a standing discontinuity. Initialize using $u = 1$ at the left end point and $u = -1$ at the right end point and zero everywhere else. Apply MacCormack's method to this problem.

4.56 Repeat Prob. 4.55 using the Beam-Warming scheme.

4.57 Determine the solution of the inviscid Burgers equation for the double-shock profile given by $u_l = 1.0$ and $u_r = 0.5$. Compute this solution using MacCormack's scheme and the Godunov method. What is the analytic solution for this set of initial conditions? Compare your numerical calculation with the analytical solution.

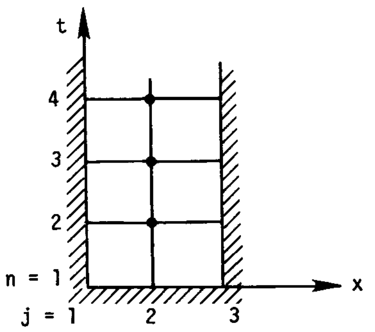

Figure P4.8

4.58 Repeat Prob. 4.49 using the schemes listed below. Remember that the stability bound for Godunov's method is $\nu \leqslant 0.5$.

(*a*) Godunov scheme

(*b*) Roe scheme (first order)

(*c*) Enquist-Osher scheme

4.59 Repeat Prob. 4.53 using the methods listed below:

(*a*) Godunov scheme

(*b*) Roe scheme (first-order) with and without the entropy fix

(*c*) Enquist-Osher scheme

4.60 Verify that the extrapolation formulas given in Section 4.4.11 give central or upwind differences for $\kappa = \pm 1$.

4.61 Solve Prob. 4.49 using the second-order Roe scheme with and without the use of limiters. Use the minmod limiter and the van Leer limiter in your calculations. Use an initial profile of $u_l = 1.0$ and $u_r = 0.5$ for this problem.

4.62 A numerical method is said to be monotone if it does not produce oscillations in the numerical solution. Schemes for solving the inviscid Burgers equation may be written in the form

$$u^{n+1} = H(u_{j-k}^n, \ldots, u_{j+k}^n)$$

The condition for monotonicity requires H to be a monotone increasing function of its arguments, i.e., $\partial H/\partial u_j \geqslant 0$. Show that the Lax method is monotone and that the first-order upwind scheme is monotone. What conditions must be satisfied to meet this condition?

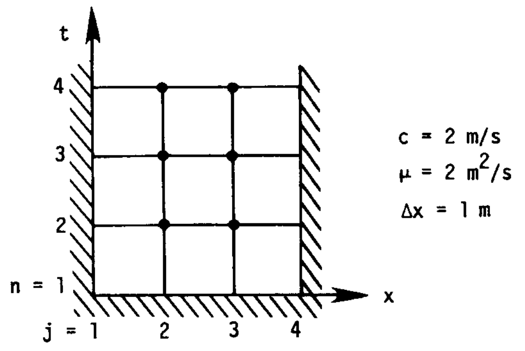

Figure P4.9

4.63 In the MUSCL approach the dependent variables are extrapolated directly to the cell boundaries, and the flux terms are recalculated. In the non-MUSCL approach, the fluxes are directly extrapolated to the interface boundaries. Repeat Prob. 4.49 using Roe's scheme with both the MUSCL and non-MUSCL approaches. Can you draw any conclusions from your results?
4.64 Solve the inviscid Burgers equation using the Lax-Wendroff method with an initial profile that is linear between the left and right boundary values of 1 and -1. This profile will become steeper as the solution progresses until a shock wave ultimately results. Perform the same experiment with the Lax method, and compare your results. Does this provide any insight into the Godunov theorem?
4.65 In Section 4.4.12 the sufficient conditions for a scheme to be TVD were given. Show that the Lax scheme satisfies these conditions, while the Lax-Wendroff scheme does not.
4.66 Solve Prob. 4.49 using the Roe-Sweby scheme. Compare your results with those obtained earlier with Roe's method using the MUSCL approach (Prob. 4.63), and comment on any differences.
4.67 Show graphically the exact steady-state solution of Eq. (4.272) for the boundary conditions

$$u(0,t) = 1$$
$$u(1,t) = 0$$

and $\mu = 0.1$.
4.68 Verify that Eq. (4.283) is an exact stationary solution of Eq. (4.282).
4.69 Derive stability conditions for the FTCS method applied to the 1-D linearized Burgers equation.
4.70 Derive the stability conditions for the upwind difference scheme given by Eq. (4.300).
4.71 Suppose the FTCS method is used to solve the linearized Burgers equation ($c = 0.5$, $\mu = 0.01$) for the initial condition

$$u(x,0) = \sin(2\pi x) \qquad 0 \leqslant x \leqslant 2$$

and periodic boundary conditions with $\Delta x = 0.02$ and $\Delta t = 0.02$. Find the amplitude error and the phase error after 20 steps.
Hint: The exact solution for the linearized Burgers equation for the above initial and boundary conditions is

$$u(x,t) = \exp(-4\pi^2\mu t)\sin[2\pi(x - ct)]$$

4.72 Use the FTCS method to solve the linearized Burgers equation for the initial condition

$$u(x,0) = 0 \qquad 0 \leqslant x \leqslant 1$$

and the boundary conditions

$$u(0,t) = 100$$
$$u(1,t) = 0$$

on a 21 grid point mesh. Find the steady-state solution for the conditions
(*a*) $r = 0.50$, $\nu = 0.25$
(*b*) $r = 0.50$, $\nu = 1.00$
(*c*) $r = 0.10$, $\nu = 0.40$
(*d*) $r = 0.05$, $\nu = 0.50$
and compare the numerical solutions with the exact solution.
4.73 Repeat Prob. 4.72 using the scheme proposed by Leonard.
4.74 Repeat Prob. 4.72 using the leap frog/DuFort-Frankel method.
4.75 Repeat Prob. 4.72 using the Allen-Cheng method.
4.76 Use the Fourier stability analysis to determine the stability limitations of the scheme proposed by Leonard, Eq. (4.302).
4.77 Determine the modified equation for the Allen-Cheng method. Retain terms up to and including u_{xxx}.

4.78 Apply the Brailovskaya scheme to the linearized Burgers equation on the computational grid shown in Fig. P4.8 and show that the steady-state value for u at $j = 2$ is

$$u_2^\infty = \lim_{n\to\infty} \sum_{i=1}^{n} \frac{1}{3^{n-i}} = \frac{3}{2}$$

Boundary conditions are $u_1^n = \frac{3}{2} = u_3^n$, and the initial condition is $u_2^1 = 1$. Do not use a digital computer to solve this problem.

4.79 Apply the Beam-Warming scheme with Euler implicit time differencing to the linearized Burgers equation on the computational grid shown in Fig. P4.9, and determine the steady-state values for u at $j = 2$ and $j = 3$. The boundary conditions are $u_1^n = 1$, $u_4^n = 4$, and the initial conditions are $u_2^1 = 0 = u_3^1$. Do not use a digital computer to solve this problem.

4.80 Apply the two-step Lax-Wendroff method to the PDE

$$u_t + F_x + uu_{xxx} = 0$$

where $F = F(u)$. Develop the final finite-difference equations.

4.81 Solve the linearized Burgers equation using

(*a*) FTCS method

(*b*) Upwind method, Eq. (4.300)

(*c*) Leonard method, Eq. (4.302)

for the initial condition

$$u(x,0) = 0 \qquad 0 \leqslant x \leqslant 1$$

and the boundary conditions

$$u(0,t) = 100$$
$$u(1,t) = 0$$

on a 21 grid point mesh. Find the steady-state solution for $r = 0.10$ and $\nu = 0.40$, and compare the numerical solutions with the exact solution.

4.82 Repeat Prob. 4.81 using the following methods

(*a*) Leap frog/DuFort-Frankel method

(*b*) Allen-Cheng method

(*c*) MacCormack method, Eq. (4.312)

4.83 Repeat Prob. 4.81 using the Briley-McDonald method with Euler implicit time differencing.

4.84 Solve the generalized Burgers equation

$$u_t + \left(\frac{1}{2} - u\right)u_x = 0.001u_{xx}$$

using

(*a*) MacCormack's scheme

(*b*) Roe's scheme

for the initial condition

$$u = \frac{1}{2}\{1 + \tanh[250(x - 20)]\} \qquad 0 \leqslant x \leqslant 40$$

and exact Dirichlet boundary conditions. Choose a 41 grid point mesh with $\Delta x = 1$, and compute to $t = 18$. Solve this problem for $\Delta t = 1.0$ and 0.5, and compare graphically with the exact stationary solution.