Chapter 2
Fundamentals

Before attempting to develop and apply numerical algorithms for the Euler and Navier-Stokes equations it is worthwhile to learn as much as possible by studying the behaviour of such methods when applied to simpler model equations. In this chapter, we will do just that, using two model equations which are linear, scalar partial differential equations (PDEs) that represent physical phenomena relevant to fluid dynamics. This chapter provides a concise summary of our earlier book *Fundamentals of Fluid Dynamics* [1], to which the reader is referred for further details.

2.1 Model Equations

2.1.1 The Linear Convection Equation

The linear convection equation provides a simple model for convection and wave propagation phenomena. It is given by

$$\frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x} = 0, \tag{2.1}$$

where $u(x, t)$ is a scalar quantity propagating with speed a, a real constant which may be positive or negative. In the absence of boundaries, for example on an infinite domain, an initial waveform retains its shape as it propagates in the direction of increasing x if a is positive and in the direction of decreasing x if a is negative. Despite its simplicity, the linear convection equation provides a stiff test for a numerical method, as it is difficult to preserve the initial waveform when it is propagated over long distances.

The linear convection equation is a good model equation in the development of numerical algorithms for the Euler equations, which include both convection and wave propagation phenomena. The one-dimensional Euler equations can be

T. H. Pulliam and D. W. Zingg, *Fundamental Algorithms in Computational Fluid Dynamics*, Scientific Computation, DOI: 10.1007/978-3-319-05053-9_2,

diagonalized so that they can be written as three equations in the form of the linear convection equation, although they of course remain nonlinear and coupled. The quantities propagating are known as *Riemann invariants*, and the speeds at which they propagate are the fluid velocity, the fluid velocity plus the speed of sound, and the fluid velocity minus the speed of sound. If the fluid velocity is positive but less than the speed of sound, i.e. the flow is subsonic, then the first two wave speeds will be positive, and the third will be negative. When using the linear convection equation as a model equation for the Euler equations, one must therefore ensure that wave speeds of arbitrary sign are considered.

If one considers a finite domain, say $0 \leq x \leq 2\pi$, then boundary conditions are required. The most natural boundary conditions are inflow-outflow conditions, which depend on the sign of a. If a is positive, then $x = 0$ is the inflow boundary, and $x = 2\pi$ is the outflow boundary. If a is negative, these roles are reversed. In both cases, $u(t)$ must be specified at the inflow boundary, but no boundary condition can be specified at the outflow boundary.

An alternative specification of boundary conditions, known as periodic boundary conditions, can be convenient for our purpose here. With periodic boundary conditions, a waveform leaving one end of the domain reenters at the other end. The domain can be visualized as a circle, and the waveform simply propagates repeatedly around the circle. This essentially eliminates any boundary information from entering the solution, which is thus determined solely by the initial condition. The use of periodic boundary conditions also permits numerical experiments with arbitrarily long propagation distances, independent of the size of the domain. Each time the initial waveform travels through the entire domain, it should return unaltered to the initial condition.

2.1.2 The Diffusion Equation

Diffusion caused by molecular motion in a continuum fluid is another important physical phenomenon in fluid dynamics. A simple linear model equation for a diffusive process is

$$\frac{\partial u}{\partial t} = \nu \frac{\partial^2 u}{\partial x^2}, \tag{2.2}$$

where ν is a positive real constant. For example, with u representing the temperature, this parabolic PDE governs the diffusion of heat in one dimension. Boundary conditions can be periodic, Dirichlet (specified u), Neumann (specified $\partial u/\partial x$), or mixed Dirichlet/Neumann. In studying numerical algorithms, it can be useful to introduce a source term into the diffusion equation as follows:

$$\frac{\partial u}{\partial t} = \nu \left[\frac{\partial^2 u}{\partial x^2} - g(x) \right]. \tag{2.3}$$

In this case, the equation has a steady-state solution that satisfies

$$\frac{\partial^2 u}{\partial x^2} - g(x) = 0. \tag{2.4}$$

2.2 Finite-Difference Methods

2.2.1 Basic Concepts: Taylor Series

We observe that the two model equations contain a number of derivative terms in space and time. In a finite-difference method, a spatial derivative at a given point in space is approximated using values of u at nearby points in space. Similarly, a temporal derivative at a specific point in time is approximated using values of u at different values of time. This is facilitated by a grid or mesh, as shown in Fig. 2.1, where the values of x at the grid points are given by x_j, and the values of t are given by t_n. Hence j is known as the spatial index and n as the temporal index. For the present exposition, we will consider equispaced meshes, and hence

$$x = x_j = j\Delta x \tag{2.5}$$
$$t = t_n = n\Delta t = nh, \tag{2.6}$$

where Δx is the spacing in x, and Δt the spacing in t, as shown in Fig. 2.1. Note that $h = \Delta t$ throughout.

Let us consider a spatial derivative initially. Assuming that a function $u(x, t)$ is known only at discrete values of x, how can one accurately approximate partial derivatives such as

$$\frac{\partial u}{\partial x} \quad \text{or} \quad \frac{\partial^2 u}{\partial x^2} \quad ? \tag{2.7}$$

From the definition of a derivative, or a simple geometric argument related to the tangent to a curve, one can easily postulate the following approximations for a first derivative:

$$\left(\frac{\partial u}{\partial x}\right)_j \approx \frac{u_{j+1} - u_j}{\Delta x} \quad \text{or} \quad \left(\frac{\partial u}{\partial x}\right)_j \approx \frac{u_j - u_{j-1}}{\Delta x}, \tag{2.8}$$

known respectively as a forward and a backward difference approximation. It is clear that these can provide accurate approximations if Δx is sufficiently small and that a suitable choice of Δx will depend on the properties of the function. A particularly astute reader may even postulate the centered difference approximation given by

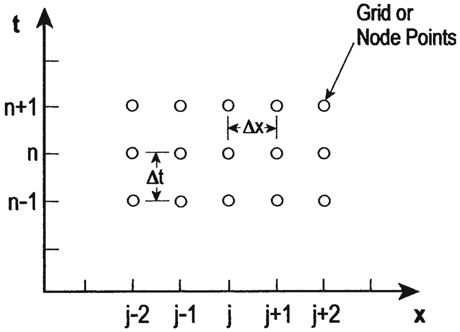


Fig. 2.1 Space–time grid arrangement

$$\left(\frac{\partial u}{\partial x}\right)_j \approx \frac{u_{j+1} - u_{j-1}}{2\Delta x}, \tag{2.9}$$

which can be justified by a geometric argument or by fitting a parabola to the three points u_{j-1}, u_j, u_{j+1} and determining the first derivative of the parabola at x_j. Finding an approximation to a second derivative is only slightly less intuitive, as one can apply a first-derivative approximation twice or determine the second derivative of the unique parabola that goes through the three points to obtain the following approximation to a second derivative:

$$\left(\frac{\partial^2 u}{\partial x^2}\right)_j \approx \frac{u_{j+1} - 2u_j + u_{j-1}}{\Delta x^2}. \tag{2.10}$$

The above intuitive approach is limited, providing no information about the accuracy of these approximations. A deeper understanding of finite-difference approximations and a general approach to deriving them can be obtained using Taylor series. Consider the following expansion of $u(x + k\Delta x) = u(j\Delta x + k\Delta x) = u_{j+k}$ about x_j, where we assume that all of the derivatives exist:

$$\begin{aligned} u_{j+k} = u_j + (k\Delta x)\left(\frac{\partial u}{\partial x}\right)_j + \frac{1}{2}(k\Delta x)^2\left(\frac{\partial^2 u}{\partial x^2}\right)_j + \dots \\ + \frac{1}{n!}(k\Delta x)^n\left(\frac{\partial^n u}{\partial x^n}\right)_j + \dots . \end{aligned} \tag{2.11}$$

For example, substituting $k = \pm 1$ into the above expression gives the Taylor series expansions for $u_{j\pm 1}$:

$$\begin{aligned} u_{j\pm 1} = u_j \pm (\Delta x)\left(\frac{\partial u}{\partial x}\right)_j + \frac{1}{2}(\Delta x)^2\left(\frac{\partial^2 u}{\partial x^2}\right)_j \pm \frac{1}{6}(\Delta x)^3\left(\frac{\partial^3 u}{\partial x^3}\right)_j \\ + \frac{1}{24}(\Delta x)^4\left(\frac{\partial^4 u}{\partial x^4}\right)_j \pm \dots . \end{aligned} \tag{2.12}$$

Subtracting u_j from the Taylor series expansion for u_{j+1} and dividing by Δx gives

$$\frac{u_{j+1}-u_j}{\Delta x} = \left(\frac{\partial u}{\partial x}\right)_j + \frac{1}{2}(\Delta x)\left(\frac{\partial^2 u}{\partial x^2}\right)_j + \dots . \tag{2.13}$$

This shows that the forward difference approximation given in (2.8) is a reasonable approximation for $\left(\frac{\partial u}{\partial x}\right)_j$ as long as Δx is small relative to some pertinent length scale. Moreover, in the limit as $\Delta x \to 0$, the leading term in the error is proportional to Δx. The *order of accuracy* of an approximation is given by the exponent of Δx in the leading error term, i.e. the lowest exponent of Δx in the error. Hence the finite-difference approximation given in (2.13) is a *first-order approximation* to a first derivative. If the mesh spacing Δx is reduced by a factor of two, the leading error term in a first-order approximation will also be reduced by a factor of two.

Similarly, subtracting the Taylor series expansion for u_{j-1} from that for u_{j+1} and dividing by $2\Delta x$ gives

$$\frac{u_{j+1}-u_{j-1}}{2\Delta x} = \left(\frac{\partial u}{\partial x}\right)_j + \frac{1}{6}\Delta x^2\left(\frac{\partial^3 u}{\partial x^3}\right)_j + \frac{1}{120}\Delta x^4\left(\frac{\partial^5 u}{\partial x^5}\right)_j \dots . \tag{2.14}$$

This shows that the centered difference approximation given in (2.9) is second-order accurate. If the mesh spacing Δx is reduced by a factor of two, the leading error term will be reduced by a factor of four. Hence, as Δx is reduced, the second-order approximation rapidly becomes more accurate than the first-order approximation. Using Taylor series expansions, one can demonstrate that the approximation to a second derivative given in (2.10) is also second-order accurate.

Finite-difference formulas can be generalized to arbitrary derivatives and arbitrary orders of accuracy. A Taylor table provides a convenient mechanism for deriving finite-difference operators (see Lomax et al. [1]). In each case, the derivative at node j is approximated using a linear combination of function values at node j and a specified number of neighbouring nodes, and the Taylor table enables one to find the coefficients that maximize the order of accuracy. For example, centered fourth-order approximations to first and second derivatives are given by

$$\left(\frac{\partial u}{\partial x}\right)_j = \frac{1}{12\Delta x}(u_{j-2} - 8u_{j-1} + 8u_{j+1} - u_{j+2}) + O(\Delta x^4) \tag{2.15}$$

$$\left(\frac{\partial^2 u}{\partial x^2}\right)_j = \frac{1}{12\Delta x^2}(-u_{j-2} + 16u_{j-1} - 30u_j + 16u_{j+1} - u_{j+2}) + O(\Delta x^4). \tag{2.16}$$

Noncentered schemes can also be useful. For example, the following is a second-order backward-difference approximation to a first derivative using data from $j-2$ to j:

$$\left(\frac{\partial u}{\partial x}\right)_j = \frac{1}{2\Delta x}\big(u_{j-2} - 4u_{j-1} + 3u_j\big) + O(\Delta x^2). \tag{2.17}$$

A *biased* third-order approximation to a first derivative using data from $j-2$ to $j+1$ is given by:

$$\left(\frac{\partial u}{\partial x}\right)_j = \frac{1}{6\Delta x}(u_{j-2} - 6u_{j-1} + 3u_j + 2u_{j+1}) + O(\Delta x^3). \tag{2.18}$$

Finally, finite-difference schemes can be further generalized to include *compact or Padé* schemes that define a linear combination of the derivatives at point j and a specified number of its neighbours as a linear combination of function values at node j and a (possibly different) specified number of neighbours. For example, the operator

$$\left(\frac{\partial u}{\partial x}\right)_{j-1} + 4\left(\frac{\partial u}{\partial x}\right)_j + \left(\frac{\partial u}{\partial x}\right)_{j+1} = \frac{3}{\Delta x}(-u_{j-1} + u_{j+1}) + O(\Delta x^4). \tag{2.19}$$

provides a fourth-order approximation to a first derivative. Compact schemes can also be easily derived using a Taylor table.

2.2.2 The Modified Wavenumber

The leading error term provides a fairly limited understanding of the accuracy of a finite-difference approximation. More detailed information can be obtained through the *modified wavenumber*. We introduce this concept by deriving the modified wavenumber for a second-order centered difference approximation, given by

$$(\delta_x u)_j = \frac{u_{j+1} - u_{j-1}}{2\Delta x}. \tag{2.20}$$

First, consider the exact first derivative of the function $\mathrm{e}^{\mathrm{i}\kappa x}$:

$$\frac{\partial \mathrm{e}^{\mathrm{i}\kappa x}}{\partial x} = \mathrm{i}\kappa \mathrm{e}^{\mathrm{i}\kappa x}. \tag{2.21}$$

Applying the operator given in (2.20) to $u_j = \mathrm{e}^{\mathrm{i}\kappa x_j}$, where $x_j = j\Delta x$, we get

$$\begin{aligned}(\delta_x u)_j &= \frac{\mathrm{e}^{\mathrm{i}\kappa\Delta x(j+1)} - \mathrm{e}^{\mathrm{i}\kappa\Delta x(j-1)}}{2\Delta x}\\ &= \frac{(\mathrm{e}^{\mathrm{i}\kappa\Delta x} - \mathrm{e}^{-\mathrm{i}\kappa\Delta x})\mathrm{e}^{\mathrm{i}\kappa x_j}}{2\Delta x}\end{aligned}$$

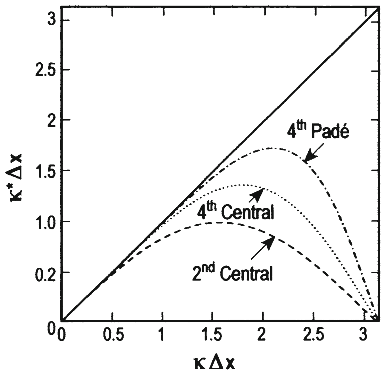


Fig. 2.2 Modified wavenumber for various schemes

$$
\begin{aligned}
&= \frac{1}{2\Delta x}[(\cos\kappa\Delta x + \mathrm{i}\sin\kappa\Delta x) - (\cos\kappa\Delta x - \mathrm{i}\sin\kappa\Delta x)]\mathrm{e}^{\mathrm{i}\kappa x_j}\\
&= \mathrm{i}\frac{\sin\kappa\Delta x}{\Delta x}\mathrm{e}^{\mathrm{i}\kappa x_j}\\
&= \mathrm{i}\kappa^*\mathrm{e}^{\mathrm{i}\kappa x_j},
\end{aligned}
\tag{2.22}
$$

where κ^* is the modified wavenumber. The modified wavenumber is so named because it appears where the wavenumber, κ, appears in the exact expression (2.21). Thus the degree to which the modified wavenumber approximates the actual wavenumber is a measure of the accuracy of the approximation.

For the second-order centered difference operator the modified wavenumber is given by

$$\kappa^* = \frac{\sin\kappa\Delta x}{\Delta x}. \tag{2.23}$$

Equation (2.23) is plotted in Fig. 2.2, along with similar relations for the standard fourth-order centered difference scheme and the fourth-order Padé scheme. The expression for the modified wavenumber provides the accuracy with which a given wavenumber component of the solution is resolved for the entire wavenumber range available in a mesh of a given size, $0 \le \kappa\Delta x \le \pi$. The value of $\kappa\Delta x$ can be related to the mesh resolution through the notion of points-per-wavelength, which is the number of grid cells per wavelength (PPW) with which a given wavenumber component of the solution is resolved, through the relation $PPW = 2\pi/\kappa\Delta x$. For example, a value of $\kappa\Delta x$ equal to $\pi/4$ corresponds to 8 points-per-wavelength, and Fig. 2.2 shows that κ^* for a second-order centered difference scheme already differs significantly from κ at this grid resolution. Hence a simulation performed with this mesh density relative to the spectral content of the function will contain substantial numerical error.

For centered difference approximations, the modified wavenumber is purely real, but in the general case it can include an imaginary component as well. Any finite-difference operator can be split into an antisymmetric and a symmetric part. For example, the operator given in (2.18) can be divided as follows:

$$\begin{aligned}(\delta_x u)_j &= \frac{1}{6\Delta x}(u_{j-2} - 6u_{j-1} + 3u_j + 2u_{j+1}) \\ &= \frac{1}{12\Delta x}[(u_{j-2} - 8u_{j-1} + 8u_{j+1} - u_{j+2}) \\ &\quad + (u_{j-2} - 4u_{j-1} + 6u_j - 4u_{j+1} + u_{j+2})].\end{aligned} \tag{2.24}$$

The antisymmetric component determines the real part of the modified wavenumber, while the imaginary part stems from the symmetric component of the difference operator. Centered difference schemes are antisymmetric; the symmetric component is zero and hence so is the imaginary component of the modified wavenumber. In the context of the linear convection equation, one can show that a numerical error in the phase speed is associated with the real part of the modified wavenumber, while an error in the amplitude of the solution is associated with the imaginary part. Thus the antisymmetric portion of the spatial difference operator determines the error in speed and the symmetric portion the error in amplitude. Note that centered schemes produce no amplitude error. Since the numerical error in the phase speed is dependent on the wavenumber, this introduces numerical dispersion, and hence phase speed error is often referred to as *dispersive error*. Similarly, amplitude error is often referred to as *dissipative* error.

2.3 The Semi-Discrete Approach

Based on the discussion in the previous section, one can see that it is possible to replace both the spatial and temporal derivatives in a PDE by finite-difference expressions and thereby reduce the PDE to a system of algebraic equations that can be solved by a computer. For various reasons it can be advantageous to consider the discretization of space and time separately. We first discretize in space to reduce the PDE to a system of ordinary differential equations (ODEs) in the general form

$$\frac{\mathrm{d}\vec{u}}{\mathrm{d}t} = \vec{F}(\vec{u}, t), \tag{2.25}$$

and then apply a time-marching method to reduce the system of ODEs to a system of algebraic equations in order to solve them. This is referred to as the *semi-discrete approach*, and the intermediate ODE form in which the spatial derivatives have been discretized but the temporal derivatives have not is known as the *semi-discrete form*. It is important to realize that some numerical algorithms for PDEs discretize time and space simultaneously and consequently have no intermediate semi-discrete form. However, many of the most widely used algorithms and all of those considered in subsequent chapters involve a separate and distinct discretization in time and space.

In the semi-discrete approach, one separates the spatial discretization step that reduces the PDE to a system of ODEs from the time-marching step that numerically solves the ODE system. By doing so, we can get a clear understanding of the impact on accuracy and stability of the spatial discretization and the time-marching method individually. This approach also enables us to take advantage of the theory associated with numerical methods for ODEs. Before we can present the semi-discrete ODE form for our model equations, we need an understanding of *matrix difference operators*.

2.3.1 Matrix Difference Operators

Consider the relation

$$(\delta_{xx}u)_j = \frac{1}{\Delta x^2}\big(u_{j+1} - 2u_j + u_{j-1}\big), \tag{2.26}$$

which is a point difference approximation to a second derivative. Now let us derive a *matrix* operator representation for the same approximation. Consider the mesh spanning the domain $0 \leq x \leq \pi$ with four interior points and boundary points labelled a and b shown below.

$$\begin{array}{rcccccc} & a & 1 & 2 & 3 & 4 & b \\ x = & 0 & - & - & - & - & \pi \\ j = & & 1 & \cdot & \cdot & M & \end{array}$$

Mesh with four interior points. $\Delta x = \pi/(M+1)$

Now impose Dirichlet boundary conditions, $u(0) = u_a,\ u(\pi) = u_b$ and use the centered difference approximation given by (2.26) at every point in the mesh. We arrive at the four equations:

$$\begin{aligned}
(\delta_{xx}u)_1 &= \frac{1}{\Delta x^2}(u_a - 2u_1 + u_2) \\
(\delta_{xx}u)_2 &= \frac{1}{\Delta x^2}(u_1 - 2u_2 + u_3) \\
(\delta_{xx}u)_3 &= \frac{1}{\Delta x^2}(u_2 - 2u_3 + u_4) \\
(\delta_{xx}u)_4 &= \frac{1}{\Delta x^2}(u_3 - 2u_4 + u_b).
\end{aligned} \tag{2.27}$$

Introducing

$$\vec{u} = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix}, \qquad \left(\vec{bc}\right) = \frac{1}{\Delta x^2} \begin{bmatrix} u_a \\ 0 \\ 0 \\ u_b \end{bmatrix} \tag{2.28}$$

and

$$A = \frac{1}{\Delta x^2} \begin{bmatrix} -2 & 1 & & \\ 1 & -2 & 1 & \\ & 1 & -2 & 1 \\ & & 1 & -2 \end{bmatrix}, \tag{2.29}$$

we can rewrite (2.27) as

$$\delta_{xx}\vec{u} = A\vec{u} + \left(\vec{bc}\right). \tag{2.30}$$

This example illustrates a matrix difference operator. Each line of a matrix difference operator is based on a point difference operator, but the point operators used from line to line are not necessarily the same. For example, boundary conditions may dictate that the lines at or near the bottom or top of the matrix be modified. In the extreme case of the matrix difference operator representing a spectral method, none of the lines is the same. The matrix operators representing the three-point central-difference approximations for a first and second derivative with Dirichlet boundary conditions on a four-point mesh are

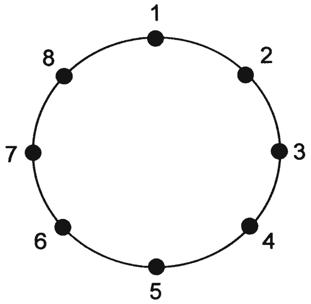
$$\delta_x = \frac{1}{2\Delta x} \begin{bmatrix} 0 & 1 & & \\ -1 & 0 & 1 & \\ & -1 & 0 & 1 \\ & & -1 & 0 \end{bmatrix}, \quad \delta_{xx} = \frac{1}{\Delta x^2} \begin{bmatrix} -2 & 1 & & \\ 1 & -2 & 1 & \\ & 1 & -2 & 1 \\ & & 1 & -2 \end{bmatrix}. \tag{2.31}$$

Each of these matrix difference operators is a square matrix with elements that are all zeros except for those along bands which are clustered around the central diagonal. We call such a matrix a *banded matrix* and introduce the notation

$$B(M : a, b, c) = \begin{bmatrix} b & c & & & \\ a & b & c & & \\ & & \ddots & & \\ & & a & b & c \\ & & & a & b \end{bmatrix} \begin{matrix} 1 \\ \\ \vdots \\ \\ M \end{matrix}, \tag{2.32}$$

where the matrix dimensions are $M \times M$. Use of M in the argument is optional, and the illustration is given for a simple *tridiagonal* matrix although any number of

Fig. 2.3 Eight points on a circular mesh



bands is a possibility. A tridiagonal matrix without constants along the bands can be expressed as $B(\vec{a}, \vec{b}, \vec{c})$. The arguments for a banded matrix are always odd in number, and the central one *always* refers to the central diagonal.

If the boundary conditions are *periodic*, the form of the matrix operator changes. Consider the eight-point periodic mesh spanning the domain $0 \le x \le 2\pi$ shown below. This can either be presented on a linear mesh with repeated entries, or more suggestively on a circular mesh as in Fig. 2.3. When the mesh is laid out on the perimeter of a circle, it does not matter where the numbering starts, as long as it "ends" at the point just preceding its starting location.

$$\begin{array}{lcccccccccccc}
\cdots & 7 & 8 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 1 & 2\ \cdots \\
x = & - & - & 0 & - & - & - & - & - & - & - & 2\pi & - \\
j = & & & 0 & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & M &
\end{array}$$

Eight points on a linear periodic mesh. $\Delta x = 2\pi/M$

The matrix that represents differencing schemes for scalar equations on a periodic mesh is referred to as a *periodic* matrix. A special subset of a periodic matrix is a *circulant* matrix, formed when the elements along the various bands are constant. Each row of a circulant matrix is shifted one element to the right of the one above it. The special case of a tridiagonal circulant matrix is given by

$$B_{\mathrm{p}}(M : a, b, c) = \begin{array}{c} \begin{bmatrix} b & c & & & a \\ a & b & c & & \\ & & \ddots & & \\ & & a & b & c \\ c & & & a & b \end{bmatrix} \end{array} \begin{array}{c} 1 \\ \\ \vdots \\ \\ M \end{array} . \tag{2.33}$$

When the standard three-point central-differencing approximation for a first derivative (see (2.31)) is used with periodic boundary conditions, it takes the form

$$(\delta_x)_\mathrm{p} = \frac{1}{2\Delta x}\begin{bmatrix} 0 & 1 & & -1 \\ -1 & 0 & 1 & \\ & -1 & 0 & 1 \\ 1 & & -1 & 0 \end{bmatrix} = \frac{1}{2\Delta x}B_\mathrm{p}(-1, 0, 1).$$

Notice that there is no boundary condition vector since this information is all interior to the matrix itself.

2.3.2 Reduction of PDEs to ODEs

Now that we have the concept of matrix difference operators, we can proceed to apply a spatial discretization to reduce PDEs to ODEs. First let us consider the model PDEs for diffusion and periodic convection described in Sect. 2.1. In these simple cases, we can approximate the space derivatives with difference operators and express the resulting ODEs with a matrix formulation. This is a simple and natural formulation when the ODEs are linear.

Model ODE for Diffusion. For example, using the three-point central-differencing scheme to represent the second derivative in the scalar PDE governing diffusion leads to the following ODE diffusion model:

$$\frac{\mathrm{d}\vec{u}}{\mathrm{d}t} = \frac{\nu}{\Delta x^2}B(1, -2, 1)\vec{u} + (\vec{bc}) \tag{2.34}$$

with Dirichlet boundary conditions folded into the $(\vec{bc})$ vector.

Model ODE for Periodic Convection. For the linear convection equation with periodic boundary conditions, the 3-point central-differencing approximation produces the ODE model given by

$$\frac{\mathrm{d}\vec{u}}{\mathrm{d}t} = -\frac{a}{2\Delta x}B_\mathrm{p}(-1, 0, 1)\vec{u}, \tag{2.35}$$

where the boundary condition vector is absent because the flow is periodic.

Equations (2.34) and (2.35) are the model ODEs for diffusion and periodic convection of a scalar in one dimension. They are linear with coefficient matrices which are independent of x and t.

The Generic Matrix Form. The generic matrix form of a semi-discrete approximation is expressed by the equation

$$\frac{\mathrm{d}\vec{u}}{\mathrm{d}t} = A\vec{u} - \vec{f}(t). \tag{2.36}$$

Note that the elements in the matrix A depend upon both the PDE and the type of differencing scheme chosen for the space terms. The vector $\vec{f}(t)$ is usually determined by the boundary conditions and possibly source terms. In general, even the Euler and

Navier–Stokes equations can be expressed in the form of (2.36). In such cases the equations are nonlinear, that is, the elements of A depend on the solution $\vec{u}$ and are usually derived by finding the Jacobian of a flux vector. Although the equations are nonlinear, linear analysis leads to diagnostics that are surprisingly accurate when used to evaluate many aspects of numerical methods as they apply to the Euler and Navier–Stokes equations.

2.3.3 Exact Solutions of Linear ODEs

In order to advance (2.25) in time, the system of ODEs must be integrated using a time-marching method. In order to analyze time-marching methods, we will make use of exact solutions of coupled systems of ODEs, which exist under certain conditions. The ODEs represented by (2.25) are said to be *linear* if F is linearly dependent on u (i.e. if $\partial F/\partial u = A$, where A is independent of u). As we have already pointed out, when the ODEs are linear they can be expressed in a matrix notation as (2.36) in which the coefficient matrix, A, is independent of u. If A *does* depend explicitly on t, the general solution *cannot* be written, whereas, if A *does not* depend explicitly on t, the general solution to (2.36) *can* be written. This holds regardless of whether or not the forcing function, $\vec{f}$, depends explicitly on t.

The exact solution of (2.36) can be written in terms of the eigenvalues and eigenvectors of A. This will lead us to a representative scalar equation for use in analyzing time-marching methods. To demonstrate this, let us consider a set of coupled, nonhomogeneous, linear, first-order ODEs with constant coefficients which might have been derived by space differencing a set of PDEs. Represent them by the equation

$$\frac{\mathrm{d}\vec{u}}{\mathrm{d}t} = A\vec{u} - \vec{f}(t). \tag{2.37}$$

Our assumption is that the $M \times M$ matrix A has a complete eigensystem[1] and can thus be transformed by the left and right eigenvector matrices, X^{-1} and X, to a diagonal matrix Λ having diagonal elements which are the eigenvalues of A. Now let us multiply (2.37) from the left by X^{-1} and insert the identity combination $XX^{-1} = I$ between A and $\vec{u}$. There results

$$X^{-1}\frac{\mathrm{d}\vec{u}}{\mathrm{d}t} = X^{-1}AX \cdot X^{-1}\vec{u} - X^{-1}\vec{f}(t). \tag{2.38}$$

Since A is independent of both $\vec{u}$ and t, the elements in X^{-1} and X are also independent of both $\vec{u}$ and t, and (2.38) can be modified to

[1] This means that the eigenvectors of A are linearly independent and thus $X^{-1}AX = \Lambda$, where X contains the right eigenvectors of A as its columns, i.e. $X = \left[\vec{x}_1\ ,\ \vec{x}_2\ \ldots\ ,\vec{x}_M\right]$, and Λ is a diagonal matrix whose elements are the eigenvalues of A.

$$\frac{\mathrm{d}}{\mathrm{d}t} X^{-1}\vec{u} = \Lambda X^{-1}\vec{u} - X^{-1}\vec{f}(t).$$

Finally, by introducing the new variables $\vec{w}$ and $\vec{g}$ such that

$$\vec{w} = X^{-1}\vec{u}, \quad \vec{g}(t) = X^{-1}\vec{f}(t), \tag{2.39}$$

we reduce (2.37) to a new algebraic form

$$\frac{\mathrm{d}\vec{w}}{\mathrm{d}t} = \Lambda\vec{w} - \vec{g}(t). \tag{2.40}$$

The equations represented by (2.40) are no longer coupled. They can be written line by line as a set of independent, single, first-order equations, thus

$$\begin{aligned} w_1' &= \lambda_1 w_1 - g_1(t) \\ &\vdots \\ w_m' &= \lambda_m w_m - g_m(t) \\ &\vdots \\ w_M' &= \lambda_M w_M - g_M(t). \end{aligned} \tag{2.41}$$

For any given set of $g_m(t)$ each of these equations can be solved separately and then recoupled, using the inverse of the relations given in (2.39):

$$\begin{aligned} \vec{u}(t) &= X\vec{w}(t) \\ &= \sum_{m=1}^{M} w_m(t)\vec{x}_m, \end{aligned} \tag{2.42}$$

where $\vec{x}_m$ is the mth column of X, i.e. the eigenvector corresponding to λ_m.

We next focus on the important subset of (2.36) when neither A nor $\vec{f}$ has any explicit dependence on t. In such a case, the g_m in (2.40) and (2.41) are also time invariant, and the solution to any line in (2.41) is

$$w_m(t) = c_m \mathrm{e}^{\lambda_m t} + \frac{1}{\lambda_m} g_m,$$

where the c_m are constants that depend on the initial conditions. Transforming back to the u-system gives

$$\begin{aligned} \vec{u}(t) &= X\vec{w}(t) \\ &= \sum_{m=1}^{M} w_m(t)\vec{x}_m \end{aligned}$$

$$
\begin{aligned}
&= \sum_{m=1}^{M} c_m \mathrm{e}^{\lambda_m t} \vec{x}_m + \sum_{m=1}^{M} \frac{1}{\lambda_m} g_m \vec{x}_m \\
&= \sum_{m=1}^{M} c_m \mathrm{e}^{\lambda_m t} \vec{x}_m + X\Lambda^{-1} X^{-1} \vec{f} \\
&= \underbrace{\sum_{m=1}^{M} c_m \mathrm{e}^{\lambda_m t} \vec{x}_m}_{\text{transient}} + \underbrace{A^{-1}\vec{f}}_{\text{steady-state}} .
\end{aligned}
\tag{2.43}
$$

Note that the steady-state solution is $A^{-1}\vec{f}$, as might be expected.

The first group of terms on the right side of this equation is referred to classically as the *complementary solution* or the solution of the homogeneous equations. The second group is referred to classically as the *particular solution* or the particular integral. In our application to fluid dynamics, it is more descriptive to refer to these groups as the *transient* and *steady-state* solutions, respectively. An alternative, but entirely equivalent, form of the solution is

$$
\vec{u}(t) = c_1 \mathrm{e}^{\lambda_1 t}\, \vec{x}_1 + \cdots + c_m \mathrm{e}^{\lambda_m t}\, \vec{x}_m + \cdots + c_M \mathrm{e}^{\lambda_M t}\, \vec{x}_M \;+\; A^{-1}\vec{f}. \tag{2.44}
$$

2.3.4 Eigenvalue Spectra for Model ODEs

It is instructive to consider the eigenvalue spectra of the ODEs formulated by central differencing the model equations for diffusion (2.34) and periodic convection (2.35). For the model diffusion equation with Dirichlet boundary conditions, the eigenvalues of A are:

$$
\begin{aligned}
\lambda_m &= \frac{\nu}{\Delta x^2}\left[-2 + 2\cos\left(\frac{m\pi}{M+1}\right)\right] \\
&= \frac{-4\nu}{\Delta x^2}\sin^2\left(\frac{m\pi}{2(M+1)}\right), \quad m = 1, 2, \ldots, M.
\end{aligned}
\tag{2.45}
$$

These eigenvalues are all real and negative, consistent with the physics of diffusion.

For periodic convection, one obtains

$$
\begin{aligned}
\lambda_m &= \frac{-\mathrm{i}a}{\Delta x}\sin\left(\frac{2m\pi}{M}\right), \quad m = 0, 1, \ldots, M-1 \\
&= -\mathrm{i}\kappa_m^* a,
\end{aligned}
\tag{2.46}
$$

where

$$\kappa_m^* = \frac{\sin \kappa_m \Delta x}{\Delta x}, \qquad m = 0, 1, \dots, M-1 \tag{2.47}$$

is the modified wavenumber, $\kappa_m = m$, and $\Delta x = 2\pi/M$. These eigenvalues are all pure imaginary, reflecting the fact that the amplitude of a waveform neither grows nor decays as it convects, a property that is preserved by centered differencing.

2.3.5 A Representative Equation for Studying Time-Marching Methods

We seek to analyze the accuracy and stability of time-marching methods applied to the systems of ODEs resulting from applying a spatial discretization to PDEs such as the Navier-Stokes equations, which take the form:

$$\frac{\mathrm{d}\vec{u}}{\mathrm{d}t} = \vec{F}(\vec{u}, t), \tag{2.48}$$

To simplify matters, we consider the simpler model equations, which lead to ODE forms such as (2.34) and (2.35) that have the generic form:

$$\frac{\mathrm{d}\vec{u}}{\mathrm{d}t} = A\vec{u} - \vec{f}(t), \tag{2.49}$$

where A is independent of u and t. To achieve a further simplification, we exploit the fact that these equations can be decoupled and study time-marching methods as applied to the following scalar ODE:

$$\frac{\mathrm{d}u}{\mathrm{d}t} = \lambda u + a\mathrm{e}^{\mu t}, \tag{2.50}$$

where λ, a, and μ are complex constants. The goal in our analysis is to study typical behavior of general situations, not particular problems. In order to evaluate time-marching methods, the parameters λ, a, and μ must be allowed to take the worst possible combination of values that might occur in the ODE eigensystem. For example, if one is interested in a time-marching method for convection dominated problems, then one should consider imaginary λs. The exact solution of the representative ODE is (for $\mu \neq \lambda$)

$$u(t) = c\,\mathrm{e}^{\lambda t} + \frac{a\mathrm{e}^{\mu t}}{\mu - \lambda}, \tag{2.51}$$

where the constant c is determined from the initial condition.

2.4 Finite-Volume Methods

2.4.1 Basic Concepts

We saw in Sect. 2.3 that a PDE can be reduced to a system of ODEs by discretizing the spatial derivatives using finite-difference approximations. A finite-volume method is an alternative spatial discretization that reduces the integral form of a conservation law to a system of ODEs. Finite-volume methods have become popular in CFD as a result, primarily, of two advantages. First, they ensure that the discretization is conservative, i.e. mass, momentum, and energy are conserved in a discrete sense. While this property can usually be obtained using a finite-difference formulation, it is obtained naturally from a finite-volume formulation. Second, finite-volume methods do not require a coordinate transformation in order to be applied on irregular meshes. As a result, they can be applied on *unstructured* meshes consisting of arbitrary polyhedra in three dimensions or arbitrary polygons in two dimensions. This increased flexibility can be advantageous in generating grids about complex geometries.

The PDE or divergence form of a conservation law can be written as

$$\frac{\partial Q}{\partial t} + \nabla \cdot \mathbf{F} = P, \tag{2.52}$$

where Q is a vector containing the set of variables which are conserved, e.g. mass, momentum, and energy, per unit volume, $\mathbf{F}$ is a set of vectors, or tensor, containing the flux of Q per unit area per unit time, P is the rate of production of Q per unit volume per unit time, and $\nabla \cdot \mathbf{F}$ is the well-known divergence operator. The same conservation law can be expressed in integral form as

$$\frac{\mathrm{d}}{\mathrm{d}t} \int_{V(t)} Q \mathrm{d}V + \oint_{S(t)} \mathbf{n} \cdot \mathbf{F} \mathrm{d}S = \int_{V(t)} P \mathrm{d}V. \tag{2.53}$$

This equation is a statement of the conservation of the conserved quantities in a finite region of space with volume $V(t)$ and surface area $S(t)$. In two dimensions, the region of space, or cell, is an area $A(t)$ bounded by a closed contour $C(t)$. The vector $\mathbf{n}$ is a unit vector normal to the surface pointing outward.

The basic idea of a finite-volume method is to satisfy the integral form of the conservation law to some degree of approximation for each of many contiguous control volumes that cover the domain of interest. Hence the function of the grid is to tessellate the domain into contiguous control volumes, and the volume V in (2.53) is that of a control volume whose shape is dependent on the nature of the grid. Examining (2.53), we see that several approximations must be made. The flux is required at the boundary of the control volume, which is a closed surface in three dimensions and a closed contour in two dimensions. This flux must then be integrated to find the net flux through the boundary. Similarly, the source term P

must be integrated over the control volume. Next a time-marching method can be applied to find the value of

$$\int_V Q \mathrm{d}V \tag{2.54}$$

at the next time step.

Let us consider each of these approximations in more detail. First, we note that the average value of Q in a cell with volume V is

$$\bar{Q} \equiv \frac{1}{V}\int_V Q \mathrm{d}V, \tag{2.55}$$

and (2.53) can be written as

$$V\frac{\mathrm{d}}{\mathrm{d}t}\bar{Q} + \oint_S \mathbf{n}\cdot\mathbf{F}\mathrm{d}S = \int_V P\mathrm{d}V \tag{2.56}$$

for a control volume that does not vary with time. Thus after applying a time-marching method, we have updated values of the cell-averaged quantities $\bar{Q}$. In order to evaluate the fluxes, which are a function of Q, at the control-volume boundary, Q can be represented within the cell by some piecewise approximation which produces the correct value of $\bar{Q}$. This is a form of interpolation often referred to as *reconstruction*. Each cell will have a different piecewise approximation to Q. When these are used to calculate $\mathbf{F}(Q)$, they will generally produce different approximations to the flux at the boundary between two control volumes, that is, the flux will be discontinuous. A nondissipative scheme analogous to centered differencing is obtained by taking the average of these two fluxes. Another approach known as flux-difference splitting is described in Sect. 2.5.

The basic elements of a finite-volume method are thus the following:

(1) Given the value of $\bar{Q}$ for each control volume, construct an approximation to $Q(x, y, z)$ in each control volume. Using this approximation, find Q at the control-volume boundary. Evaluate $\mathbf{F}(Q)$ at the boundary. Since there is a distinct approximation to $Q(x, y, z)$ in each control volume, two distinct values of the flux will generally be obtained at any point on the boundary between two control volumes.
(2) Apply some strategy for resolving the discontinuity in the flux at the control-volume boundary to produce a single value of $\mathbf{F}(Q)$ at any point on the boundary. This issue is discussed in Sect. 2.5.
(3) Integrate the flux to find the net flux through the control-volume boundary using some sort of quadrature.
(4) Advance the solution in time using a time-marching method to obtain new values of $\bar{Q}$.

The order of accuracy of the method is dependent on each of the approximations.

In order to include diffusive fluxes, the following relation between ∇Q and Q is sometimes used:

$$\int_V \nabla Q \mathrm{d}V = \oint_S \mathbf{n} Q \mathrm{d}S \tag{2.57}$$

or, in two dimensions,

$$\int_A \nabla Q \mathrm{d}A = \oint_C \mathbf{n} Q \mathrm{d}l, \tag{2.58}$$

where the unit vector **n** points outward from the surface or contour.

2.4.2 One-Dimensional Examples

We restrict our attention to a scalar dependent variable u and a scalar flux f, as in the model equations. We consider an equispaced grid with spacing Δx. The nodes of the grid are located at $x_j = j\Delta x$ as usual. Control volume j extends from $x_j - \Delta x/2$ to $x_j + \Delta x/2$, as shown in Fig. 2.4. This is referred to as a node centered scheme in contrast to a cell-centered scheme, where the control volume would extend from x_j to x_{j+1}. With respect to the discussion in this section, these two approaches are identical. We will use the following notation:

$$x_{j-1/2} = x_j - \Delta x/2, \quad x_{j+1/2} = x_j + \Delta x/2, \tag{2.59}$$

$$u_{j\pm 1/2} = u(x_{j\pm 1/2}), \quad f_{j\pm 1/2} = f(u_{j\pm 1/2}). \tag{2.60}$$

With these definitions, the cell-average value becomes

$$\bar{u}_j(t) \equiv \frac{1}{\Delta x} \int_{x_{j-1/2}}^{x_{j+1/2}} u(x, t) \mathrm{d}x, \tag{2.61}$$

and the integral form becomes

$$\frac{\mathrm{d}}{\mathrm{d}t}(\Delta x \bar{u}_j) + f_{j+1/2} - f_{j-1/2} = \int_{x_{j-1/2}}^{x_{j+1/2}} P \mathrm{d}x. \tag{2.62}$$

The integral form of the linear convection equation is obtained with $f = au$ and $P = 0$, while the integral form of the diffusion equation is obtained with $f = -\nu\nabla u = -\nu\partial u/\partial x$ and $P = 0$.

A Second-Order Approximation to the Convection Equation. With $a = 1$, the integral form of the linear convection equation becomes

$$\Delta x \frac{\mathrm{d}\bar{u}_j}{\mathrm{d}t} + f_{j+1/2} - f_{j-1/2} = 0 \tag{2.63}$$

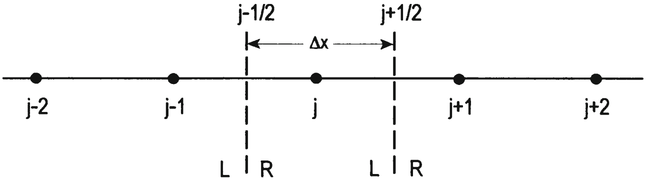


Fig. 2.4 Control volume in one dimension

with $f = u$. We choose a piecewise-constant approximation to $u(x)$ in each cell such that

$$u(x) = \bar{u}_j \quad x_{j-1/2} \le x \le x_{j+1/2}. \tag{2.64}$$

Evaluating this at $j + 1/2$ gives

$$f^{\mathrm{L}}_{j+1/2} = f(u^{\mathrm{L}}_{j+1/2}) = u^{\mathrm{L}}_{j+1/2} = \bar{u}_j, \tag{2.65}$$

where the L indicates that this approximation to $f_{j+1/2}$ is obtained from the approximation to $u(x)$ in the cell to the *left* of $x_{j+1/2}$, as shown in Fig. 2.4. The cell to the *right* of $x_{j+1/2}$, which is cell $j + 1$, gives

$$f^{\mathrm{R}}_{j+1/2} = \bar{u}_{j+1}. \tag{2.66}$$

Similarly, cell j is the cell to the right of $x_{j-1/2}$, giving

$$f^{\mathrm{R}}_{j-1/2} = \bar{u}_j \tag{2.67}$$

and cell $j - 1$ is the cell to the left of $x_{j-1/2}$, giving

$$f^{\mathrm{L}}_{j-1/2} = \bar{u}_{j-1}. \tag{2.68}$$

We have now accomplished the first step from the list in Sect. 2.4.1; we have defined the fluxes at the cell boundaries in terms of the cell-average data. In this example, the discontinuity in the flux at the cell boundary is resolved by taking the average of the fluxes on either side of the boundary. Thus

$$\hat{f}_{j+1/2} = \frac{1}{2}(f^{\mathrm{L}}_{j+1/2} + f^{\mathrm{R}}_{j+1/2}) = \frac{1}{2}(\bar{u}_j + \bar{u}_{j+1}) \tag{2.69}$$

and

$$\hat{f}_{j-1/2} = \frac{1}{2}(f^{\mathrm{L}}_{j-1/2} + f^{\mathrm{R}}_{j-1/2}) = \frac{1}{2}(\bar{u}_{j-1} + \bar{u}_j), \tag{2.70}$$

where $\hat{f}$ denotes a *numerical* flux which is an approximation to the exact flux.

Substituting (2.69) and (2.70) into the integral form, (2.63), we obtain

$$\begin{aligned} \Delta x \frac{\mathrm{d}\bar{u}_j}{\mathrm{d}t} &+ \frac{1}{2}(\bar{u}_j + \bar{u}_{j+1}) - \frac{1}{2}(\bar{u}_{j-1} + \bar{u}_j) \\ &= \Delta x \frac{\mathrm{d}\bar{u}_j}{\mathrm{d}t} + \frac{1}{2}(\bar{u}_{j+1} - \bar{u}_{j-1}) = 0. \end{aligned} \tag{2.71}$$

With periodic boundary conditions, this point operator produces the following semi-discrete form:

$$\frac{\mathrm{d}\vec{\bar{u}}}{\mathrm{d}t} = -\frac{1}{2\Delta x} B_\mathrm{p}(-1, 0, 1)\vec{\bar{u}} \tag{2.72}$$

This is identical to the expression obtained using second-order centered differences, except it is written in terms of the cell average $\vec{\bar{u}}$, rather than the nodal values, $\vec{u}$. Hence our analysis and understanding of the eigensystem of the matrix $B_\mathrm{p}(-1, 0, 1)$ is relevant to finite-volume methods as well as finite-difference methods. Since the eigenvalues of $B_\mathrm{p}(-1, 0, 1)$ are pure imaginary, we can conclude that the use of the average of the fluxes on either side of the cell boundary, as in (2.69) and (2.70), leads to a nondissipative finite-volume method.

A Fourth-Order Approximation to the Convection Equation. A fourth-order spatial discretization can be obtained by replacing the piecewise-constant approximation in Sect. 2.4.2 with a piecewise-quadratic approximation as follows

$$u(\xi) = a\xi^2 + b\xi + c, \tag{2.73}$$

where ξ is again equal to $x - x_j$. The three parameters a, b, and c are chosen to satisfy the following constraints:

$$\begin{aligned} \frac{1}{\Delta x}\int_{-3\Delta x/2}^{-\Delta x/2} u(\xi)\mathrm{d}\xi &= \bar{u}_{j-1} \\ \frac{1}{\Delta x}\int_{-\Delta x/2}^{\Delta x/2} u(\xi)\mathrm{d}\xi &= \bar{u}_j \\ \frac{1}{\Delta x}\int_{\Delta x/2}^{3\Delta x/2} u(\xi)\mathrm{d}\xi &= \bar{u}_{j+1}. \end{aligned} \tag{2.74}$$

These constraints lead to

$$\begin{aligned} a &= \frac{\bar{u}_{j+1} - 2\bar{u}_j + \bar{u}_{j-1}}{2\Delta x^2} \\ b &= \frac{\bar{u}_{j+1} - \bar{u}_{j-1}}{2\Delta x} \end{aligned}$$

$$c = \frac{-\bar{u}_{j-1} + 26\bar{u}_j - \bar{u}_{j+1}}{24}. \tag{2.75}$$

It is left as an exercise for the reader to show that this reconstruction leads to the following form:

$$\Delta x \frac{\mathrm{d}\bar{u}_j}{\mathrm{d}t} + \frac{1}{12}(-\bar{u}_{j+2} + 8\bar{u}_{j+1} - 8\bar{u}_{j-1} + \bar{u}_{j-2}) = 0, \tag{2.76}$$

which is analogous to a fourth-order centered finite-difference scheme.

A Second-Order Approximation to the Diffusion Equation. In this section, we describe two approaches to deriving a finite-volume approximation to the diffusion equation. The first approach is simpler to extend to multidimensions, while the second approach is more suited to extension to higher-order accuracy.

With $\nu = 1$, the integral form of the diffusion equation is

$$\Delta x \frac{\mathrm{d}\bar{u}_j}{\mathrm{d}t} + f_{j+1/2} - f_{j-1/2} = 0 \tag{2.77}$$

with $f = -\nabla u = -\partial u/\partial x$. Also, (2.58) becomes

$$\int_a^b \frac{\partial u}{\partial x} \mathrm{d}x = u(b) - u(a). \tag{2.78}$$

We can thus write the following expression for the average value of the gradient of u over the interval $x_j \le x \le x_{j+1}$:

$$\frac{1}{\Delta x} \int_{x_j}^{x_{j+1}} \frac{\partial u}{\partial x} \mathrm{d}x = \frac{1}{\Delta x}(u_{j+1} - u_j). \tag{2.79}$$

The value of a continuous function at the center of a given interval is equal to the average value of the function over the interval to second-order accuracy. Hence, to second-order, we can write

$$\hat{f}_{j+1/2} = -\left(\frac{\partial u}{\partial x}\right)_{j+1/2} = -\frac{1}{\Delta x}(\bar{u}_{j+1} - \bar{u}_j). \tag{2.80}$$

Similarly,

$$\hat{f}_{j-1/2} = -\frac{1}{\Delta x}(\bar{u}_j - \bar{u}_{j-1}). \tag{2.81}$$

Substituting these into the integral form (2.77), we obtain

$$\Delta x \frac{\mathrm{d}\bar{u}_j}{\mathrm{d}t} = \frac{1}{\Delta x}(\bar{u}_{j-1} - 2\bar{u}_j + \bar{u}_{j+1}) \tag{2.82}$$

or, with Dirichlet boundary conditions,

$$\frac{\mathrm{d}\vec{\bar{u}}}{\mathrm{d}t} = \frac{1}{\Delta x^2} B(1, -2, 1)\vec{\bar{u}} + \left(\vec{bc}\right). \tag{2.83}$$

This provides a semi-discrete finite-volume approximation to the diffusion equation, and we see that the properties of the matrix $B(1, -2, 1)$ are relevant to the study of finite-volume methods as well as finite-difference methods.

For our second approach, we use a piecewise-quadratic approximation as in Sect. 2.4.2. From (2.73) we have

$$\frac{\partial u}{\partial x} = \frac{\partial u}{\partial \xi} = 2a\xi + b \tag{2.84}$$

with a and b given in (2.75). With $f = -\partial u/\partial x$, this gives

$$f^{\mathrm{R}}_{j+1/2} = f^{\mathrm{L}}_{j+1/2} = -\frac{1}{\Delta x}(\bar{u}_{j+1} - \bar{u}_j) \tag{2.85}$$

$$f^{\mathrm{R}}_{j-1/2} = f^{\mathrm{L}}_{j-1/2} = -\frac{1}{\Delta x}(\bar{u}_j - \bar{u}_{j-1}). \tag{2.86}$$

Notice that there is no discontinuity in the flux at the cell boundary. This produces

$$\frac{\mathrm{d}\bar{u}_j}{\mathrm{d}t} = \frac{1}{\Delta x^2}(\bar{u}_{j-1} - 2\bar{u}_j + \bar{u}_{j+1}), \tag{2.87}$$

which is identical to (2.82).

2.5 Numerical Dissipation and Upwind Schemes

For a given order of accuracy, centered difference schemes produce the lowest coefficient of the leading truncation error term in comparison with one-sided and biased schemes. Moreover, a centered difference approximation correctly mimics the physics of convection and diffusion. In particular, a centered approximation to a first derivative is nondissipative, i.e. the eigenvalues of the associated matrix operator are pure imaginary. No aphysical numerical dissipation is introduced. Nevertheless, in the numerical solution of many practical problems, a small well-controlled amount of numerical dissipation is desirable and possibly even necessary for stability.

In a linear problem, there exist modes that are inaccurately resolved, as demonstrated by the modified wavenumbers shown in Fig. 2.2. If these modes are introduced into a simulation somehow, for example by the initial conditions, and there exists no mechanism to damp them, then they will persist and potentially contaminate the solution. It is preferable to damp these under-resolved solution components. In processes governed by nonlinear equations, such as the Navier–Stokes equations, there can be a continual production of high-frequency components of the solution, leading, for example, to the formation of shock waves. In a real physical problem, the production of high frequencies is eventually limited by viscosity. However, in practical simulations, the smallest length scales where the physical damping occurs are often under resolved. Unless the relevant length scales are resolved, some form of added numerical dissipation is required. Since the addition of numerical dissipation is tantamoun to intentionally introducing nonphysical behavior, it must be carefully controlled such that the error introduced is not excessive.

2.5.1 Numerical Dissipation in the Linear Convection Equation

One means of introducing numerical dissipation is through the use of one-sided differencing in the inviscid flux terms. For example, consider the following point operator for the spatial derivative term in the linear convection equation:

$$\begin{aligned} -a(\delta_x u)_j &= \frac{-a}{2\Delta x}[-(1+\beta)u_{j-1} + 2\beta u_j + (1-\beta)u_{j+1}] \\ &= \frac{-a}{2\Delta x}[(-u_{j-1} + u_{j+1}) + \beta(-u_{j-1} + 2u_j - u_{j+1})]. \end{aligned} \tag{2.88}$$

The second form shown divides the operator into an antisymmetric component $(-u_{j-1} + u_{j+1})/2\Delta x$ and a symmetric component $\beta(-u_{j-1} + 2u_j - u_{j+1})/2\Delta x$. The antisymmetric component is the second-order centered difference operator. With $\beta \neq 0$, the operator is only first-order accurate. A *backward* difference operator is given by $\beta = 1$, and a *forward* difference operator is given by $\beta = -1$.

For periodic boundary conditions, the corresponding matrix operator is

$$-a\delta_x = \frac{-a}{2\Delta x} B_{\mathrm{p}}(-1-\beta, 2\beta, 1-\beta).$$

The eigenvalues of this matrix are

$$\lambda_m = \frac{-a}{\Delta x}\left\{\beta\left[1 - \cos\left(\frac{2\pi m}{M}\right)\right] + \mathrm{i}\sin\left(\frac{2\pi m}{M}\right)\right\}, \quad m = 0, 1, \ldots, M-1.$$

If a is positive, the forward difference operator ($\beta = -1$) produces $\Re(\lambda_m) > 0$, the centered difference operator ($\beta = 0$) produces $\Re(\lambda_m) = 0$, and the backward difference operator produces $\Re(\lambda_m) < 0$. Hence the forward difference operator is inherently unstable, while the centered and backward operators are inherently stable. If a is negative, the roles are reversed.

In order to devise a spatial discretization that is stable independent of the sign of a, we can rewrite the linear convection equation as

$$\frac{\partial u}{\partial t} + (a^+ + a^-)\frac{\partial u}{\partial x} = 0, \quad a^\pm = \frac{a \pm |a|}{2}. \tag{2.89}$$

If $a \geq 0$, then $a^+ = a \geq 0$ and $a^- = 0$. Alternatively, if $a \leq 0$, then $a^+ = 0$ and $a^- = a \leq 0$. Now we can safely use a backward difference approximation for the a^+ (≥ 0) term and a forward difference approximation for the a^- (≤ 0) term. This is the basic concept behind upwind methods, that is some decomposition or splitting of the fluxes into terms which have positive and negative characteristic speeds so that appropriate differencing schemes can be chosen for each.

The above approach can be written in a different, but entirely equivalent, manner. From (2.88), we see that a stable discretization is obtained with $\beta = 1$ if $a \geq 0$ and with $\beta = -1$ if $a \leq 0$. This is achieved by the following point operator:

$$-a(\delta_x u)_j = \frac{-1}{2\Delta x}[a(-u_{j-1} + u_{j+1}) + |a|(-u_{j-1} + 2u_j - u_{j+1})]. \tag{2.90}$$

Any symmetric component in the spatial operator introduces dissipation (or amplification). Therefore, one could choose $\beta = 1/2$ in (2.88), for example, leading to the following operator:

$$-a(\delta_x u)_j = \frac{-1}{2\Delta x}[a(-u_{j-1} + u_{j+1}) + \frac{1}{2}|a|(-u_{j-1} + 2u_j - u_{j+1})]. \tag{2.91}$$

The resulting spatial operator is not one-sided, but it is dissipative.

Similarly, biased schemes use more information on one side of the grid node than the other. For example, a third-order backward-biased scheme is given by

$$\begin{aligned}(\delta_x u)_j &= \frac{1}{6\Delta x}(u_{j-2} - 6u_{j-1} + 3u_j + 2u_{j+1}) \\ &= \frac{1}{12\Delta x}[(u_{j-2} - 8u_{j-1} + 8u_{j+1} - u_{j+2}) \\ &\quad + (u_{j-2} - 4u_{j-1} + 6u_j - 4u_{j+1} + u_{j+2})].\end{aligned} \tag{2.92}$$

The antisymmetric component of this operator is the fourth-order centered difference operator. The symmetric component approximates $\Delta x^3 u_{xxxx}/12$. Therefore, this operator produces fourth-order accuracy in phase, with a third-order dissipative term. Note that the antisymmetric portion of the first-derivative operator always has an even order of accuracy, while the symmetric portion always has an odd order.

2.5.2 Upwind Schemes

In Sect. 2.5.1, we saw that numerical dissipation can be introduced in the spatial difference operator by using one-sided difference schemes or, more generally, by adding a symmetric component to the spatial operator. With this approach, the direction of the one-sided operator (i.e. whether it is a forward or a backward difference) and the sign of the symmetric component depend on the sign of the wave speed. When a *hyperbolic system* of equations is being solved, the wave speeds can be both positive and negative. For example, the eigenvalues of the flux Jacobian for the one-dimensional Euler equations are u, $u+a$, $u-a$, where u is the fluid velocity and a is the speed of sound. When the flow is subsonic, these are of mixed sign. In order to apply one-sided differencing schemes to such systems, some form of splitting is required.

Flux-Vector Splitting. Consider a linear, constant-coefficient, hyperbolic system of partial differential equations given by

$$\frac{\partial u}{\partial t} + \frac{\partial f}{\partial x} = \frac{\partial u}{\partial t} + A\frac{\partial u}{\partial x} = 0, \tag{2.93}$$

where $f = Au$, and A is diagonalizable with real eigenvalues. This system can be decoupled into characteristic equations of the form

$$\frac{\partial w_i}{\partial t} + \lambda_i \frac{\partial w_i}{\partial x} = 0, \tag{2.94}$$

where the wave speeds, λ_i, are the eigenvalues of the Jacobian matrix, A, and the w_is are the characteristic variables. In order to apply a one-sided (or biased) spatial differencing scheme, we need to apply a backward difference if the wave speed, λ_i, is positive, and a forward difference if the wave speed is negative.

To accomplish this, we split the matrix of eigenvalues, Λ, into two components such that

$$\Lambda = \Lambda^+ + \Lambda^-, \tag{2.95}$$

where

$$\Lambda^+ = \frac{\Lambda + |\Lambda|}{2}, \quad \Lambda^- = \frac{\Lambda - |\Lambda|}{2}. \tag{2.96}$$

With these definitions, Λ^+ contains the positive eigenvalues and Λ^- contains the negative eigenvalues. With the additional definitions[2]

$$A^+ = X\Lambda^+X^{-1}, \quad A^- = X\Lambda^-X^{-1}, \tag{2.97}$$

we can define the split flux vectors as

[2] With these definitions, A^+ has all nonnegative eigenvalues, and A^- has all nonpositive eigenvalues.

$$f^+ = A^+u, \quad f^- = A^-u. \tag{2.98}$$

Noting that $f = f^+ + f^-$, we can rewrite the original system in terms of the split flux vectors as

$$\frac{\partial u}{\partial t} + \frac{\partial f^+}{\partial x} + \frac{\partial f^-}{\partial x} = 0. \tag{2.99}$$

The spatial terms have been split into two components according to the sign of the wave speeds. A dissipative scheme is obtained by applying backward differencing to the $\frac{\partial f^+}{\partial x}$ term and forward differencing to the $\frac{\partial f^-}{\partial x}$ term.

Flux-vector splitting [2, 3] can also be used with a finite-volume method. Referring back to Sect. 2.4, recall that in a finite-volume method there exists a discontinuity in the flux at a control-volume boundary. When we took the average of the two fluxes at the interface, we obtained a nondissipative finite-volume discretization analogous to centered differencing. In order to develop a dissipative scheme, we can instead choose f^+ from the state to the left of the interface and f^- from the right state. This leads to the following upwind numerical flux:

$$\hat{f}_{j+1/2} = (f^+)^L + (f^-)^R, \tag{2.100}$$

which leads to a finite-volume method that is analogous to the flux-vector-split finite-difference scheme described above.

Flux-Difference Splitting. With flux-difference splitting [4], the numerical flux is given by

$$\hat{f}_{j+1/2} = \frac{1}{2}\left(f^{\mathrm{L}} + f^{\mathrm{R}}\right) + \frac{1}{2}|A|\left(u^{\mathrm{L}} - u^{\mathrm{R}}\right), \tag{2.101}$$

where

$$|A| = X|\Lambda|X^{-1}. \tag{2.102}$$

It is straightforward to show that in the linear, constant-coefficient case this is entirely equivalent to (2.100).

2.5.3 *Artificial Dissipation*

We have seen that numerical dissipation can be introduced by using one-sided differencing schemes together with some form of flux splitting. We have also seen that such dissipation can be introduced by adding a symmetric component to an antisymmetric (dissipation-free) operator. Thus we can generalize the concept of upwinding to include any scheme in which the symmetric portion of the operator is treated in such a manner as to be truly dissipative.

For example, consider the operator

$$\delta_x f = \delta_x^{\mathrm{a}} f + \delta_x^{\mathrm{s}}(|A|u), \tag{2.103}$$

where δ_x^{a} and δ_x^{s} are antisymmetric and symmetric difference operators, and $|A|$ is defined in (2.102). The second spatial term is known as *artificial dissipation*. With appropriate choices of δ_x^{a} and δ_x^{s}, this approach can be identical to the upwind approach.

It is common to use the following operator for δ_x^{s}

$$\left(\delta_x^{\mathrm{s}} u\right)_j = \frac{\epsilon}{\Delta x}(u_{j-2} - 4u_{j-1} + 6u_j - 4u_{j+1} + u_{j+2}), \tag{2.104}$$

where ϵ is a problem-dependent coefficient. This symmetric operator approximates $\epsilon \Delta x^3 u_{xxxx}$ and thus introduces a third-order dissipative term. With an appropriate value of ϵ, this often provides sufficient damping of high frequency modes without greatly affecting the low frequency modes. A more complicated treatment of the numerical dissipation is required near shock waves and other discontinuities; this subject is dealt with in later chapters.

2.6 Time-Marching Methods for ODEs

2.6.1 Basic Concepts: Explicit and Implicit Methods

After discretizing the spatial derivatives in the governing PDEs (such as the Navier–Stokes equations), we obtain a coupled system of nonlinear ODEs in the form

$$\frac{\mathrm{d}\vec{u}}{\mathrm{d}t} = \vec{F}(\vec{u}, t). \tag{2.105}$$

These can be integrated in time using a time-marching method to obtain a time-accurate solution to an *unsteady* flow problem. For a *steady* flow problem, spatial discretization leads to a coupled system of nonlinear algebraic equations in the form

$$\vec{F}(\vec{u}) = 0. \tag{2.106}$$

As a result of the nonlinearity of these equations, some sort of iterative method is required to obtain a solution. For example, one can consider the use of Newton's method, which is widely used for nonlinear algebraic equations. This produces an iterative method in which a coupled system of linear algebraic equations must be solved at each iteration. Alternatively, one can consider a time-dependent path to the steady state and use a time-marching method to integrate the unsteady form of the equations until the solution is sufficiently close to the steady solution. The subject of the present section, time-marching methods for ODEs, is thus relevant to both steady and unsteady flow problems. When using a time-marching method to compute steady flows, the goal is simply to remove the transient portion of the solution as quickly as possible; time-accuracy is not required. This motivates the study of stability and stiffness, topics which are discussed in the next section.

Application of a time-marching method to an ODE produces an ordinary *difference* equation (OΔE). Simple OΔEs can be easily solved, so we can develop exact solutions for the model OΔEs arising from the application of time-marching methods to the model ODEs. Using these exact solutions, we can analyze and understand the stability and accuracy properties of various time-marching methods.

Based on the discussion in Sect. 2.3, we will consider scalar ODEs given by

$$\frac{\mathrm{d}u}{\mathrm{d}t} = u' = F(u, t), \tag{2.107}$$

bearing in mind that the analysis applies directly to the solution of systems of ODEs. As in Sect. 2.2, we use the convention that the n subscript, or the (n) superscript, always denotes a discrete time value, and h represents the time interval Δt. Combining this notation with (2.107) gives

$$u'_n = F_n = F(u_n, t_n), \quad t_n = nh.$$

Often we need a more sophisticated notation for intermediate time steps involving intermediate solutions denoted by $\tilde{u}$, $\bar{u}$, etc. For these we use the notation

$$\tilde{u}'_{n+\alpha} = \tilde{F}_{n+\alpha} = F(\tilde{u}_{n+\alpha}, t_n + \alpha h).$$

The methods we study are to be applied to linear or nonlinear ODEs, but the methods themselves are formed by linear combinations of the dependent variable and its derivative at various time intervals. They are represented conceptually by

$$u_{n+1} = f\left(\beta_1 h u'_{n+1}, \beta_0 h u'_n, \beta_{-1} u'_{n-1}, \ldots, \alpha_0 u_n, \alpha_{-1} u_{n-1}, \ldots\right). \tag{2.108}$$

With an appropriate choice of the αs and βs, these methods can be constructed to give a local Taylor series accuracy of any order. A method is said to be *explicit* if $\beta_1 = 0$ and *implicit* otherwise. An *explicit* method is one in which the new predicted solution is only a function of known data, for example, u'_n, u'_{n-1}, u_n, and u_{n-1} for a method using two previous time levels, and therefore the time advance is simple. For an *implicit* method, the new predicted solution is also a function of the time derivative at the new time level, that is, u'_{n+1}. As we shall see, for systems of ODEs and nonlinear problems, implicit methods require more complicated strategies to solve for u_{n+1} than explicit methods.

Most time-marching methods in current use in CFD fall into one of three categories: *linear multistep methods*, *predictor–corrector methods*, and *Runge–*

Kutta methods. For the purpose of our analysis, we group predictor–corrector and Runge–Kutta methods together under the heading *multi–stage methods.*

In a linear multistep method, the solution at the new time level is a linear combination of the solution and its derivative at various time levels. In other words, (2.108) becomes

$$u_{n+1} = \beta_1 h u'_{n+1} + \beta_0 h u'_n, \beta_{-1} u'_{n-1} + \cdots + \alpha_0 u_n + \alpha_{-1} u_{n-1} + \cdots \tag{2.109}$$

Specific linear multistep methods are associated with specific choices of the αs and βs. In order to establish the order of accuracy, one can perform a Taylor series expansion of the right-hand side of (2.109) with particular values of the αs and βs and compare it to the Taylor series expansion of u_{n+1}. The order of accuracy of the method is the lowest exponent of h in the difference minus one. Similarly, one can use a Taylor table to derive a method by choosing which αs and βs will be permitted to have nonzero values; the Taylor table facilitates the derivation of the α and β values that maximize the order of accuracy. For example, the most basic time-marching method, which we will call the explicit Euler method, is found with all αs and βs set to zero with the exception of α_0 and β_0. In order to maximize the order of accuracy, one must choose $\alpha_0 = \beta_0 = 1$, which gives

$$u_{n+1} = u_n + h u'_n + O(h^2). \tag{2.110}$$

Since the leading error term is $O(h^2)$, this method is first order. This means that if an ODE is solved with this method over a specific time interval using first a specific value of h and then with $h/2$, the error in the solution at the end of the time interval will be reduced by a factor of two.

Further examples of linear multistep methods commonly used in CFD applications are given below[3]:

Explicit Methods.

$$\begin{array}{lc}
u_{n+1} = u_{n-1} + 2h u'_n & \text{Leapfrog} \\
u_{n+1} = u_n + \frac{1}{2} h \left[3u'_n - u'_{n-1} \right] & \text{AB2} \\
u_{n+1} = u_n + \frac{h}{12} \left[23u'_n - 16u'_{n-1} + 5u'_{n-2} \right] & \text{AB3}
\end{array}$$

Implicit Methods.

$$\begin{array}{lc}
u_{n+1} = u_n + h u'_{n+1} & \text{Implicit Euler} \\
u_{n+1} = u_n + \frac{1}{2} h \left[u'_n + u'_{n+1} \right] & \text{Trapezoidal (AM2)} \\
u_{n+1} = \frac{1}{3} \left[4u_n - u_{n-1} + 2h u'_{n+1} \right] & \text{2nd-order Backward} \\
u_{n+1} = u_n + \frac{h}{12} \left[5u'_{n+1} + 8u'_n - u'_{n-1} \right] & \text{AM3}
\end{array}$$

[3] Where the notation AB2 refers to the 2nd-order Adams-Bashforth method and AM2 refers to the second-order Adams-Moulton method, etc.

Predictor–corrector methods constructed to time-march linear or nonlinear ODEs are composed of sequences of linear multistep methods, each of which is referred to as a family in the solution process. There may be many families in the sequence, and usually the final family has a higher Taylor-series order of accuracy than the intermediate ones. Their use is motivated by ease of application and increased efficiency. In a simple two–stage predictor–corrector method, the solution is initially *predicted* at the next time level or some intermediate time using a linear multistep method. It is then *corrected* by applying another linear multistep method that involves applying the derivative function $F(u, t)$ at the predicted u and the appropriate value of t. For example, the second-order predictor–corrector method we will call MacCormack's method[4] can be written as

$$\begin{aligned}\tilde{u}_{n+1} &= u_n + hu'_n \\ u_{n+1} &= \frac{1}{2}(u_n + \tilde{u}_{n+1} + h\tilde{u}'_{n+1}).\end{aligned} \tag{2.111}$$

The predicted solution $\tilde{u}_{n+1}$ is obtained at t_{n+1} using the explicit Euler method, while the correction is obtained using the implicit trapezoidal method (see examples above) with u'_{n+1} replaced by $\tilde{u}'_{n+1}$. The method is explicit, since $\tilde{u}_{n+1}$ is computed before $\tilde{u}'_{n+1}$ is needed. Note that in order to advance one time step, two evaluations of the derivative function $F(u, t)$ are required, $F(u_n, t_n)$ in the predictor and $F(\tilde{u}_{n+1}, t_{n+1})$ in the corrector. Since evaluating the derivative function is typically the greatest computing expense in the application of a time-marching method, this means that the cost per time step of MacCormack's method is nominally twice that of a linear multistep method, where only one derivative function evaluation is needed per time step.[5]

Runge–Kutta methods are another important subset of multi-stage methods. The most popular is the classical explicit fourth-order Runge–Kutta method, which can be written in a notation consistent with the predictor–corrector example as

$$\begin{aligned}\widehat{u}_{n+1/2} &= u_n + \frac{1}{2}hu'_n \\ \tilde{u}_{n+1/2} &= u_n + \frac{1}{2}h\widehat{u}'_{n+1/2} \\ \overline{u}_{n+1} &= u_n + h\tilde{u}'_{n+1/2} \\ u_{n+1} &= u_n + \frac{1}{6}h\Big[u'_n + 2\Big(\widehat{u}'_{n+1/2} + \tilde{u}'_{n+1/2}\Big) + \overline{u}'_{n+1}\Big].\end{aligned} \tag{2.112}$$

[4] Here we discuss only MacCormack's *time-marching* method. The method commonly referred to as MacCormack's method is a fully-discrete method [5].

[5] If a linear multistep method requires, for example, $F(u_{n-1}, t_{n-1})$, this can be calculated at a previous time step and stored.

This method requires four derivative function evaluations per time step. As described below, the analysis and derivation of multi-stage methods is more involved than that for linear multistep methods.

2.6.2 Converting Time-Marching Methods to OΔEs

In Sect. 2.3.5 we chose a representative scalar ODE for the study of time-marching methods given by

$$\frac{\mathrm{d}u}{\mathrm{d}t} = \lambda u + a\mathrm{e}^{\mu t}, \tag{2.113}$$

where λ, a, and μ are complex constants. This equation has the following exact solution (for $\mu \neq \lambda$):

$$u(t) = c\,\mathrm{e}^{\lambda t} + \frac{a\mathrm{e}^{\mu t}}{\mu - \lambda}, \tag{2.114}$$

where the constant c is determined from the initial condition. Of course, one would not normally apply a numerical method to solve an equation for which one can derive the exact solution. Our purpose here is to analyze and evaluate time-marching methods, and a model ODE with a known solution plays an important role in this process. Using the theory of OΔEs we can obtain a closed form solution for the *numerical* solution obtained when a given time-marching method is used to solve the representative ODE. Rather than having to conduct a series of numerical experiments in order to understand the properties of a time-marching method, we can use this closed form solution to obtain these properties as an explicit function of the parameters h, λ, a, and μ. Hence, the theory of OΔEs provides a powerful tool for analyzing and deriving time-marching methods.

For example, consider the application of the explicit Euler method (2.110) to the representative ODE. Noting that $t_n = hn$, one obtains

$$\begin{aligned} u_{n+1} &= u_n + h(\lambda u_n + a\mathrm{e}^{\mu hn}) \\ &= (1 + \lambda h)u_n + ha\mathrm{e}^{\mu hn}. \end{aligned} \tag{2.115}$$

This is a first-order inhomogenous OΔE that can be written in the general form

$$u_{n+1} = \sigma u_n + \hat{a}b^n, \tag{2.116}$$

where σ, $\hat{a}$, and b are, in general, complex parameters. The independent variable is n rather than t, and, since the equations are linear and have constant coefficients, σ is not a function of either n or u. The exact solution of (2.116) is (for $b \neq \sigma$):

$$u_n = c_1 \sigma^n + \frac{\hat{a} b^n}{b - \sigma}, \tag{2.117}$$

where c_1 is a constant determined by the initial conditions. That (2.117) is a solution to (2.116) can be easily verified by substitution, and the reader is encouraged to do so.

Applying the exact OΔE solution (2.117) to the OΔE obtained by applying the explicit Euler method to the representative ODE (2.115), one obtains the exact numerical solution:

$$u_n = c_1 (1 + \lambda h)^n + \frac{h a \mathrm{e}^{\mu h n}}{\mathrm{e}^{\mu h} - 1 - \lambda h}. \tag{2.118}$$

This can be compared directly with the exact ODE solution rewritten as

$$u(t) = c\,(\mathrm{e}^{\lambda h})^n + \frac{a \mathrm{e}^{\mu h n}}{\mu - \lambda}. \tag{2.119}$$

In particular, comparing the homogeneous solutions

$$c_1 (1 + \lambda h)^n \approx c\,(\mathrm{e}^{\lambda h})^n, \tag{2.120}$$

where $c_1 = c$, shows that $\sigma = 1 + \lambda h$ is an approximation to $\mathrm{e}^{\lambda h}$. Given that the Taylor series expansion of $\mathrm{e}^{\lambda h}$ about $\lambda h = 0$ is

$$\mathrm{e}^{\lambda h} = 1 + \lambda h + \frac{1}{2} \lambda^2 h^2 + \cdots + \frac{1}{k!} \lambda^k h^k + \cdots \quad , \tag{2.121}$$

the error in the approximation is $O(h^2)$, consistent with the fact that the explicit Euler method is a first-order method. With a little more algebra,[6] one can readily show that the particular solution in (2.118) is also a first-order approximation to the exact particular solution.

Let us examine the homogeneous OΔE solution in more detail. Consider as an example $\lambda = -1$. The exact ODE homogeneous solution is simply $c\mathrm{e}^{-t}$. The homogeneous solution for the explicit Euler OΔE is

$$u_n = c_1 (1 - h)^n. \tag{2.122}$$

For small h this is a good approximation, consistent with the fact that $\sigma \approx \mathrm{e}^{\lambda h}$. However, for $h = 1$, the homogeneous solution becomes $u_n = 0$ after one step. This is completely inaccurate but at least provides the correct homogeneous solution as $n \to \infty$. With $h = 2$, the solution oscillates between 1 and -1, and for $h > 2$,

[6] One must expand both the exact ODE particular solution and the exact OΔE particular solution in Taylor series and compare on a term by term basis starting with the lowest power of h.

the solution grows without bound as $n \to \infty$. Generalizing this to arbitrary λ, the solution grows without bound if $|\sigma| = |1 + \lambda h| > 1$.[7]

Next consider the application of the implicit Euler method

$$u_{n+1} = u_n + hu'_{n+1} \tag{2.123}$$

to the representative ODE. The resulting OΔE is

$$u_{n+1} = \frac{1}{1 - \lambda h} u_n + \frac{1}{1 - \lambda h} h e^{\mu h} a e^{\mu h n}. \tag{2.124}$$

This can once again be compared to the form (2.116) to obtain the exact OΔE solution

$$u_n = c_1 \left(\frac{1}{1 - \lambda h} \right)^n + a e^{\mu h n} \cdot \frac{h e^{\mu h}}{(1 - \lambda h) e^{\mu h} - 1}. \tag{2.125}$$

In this case $\sigma = 1/(1 - \lambda h)$. Although again a first-order approximation to $e^{\lambda h}$, this leads to quite different behaviour than $\sigma = 1 + \lambda h$ obtained for the explicit Euler method. For example, with $\lambda = -1$ as in our previous example, the solution will not become unbounded even as $h \to \infty$.

This approach based on (2.116) and its solution (2.117) enables us to study one-step linear multistep methods, which are linear multistep methods that use data only at time levels $n + 1$ and n. For linear multistep methods of two steps or more and multistage methods, a more general theory is needed. This is achieved by writing the OΔE obtained by applying a time-marching method to the representative ODE in the following *operational form*:

$$P(E) u_n = Q(E) \cdot a e^{\mu h n}. \tag{2.126}$$

The terms $P(E)$ and $Q(E)$ are polynomials in E referred to as the *characteristic polynomial* and the *particular polynomial*, respectively. The *shift operator* E is defined formally by the relations

$$u_{n+1} = E u_n, \quad u_{n+k} = E^k u_n$$

and also applies to exponents, thus

$$b^\alpha \cdot b^n = b^{n+\alpha} = E^\alpha \cdot b^n,$$

where α can be any fraction or irrational number.

[7] Recall that λ and hence σ are in general complex.

The general solution of (2.126) can be expressed as

$$u_n = \sum_{k=1}^{K} c_k(\sigma_k)^n + a\mathrm{e}^{\mu hn} \cdot \frac{Q(\mathrm{e}^{\mu h})}{P(\mathrm{e}^{\mu h})}, \tag{2.127}$$

where σ_k are the K roots of the characteristic polynomial, $P(\sigma) = 0$. An important subset of this solution occurs when $\mu = 0$, representing a time-invariant particular solution, or a steady state. In such a case

$$u_n = \sum_{k=1}^{K} c_k(\sigma_k)^n + a \cdot \frac{Q(1)}{P(1)}. \tag{2.128}$$

We shall illustrate the application of (2.126) and (2.127) with two examples, a two-step multistep method and a multistage method, MacCormack's predictor-corrector method (2.111).

Consider first the *leapfrog method*, a second-order explicit two-step multistep method given by[8]

$$u_{n+1} = u_{n-1} + 2hu'_n. \tag{2.129}$$

Applying it to the representative ODE gives

$$u_{n+1} = u_{n-1} + 2h(\lambda u_n + a\mathrm{e}^{\mu hn}). \tag{2.130}$$

After rearranging and introducing the shift operator ($u_{n+1} = Eu_n$, $u_{n-1} = E^{-1}u_n$,) we obtain

$$(E - 2\lambda h - E^{-1})u_n = 2ha\mathrm{e}^{\mu hn}, \tag{2.131}$$

which is in the form (2.126) with

$$P(E) = E - 2\lambda h - E^{-1}, \quad Q(E) = 2h. \tag{2.132}$$

Setting $P(\sigma) = 0$ gives the relation

$$\sigma^2 - 2\lambda h\sigma - 1 = 0, \tag{2.133}$$

which produces two σ roots:

$$\sigma_{1,2} = \lambda h \pm \sqrt{\lambda^2 h^2 + 1}. \tag{2.134}$$

[8] The reader should observe the relationship between this time-marching method and the second-order centered difference approximation to a first derivative.

Thus the OΔE solution is

$$\begin{aligned} u_n = c_1(\lambda h + \sqrt{\lambda^2 h^2 + 1})^n + c_2(\lambda h - \sqrt{\lambda^2 h^2 + 1})^n \\ + a\mathrm{e}^{\mu h n} \cdot \frac{2h}{\mathrm{e}^{\mu h} - 2\lambda h - \mathrm{e}^{-\mu h}}. \end{aligned} \tag{2.135}$$

This OΔE solution has an important difference from that obtained for the explicit and implicit Euler methods: two σ-roots. Only one of them approximates $\mathrm{e}^{\lambda h}$. In this case $\sigma_1 = \lambda h + \sqrt{\lambda^2 h^2 + 1}$ can be expanded in a Taylor series to show that it is a second-order approximation to $\mathrm{e}^{\lambda h}$. The root with this property is known as the *principal root*, and the other root or roots are known as *spurious roots*. There are two constants in the OΔE solution, but only one initial condition. This reflects the fact that a method requiring data at time level $n - 1$ or earlier is not self starting. At the first time step $n = 0$, $u_n = u_0$ is known from the initial condition, but u_{n-1} is not known. Therefore, such methods are normally started using a self-starting method for the first step or steps, as required, and this provides the second necessary constant. If the method is started in this manner, the coefficients of the spurious roots will have small (but not zero) magnitudes.

As our final example, we will derive the solution to the OΔE obtained by applying MacCormack's explicit predictor–corrector method (2.111) to the representative ODE. This methodology can be followed to analyze Runge-Kutta methods as well.[9] Applying MacCormack's method to the representative equation gives

$$\begin{aligned} \tilde{u}_{n+1} - (1 + \lambda h)u_n = ah\mathrm{e}^{\mu h n} \\ -\tfrac{1}{2}(1 + \lambda h)\tilde{u}_{n+1} + u_{n+1} - \tfrac{1}{2}u_n = \tfrac{1}{2}ah\mathrm{e}^{\mu h(n+1)}, \end{aligned} \tag{2.136}$$

which is a coupled set of linear OΔEs with constant coefficients. The second line in (2.136) is obtained by noting that

$$\begin{aligned} \tilde{u}'_{n+1} &= F(\tilde{u}_{n+1}, t_n + h) \\ &= \lambda\tilde{u}_{n+1} + a\mathrm{e}^{\mu h(n+1)}. \end{aligned} \tag{2.137}$$

Introducing the shift operator E, we obtain

$$\begin{bmatrix} E & -(1 + (\mathrm{e}^{\mu h})) \\ -\frac{1}{2}(1 + (\mathrm{e}^{\mu h}))E & E - \frac{1}{2} \end{bmatrix} \begin{bmatrix} \tilde{u} \\ u \end{bmatrix}_n = h \cdot \begin{bmatrix} 1 \\ \frac{1}{2}E \end{bmatrix} \tilde{u}. \tag{2.138}$$

This system has a solution for both the intermediate family $\tilde{u}_n$ and the final family u_n. Since we are interested only in the final family, we can use Cramer's rule to obtain the operational form (2.126) as follows:

[9] In fact, MacCormack's method can be considered a second-order Runge-Kutta method.

$$P(E) = \det \begin{bmatrix} E & -(1+\lambda h) \\ -\frac{1}{2}(1+\lambda h)E & E-\frac{1}{2} \end{bmatrix} = E\left(E - 1 - \lambda h - \frac{1}{2}\lambda^2 h^2\right)$$

$$Q(E) = \det \begin{bmatrix} E & h \\ -\frac{1}{2}(1+\lambda h)E & \frac{1}{2}hE \end{bmatrix} = \frac{1}{2}hE(E + 1 + \lambda h).$$

The σ-root is found from

$$P(\sigma) = \sigma\left(\sigma - 1 - \lambda h - \frac{1}{2}\lambda^2 h^2\right) = 0,$$

which has only one nontrivial root

$$\sigma = 1 + \lambda h + \frac{1}{2}\lambda^2 h^2. \tag{2.139}$$

The complete solution can therefore be written

$$u_n = c_1\left(1 + \lambda h + \frac{1}{2}\lambda^2 h^2\right)^n + a\mathrm{e}^{\mu h n} \cdot \frac{\frac{1}{2}h\left(\mathrm{e}^{\mu h} + 1 + \lambda h\right)}{\mathrm{e}^{\mu h} - 1 - \lambda h - \frac{1}{2}\lambda^2 h^2}. \tag{2.140}$$

The σ-root is clearly a second-order approximation to $\mathrm{e}^{\lambda h}$, and the particular solution can also be shown to be a second-order approximation of the particular solution in (2.114). This example provides a template that can be used for the derivation and analysis of predictor–corrector and Runge-Kutta methods up to third order. Runge-Kutta methods of order four and higher must be derived based on a nonlinear ODE.

We are now in a position to generalize what we have learned about the σ-roots associated with a time-marching method. Recall that we intend to apply time-marching methods to systems of ODEs generated by discretizing the spatial derivatives in a PDE. For the linear, constant-coefficient systems of ODEs associated with our model equations, which are in the form (2.36), the solution can be written in the form (2.44), which we rewrite here as follows, noting that $t = nh$:

$$\vec{u}(t) = c_1\left(\mathrm{e}^{\lambda_1 h}\right)^n \vec{x}_1 + \cdots + c_m\left(\mathrm{e}^{\lambda_m h}\right)^n \vec{x}_m + \cdots + c_M\left(\mathrm{e}^{\lambda_M h}\right)^n \vec{x}_M \;+\; P.S., \tag{2.141}$$

where the λ_m and $\vec{x}_m$ are the eigenvalues and eigenvectors of the A matrix in the ODE system, and for the present we are not interested in the form of the particular solution ($P.S.$).

Both the explicit Euler and MacCormack methods are one-root methods; they produce one σ-root for each λ-root. If we use such a method to time march the system of ODEs, the solution of the resulting OΔEs is

$$\vec{u}_n = c_1(\sigma_1)^n\,\vec{x}_1 + \cdots + c_m(\sigma_m)^n\,\vec{x}_m + \cdots + c_M(\sigma_M)^n\,\vec{x}_M \;+\; P.S., \tag{2.142}$$

where the c_m and the $\vec{x}_m$ in the two equations are identical, and σ_m is an approximation to $\mathrm{e}^{\lambda h}$ that depends on the specific time-marching method. If the method produces one or more spurious σ-roots for each λ, as in our example of the leapfrog method, then the OΔE solution is

$$\begin{aligned}\vec{u}_n = {} & c_{11}(\sigma_1)_1^n\,\vec{x}_1 + \cdots + c_{m1}(\sigma_m)_1^n\,\vec{x}_m + \cdots + c_{M1}(\sigma_M)_1^n\,\vec{x}_M + P.S.\\ & + c_{12}(\sigma_1)_2^n\,\vec{x}_1 + \cdots + c_{m2}(\sigma_m)_2^n\,\vec{x}_m + \cdots + c_{M2}(\sigma_M)_2^n\,\vec{x}_M\\ & + c_{13}(\sigma_1)_3^n\,\vec{x}_1 + \cdots + c_{m3}(\sigma_m)_3^n\,\vec{x}_m + \cdots + c_{M3}(\sigma_M)_3^n\,\vec{x}_M\\ & + \text{etc., if there are more spurious roots.}\end{aligned} \tag{2.143}$$

The σ-root that approximates $\mathrm{e}^{\lambda_m h}$ is referred to as the *principal* σ-root, and designated $(\sigma_m)_1$. Application of the same time-marching method to all of the equations in a coupled system of linear ODEs in the form of (2.36) always produces one principal σ-root for every λ-root that satisfies the relation

$$\sigma = 1 + \lambda h + \frac{1}{2}\lambda^2 h^2 + \cdots + \frac{1}{k!}\lambda^k h^k + O\big(h^{k+1}\big), \tag{2.144}$$

where k is the order of the time-marching method. This property can be stated regardless of the details of the time-marching method, knowing only that its leading error is $O\big(h^{k+1}\big)$. Thus the principal root is an approximation to $\mathrm{e}^{\lambda h}$ up to $O\big(h^k\big)$.

Spurious roots arise if a method uses data from time level $n-1$ or earlier to advance the solution from time level n to $n+1$. Such roots originate entirely from the numerical approximation of the time-marching method and have nothing to do with the ODE being solved. However, generation of spurious roots does not, in itself, make a method inferior. In fact, many very accurate methods in practical use for integrating some forms of ODEs have spurious roots. Based on the starting technique, the magnitudes of the coefficients of the spurious roots will be small but nonzero. If the spurious roots themselves have amplitudes less than unity, they will not grow and hence will not contaminate the solution. Thus while spurious roots must be considered in *stability* analysis, they play virtually no role in *accuracy* analysis. Table 2.1 shows the λ–σ relations for various methods.

2.6.3 Implementation of Implicit Methods

Although the approach we have presented for analyzing time-marching methods based on the representative ODE is a powerful means of understanding the behaviour of time-marching methods, it obscures some aspects of the implementation of implicit methods to systems of nonlinear ODEs. These are introduced here.

Table 2.1 Some λ–σ relations

1.	$\sigma - 1 - \lambda h = 0$	Explicit Euler
2.	$\sigma^2 - 2\lambda h\sigma - 1 = 0$	Leapfrog
3.	$\sigma^2 - (1 + \frac{3}{2}\lambda h)\sigma + \frac{1}{2}\lambda h = 0$	AB2
4.	$\sigma^3 - (1 + \frac{23}{12}\lambda h)\sigma^2 + \frac{16}{12}\lambda h\sigma - \frac{5}{12}\lambda h = 0$	AB3
5.	$\sigma^{(}1 - \lambda h) - 1 = 0$	Implicit Euler
6.	$\sigma(1 - \frac{1}{2}\lambda h) - (1 + \frac{1}{2}\lambda h) = 0$	Trapezoidal
7.	$\sigma^2(1 - \frac{2}{3}\lambda h) - \frac{4}{3}\sigma + \frac{1}{3} = 0$	2nd-order backward
8.	$\sigma^2(1 - \frac{5}{12}\lambda h) - (1 + \frac{8}{12}\lambda h)\sigma + \frac{1}{12}\lambda h = 0$	AM3
9.	$\sigma^2 - (1 + \frac{13}{12}\lambda h + \frac{15}{24}\lambda^2h^2)\sigma + \frac{1}{12}\lambda h(1 + \frac{5}{2}\lambda h) = 0$	ABM3
10.	$\sigma^3 - (1 + 2\lambda h)\sigma^2 + \frac{3}{2}\lambda h\sigma - \frac{1}{2}\lambda h = 0$	Gazdag
11.	$\sigma - 1 - \lambda h - \frac{1}{2}\lambda^2h^2 = 0$	RK2
12.	$\sigma - 1 - \lambda h - \frac{1}{2}\lambda^2h^2 - \frac{1}{6}\lambda^3h^3 - \frac{1}{24}\lambda^4h^4 = 0$	RK4
13.	$\sigma^2(1 - \frac{1}{3}\lambda h) - \frac{4}{3}\lambda h\sigma - (1 + \frac{1}{3}\lambda h) = 0$	Milne 4th

Application to Systems of Equations. Consider the application of the implicit Euler method to our generic system of equations given by

$$\vec{u}' = A\vec{u} - \vec{f}(t), \tag{2.145}$$

where $\vec{u}$ and $\vec{f}$ are vectors, and we still assume that A is not a function of $\vec{u}$ or t. One obtains the following system of algebraic equations that must be solved at each time step:

$$(I - hA)\vec{u}_{n+1} - \vec{u}_n = -h\vec{f}(t + h) \tag{2.146}$$

or

$$\vec{u}_{n+1} = (I - hA)^{-1}[\vec{u}_n - h\vec{f}(t + h)]. \tag{2.147}$$

The inverse is not actually performed; rather we solve (2.146) as a linear system of equations. For our one-dimensional examples, the system of equations which must be solved is tridiagonal (e.g. for periodic convection, $A = -aB_\mathrm{p}(-1, 0, 1)/2\Delta x$), and hence its solution is inexpensive, but in multiple dimensions the bandwidth can be very large. In general, the cost per time step of an implicit method is thus larger than that of an explicit method. The primary area of application of implicit methods is in the solution of *stiff* ODEs; this is further discussed in Sect. 2.7.

Application to Nonlinear Equations. Now consider the general *nonlinear* scalar ODE given by

$$\frac{\mathrm{d}u}{\mathrm{d}t} = F(u, t). \tag{2.148}$$

Application of the implicit Euler method gives

$$u_{n+1} = u_n + hF(u_{n+1}, t_{n+1}). \tag{2.149}$$

This is a nonlinear difference equation which requires a nontrivial method to solve for u_{n+1}. There are several different approaches one can take to solving this nonlinear difference equation. An iterative method, such as Newton's method, can be used. Other alternatives include *local linearization* and *dual time stepping*.

In order to implement a local linearization, we expand $F(u, t)$ about some reference point in time. Designate the reference value by t_n and the corresponding value of the dependent variable by u_n. A Taylor series expansion about these reference quantities gives

$$F(u, t) = F_n + \left(\frac{\partial F}{\partial u}\right)(u - u_n) + \left(\frac{\partial F}{\partial t}\right)(t - t_n) + O(h^2). \tag{2.150}$$

This represents a second-order, locally-linear approximation to $F(u, t)$ that is valid in the vicinity of the reference station t_n and the corresponding $u_n = u(t_n)$. With this we obtain the locally (in the neighborhood of t_n) linear representation of (2.148), namely

$$\frac{\mathrm{d}u}{\mathrm{d}t} = \left(\frac{\partial F}{\partial u}\right)_n u + \left[F_n - \left(\frac{\partial F}{\partial u}\right)_n u_n\right] + \left(\frac{\partial F}{\partial t}\right)_n (t - t_n) + O(h^2). \tag{2.151}$$

As an example of how such an expansion can be used, consider the mechanics of applying the trapezoidal method for the time integration of (2.148). The trapezoidal method is given by

$$u_{n+1} = u_n + \frac{1}{2}h(F_{n+1} + F_n). \tag{2.152}$$

Using (2.150) to evaluate $F_{n+1} = F(u_{n+1}, t_{n+1})$, one finds

$$u_{n+1} = u_n + \frac{1}{2}h\left[F_n + \left(\frac{\partial F}{\partial u}\right)_n (u_{n+1} - u_n) + h\left(\frac{\partial F}{\partial t}\right)_n + O(h^2) + F_n\right]. \tag{2.153}$$

Note that the $O(h^2)$ term within the brackets (which is due to the local linearization) is multiplied by h and therefore preserves the second-order accuracy of the trapezoidal method. The local time linearization updated at the end of each time step and the trapezoidal time-marching method combine to make a *second-order-accurate* numerical integration process. There are, of course, other second-order implicit time-marching methods that can be used. The important point to be made here is that local linearization updated at each time step has not reduced the order of accuracy of a second-order time-marching process. Extension to systems of equations is straightforward, with $\left(\frac{\partial F}{\partial u}\right)_n$ representing a Jacobian matrix.

A useful reordering of the terms in (2.153) results in the expression

$$\left[1-\frac{1}{2}h\left(\frac{\partial F}{\partial u}\right)_n\right]\Delta u_n = hF_n + \frac{1}{2}h^2\left(\frac{\partial F}{\partial t}\right)_n, \tag{2.154}$$

which is known as the *delta form*. In many fluid mechanics applications the nonlinear function F is not an *explicit* function of t. In such cases the partial derivative of $F(u)$ with respect to t is zero, and (2.154) simplifies to the second-order-accurate expression

$$\left[1-\frac{1}{2}h\left(\frac{\partial F}{\partial u}\right)_n\right]\Delta u_n = hF_n. \tag{2.155}$$

Following the same steps with the implicit Euler method and again assuming that F is not an explicit function of time, we arrive at the form

$$\left[1-h\left(\frac{\partial F}{\partial u}\right)_n\right]\Delta u_n = hF_n. \tag{2.156}$$

We see that the only difference between the implementation of the trapezoidal method and the implicit Euler method is the factor of $1/2$ in the brackets of the left side of (2.155) and (2.156). While a method of second-order accuracy or higher is preferred for unsteady problems, the first-order implicit Euler method is an excellent choice for steady problems.

Consider the limit $h \to \infty$ of (2.156) obtained by dividing both sides by h and setting $1/h = 0$. There results

$$-\left(\frac{\partial F}{\partial u}\right)_n \Delta u_n = F_n \tag{2.157}$$

or

$$u_{n+1} = u_n - \left[\left(\frac{\partial F}{\partial u}\right)_n\right]^{-1} F_n. \tag{2.158}$$

This is the well-known Newton method for finding the roots of the nonlinear equation $F(u) = 0$.

Finally, we illustrate the dual time-stepping approach by applying it to the trapezoidal method. The algebraic equation that must be solved at each time step is given by (2.152). Hence u_{n+1} is the solution to

$$G(u) = 0, \tag{2.159}$$

where

$$G(u) = -u + u_n + \frac{1}{2}h\left(F(u) + F(U_n)\right). \tag{2.160}$$

While Newton's method provides one option for solving such an equation, another approach is to consider u_{n+1} to be the steady solution of the following ODE:

$$\frac{\mathrm{d}u}{\mathrm{d}\tau} = G(u), \tag{2.161}$$

where τ is often referred to as pseudo time. One can use an appropriate time-marching method to solve this ODE, and typically the method would be optimized for obtaining steady solutions efficiently. Note that $\Delta\tau$ can be selected for rapid convergence to the steady solution of (2.161), while h determines the time accuracy of the trapezoidal method. If an *explicit* time-marching method is used to solve (2.161) for a system of equations, then one has an implementation of an implicit method that does not require the solution of a linear system of algebraic equations at each time step.

2.7 Stability Analysis

Stability of numerical algorithms for the solution of PDEs is an important and complex topic. Here we will simplify matters and consider only time-dependent ODEs and OΔEs in which the coefficient matrices are independent of both u and t. We will refer to such matrices as *stationary*. In the preceding sections, we developed the representative forms of ODEs generated from the basic PDEs by the semi-discrete approach, and then the OΔEs generated from the representative ODEs by application of time-marching methods. These are represented by

$$\frac{\mathrm{d}\vec{u}}{\mathrm{d}t} = A\vec{u} - \vec{f}(t) \tag{2.162}$$

and

$$\vec{u}_{n+1} = C\vec{u}_n - \vec{g}_n, \tag{2.163}$$

respectively. For a one-step method, the latter form is obtained by applying a time-marching method to the generic ODE form in a fairly straightforward manner. For example, the explicit Euler method leads to $C = I + hA$, and $\vec{g}_n = h\vec{f}(nh)$. Methods involving two or more steps can always be written in the form of (2.163) by introducing new dependent variables. Note also that only methods in which the time and space discretizations are treated separately can be written in an intermediate semi-discrete form such as (2.162). The fully-discrete form, (2.163), and the associated stability definitions and analysis are applicable to all methods.

Our definitions of stability are based entirely on the behavior of the homogeneous parts of (2.162) and (2.163). The stability of (2.162) depends entirely on the eigensystem[10] of A. The stability of (2.163) can often also be related to the eigensystem of its matrix. However, in this case the situation is not quite so simple since, in our applications to partial differential equations (especially hyperbolic ones), a stability definition can depend on both the time and space differencing. Analysis of these eigensystems has the important added advantage that it gives an estimate of the *rate* at which a solution approaches a steady-state if a system is stable. We will consider only systems with complete eigensystems; for a discussion of defective systems, see Lomax et al. [1]. Note that a complete system can be arbitrarily close to a defective one, in which case practical applications can make the properties of the latter appear to dominate.

If A and C are stationary, we can estimate their fundamental properties. For example, in Sect. 2.3.4, we found from our model ODEs for diffusion and periodic convection what could be expected for the eigenvalue spectra of practical physical problems containing these phenomena. They are important enough to be summarized by the following:

- For diffusion-dominated flows, the λ-eigenvalues tend to lie along the negative real axis.
- For convection-dominated flows, the λ-eigenvalues tend to lie along the imaginary axis.

2.7.1 Inherent Stability of ODEs

Here we state the standard stability criterion used for ordinary differential equations:

$$\begin{aligned}&\text{For a } \textit{stationary} \text{ matrix } A, \text{ (2.162) is } \textit{inherently stable} \text{ if,}\\ &\text{when } \vec{f} \text{ is constant, } \ \vec{u} \text{ remains bounded as } t \to \infty.\end{aligned} \tag{2.164}$$

Note that inherent stability depends only on the transient solution of the ODEs.

If a matrix has a complete eigensystem, all of its eigenvectors are linearly independent, and the matrix can be diagonalized by a similarity transformation. In such a case it follows at once from (2.141), for example, that the ODEs are inherently stable if and only if

$$\Re(\lambda_m) \leq 0 \quad \text{for all } m. \tag{2.165}$$

This states that, for inherent stability, all of the λ eigenvalues must lie on, or to the left of, the imaginary axis in the complex λ plane. This criterion is satisfied for the model ODEs representing both diffusion and periodic convection.

[10] This is *not* the case if the coefficient matrix depends on t, even if it is linear.

2.7.2 Numerical Stability of OΔEs

The OΔE companion to (2.164) is:

$$\begin{aligned}&\text{For a } \textit{stationary} \text{ matrix } C, \text{(2.163) is } \textit{numerically stable} \text{ if,}\\ &\text{when } \vec{g} \text{ is constant, } \vec{u}_n \text{ remains bounded as } n \to \infty.\end{aligned} \tag{2.166}$$

We see that numerical stability depends only on the transient solution of the OΔEs. This definition of stability is sometimes referred to as asymptotic or time stability.

Consider a set of OΔEs governed by a complete eigensystem. The stability criterion, according to the condition set in (2.166), follows at once from a study of (2.142) and its companion for multiple σ-roots, (2.143). Clearly, for such systems a time-marching method is numerically stable if and only if

$$\left|(\sigma_m)_k\right| \leq 1 \quad \text{for all } m \text{ and } k. \tag{2.167}$$

This condition states that, for numerical stability, all of the σ eigenvalues (both principal and spurious, if there are any) must lie on or inside the unit circle in the complex σ-plane.

The most important aspect of numerical stability occurs under conditions when:

- one has inherently stable, coupled systems with λ-eigenvalues having widely separated magnitudes,

or

- we seek only to find a steady-state solution using a path that includes the unwanted transient.

In both of these cases there exist in the eigensystems relatively large values of $|\lambda h|$ associated with eigenvectors that we wish to drive through the solution process without any regard for their individual accuracy. This situation is the major motivation for the study of numerical stability and leads to the subject of stiffness discussed later in this section.

2.7.3 Unconditional Stability, A-stable Methods

A numerical method is *unconditionally stable* if it is stable for all ODEs that are inherently stable. A method with this property is said to be *A-stable*. It can be proved that the order of an A-stable linear multistep method *cannot exceed two*, and, furthermore that of all 2nd-order A-stable methods, the trapezoidal method has the smallest truncation error.

2.7.4 Stability Contours in the Complex λh Plane.

A convenient way to present the stability properties of a time-marching method is to plot the locus of the complex λh for which $|\sigma| = 1$, such that the resulting contour goes through the point $\lambda h = 0$. Here $|\sigma|$ refers to the maximum absolute value of any σ, principal or spurious, that is a root to the characteristic polynomial for a given λh. It follows from Sect. 2.7.2 that on one side of this contour the numerical method is stable, while on the other, it is unstable. We refer to it, therefore, as a *stability contour*.

Consider, for example, the explicit Euler method, for which

$$\sigma = 1 + \lambda h = 1 + \lambda_r h + \mathrm{i}\lambda_i h, \tag{2.168}$$

where λ_r and λ_i denote the real and imaginary parts of λ. Setting $|\sigma| = 1$ leads to

$$(1 + \lambda_r h)^2 + (\lambda_i h)^2 = 1, \tag{2.169}$$

which is the equation of a unit circle in the complex λh plane centered at $(-1, 0)$. The explicit Euler method is stable for λh values on or inside this circle. This means that it is unstable for the model periodic convection ODE and convection-dominated problems in general. For the model diffusion ODE it is *conditionally stable*. The time step must be chosen such that the eigenvalue of largest magnitude, which is given by

$$\lambda = \frac{\nu}{\Delta x^2}\left[-2 + 2\cos\left(\frac{M\pi}{M+1}\right)\right] \tag{2.170}$$

lies on or inside the unit circle. This gives

$$h \le \frac{\Delta x^2}{\nu\left[1 - \cos\left(\frac{M\pi}{M+1}\right)\right]} \approx \frac{\Delta x^2}{2\nu}, \tag{2.171}$$

or

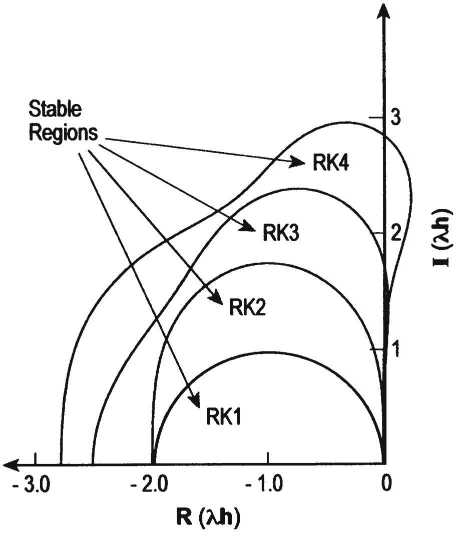
$$\frac{\nu h}{\Delta x^2} \le \frac{1}{2}, \tag{2.172}$$

where $\nu h/\Delta x^2$ is often referred to as the *Von Neumann number*.

The stability contour of the explicit Euler method is typical of all stability contours for explicit methods in the following two ways:

(1) The contour encloses a finite portion of the left-half complex λh-plane.
(2) The region of stability is *inside* the boundary, and therefore, it is conditional.

Fig. 2.5 Stability contours for explicit Runge–Kutta methods



Stability contours for explicit Runge-Kutta methods of orders one through four are shown in Fig. 2.5.[11] Notice that the contours of the third- and fourth-order Runge–Kutta methods include a portion of the imaginary axis out to ± 1.9i and $\pm 2\sqrt{2}$i, respectively, and hence are suitable for convection-dominated problems.

The eigenvalues of the ODE system arising from the application of second-order centered differencing to the periodic convection PDE are given in (2.46). The maximum magnitude is $|a|/\Delta x$, which leads to the following time step restriction when this system is solved using the fourth-order Runge-Kutta method:

$$\frac{|a|h}{\Delta x} \leq 2\sqrt{2}, \tag{2.173}$$

where $|a|h/\Delta x$ is known as the *Courant* or *CFL number*.

For the implicit Euler method, one can easily show that the stability contour is a unit circle centered at $(1, 0)$ with the unstable region being *inside* the circle. This means that the method is numerically stable even when the ODEs that it is being used to integrate are inherently unstable and is typical of many stability contours for unconditionally stable implicit methods. For the trapezoidal method, the stability boundary is the imaginary axis, so it is stable for λh lying on or to the left of this axis. Hence its stability condition precisely mimics that of the ODE system.

[11] The method labelled RK1 is the explicit Euler method.

2.7.5 Fourier Stability Analysis

The most popular form of stability analysis for numerical schemes is the Fourier or Von Neumann approach. This analysis is usually carried out on point operators, and it does not depend on an intermediate stage of ODEs. Strictly speaking it applies only to difference approximations of PDEs that produce OΔEs which are linear, have no space or time varying coefficients, and have periodic boundary conditions. In practical application it is often used as a guide for estimating the worthiness of a method for more general problems. It serves as a fairly reliable *necessary* stability condition, but it is by no means a *sufficient* one.

One takes data from a "typical" point in the flow field and uses this as constant throughout time and space according to the assumptions given above. Then one imposes a spatial harmonic as an initial value on the mesh and asks the question: Will its amplitude grow or decay in time? The answer is determined by finding the conditions under which

$$u(x, t) = \mathrm{e}^{\alpha t} \cdot \mathrm{e}^{\mathrm{i}\kappa x} \tag{2.174}$$

is a solution to the *difference* equation, where κ is real and $\kappa\Delta x$ lies in the range $0 \leq \kappa\Delta x \leq \pi$. Since, for the general term,

$$u_{j+m}^{(n+\ell)} = \mathrm{e}^{\alpha(t+\ell\Delta t)} \cdot \mathrm{e}^{\mathrm{i}\kappa(x+m\Delta x)} = \mathrm{e}^{\alpha\ell\Delta t} \cdot \mathrm{e}^{\mathrm{i}\kappa m\Delta x} \cdot u_j^{(n)},$$

the quantity $u_j^{(n)}$ is common to every term and can be factored out. In the remaining expressions, we find the term $\mathrm{e}^{\alpha\Delta t}$, which we represent by σ, thus

$$\sigma \equiv \mathrm{e}^{\alpha\Delta t}.$$

Then, since $\mathrm{e}^{\alpha t} = \left(\mathrm{e}^{\alpha\Delta t}\right)^n = \sigma^n$, it is clear that:

$$\text{For numerical stability } |\sigma| \leq 1 \tag{2.175}$$

and the problem is to solve for the σs produced by any given method and, as a necessary condition for stability, make sure that, in the worst possible combination of parameters, (2.175) is satisfied.

The procedure can best be explained by an example. Consider the following fully-discrete point operator for the model diffusion equation:

$$u_j^{(n+1)} = u_j^{(n-1)} + \nu\frac{2\Delta t}{\Delta x^2}\left(u_{j+1}^{(n)} - 2u_j^{(n)} + u_{j-1}^{(n)}\right), \tag{2.176}$$

which is obtained by combining second-order centered differencing with the leapfrog method. Substitution of (2.174) into (2.176) gives the relation

$$\sigma = \sigma^{-1} + \nu \frac{2\Delta t}{\Delta x^2}\left(\mathrm{e}^{\mathrm{i}\kappa\Delta x} - 2 + \mathrm{e}^{-\mathrm{i}\kappa\Delta x}\right)$$

or

$$\sigma^2 + \underbrace{\left[\frac{4\nu\Delta t}{\Delta x^2}(1 - \cos\kappa\Delta x)\right]}_{2\mathrm{b}}\sigma - 1 = 0. \tag{2.177}$$

Thus (2.174) is a solution of (2.176) if σ is a root of (2.177). The two roots of (2.177) are

$$\sigma_{1,2} = -b \pm \sqrt{b^2 + 1},$$

from which it is clear that one $|\sigma|$ is always > 1. We find, therefore, that by the Fourier stability test, this method is unstable for all ν, κ and Δt. The same conclusion can be gleaned from a knowledge of the stability contour for the leapfrog method and the eigenvalues of the diffusion ODE system. The leapfrog method is stable only for pure imaginary eigenvalues with amplitude less than or equal to unity. The diffusion ODE eigenvalues are strictly real and hence cannot be brought into the stable region of the leapfrog method by any choice of h.

2.7.6 Stiffness of Systems of ODEs

The concept referred to as “stiffness” comes about from the numerical analysis of mathematical models constructed to simulate dynamic phenomena containing widely different time scales. The difference between the dynamic scales translates into a difference in the magnitudes of the eigenvalues of the ODE system. The concept of stiffness in CFD arises from the fact that we often do not need accurate *time resolution* of eigenvectors associated with the large $|\lambda_m|$ in the transient solution, although these eigenvectors must remain coupled into the system to maintain the accuracy of the *spatial resolution*. For example, recall the modified wavenumber for a second-order centered difference approximation of a first derivative depicted in Fig. 2.2. For wavenumbers $\kappa\Delta x$ greater than unity the approximation is very inaccurate. Therefore there is no reason to time-march the eigenvectors associated with these wavenumbers with a high degree of accuracy. However, these components of the solution must be time-marched in a stable manner so that they do not contaminate the solution.

This situation is depicted graphically in Fig. 2.6 for the explicit Euler method. All eigenvalues, whether they must be accurately time resolved or not, must lie within the stable region of the time-marching method. In addition, those eigenvalues that correspond to eigenvectors for which accurate time resolution is required must lie within a region near the origin where the principal σ-root is a sufficiently accurate approximation to $\mathrm{e}^{\lambda h}$ for the purposes of the specific simulation (labelled the accurate

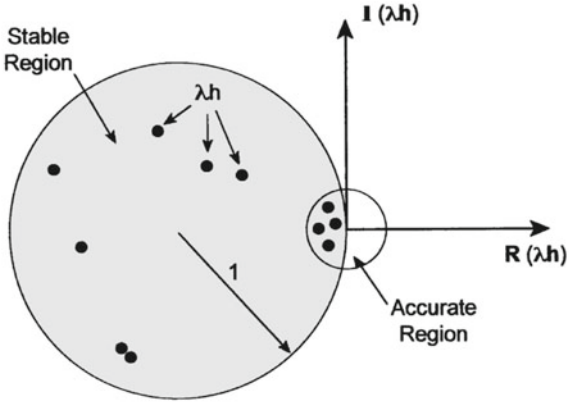


Fig. 2.6 Stable and accurate regions for the explicit Euler method

region). In the figure, the time step has been chosen so that time accuracy is given to the eigenvectors associated with the eigenvalues lying in the small circle, and stability without time accuracy is given to those associated with the eigenvalues lying outside of the small circle but still inside the large circle.

We term the eigenvalues corresponding to eigenvectors for which time accuracy is required the *driving* eigenvalues, and those for which only stability is required are termed *parasitic* eigenvalues. Unfortunately, although time accuracy requirements are dictated by the driving eigenvalues, numerical stability requirements are dictated by the parasitic ones. If a time step *h* chosen on the basis of stability requirements is sufficiently small that the driving eigenvalues fall within the accurate region, then the time step choice is described as *stability limited.* Similarly, if a time step *h* chosen on the basis of accuracy requirements is sufficiently small that the parasitic eigenvalues fall within the stable region, then the time step choice is described as *accuracy limited.*

The stiffness of an ODE system is related to the ratio of the magnitude of the largest parasitic eigenvalue to that of the largest driving eigenvalue. If this ratio is large, the system is stiff, and, if a conditionally stable time-marching method is used, the time step selection will be severely constrained by stability requirements. In other words, the time step necessary for stability is much smaller than that required for accuracy of the driving eigenvalues, and the simulation can be inefficient, requiring many more time steps than are actually needed for accurate resolution of the driving modes. In such instances, unconditionally stable implicit methods become preferable, as the time step can be selected solely on the basis of the accuracy requirements. Since implicit methods typically require more computation per time step, the comparison depends on the degree of stiffness of the problem. As the degree of stiffness increases, the advantage tilts toward implicit methods, as the reduced number of time steps begins to outweigh the increased cost per time step.

References

1. Lomax, H., Pulliam, T.H., Zingg, D.W.: Fundamentals of Computational Fluid Dynamics. Springer, Berlin (2001)
2. Steger, J.L., Warming, R.F.: Flux vector splitting of the inviscid gas dynamic equations with applications to finite difference methods. J. Comput. Phys. **40**, 263–293 (1981)
3. Van Leer, B.: Flux vector splitting for the Euler equations. In: Proceedings of the 8th International Conference on Numerical Methods in Fluid Dynamics, Springer-Verlag, Berlin (1982)
4. Roe, P.L.: Approximate Riemann solvers, parameter vectors, and difference schemes. J. Comput. Phys. **43**, 357–372 (1981)
5. MacCormack, R.W.: The effect of viscosity in hypervelocity impact cratering, AIAA Paper 69-354 (1969)