son

THE HANDBOOK OF FLUID DYNAMICS

THE HANDBOOK OF FLUID DYNAMICS

Edited by

Richard W. Johnson

THE HANDBOOK OF FLUID DYNAMICS

Edited by

Richard W. Johnson

CRC Press
Boca Raton Boston London New York Washington, D.C.

Acquiring Editor: *Bob Stern*
Production Manager: *Suzanne Lassandro*
Project Editor: *Susan Fox*
Cover Design: *Dawn Boyd*

Library of Congress Cataloging-in-Publication Data

Catalog record is available from the Library of Congress.

International Standard Book Number 0-8493-2509-9
Printed in the United States of America 1 2 3 4 5 6 7 8 9 0
Printed on acid-free paper

31

Solution Methods for the Incompressible Navier–Stokes Equations

Wei Shyy
University of Florida

Rajat Mittal
University of Florida

31.1 Introduction

In this chapter, we will offer an overview of the various issues related to numerical solutions of the Navier–Stokes equations, with emphasis on the incompressible regime. These equations are a set of coupled, non-linear, mixed elliptic-parabolic partial differential equations that require special techniques for numerical solution. We will first review the major characteristics of these equations and difficulties associated with solving them numerically. In particular, the role of the pressure and the lack of a unique differential equation governing it will be discussed. Then, we will summarize the major issues arising from solving these flow problems, including treatments of time-dependency, algorithms proposed to handle the pressure distribution and the continuity constraint, alternative discretization schemes for the non-linear convection terms, procedures for numerically solving the system of algebraic equations result-

0-8493-2509-9/98/$0.00+$.50

ing from the discretization procedure, the outflow boundary condition, and treatments of complex geometries.

Pressure and Continuity Equation

In primitive variables, the differential equations governing the incompressible, constant property Newtonian fluid flows are

$$\nabla \cdot \boldsymbol{u} = 0 \text{ in } \Omega \tag{31.1}$$

$$\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u} \cdot \nabla \boldsymbol{u} = -\nabla p + \nu \nabla^2 \boldsymbol{u} \text{ in } \Omega \tag{31.2}$$

$$\boldsymbol{u}(x,t) = \boldsymbol{v}(x,t) \text{ on } \partial\Omega \tag{31.3}$$

where Ω denotes the volume and $\partial\Omega$ the corresponding bounding surface of the flow under consideration. Furthermore, u and p denote the velocity and pressure, respectively, and v is the prescribed velocity on the boundary. The numerical solution of the above equations is complicated by the fact that there is no evolution equation for pressure and pressure is instead constrained by Eq. (31.1). This observation had lead to methods such as pseudo-compressibility [Chorin, 1967; Peyret and Taylor, 1983; Hirsch, 1990] and penalty methods [Temam, 1978; Hughes et al., 1979; Braaten and Shyy, 1986] where an artificial equation for pressure is devised. Alternatively, one can also manipulate the momentum and continuity equations and either retain every term to formulate a Poisson type of pressure equation [Harlow and Welch, 1965; Peyret and Taylor, 1983], or simplify some terms to formulate a predictor–corrector type procedure to update pressure and velocity fields in sequence [Chorin, 1968; Pantankar, 1980; Issa, 1986; Dukowicz and Dvinsky, 1992]. All these methods are capable of producing both steady-state and time-dependent solutions.

Another alternative is to use the vorticity–streamfunction formulation, which does not require the solution of the pressure field. In the past, due to limited computing power, most viscous calculations were two-dimensional and the vorticity–streamfunction formulation was quite widely used. However, the extension of this formulation to three-dimensional flows in not straightforward and requires separate formulations. In view of its lack of unified capability to handle two- and three-dimensional problems, this approach has seen diminished attention in favor of the primitive variable approach.

Conservative Versus Nonconservative Formulations

When considering the various possible choices of velocity variables in the framework of the finite volume formulation, a full conservation law form of the governing equations is desirable as it can satisfy the physical laws more easily and accurately. This consideration has a particularly important implication on the convection terms of the momentum equations since they are nonlinear and are usually a major source of numerical difficulty. In Cartesian coordinates, the convection terms in the momentum equations are of the form $(\rho uu)_x + (\rho vu)_y$, which is fully conservative. In a curvilinear coordinate system, these terms can be transformed in a straightforward manner.

One of the most basic tests of the numerical accuracy of any computational algorithm can be made by generating a grid system with arbitrary skewness and nonuniformity and then using this grid system to check numerical accuracy by solving a uniform flow field of say, $\rho = 1$, $u = 1$, and $v = 1$. With this condition, the continuity equation is trivially satisfied in the differential sense. Hence it serves as a good case to test whether an algorithm can honor the geometric aspect of the conservation laws in a discrete form. Here we call this requirement the geometric conservation law [Thomas and Lombard, 1979; Shyy and Vu, 1991], since the governing equations retain the conservation law form but contain only the

geometric quantities. The transformed form of the two-dimensional steady-state continuity equation in curvilinear coordinates becomes

$$\left(\rho U u\right)_{\xi} + \left(\rho V u\right)_{\eta} = 0 \tag{31.4}$$

where U and V are the contra-variant velocity components [Shyy et al., 1985], to be discussed later in Section 31.7. With the uniform flowfield the above equation is reduced to

$$\left(y_{\eta} - x_{\eta}\right)_{\xi} + \left(-y_{\xi} + x_{\xi}\right)_{\eta} = 0 \tag{31.5}$$

If a consistent finite volume formulation is adopted by approximating the derivative of the metric terms in Eq. (31.5), with the difference between two end points of the mesh line, then it becomes

$$\begin{aligned}&\left[\left(y_{i+1,j+1} - y_{i+1,j}\right) - \left(x_{i+1,j+1} - x_{i+1,j}\right)\right] - \left[\left(y_{i,j+1} - y_{i,j}\right) - \left(x_{i,j+1} - x_{i,j}\right)\right] \\ &- \left[-\left(y_{i+1,j+1} - y_{i,j+1}\right) + \left(x_{i+1,j+1} - x_{i,j+1}\right)\right] - \left[-\left(y_{i+1,j} - y_{i,j}\right) + \left(x_{i+1,j} - x_{i,j}\right)\right] = 0\end{aligned} \tag{31.6}$$

which is satisfied *exactly* regardless of how skew or nonuniform the mesh may be. However, if one chooses to use other discretization schemes, say, one-side formulas, to treat the derivatives, then there is no guarantee that Eq. (31.5) can be satisfied.

Another related issue concerns the conservation of kinetic energy (in the limit of zero viscosity) by the discretized equations. Since unbounded growth of energy implies instability, the issue of kinetic energy conservation is intimately linked to the stability of the discretized equations. Energy conservation (or the lack thereof) is connected to the accumulation of aliasing errors that occur due to the discrete evaluation of nonlinear terms [Ferzigier and Peric, 1996]. Since kinetic energy is not an independent variable, the energy conservation requirement presents itself as a constraint which has to be satisfied by the discretized equation set comprised of the continuity and momentum equations. Four approaches can be taken to deal with this issue: (1) Addition of artificial viscosity through upwinding or filtering [Shyy et al., 1992] in order to damp the high-wavenumber components of the velocity field (2) using special spatial discretization schemes that automatically conserve energy [Morinishi, 1995], (3) using special forms of the nonlinear convection terms [Zang, 1991], and (4) dealiasing the calculation [Canuto et al., 1987]. Kinetic energy conservation can be especially important in time accurate simulations of turbulent flows and a number of studies have documented the effect of energy conservation errors on turbulent flow [Horiuti, 1987; Kravchenko and Moin, 1997].

31.2 Time Discretization Schemes for Unsteady Problems

Stability Consideration

Time discretization schemes are usually chosen on the basis of stability, accuracy and required computational effort (CPU time and storage). The accuracy of the temporal discretization can be determined in a straightforward manner from the truncation error. Usually it is a balance between the time-step restriction imposed by the stability of the scheme and the required computational effort that determines the choice of the scheme for a particular flow configuration. Based on a one-dimensional analysis, explicit treatment of the convective and viscous terms results in a time step restriction given by $\Delta t < K\,\Delta x/u$ and $\Delta t < K'(\Delta x)^2/\nu$, respectively, where K and K' are typically $O(1)$ constants which depends on the particular time and spatial discretization scheme. Δt and Δx denote the time-step size and grid spacing, respectively, and u is the convective velocity.

The convective stability requirement imposes severe restriction on the size of the time step if a small grid spacing is encountered in the direction of the flow. In many flow problems the streamwise gradients are small relative to the stream-normal direction and consequently the streamwise grid spacing required to resolve these gradients is relatively large. This relieves the severity of the convective stability constraints and allows the use of explicit schemes for the convective term. There are, however, cases where one is forced to use a grid which is unduly fine in the flow direction. An example is shown in Fig. 31.1. The figure shows the topology of a typical structured grid used for computing flow over a backward facing step. The key feature is that the wall-normal resolution required to resolve the boundary layer on the lower wall of the inlet is carried into the downstream region. In this region there can be a considerable vertical velocity (as indicated by the arrows) even in the mean, and this coupled with the small vertical spacing can lead to severe convective stability constraints. In such situations implicit treatment of the convective terms become imperative.

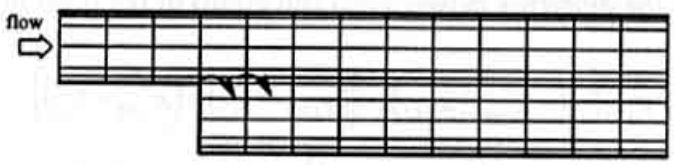

FIGURE 31.1 An example of a flow configuration that would benefit from fully implicit treatment of the momentum equation. Arrows indicate flow in the verticle direction.

In contrast to the convective stability, the viscous stability constraint does not depend directly on the velocity field and is basically determined by the smallest grid spacing in the domain. At first glance it seems that the viscous stability restriction would not be severe at high Reynolds numbers but it turns out that flows at high Reynolds numbers typically contain thin boundary/shear layers and adequate resolution of these layers requires small grid spacing and this results in a severe viscous stability constraint. Therefore, computations at all Reynolds numbers can be severely affected by the viscous stability constraint. Furthermore, explicit treatment of the viscous terms eliminates the possibility of imposing the velocity boundary conditions directly, and this is unsatisfactory in most cases. Due to these reasons, explicit treatment of all the viscous terms is generally not a viable alternative and thus, fully explicit schemes will not be discussed further. However, explicit treatment of some portions of the viscous terms is sometimes possible and indeed preferable, and this will be discussed in the last subsection in Section 31.2. It should also be pointed out that pressure *has* to be treated in an implicit manner in order satisfy divergence of the new velocity field.

Fully Implicit Schemes

These refer to time discretization schemes that include of the transport terms (convective, diffusive, and pressure) at the new time level and typically allow the use of large time steps. The Adams–Moulton family of schemes is widely used and a J_{AM}^{th}-order discretization of the momentum equation as per these schemes can be written as

$$u^{n+1} = u^n + \Delta t \sum_{j=0}^{j=J_{AM}-1} A_j^{AM}\left[-N\left(u^{n+1-j}\right) + vL\left(u^{n+1-j}\right) - G\left(p^{n+1-j}\right)\right] \tag{31.7}$$

where N, L, and G are discrete versions of the non-linear convection, linear viscous, and gradient operators, respectively, and A_j^{AM} represent the appropriate weights for the different schemes. For J_{AM} equal to 1 and $A_0^{AM} = 1$ we get the backward Euler (BE) scheme, which is first-order accurate and is unconditionally stable for convection–diffusion problems. This implies that in principle, arbitrarily large time steps can be used for advancing the solution in time, however, in practice, accuracy requirements usually impose some restriction on the time-step.

For J_{AM} equal to 2 and $A_0^{AM} = A_1^{AM} = 1/2$ we get the popular Crank–Nicolson (CN) scheme, which is second-order accurate and is also unconditionally stable for convection–diffusion problems. This scheme does not require any additional computational effort over the BE scheme and second-order accuracy is obtained at the expense of storing the right-hand side at the previous time level. One deficiency of this method is that unlike the BE scheme which produces smooth solutions for all Δt, the CN scheme damps high-frequency components very weakly and oscillatory solutions can be obtained. These properties make the BE scheme popular for calculations of steady flow or flows that vary slowly in time, whereas the CN scheme finds use mostly in time-accurate calculations of highly unsteady flows. Higher-order schemes of the Adams–Moulton family can also be obtained, and these are found to be only conditionally stable [Canuto et al., 1987].

Implicit schemes provide relief from stability constraints and allow larger time-steps, but this comes at the cost of significantly increased computational effort at every time step. Equation (31.7) along with the incompressibility constraint represents a large, coupled system of nonlinear equations that are virtually impossible to solve directly. In particular, the implicit treatment of the nonlinear convective terms necessitates the use of iterative methods (which will be discussed in Section 5), and the number of iterations required to advance the solution from one time level to the next determines the viability of the implicit method. In many cases, by using a scheme where the convective term is treated explicitly and which is therefore constrained by the CFL condition [Shyy, 1994], it is possible to advance the solution over a given time interval with less total computational effort than would be required by a fully implicit method.

Mixed Implicit–Explicit Schemes

The discussion in the previous section leads us naturally to mixed schemes that treat the nonlinear convective terms explicitly and the linear viscous and pressure terms implicitly. The Adams–Bashforth (AB) family of explicit schemes is widely used and the discretized momentum equation with convective term discretized by a J_{AB}^{th}-order AB scheme can be written

$$u^{n+1} = u^n - \Delta t \sum_{j=0}^{j=J_{AB}-1} A_j^{AB} N\left(u^{n-j}\right) + \Delta t \sum_{j=0}^{j=J_{AM}-1} A_j^{AM}\left[\nu L\left(u^{n+1-j}\right) - G\left(p^{n+1-j}\right)\right] \tag{31.8}$$

where A_j^{AB} are the coefficients for the various schemes. For J_{AB} equal to 1 and $A_0^{AB} = 1$ we get the simple first-order accurate forward Euler (FE) scheme. For J_{AB} equal to 2 the highly popular second-order Adams–Bashforth (AB2) scheme is obtained. The coefficients for this scheme are $A_0^{AB} = \frac{3}{2}$ and $A_1^{AB} = -\frac{1}{2}$ AB2 is weakly unstable for pure advection problems, but this is usually not a concern for most viscous flow simulations or simulations of Euler flows that employ dissipative numerical schemes. Higher-order (third- and fourth-order) schemes do not suffer from this deficiency but are more restrictive on the size of the time-step.

It should be pointed out that AB2 and higher order AB schemes require information from more than one previous time level and are thus not self starting, i.e., for such schemes self-starting methods such as Runge–Kutta (RK) have to be used [Press et al., 1987; Canuto et al., 1987]. The RK family of schemes, which are akin to predictor–corrector schemes, are also quite popular for flow calculations and schemes up to 4th order have been used. A RK scheme of a given order is more accurate and stable than an AB scheme of the same order. Furthermore, in contrast to AB schemes, the stability of RK schemes improves as the order increases. The added stability and accuracy of RK schemes, however, comes at the cost of more computations since a N^{th} order RK scheme requires N evaluations of the discretized set of equations per time step.

Mixed implicit–explicit schemes are particularly useful in flows where the time-step size is limited to some extent by accuracy conditions. This is sometimes the case in time-accurate calculations of turbulent flows either through direct numerical simulation (DNS) or large-eddy simultion (LES) where the need

to resolve the small time-scales imposes a relatively severe upper limit on the size of the time-step [Choi and Moin, 1994]. In simulations of such flows large time-steps that are allowed by the implicit treatment of the convective terms cannot be used in the simulation due to accuracy constraints, and implicit treatment of the convective term does not yield any significant benefits.

Mixed implicit–explicit schemes that use a combination of AB (or RK) and CN have become quite popular for time-accurate calculations of highly unsteady flows [Orszag and Kells, 1980; Kim and Moin, 1985; Karniadakis and Triantafyllou, 1992; Mittal and Balachandar, 1995, 1996; Kravchenko and Moin, 1997]. These mixed schemes require inversion of only linear operators, and this can be accomplished through either direct or iterative methods. For multidimensional problems, inversion of the full linear viscous operator can still be a formidable task, and special techniques for inverting such equations will be discussed in Section 31.5. For many problems, however, most of the viscous stability constraint comes from the wall-normal direction, since small grid spacing is required in this direction near the wall to resolve the boundary layer. In such flows, wall-tangential viscous terms can be treated explicitly without imposing a significant penalty on the size of the time step. This results in a considerable saving in CPU time, and such schemes have been used successfully before in a number of different flows [Malik et al., 1985; Mittal and Balachandar, 1996; Mittal, 1996; Fatica and Mittal, 1996].

31.3 Imposition of the Continuity Constraint

For a 3-D problem with N grid points, Eq. (31.7) or (31.8) along with the continuity constraint represents a large $4N \times 4N$ system. Thus, even with the nonlinear terms treated explicitly, the pressure strongly couples the three-momentum equations and results in a large set of coupled equations. The inversion of the full coupled system has become viable only in the last decade and has been used only for simple flows. Examples include the direct inversion technique used of Moin and Kim [1980] for stimulating turbulent channel flow and the iterative technique of Malik et al. [1985] also for channel flow. Almost all other methods devised for solving the incompressible Navier–Stokes equations try in one form or another to break the solution of the above equation set into successive solution of smaller sets of equations that are more amenable to solution on the computer. The key to developing these methods is to decouple the computation of pressure from that of the velocity field so that each velocity component and pressure can be solved for separately. A number of methods that fit this category have been developed and utilized in a variety of flow configurations and these will form the focus of this section.

Coupled Methods

These represent the most straightforward but computationally expensive class of methods for solving the incompressible Navier–Stokes equations and include the methods indicated in the previous paragraph. Klieser and Schumann [1980] have proposed an elegant technique for solving the above coupled system of equations. In this technique the equations are discretized in time using a mixed explicit–implicit scheme where viscous and pressure terms are treated implicitly and the nonlinear convection terms are treated explicitly. A discretized Poisson equation for pressure is obtained by combining the continuity constraint with the momentum equation. The discrete set of equations can then be written as

$$vL\left(u^{n+1}\right) - \frac{1}{\Delta t}u^{n+1} - Gp^{n+1} = r\left(u^{n}\right) \text{ in } \Omega \tag{31.9}$$

with

$$u^{n+1} = v^{n+1} \text{ on } \partial\Omega \tag{31.10}$$

and

$$DGp^{n+1} = -Dr \text{ in } \Omega \tag{31.11}$$

with

$$Du^{n+1} = 0 \text{ on } \partial\Omega \tag{31.12}$$

where r represents all the explicit terms and D the discrete divergence operator. Furthermore u and p now represent vectors containing discrete values of the velocity and pressure. The solution of Eq. (31.11) poses problems, since the boundary condition for pressure that would satisfy Eq. (31.12) is not known *a priori*. However, the linearity of the above set of discrete equations allows for the use of a Green's function (or *influence matrix*) technique where the solution can be decomposed into

$$\begin{bmatrix} u^{n+1} \\ p^{n+1} \end{bmatrix} = \begin{bmatrix} u^p \\ p^p \end{bmatrix} + \sum_{j=1}^{N_b} \gamma_j \begin{bmatrix} u_j^b \\ p_j^b \end{bmatrix} \tag{31.13}$$

where (u^p, p^p) represents the particular solution of the above equations with a homogeneous Dirichlet pressure boundary condition instead of Eq. (31.12). Furthermore, (u_j^b, p_j^b) represents a homogeneous solution (i.e., with $r = 0$) to the above equations with a homogeneous velocity boundary condition and $p_j^b = \delta_{ij}$ on the boundary where index i goes over all the N_b boundary points. In Eq. (31.13), γ_j represent the unknown "*influence*" coefficients, and these can be computed by utilizing the boundary constraint expressed by Eq. (31.12). This requires the inverse of a $N_b \times N_b$ "*influence matrix*," which can be precomputed once and stored at the start of the calculation. At every time step the particular solution (u^p, p^p) is obtained and from this the *influence coefficients* (which are in fact the correct boundary values of pressure) are computed and the pressure Poisson equation is solved again with the correct pressure boundary condition. Equation (31.9) is then solved for the velocity field. Thus at every time step, the exact solution to the fully coupled discretized system can be obtained at the cost of two inversions of the discretized momentum and pressure Poisson equations. Further details and applications of this method can be found in Klieser and Schumann [1980], Canuto et al. [1987], Deville et al. [1984], Ku et al. [1987], Madabhushi et al. [1993], Marcus [1984] and Mittal and Balachandar [1996].

The Penalty Formulation

This approach has been comprehensively investigated by Temam [1978], Hughes et al. [1979], Cuvelier et al. [1986], and Braaten and Shyy [1986]. In the penalty formulation, the continuity equation is viewed as an incompressibility constraint that the velocity field must obey. The result of the penalty formulation is an approximate relationship between the pressure field and the velocity field, which replaces the continuity equation. This penalty relation takes the form

$$\frac{p}{\lambda} + \frac{\partial u_i}{\partial x_i} = 0 \tag{31.14}$$

for constant property flows, where the penalty parameter

$$\lambda = \mathrm{Re} \cdot \mu \cdot \beta \tag{31.15}$$

and β is an arbitrarily large number, say, 10^7 to 10^{10}; *Re* and μ are, respectively, the Reynolds number and viscosity. In the high λ limit, the solution of the penalty formulation converges to that of the original Navier–Stokes equations (for a proof of the Stokes problem, see [Cuvelier et al., 1986]). In practice, the pressure is obtained at the penalty of not exactly satisfying the divergence-free condition of the velocity

field, and hence the name "penalty formulation." A major computational advantage of the penalty formulation is that pressure appears explicitly in the continuity equation. Hence, no extra manipulation is needed for treating the continuity and momentum equations. Alternatively, one can also substitute the approximate penalty relationship, given above, into the momentum equation and eliminate the pressure term altogether. Since the penalty parameter is large, care should be taken to avoid creating inaccuracy in the pressure field. This formulation is attractive in a coupled system, but is not necessarily most competitive in terms of computational efficiency if cast in curvilinear coordinates [Braaten and Shyy, 1986]. Furthermore, for more complex problems involving variable fluid properties, the approximate penalty formulation replacing the continuity equation needs to be revised.

The Artificial Compressibility Method

The artificial compressibility method was originally proposed by Chorin [1967], and has since, been extensively developed and applied to a host of fluid flow problems [Kwak et al., 1986; Hosangadi et al., 1990; Shuen et al., 1993]. It introduces an artificial pressure term into the continuity equation:

$$\frac{\partial p}{\partial t}+\beta^2\frac{\partial u_i}{\partial x_i}=0 \tag{31.16}$$

where β can be interpreted as related to an artificial speed of sound, such that

$$p=a^2\cdot\rho \tag{31.17}$$

Obviously the artificial speed of sound depends on the choice of β. As demonstrated by Eq. (31.14) and (31.16), the penalty formulation and the artificial compressibility method share some commonality. However, by introducing a pseudo transient term, the artificial compressibility method exhibits a wave equation appearance. To see the method more clearly, consider the one-dimensional inviscid system, governed by

$$\frac{\partial}{\partial\tau}\begin{bmatrix}p\\u\end{bmatrix}+[A]\frac{\partial}{\partial x}\begin{bmatrix}p\\u\end{bmatrix}=0 \tag{31.18}$$

where

$$[A]=\begin{bmatrix}0 & \beta^2\\1 & 2u\end{bmatrix} \tag{31.19}$$

is the coefficient matrix, and τ is the artificial time coordinate. One can perform a similarity transform to decouple the two equations given in the system (31.18). Then the eigenvalues of the matrix [A] define the speed of sound in the transformed, decoupled equations [Shyy, 1994]. The eigenvalues are

$$\lambda_{\pm}=u\pm\sqrt{u^2+\beta^2} \tag{31.20}$$

which are real and of opposite signs. Hence, the artificial compressibility method results in a hyperbolic system of equations even for steady-state problems. The effective speed of sound is

$$a=\sqrt{u^2+\beta^2} \tag{31.21}$$

which means that the artificial speed of sound now depends on the local fluid velocity, u. If we simply adopt the unsteady, inviscid compressible flow equations and take the low Mach number limit to seek the solution to the incompressible flow, the disparity between the physical speed of sound and the local fluid velocity is very large, causing extremely stringent requirement in choosing the time step size. The artificial compressibility method modifies the value of the speed of sound computationally to reduce the stringent condition imposed by the zero Mach number condition. Hence, it is desirable to ensure that a and u are comparable. However, it is not always straightforward to choose appropriate values of β. For more information, see Kwak et al. [1986], Shuen et al. [1993].

The above discussion is aimed at solving steady-state flow problems. For time-dependent problems, one can introduce the dual time-stepping concept. Specifically, we can rewrite the system (31.18) by incorporating the physical time coordinate, in addition to the artificial time coordinate:

$$\frac{\partial}{\partial t}\begin{bmatrix}0\\u\end{bmatrix}+\frac{\partial}{\partial \tau}\begin{bmatrix}p\\u\end{bmatrix}+[A]\frac{\partial}{\partial x}\begin{bmatrix}p\\u\end{bmatrix}=0 \tag{31.22}$$

where t is the physical time coordinate and τ is the artificial time coordinate. To obtain a time accurate solution, we simply march along the physical time, t, and at every time instant, we march along the artificial time, τ until the solution converges in τ. Then we go on to the next time instant in the physical coordinate, t. Similarly, if we deal with variable density flows (but still at zero Mach number), then, we can simply add the density terms, in both space and time derivatives, to the governing equations, and solve the system of equations accordingly.

The Projection Method

An efficient alternative to solving the incompressible Navier–Stokes equations is through the so called *fractional-step* or *time-split* method, which relies on the idea of operator splitting [Yanenko, 1971] to decouple the computation of the pressure from that of the velocity field. The method was proposed independently by Chorin [1968] and Temam [1969] and has since then become quite popular for time-accurate solution of the incompressible Navier–Stokes equations. There are a number of different version of this method, but the basic idea can be described as follows: Given the velocity and pressure field at time level 'n' the velocity at the next time level '$n+1$' is obtained by first advancing through a convection diffusion step and computing an intermediate velocity field u^*. For instance, if a mixed implicit–explicit scheme is used when the discretized convection-diffusion equation is given by

$$\frac{u^*-u^n}{\Delta t}=-\sum_{j=0}^{j=J_{AB}-1}A_j^{AB}N\left(u^{n-j}\right)+\Delta t\sum_{j=0}^{j=J_{AM}-1}A_j^{AM}vL\left(u^{n+1-j}\right) \tag{31.23}$$

with

$$u^*=v^{n+1}\ \text{on}\ \partial\Omega \tag{31.24}$$

Note that in Eq. (31.23), u^* would be used in the implicit terms instead of u^{n+1}. This step is followed by the pressure correction step

$$\frac{u^{n+1}-u^*}{\Delta t}=-Gp^{n+1} \tag{31.25}$$

which is accompanied with the boundary condition

$$u^{n+1} \cdot \tilde{n} = v^{n+1} \cdot \tilde{n} \tag{31.26}$$

and the condition

$$Du^{n+1} = 0 \tag{31.27}$$

In Eq. (31.26) $\tilde{n}$ denotes a unit vector normal to the boundary. Note that the pressure correction consists of a set of inviscid equations and is thus well posed only if the normal velocity is specified at the boundary. Equations (31.25) and (31.27) together give the following Poisson equation for pressure:

$$DGp^{n+1} = \frac{Du^*}{\Delta t} \tag{31.28}$$

which is solved with a homogeneous Neumann pressure boundary condition implied by Eq. (31.24), (31.25), and (31.26). Thus Eq. (31.23) is first solved for the intermediate velocity and following that the pressure Poisson equation (31.28) is solved. The final step is to use Eq. (31.25) to correct the velocity. It should be pointed out that the intermediate velocity field is nonsolenoidal and the pressure correction step can be viewed as the projection of this velocity field into solenoidal space. This is why these type of splitting methods when applied to the incompressible Navier–Stokes equations are also called "projection" methods.

The simplicity and efficiency of the above method comes at the cost of errors incurred due to the splitting of the Navier–Stokes equations. First, in the method outlined above the final velocity satisfies the boundary condition on the tangential velocity only up to $O(\Delta t)$. However, this is easily remedied; for instance this error can be reduced to $O(\Delta t^3)$ by employing the following boundary condition for the intermediate tangential velocity:

$$u^* \cdot \tilde{t} = v^{n+1} \cdot \tilde{t} + \Delta t\left(2Gp^n - Gp^{n-1}\right) \cdot \tilde{t} \tag{31.29}$$

where $\tilde{t}$ is a unit vector tangential to the boundary [Kim & Moin, 1985; Canuto et al., 1987].

Furthermore, it can also be shown that the splitting procedure can affect the accuracy of the final solution. For instance when a second-order time discretization is used, it can be shown that the pressure obtained from solving Eq. (31.28) differs from the real pressure by $O(\nu\Delta t)$, whereas the error in velocity due to the splitting is $O(\Delta t^2)$ [Kim and Moin, 1985]. For flows at reasonably high Reynolds numbers, the error in pressure might be acceptable, but there also exist variations of this method that allow for higher accuracy in the pressure computation [Orszag et al., 1986; Marcus, 1984; Karniadakis et al., 1991; Mittal and Balachandar, 1996]. The splitting error is intimately coupled with the boundary conditions that are used for the intermediate velocity and pressure. The issue of intermediate velocity and pressure boundary conditions exists in the first place because the continuous equations are split *before* they are discretized. If on the other hand the equations are discretized and velocity boundary conditions included into the discretized system before splitting is accomplished, then the issue of intermediate boundary conditions does not arise. Dukowicz and Dvinsky, [1992] have outlined such an approach where they obtain a consistent set of decoupled equations by viewing the splitting as an approximate factorization of the discretized Navier–Stokes equations. This approach provides the most satisfactory resolution of the issue of splitting errors. Other approaches for restoring the accuracy of the time-discretization scheme can be found in Van Kan [1986] and Perot [1993].

It should be pointed out that even though the fractional step method discussed here uses a mixed explicit–implicit scheme, inclusion of a fully implicit scheme for the convection–diffusion terms does not pose any special problems. Choi et al. [1993] have used a Newton iteration technique in conjunction

with a fractional step scheme to simulate turbulent flow over a riblet mounted surface where fully implicit treatment of the convection–diffusion terms allowed larger time-steps to be used.

The Pressure-Correction Method (SIMPLE)

This is another class of method that solves the incompressible Navier–Stokes equations and has been widely used to solve a variety of flow problems. This method was originally envisioned for computing convection dominated flows and was therefore designed as a fully implicit method, i.e., where both the convection and diffusion terms were treated implicitly. The implicit time advancement is particularly useful for computing steady flows, since large time-steps can be used to approach the desired solution. This method solves the fully implicit equations iteratively though a set of guesses and corrections to the velocity and pressure field that successively move closer to the solution of the full implicit system given in Eq. (31.7). Consider here the solution to the steady state problem and suppose that at iteration level $m-1$ we obtain a solution (u^{m-1}, p^{m-1}) that satisfies the *linearized* momentum and continuity equations. "Linearized" implies that the coefficients in the nonlinear operator that depend on the velocity field are lagged behind in the iteration process, i.e., they depend on the velocity at level $m-2$. Now the coefficients in the nonlinear operator can be updated with the latest available velocity field, and the idea is to come up with a solution at the next iteration level (u^m, p^m) that satisfies the following set of equations:

$$M^{u^{m-1}}\left(u^m\right) = -G\left(p^m\right) \tag{31.30}$$

$$Du^m = 0 \tag{31.31}$$

$M^{u^{m-1}} = (N^{u^{m-1}} - \nu L)$ represents the convection–diffusion operator that has been linearized about u^{m-1}. An estimate for the velocity field that satisfies the above momentum equation can be obtained by using a guess for the pressure field, which is usually taken to be the pressure at the previous iteration level. Thus the guessed velocity field u^{m^*} satisfies the following equation:

$$M^{u^{m-1}}\left(u^{m^*}\right) = -G\left(p^{m-1}\right) \tag{31.32}$$

Since the pressure in the above equation is not the correct pressure, the guessed velocity field will not satisfy the continuity equation. The required velocity and pressure fields can be obtained by adding corrections to the guessed velocity and pressure fields, viz.,

$$p^m = p^{m-1} + p'; \qquad u^m = u^{m^*} + u' \tag{31.33}$$

Now using Eq. (31.30) and (31.32) an equation for the velocity and pressure corrections can be obtained, and this is given by

$$M^{u^{m-1}}\left(u'\right) = -G\left(p'\right) \tag{31.34}$$

For reasons that will become clear soon, the convection–diffusion operator can be split as $M^{u^{m-1}} = M_D^{u^{m-1}} - M'^{u^{m-1}}$ where $M_D^{u^{m-1}}$ is the diagonal part of the operator and $M'^{u^{m-1}}$ is the off-diagonal part that couples the velocity field at one grid location to the neighboring points. Equation (31.34) can then be written as

$$M_D^{u^{m-1}}\left(u'\right) = M'^{u^{m-1}}\left(u'\right) - G\left(p'\right) \tag{31.35}$$

At this point the SIMPLE algorithm proceeds by neglecting the first term on the RHS of the above equation and the velocity correction is then given by

$$u' = -\left[M_D^{u^{m-1}}\right]^{-1} G(p') \tag{31.36}$$

The corrected velocity field is then given by

$$u^m = u^{m^*} - \left[M_D^{u^{m-1}}\right]^{-1} G(p') \tag{31.37}$$

Now using Eq. (31.31) we get

$$D\left[M_D^{u^{m-1}}\right]^{-1} G(p') = D\left(u^{m^*}\right) \tag{31.38}$$

which is nothing but a discretized Poisson equation for the pressure correction. Thus Eq. (31.32) is solved first followed by the pressure correction equation (31.38). Finally, Eq. (31.36) is used to obtain the corrected velocity which is divergence free. At this point one iteration is complete and the procedure is now repeated at the new iteration level. Iterations are carried out until the solution satisfies the fill implicit equations to within some predetermined error bound. It should be pointed out that the velocity at the end of each iteration satisfies the continuity equation *exactly*, which is absolutely essential for computation of incompressible flows.

The great simplicity of the SIMPLE algorithm comes from neglecting the operator that couples neighboring velocity values in the equation for the velocity correction [Eq. (31.35)]. However, this can also cause slow convergence of the conventional SIMPLE method [Braaten and Shyy, 1986] for highly coupled systems. The elliptic nature of the pressure field makes it especially sensitive to the omission of the velocity coupling term, and it has been found that the above solution procedure tends to overpredict the pressure correction and underrelaxation has to be resorted to in order to stabilize the iterative procedure. It is useful to point out that this algorithm does not suffer from any ambiguity regarding the boundary conditions. The momentum equation (31.32) is solved with the correct velocity boundary conditions and the pressure correction equation is solved with a homogeneous Neumann boundary condition. This results in the finite slip velocity at the end of the iteration that is proportional to the tangential gradient of the pressure correction. However, as the iteration converges, both pressure and velocity corrections approach small values and so does the slip velocity.

SIMPLER and PISO

There also exist a number of variations of the SIMPLE algorithm that are designed to mitigate the deficiencies of the SIMPLE algorithm. A popular variation is the SIMPLER method [Patankar, 1980], where the velocity correction is obtained as in the SIMPLE algorithm but the pressure is recomputed so as to include the influence of the *corrected* velocity field. This method generally converges faster than SIMPLE and does not require underrelaxation. Other variations such as SIMPLEC [Van Doormal and Raithby, 1984] and PISO [Issa, 1986; Issa et al., 1986] have also been used, and the interested reader is referred to the respective papers. SIMPLEC basically attempts to account for the incompatible time scales between pressure and velocity fields, due to the truncation procedure introduced by SIMPLE. SIMPLER and PISO offer more direct coupling between pressure and velocity in the course of devising the pressure equation. For steady-state problems, the two methods are essentially the same [Braaten, 1984; Braaten and Shyy, 1986]. In essence, the PISO method employs the Poisson-based pressure equation, which directly incorporates the mass continuity constraint, similar to the earlier concept developed for the marker-and-cell (MAC) method [Harlow and Welch, 1965]. However, in solving for the pressure field,

PISO resorts to the noniterative approach by adopting a series of predictor–corrector steps with intermediate solutions of the velocity and pressure fields. By following this approach, PISO can handle both steady-state and time-dependent problems in the same framework. Furthermore, in PISO (and SIMPLER), similar to the SIMPLE method, the solution procedure is sequentially conducted between momentum and pressure equations. This way, PISO can separate the implicit steps between the pressure and velocity variables, while still retaining all the terms in each governing equation without truncating them.

Remarks on Various Solvers

Clearly, a number of alternative solvers are available for computing fluid flow problems. In principle, all the techniques reviewed in this section are applicable to low Mach number flows, with constant or variable properties. In practice, however, they have demonstrated uneven performance characteristics in terms of accuracy, efficiency, and robustness. Strictly speaking, the accuracy issue should be separated from the pressure–velocity treatment, except for splitting errors associated with fractional-step methods. Unfortunately, quite often one equates accuracy associated with, say, convection treatments, with that of the basic flow solver. Since very little of a flow solver's performance characteristics can be rigorously proved, it is an extremely challenging task to compare the various solvers and to pick out the best one. While valuable efforts have been made in this regard (e.g., [Merkle et al., 1992]), it seems clear that the relative merit of a given solver will depend on the type of the flow being computed (e.g., Reynolds number, body force, density variation), the complexity of the flow configuration, the boundary conditions, and the time-dependency of the problem. Qualitatively, we summarize some known characteristics of these solvers in the following.

Fractional step methods have found wide use in time-accurate simulations of unsteady flows. These methods have been used in a variety of flows including internal/confined as well as external flows and have been used in conjunction with almost all types of spatial discretization schemes including finite difference, finite volume, finite element, spectral and spectral element. The splitting error associated with the conventional fractional step schemes, however, makes it unsuitable for boundary driven flows such as Taylor–Couette flow [Marcus, 1984] and turbulent thermal convection and the method has to be modified to handle such flows. Furthermore, projection methods have not been extensively applied to flows that have strong flow dependent source terms, such as reactive flows. However, a number of improvements have been proposed to the conventional fractional step schemes, and it remains to be seen to what extent these remove the deficiencies inherent in the splitting approach.

For cases where the splitting errors are unacceptable, one has to resort to either coupled methods or the SIMPLE, SIMPLER, and PISO types of algorithms. For mixed explicit–implicit time discretization, the influence matrix technique is a viable option, but this method is limited to constant viscosity flows. The SIMPLE family of schemes has been used widely for simulation of steady flows. The extension to unsteady flows is quite straightforward, but the actual computation of unsteady flows using this technique can be quite time-consuming. The fully implicit time discretization, however, results in a very robust solution procedure, and strong flow dependent source terms do not degrade the integrity of the solution algorithm. It has also been found that when the fluid flow geometry is complicated/irregular, the performance of this method does not deteriorate significantly. Owing to the intrinsic robustness of this scheme it has been applied to solve some highly challenging practical problems involving combustion, turbulence, 2nd phase change [Patankar, 1980; Shyy, 1994], and has also been extended to high Mach number flow problems [Shyy, 1994].

The artificial compressibility method has also been applied to numerous flow problems, including those that involve moving boundaries [Kiris et al., 1991] and combustion [Hosangadi, et al., 1990]. To combine it with the compressibility effects, one needs to distinguish the physical and the nonphysical portions of governing laws to determine the speed of sound appropriately [Ramshaw and Mosseau, 1991].

31.4 Convection Schemes for Complex Flow Problems

In addition to pressure and continuity constraints, the convection term is a distinguishing feature of fluid flow computations. The difficulties of treating convection are that (1) it is nonlinear, and (2) it is a first derivative term, representing a wave characteristic in contrast to the second-order dissipation term. It is a challenging task to reproduce the intrinsic feature of convection with high fidelity. In general, one is faced with either excessive numerical smearing or undesirable oscillations in the vicinity of sharp gradients. In the following, we summarize the characteristics of several convection schemes that have often been applied to convection-dominated flow problems encountered in engineering practices, as documented in Shyy [1994].

Here for simplicity, we shall use a uniform Cartesian grid to illustrate the salient points; the formulas on curvilinear nonuniform grids can be derived based on identical concepts. For example, the convection flux on the east interface of the cell centered at the location P is estimated as

$$F_e = (\rho u)_e \delta y = \frac{(\rho_P + \rho_E)}{2} \frac{(u_P + u_E)}{2} \delta y \tag{31.39}$$

where δy is the height of the cell interface, subscripts e and E designate, respectively, the quantity at the interface and the quantity at the nodal point to the right of the point P. In the index notation, P, e, and E correspond to, respectively, i, $i + 1/2$, and $i + 1$. In devising and interpreting the property of a convection scheme, one can resort to either an algebraic approach utilizing, say, the standard central difference scheme and an added amount of numerical diffusion [MacCormack and Baldwin, 1975; Jameson et al., 1981; Harten, 1983] or a geometric approach utilizing different interpolation profiles to construct the flux distribution [Boris and Book, 1973; Hirsch, 1990].

In the framework of an algebraic approach, for example, for constant u,

$$L_{u1}(\phi_{ij}) = L_{c2}(\phi_{ij}) - \frac{u\Delta x}{2} D_{c2}(\phi_{ij}) \tag{31.40}$$

where L_{u1}, L_{u2}, D_{c2} are, respectively, first-order upwinding for $(u\phi)_x$, second-order central differencing for $(u\phi)_x$, and second-order central differencing for ϕ_{xx}. Corresponding interpretations can also be made for the QUICK (quadratic upstream interpolation for convective kinematics) and the second-order upwind schemes [Shyy, 1994]. If the fourth-order central difference operator approximating $(u\phi)_x$ and the second-order central difference operator approximating ϕ_{xxx} are defined as $L_{c4}(\phi_{ij})$ and $D_{c4}(\phi_{ij})$, respectively, and the QUICK approximation [Leonard, 1979] to $(u\phi)_x$ as L_{QU}, then

$$L_{QU}(\phi_{ij}) = \frac{1}{4} L_{c2}(\phi_{ij}) + \frac{3}{4} L_{c4}(\phi_{ij}) + \frac{u(\Delta x)^3}{16} D_{c4}(\phi_{ij}) \tag{31.41}$$

showing that the QUICK scheme is equivalent to using 25% second-order accurate central differencing, 75% fourth-order accurate central differencing, and some fourth derivative damping. On the other hand, if the second-order upwind approximation to $(u\phi)_x$ is defined as L_{u2}, then

$$L_{u2}(\phi_{ij}) = -2L_{c2}(\phi_{ij}) + 3L_{c4}(\phi_{ij}) + \frac{u(\Delta x)^3}{4} D_{c4}(\phi_{ij}) \tag{31.42}$$

or, the second-order upwind scheme is equivalent to using 300% fourth-order accurate central differencing, then subtracting 200% second-order central differencing and adding some fourth derivative damping.

In the following, the geometric interpolation procedure will be adopted to facilitate the derivation of the various schemes. Different convection schemes use different interpolations, including the first- and second-order upwind schemes, second-order central difference scheme, and the QUICK scheme are illustrated in Figure 31.2.

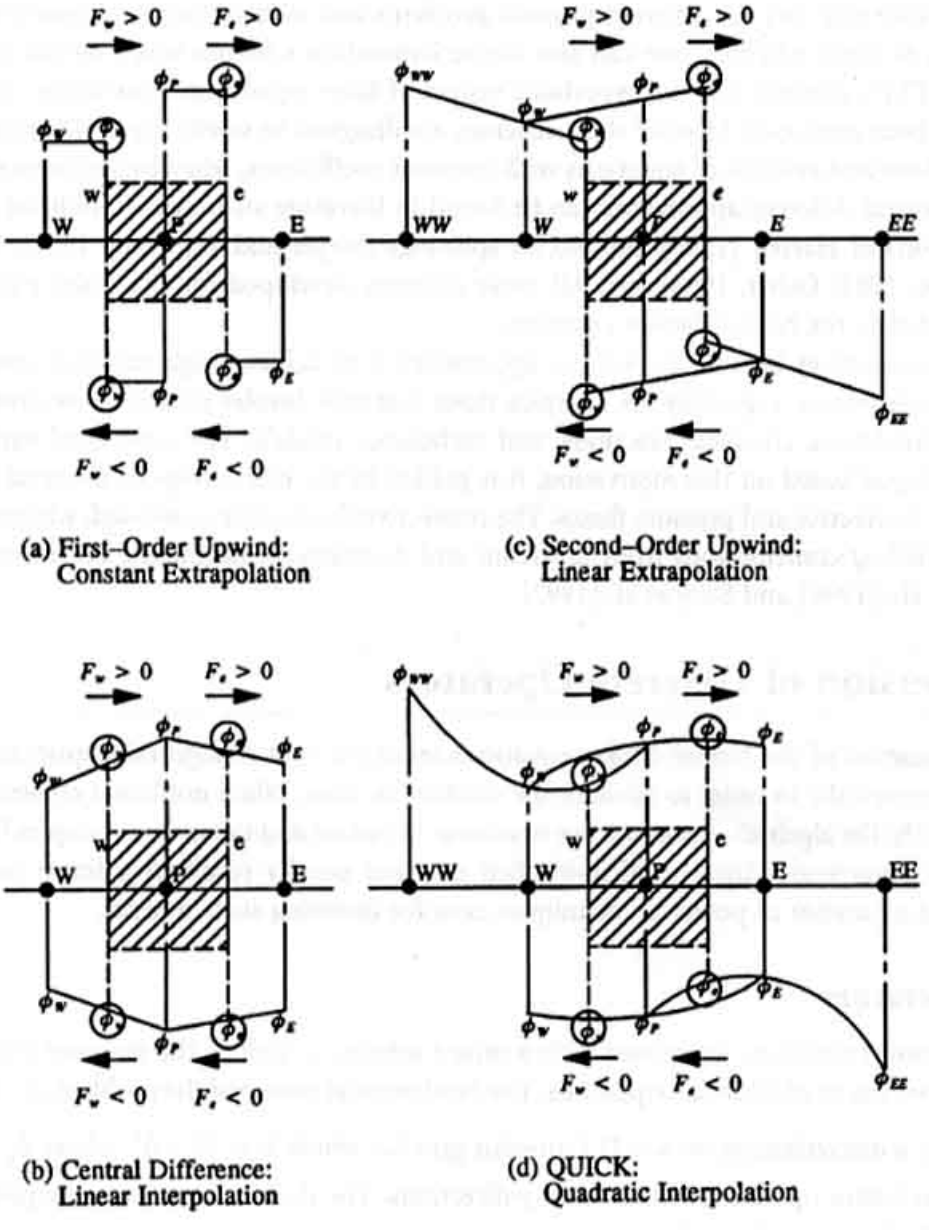

FIGURE 31.2 Schematic of geometric interpolation/extrapolation employed by the various convection schemes.

Compared to the first-order upwind scheme, the shape function utilized for the extrapolation procedure for other three schemes are of a higher order, allowing variations in the solution profile between the adjacent nodes. With the use of a linear extrapolation, instead of a piecewise constant extrapolation, the second-order upwind scheme is capable of accommodating flow fields that span a wide range of the cell Reynolds number. Many alternatives can be devised to implement the second-order upwind scheme and the QUICK schemes; however, by utilizing the conservative, finite-volume treatments, the guidelines for devising appropriate implementations of such schemes become clearer [Thakur and Shyy, 1993; Shyy, 1994].

As for the local order of accuracy, if we do a Taylor series expansion about the velocity at the point P, we get a second-order accurate scheme. Compared to the first- and second-order upwind schemes, the QUICK scheme assumes a parabolic shape profile for conducting the interpolation procedure. From this

viewpoint, it seems that although QUICK is still a second-order scheme in terms of the formal order of accuracy, it can yield a higher degree of actual accuracy for many practical computations. On the other hand, it is well known that the performance of a convection scheme is not necessarily correlated with either the formal order or the complexity of the scheme [Shyy, 1994]. It is not appropriate to make a universal assertion regarding the relative performance among these higher-order upwind schemes, since their performance can vary for different physical problems and with varying grid resolutions.

In addition to these schemes, one can also derive convection schemes based on the total-variation-diminishing (TVD) concept. For the hyperbolic system of Euler equations, a number of high resolution schemes have been proposed. Most of these schemes are designed to satisfy the TVD property for scalar conservation laws and systems of equations with constant coefficients, whereby spurious oscillations are suppressed. Several different approaches can be found in literature such as the modified flux approach (scalar diffusion) of Harten [1983], flux vector splittings [Steger and Warming, 1981], flux difference splittings [Roe, 1981; Osher, 1984], etc. All these schemes developed for the Euler equations can be directly extended to the Navier–Stokes equations.

The main motivation behind the various approaches is to achieve high accuracy and efficiency in numerical computations, especially for complex flows that may involve strong convective effects, sharp gradients, recirculation, chemical reactions, and turbulence models. The controlled variation scheme (CVS) is developed based on this motivation. It is guided by the eigenvalues of the total flux as well as the individual convective and pressure fluxes. The convective flux is fully upwinded, whereas the pressure flux is split yielding contributions from upstream and downstream neighbors. For more information, see Thakur et al., [1996] and Shyy et al. [1997].

31.5 Inversion of Discrete Operators

The discretization of the Navier–Stokes equations results in a set of algebraic equations that need to be inverted numerically in order to advance the solution in time. When nonlinear convection terms are treated implicitly, the algebraic equations are nonlinear in nature and iterative techniques have to be used to solve these equations. Mixed explicit–implicit schemes usually result in a linear set of algebraic equations, and a number of powerful techniques exist for inverting such systems.

Linear Operators

The momentum equation discretized with a mixed scheme as well as the pressure Poisson equation result in a linear system of algebraic equations. The fundamental nature of the problem can be understood by considering a discretization on a 2-D Cartesian grid for which $L = \delta_x^2 + \delta_y^2$ where $\delta_x + \delta_y$ represent the discrete derivative operators in the x and y directions. The discrete momentum equation for the u_i component of velocity can be written

$$\left[I - \Delta t \nu\left(\delta_x^2 + \delta_y^2\right)\right]\left\{u_i\right\} = \left\{r_i\right\} \tag{31.43}$$

whereas the discrete Poisson equation for pressure can be written as

$$\left[\delta_x^2 + \delta_y^2\right]\left\{p\right\} = \left\{s\right\} \tag{31.44}$$

If N_x and N_y are the number of grid points in the x and y directions, respectively, and $N = N_x N_y$ is the total number of grid points, then the matrix on the LHS of the above two equation are of size $N \times N$ and $\{u_i\}$, $\{r_i\}$, $\{p\}$, and $\{s\}$ are vectors of size N that contain the discrete values of the velocity, explicit term, pressure, and source term, respectively. The above systems of equations can also be written as

$$[A]\{\phi\} = \{b\} \tag{31.45}$$

where $[A]$ represents either of the linear operator in the above equation and $\{\phi\}$ and $\{b\}$ the corresponding variable and source term, respectively. A number of direct techniques can be used to invert this linear algebraic system. The most basic method is Gauss elimination. The core of this method is a series of linear row/column operations that transform the matrix $[A]$ into a triangular matrix from which the solution of Eq. (31.45) can be obtained easily. Consider Eq. (31.45) rewritten as

$$\begin{bmatrix} A_{11} & A_{12} & A_{13} & \cdots & A_{1N} \\ A_{21} & A_{22} & A_{23} & \cdots & A_{2N} \\ A_{31} & A_{32} & A_{33} & \cdots & A_{3N} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ A_{N1} & A_{N2} & A_{N3} & \cdots & A_{NN} \end{bmatrix} \begin{Bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \\ \vdots \\ \phi_N \end{Bmatrix} = \begin{Bmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_N \end{Bmatrix} \tag{31.46}$$

To transform the above system into an upper triangular one we start by zeroing out the elements A_{i1}; $i = 2,3, \cdots, N$. For instance A_{21} can be zeroed by subtracting from (*row*-1) $\times A_{i,1}/A_{1,1}$ from (*row*-2). The corresponding element on the RHS b_2 is also changed by subtracting $b_1 \times A_{21}/A_{11}$ from it. The first elements of the other rows can also be zeroed out in a similar manner. At the end of these operations one obtains the following altered system of equations:

$$\begin{bmatrix} A_{11} & A_{12} & A_{13} & \cdots & A_{1N} \\ 0 & A'_{22} & A'_{23} & \cdots & A'_{2N} \\ 0 & A'_{32} & A'_{33} & \cdots & A'_{3N} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 0 & A'_{N2} & A'_{N3} & \cdots & A'_{NN} \end{bmatrix} \begin{Bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \\ \vdots \\ \phi_N \end{Bmatrix} = \begin{Bmatrix} b_1 \\ b'_2 \\ b'_3 \\ \vdots \\ b'_N \end{Bmatrix} \tag{31.47}$$

where the primes indicate the elements altered by the row operations. The above procedure can now be applied recursively to the reduced system enclosed by the dashed lines to zero out the subdiagonal elements in the successive columns such that at the end one obtains a upper triangular system of equations given by

$$\begin{bmatrix} A_{11} & A_{12} & A_{13} & \cdots & A_{1N} \\ 0 & A_{22} & A_{23} & \cdots & A_{2N} \\ 0 & 0 & A_{33} & \cdots & A_{3N} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & 0 & \cdots & A_{NN} \end{bmatrix} \begin{Bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \\ \vdots \\ \phi_N \end{Bmatrix} = \begin{Bmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_N \end{Bmatrix} \tag{31.48}$$

where the primes have been removed and the A's now represent the elements of the transformed triangular matrix and b's the transformed elements of the RHS vector. Once the triangular matrix is obtained the solution can be obtained easily by back-substitution. This is initiated by computing $\phi_N = b_N/A_{NN}$ and followed by successively computing ϕ_{N-1} to ϕ_1. This procedure is quite straightforward but unfortunately suffers from two main deficiencies. Firstly the operation count for the above method is $O(N^3)$, which makes it impractical for most problems in fluid dynamics. Second, the procedure requires *a priori* knowledge of the vector on the RHS, which makes the method very expensive when the system of equations has to be solved with multiple RHS vectors.

A solution to the second problem is resorting to the method of LU decomposition where matrix $[A]$ is factored into a lowered triangular matrix $[L]$ and an upper triangular $[U]$. Thus the system of equations can be written as

$$[A]\{\phi\} = [L][U]\{\phi\} = \{b\} \tag{31.49}$$

The solution of the above set of equations can be obtained by solving for a vector $\{\xi\}$ such that

$$[L]\{\xi\} = \{b\} \tag{31.50}$$

followed by

$$[U]\{\phi\} = \{\xi\} \tag{31.51}$$

The usefulness of the above decomposition becomes clear when we recognize that the solution of Eq. (31.50) and (31.51) is trivial and follows along the line of the back-substitution procedure. It should be pointed out that inverting the above two equations is also a $O(N^3)$ process and at first glance does not represent any significant advantage over Gauss elimination. However, LU decomposition does not require *a priori* knowledge of the right-hand side, and once the LU decomposition has been computed, we can solve for as many right-hand sides as we want. This is a situation that is actually encountered in unsteady flows where the same operator needs to be inverted at every time step albeit with different source terms on the right-hand side. The same is true for iterative solution procedures.

Another method that has been used for direct inversion of equations like (31.45) primarily in conjunction with spectral-method based solvers is the matrix-diagonalization technique of Haidvogel and Zang [1979]. Consider the pressure Poisson equation (31.44), which can be rewritten as

$$[D_x^2][\phi] + [\phi][D_y^2]^T = [s] \tag{31.52}$$

where $[D_x^2]$ is a $N_x \times N_x$ matrix that represents the discrete second derivative in the x direction and $[D_y^2]$ is the $N_y \times N_y$ second derivative matrix in the y direction. Furthermore, $[\phi]$ and $[b]$ are $N_x \times N_y$ matrices such that the (ij) element of these matrices denotes the discrete value at the (x_i, y_j) location. The derivative matrices can be decomposed as

$$[D_x^2] = [M][\lambda][M]^{-1}; \qquad [D_y^2]^T = [N][\beta][N]^{-1} \tag{31.53}$$

where $[M]$ and $[N]$ represent the eigenvector matrices of $[D_x^2]$ and $[D_y^2]$, respectively, and $[\lambda]$ and $[\beta]$ are the corresponding diagonal eigenvalue matrices. The decomposition is substituted into (31.52) and after a little bit of manipulation can be written as

$$[\lambda][\phi'] + [\phi'][\beta] = [s'] \tag{31.54}$$

where

$$[\phi'] = [M]^{-1}[\phi][N]; \qquad [s'] = [M]^{-1}[s][N] \tag{31.55}$$

The solution to Eq. (31.54) can be easily obtained as

$$\phi'_{ij} = \frac{s'_{ij}}{\lambda_i + \beta_j} \tag{31.56}$$

and $[\phi]$ can then be obtained from Eq. (31.55). The eigenvector and eigenvalue matrices can be precomputed and stored, and the solution can then be obtained by a series of matrix–matrix multiplications. The extension of this technique to Eq. (31.43) is also straightforward. Like LU factorization, this technique does not require *a priori* knowledge of the RHS. Furthermore, for problems with a third periodic direction, the memory requirement and computational cost for implementing this technique is usually only a fraction of that required by the LU factorization procedure. However, it should be pointed out that this technique is only applicable when the operator to be inverted is separable in terms of the independent variables. Furthermore, straightforward extension of this technique to include a third (nonperiodic) direction is not possible.

There is another family of highly efficient techniques that can provide approximate solution of Eq. (31.43) which are categorized as approximate factorization and ADI (alternating direction implicit). The approximate factorization of the equation can be accomplished by writing Eq. (31.43) as

$$\left[I - \Delta t\, \nu \delta_x^2\right]\left[I - \Delta t\, \nu \delta_y^2\right]\{u_i\} = \{r_i\} - (\Delta t\, \nu)^2 \left[\delta_x^2\right]\left[\delta_y^2\right]\{u_i\} \tag{31.57}$$

If one thinks of the above equation as being discretized with a first-order backward Euler scheme, then the last term can be safely neglected, since it is $O(\Delta t^2)$. The approximately factored equations can then be solved in two steps as follows:

$$\left[I - \Delta t\, \nu \delta_x^2\right]\{u_i^*\} = \{r_i\} \tag{31.58}$$

followed by

$$\left[I - \Delta t\, \nu \delta_y^2\right]\{u_i\} = \{u_i^*\} \tag{31.59}$$

Such an approximate factorization can also be obtained for a second-order Crank–Nicolson scheme. The efficiency of this scheme comes from the fact that the approximately factored system requires the inversion of a smaller system of equations. For instance the solution for the above equations can be obtained by inverting one $N_x \times N_x$ matrix in Eq. (31.58) and a $N_y \times N_y$ matrix in Eq. (31.59) instead of a $N \times N$ that is required by Eq. (31.43). Quite often the solution to the two equations above can be obtained by direct inversion. This is especially the case when a second-order central difference scheme is used for spatial differentiation. In that case the matrix on the LHS of Eq. (31.58) and (31.59) is tridiagonal and can be inverted with ease using the Thomas algorithm [Press et al., 1987].

The alternating direction implicit (ADI) techniques is similar to approximate factorization in that here too the idea is to obtain the approximate solution of Eq. (31.43) by solving a series of smaller problems. Here the solution is advanced from time step n to $n + 1$ through an intermediate solution at $n + 1/2$ by solving the following equation:

$$\left[I - \frac{\Delta t}{2}\nu\delta_x^2\right]\left\{u_i^{n+\frac{1}{2}}\right\} = \left\{r_i\left(u^n\right)\right\} + \frac{\Delta t}{2}\nu\left[\delta_y^2\right]\{u_i^n\} \tag{31.60}$$

The solution at $n + 1$ is then obtained by solving the following equation:

$$\left[I-\frac{\Delta t}{2}\nu\delta_y^2\right]\left\{u_i^{n+1}\right\}=\left\{r_i\left(u^{n+\frac{1}{2}}\right)\right\}+\frac{\Delta t}{2}\nu\left[\delta_x^2\right]\left\{u_i^{n+\frac{1}{2}}\right\} \tag{31.61}$$

Thus the first step is implicit in x and explicit in y, whereas the second step is implicit in y and explicit in x. The above equations are similar in structure to the approximately factored equations, and direct methods can often be used for their inversion. It should be pointed out that ADI and approximate factorization schemes are usually designed so as to retain the accuracy and stability characteristics of the original discrete equations, but there are exceptions where stability characteristics may be lost [Shyy, 1994].

The pressure Poisson equation does not contain the time-derivative term so that methods such as approximate factorization or ADI cannot be directly used for solving this equation. However, this suggests that addition of a time derivative term to the Poisson equation might lead to solution methods for these equations. The idea would then be to integrate the resulting equation to a steady state where the solution to the original Poisson equation would be recovered. Consider for instance the addition of a time-derivative to the Poisson equation and the resulting equation discretized by a forward Euler scheme in time and a central scheme in space. The discrete equation can then be written as

$$p_{i,j}^{n+1}=p_{i,j}^{n}+\frac{\Delta t}{\Delta^2}\left(p_{i+1,j}^{n}+p_{i-1,j}^{n}+p_{i,j+1}^{n}+p_{i,j-1}^{n}-4p_{i,j}^{n}\right)-\Delta t\, s_{i,j} \tag{31.62}$$

where $\Delta x=\Delta y=\Delta$. Now in two dimensions, stability limits the maximum allowable time-step to $\frac{\Delta^2}{4}$ and when this time-step is used the above equation becomes

$$p_{i,j}^{n+1}=\frac{1}{4}\left(p_{i+1,j}^{n}+p_{i-1,j}^{n}+p_{i,j+1}^{n}+p_{i,j-1}^{n}\right)-\frac{\Delta^2}{4}s_{i,j} \tag{31.63}$$

This equation can now be viewed as an iterative scheme where the average of the four neighbors plus the contribution from the source term is used to compute a better estimate of the solution. This iteration can be carried out until a "steady state" is reached, i.e., when the difference between p^n and p^{n+1} becomes negligible. At that point, we obtain the solution to the original Poisson equation.

The above iterative method is in fact the Jacobi method, which can be considered the father of all iterative methods. Even though the above method converges too slowly to be practical, it provides the starting point for a number of other powerful iterative methods. It turns out that the operation count or memory storage makes direct inversion of discrete operators impractical for many practical problems and one has then to resort to iterative schemes. Furthermore, operators that result from the discretization of the governing equations are generally sparse, and most of the multiplicative operations in the direct inversion process are wasted on elements which are in fact zero. Finally, even though the storage required for these sparse matrices may be small, the inverse of these matrices are typically full and require significantly more storage. In contrast, iterative techniques do not store the inverse of the operator matrices and access the operator matrix only through it multiplication to a vector, an operation that can be made highly efficient by utilizing the sparsity of the matrix.

A simple twist of the Jacobi method leads to the Gauss–Siedel method, where the latest available estimate of the solution is used in computing the average on the RHS. The iteration equation then becomes

$$p_{i,j}^{n+1}=\frac{1}{4}\left(p_{i+1,j}^{n}+p_{i-1,j}^{n+1}+p_{i,j+1}^{n}+p_{i,j-1}^{n+1}\right)-\frac{\Delta^2}{4}s_{i,j} \tag{31.64}$$

Thus, in the Gauss–Siedel method the latest available estimate is used in forming the average. The Gauss–Siedel method converges twice as fast as the Jacobi method but is still too slow to be practical for large problems. This simple method however forms the backbone of a number of extremely powerful iterative methods. Consider the decomposition of the matrix $[A]$ into a lower part $[L]$, a diagonal part $[D]$, and an upper part $[U]$ such that $[A] = [L] + [D] + [U]$. The Jacobi method can then be written in matrix-vector notation as

$$[D]\{p^{n+1}\} = -[L+U]\{p^n\} + \{s\} \tag{31.65}$$

whereas the Gauss–Siedel method corresponds to

$$[L+D]\{p^{n+1}\} = -[U]\{p^n\} + \{s\} \tag{31.66}$$

Now consider the decomposition of matrix $[A]$ into $[B] + [C]$. A general iterative scheme can then be written as

$$[B]\{\phi^{n+1}\} = -[C]\{\phi^n\} + \{b\} \tag{31.67}$$

which can further be written as

$$\{\phi^{n+1}\} = [Q]\{\phi^n\} + [B]^{-1}\{b\} \tag{31.68}$$

where $[Q] = -[B]^{-1}[C]$. Thus for Jacobi method $[Q] = -[D]^{-1}[L + U]$ and for Gauss–Siedel method $[Q] = -[L + D]^{-1}[U]$. The matrix $[Q]$ is called the *iteration matrix*, and this matrix is the key element of the iterative scheme. It can be shown that for an iterative scheme to converge, all the eigenvalues of this matrix should have a magnitude less than one. The residual which is a one measure of the difference between the iterative solution and the exact solution, is then guaranteed to reduce as the iteration proceeds. In fact, the factor by which the residual is reduced in one iteration is proportional to the largest eigenvalue (the magnitude of which is also called the *spectral radius* of the operator) of the iteration matrix. Therefore, for the iteration to converge reasonably fast, it is crucial that the iteration matrix be designed such that the spectral radius is significantly smaller than one. For a sufficiently large number of grid points, it can be shown that the spectral radius of Gauss–Siedel literation matrix for the 2-D Laplace equation with Dirichlet boundary condition is approximately equal to $(1 - \pi^2/N)$ where N is the total number of mesh points. Thus it is clear that for large N, the Gauss–Siedel method will converge very slowly. If we consider a mesh with $N = 10^5$, then to reduce the residual by about four orders of magnitude, it would require about 10^5 iterations! Fortunately, with just a little bit of effort, it is usually possible to greatly accelerate the convergence of the iteration procedure. Successive overrelaxation (SOR), multigrid [Briggs, 1977; Hackbusch, 1985], and preconditioning [Canuto et al., 1987; Barrette et al., 1994] are some of the important techniques that can be used in conjunction with a basic iterative technique like Gauss–Siedel, and these can result in tremendous acceleration of the iterative procedure. These and other convergence acceleration techniques will be discussed in Chapter 32.

There exists another class of methods called conjugate gradient (CG) methods that can be used for solving the set of equations given by Eq. (31.45). Instead of approaching the solution of Eq. (31.45) directly, these methods attempt to minimize the following function:

$$f(\phi) = \frac{1}{2}(\phi)^T[A]\{\phi\} - \{\phi\}^T\{b\} \tag{31.69}$$

The above function is minimized when

$$\nabla f(\phi) = [A]\{\phi\} - \{b\} = 0 \tag{31.70}$$

and it becomes clear that minimization of the function in Eq. (31.69) is equivalent to solving our linear set of equation. Now suppose that we are trying to solve the minimization problem though a process where we guess the solution vector and then use the resulting residual (in some yet unprescribed manner) to make the next guess for the solution. Let the guess to the solution be denoted by $[\phi_c]$. Then it can be shown that the corresponding residual $[d_c]$ is related to the gradient of the function as

$$\{d_c\} = \{b\} - [A]\{\phi_c\} = -\nabla f(\phi_c) \tag{31.71}$$

Since the function decreases most rapidly in the direction of the negative gradient, it seems natural that the next guess to the solution be chosen as $[\phi_c + ad_c]$ where a can be chosen so as to minimize $f(\phi_c + ad_c)$. This corresponds to the line minimization of the function $f(\phi)$ along the direction $[d_c]$. When this procedure is carried out successively along resultant residual directions, it corresponds to the well known method of steepest descent and can in principle be used to arrive at the solution of the linear system of equations. However, it has been found that the convergence of this method can be extremely poor in some cases [Press et al., 1987]. The problem with this method lies in the fact that line minimization along one residual direction does not preserve the minimization along previous directions, and one might actually undo some of the work done in previous line minimizations.

The idea then is to come up with minimization (or search) directions that do not "interfere" with the previous search directions. Suppose that we have moved the solution in the direction $\{m\}$ to a minimum. Now by definition the gradient of the function at this new location is orthogonal to the direction $\{m\}$. Now instead of moving in this orthogonal direction like the steepest descent method, we move in a direction that will not interfere with our minimization along $\{m\}$. This can be accomplished if we move in a direction such that the change in the gradient stays orthogonal to $\{m\}$. If this new direction is $\{n\}$ then from Eq. (31.71) it is clear that the change in the gradient due to movement in this direction is $[A]\{n\}$, and the condition that this change in gradient stay orthogonal to $\{m\}$ implies that $\{m\}^T[A]\{n\} = 0$. This then provides the recipe for obtaining "noninterfering" or "conjugate" directions. In the actual conjugate gradient algorithm, the new direction is chosen such that it minimizes the function over the current as well *all* the previous directions. That this is possible is in itself quite unexpected! However, the fact that in most cases the conjugate directions are linearly independent leads to the even more remarkable discovery that for a system with N degrees of freedom the *exact* solution can be obtained in N line minimizations. However, this is only true in principle but is never achievable on computers where roundoff error spoil the "exactness" of the search procedure. The conjugate gradient method should thus be regarded as an iterative method. The actual conjugate gradient algorithm will not be presented here, since detailed descriptions can be found in Golub and Van Loan [1993], Press et al. [1987], and Barrette et al. [1994]. It should, however, be pointed out that there exist a number of different variation of conjugate gradient algorithm. In addition to the basic CG method, which is applicable only to symmetric positive definite matrices, the biconjugate gradient (BiCG) method can handle nonsymmetric matrices. However, BiCG method can have irregular convergence, and other methods such as quasi-minimal residual (QMR) and biconjugate gradient stabilized (Bi-CGSTAB) have been devised in order to mitigate this deficiency. Generalized minimal residual (GMRES) is another descent method that can be used for general nonsymmetric matrices [Saad and Schultz, 1986]. Details of the implementation of all these methods can be found in Barrette et al., [1994]. The above methods are commonly referred to as Krylov subspace based methods. Most of these methods become viable only in conjunction with suitable preconditioners, and quite often the performance can depend critically on the particular choice of the preconditioner [Barrette et al., 1994]. For information related to fluid flow computations, see Johnson et al. [1997], Knoll et al. [1996], and McHugh et al. [1996].

Nonlinear Operators

Nonlinear operators are encountered when a fully implicit method is used for the time discretization of the governing equations. More specifically, nonlinearity comes from the implicit treatment of the nonlinear convection terms and any source terms that might have a nonlinear dependence on the flow variables. A simple iterative scheme can be devised where the nonlinear term can be linearized about the latest guess of the velocity field. The resulting linear equation can then be solved using any of the techniques discussed above and a better estimate of the velocity field obtained. This approach is taken in many methods, such as the SIMPLE algorithm [Patankar, 1980], where the nonlinear system that results from the momentum equation is linearized at every iteration level. Faster convergence can be obtained if more sophisticated linearization schemes are used. The Newton–Raphson iterative method for finding roots of a nonlinear system of equations can be used to invert the nonlinear system that results from the fully implicit treatment of the convection–diffusion terms. Consider for example the convection–diffusion equation discretized by a fully implicit backward Euler scheme. The discretized convection diffusion equation can be written as

$$f(u) = u + \Delta t\left(N(u) - \nu L(u)\right) - r = 0 \tag{31.72}$$

where u is the solution at the current step and r accounts for all the explicit terms. Now we can view the time advancement of the above fully implicit system as an exercise in finding the root of $f(u)$. In the Newton's method we begin by expanding the function in a Taylor series as

$$f(u+\delta u) = f(u) + \nabla f\big|_u \cdot \delta u + O(\delta u^2) \tag{31.73}$$

If the $O(\delta u^2)$ is neglected then by equating $f(u + \delta u)$ to zero, we get an equation for the correction δu. The gradient term on the RHS of the above equation is called the *Jacobian matrix* of the system. Given a guess u^k at the k^{th} iteration, the following equation is solved for the correction:

$$\nabla f\big|_k \cdot \delta u = -f(u^k) \tag{31.74}$$

The above equation is linear and is amenable to solution by any of the techniques discussed in the previous subsection. Once the above equation is solved, a new guess for the solution vector is obtained as $u^{k+1} = u^k + \delta u$. As a result of this correction, the vector function $f(u)$ moves closer to zero. This iteration is carried out until convergence is reached and the final solution is obtained.

It can be shown that if the guess is sufficiently close to the correct solution, then the Newton iteration converges quadratically. This method was used for DNS of turbulent flows by Choi et al. [1993], and there it was found to work quite well. However, there are two main problems with this iteration scheme. First, there is no guarantee that the iteration will converge, or converge to the right answer. In fact, solution by this type of iterative technique requires a good initial guess for the scheme to converge to the right answer. Usually, the initial guess is taken to be the velocity at the previous time step, and for sufficiently small time steps, this provides a reasonable initial guess. However, in some configurations where the flow might change rapidly in a localized region, the Newton iteration method might require a small time-step to converge to the right solution. This could possibly nullify the advantage of using an implicit scheme. The second problem with these type of iterative schemes for the momentum equation is that it is usually difficult to decide on an adequate convergence criterion. If the convergence criterion is not strict enough, this can lead to numerical instability, whereas too severe a convergence criterion can make the solution extremely expensive. The convergence criterion is typically decided based on trial and error, but can vary significantly from flow to flow and even for a given flow at different flow parameters.

Fortunately there exists a class of methods for nonlinear equations that are *globally convergent*, i.e., they converge to the right solution from almost any initial guess. These are in fact the conjugate gradient family of methods, which were designed initially for the purpose of minimizing nonlinear functions and were only later used for solving linear systems of equations. The problem with the Newton iterative method is that the correction that is computed from Eq. (31.74) does not necessarily move toward the zero of the function $f(u)$. A good discussion of this can be found in Press et al. [1987]. Globally convergent methods, on the other hand, guarantee that with every correction or iteration, the solution will move in the right direction. To understand this consider for the time being the scalar nonlinear function $f(u)$ where u is a N-component vector. Extension of the technique to a vector function is in principle quite straightforward. If the origin of the coordinate system lies at v, then the scalar function can be approximated at a neighboring point u by a truncated Taylor series expansion about v as

$$f(u+v) = f(v) + \nabla f\big|_v \cdot u + \frac{1}{2} u \cdot \nabla\nabla f\big|_v \cdot u \tag{31.75}$$

where cubic and higher term are neglected. The object here is to deduce a u that will minimize the above function. Consistent with the above approximation, the gradient of the function can then be approximated as $\nabla f(u) = \nabla\nabla f|_v . u + \nabla f|_v$. If Eq.(31.75) is written in matrix-vector notation as

$$f\{u\} = c - \{u\}^T\{r\} + \frac{1}{2}\{u\}^T[A]\{u\} \tag{31.76}$$

where

$$c \equiv f\{v\}; \quad \{r\} \equiv -\nabla f\big|_v \quad \text{and} \quad [A] \equiv \nabla\nabla f\big|_v \tag{31.77}$$

then the approximate gradient can be written as

$$\nabla f = [A]\{u\} - \{r\} \tag{31.78}$$

Since the function will be at an extremum when the gradient vanishes, the minimization problem becomes equivalent to solving $\nabla f = [A]\{u\} - \{r\} = 0$. Thus, the connection between the solution of linear and nonlinear equations through descent methods becomes clear. It should be pointed out that for functions that are not quadratic, more than one pass of N line minimizations will have to be performed. Therefore, the CG method when used for the purpose of solving a set of nonlinear equations is a true iterative method. However, with this method, convergence is guaranteed from any initial guess, although quadratic convergence is obtained only in the neighborhood of the actual minimum.

Other techniques capable of significantly accelerating the convergence of the iterative solver are available, notably the multigrid method [Briggs, 1977; Hackbusch, 1985; Shyy, 1994; Ferziger and Peric, 1996]. Extensive discussions of these techniques are given elsewhere, and will not be presented here.

31.6 Outflow Boundary Conditions

Another issue that needs to be discussed in fluid flow computations is regarding boundary conditions at outflow boundaries. Often we encounter problems in fluid dynamics where the computational domain has to be truncated in order to make the problem tractable. Thus, artificial computational boundaries are created and boundary conditions have to be applied on these boundaries in order to get a closed system of equations. In general, no natural outflow boundary conditions exist, and one has to contend with *devising* appropriate boundary conditions. In most cases, the objectives in devising the outflow

boundary condition are that it should (1) enforce global mass conservation [Shyy, 1994], and (2) allow flow disturbances to propagate out of the computational domain with minimum reflection so that the spurious effect of artificially truncating the boundary is minimized. The first requirement leads to procedures for correcting velocity profiles at the boundary, and the second requirement leads to the so-called *nonreflecting* boundary conditions. The most popular nonreflective boundary is the convective boundary condition, which neglects the diffusive effect near the boundary and assumes the flow to be purely convective in nature. Thus for a 2-D problem if u_n is the velocity component normal to the outflow boundary, then the convective boundary condition that is applied at the boundary is given by

$$\frac{\delta u}{\delta t} + u_n \frac{\delta u}{\delta n} = 0 \tag{31.79}$$

This simple boundary condition works quite well in convecting large disturbances out of the computational domain with minimum reflections and has been used successfully in internal as well as external flows. However, it has been found that when such simple boundary conditions are used in conjunction with higher-order methods such as spectral methods, they can lead to the generation of spurious oscillations [Abarbanel, 1991]. Spectral methods are notoriously sensitive to discontinuities in the flow field, and the abrupt imposition of the convective condition results in spurious oscillations associated with the Gibbs phenomenon. The convective boundary condition can instead by applied smoothly, and this leads to the buffer domain technique [Streett and Macaraeg, 1989; Karniadakis and Triantafyllou, 1992; Mittal and Balachandar, 1996]. In this technique, the momentum equation are smoothly parabolized near the outflow boundary by multiplying the diffusion terms normal to the outflow boundary by a function that is equal to one in most of the domain and smoothly goes to zero near the outflow boundary. Since the equation is parabolized at the outflow boundary, it no longer requires a boundary condition, and the governing equation itself can be solved at the boundary. Any possible feedback due to the elliptic nature of pressure can also be minimized by attentuating the source terms of the pressure Poisson equation in a manner similar to the diffusion term. Details of the implementation of this method for single as well as multidomain methods can be found in Street and Macaraeg [1989], Joslin et al. [1992], Mittal [1995] and Mittal and Balachandar [1996].

3.7 Complex Geometries

In solving fluid flow problems, complex geometries are a rule rather than an exception. The basic problem in complex geometries arises from the requirement of satisfying the boundary conditions (no-slip, no-penetration, no-shear, etc.) at geometrically complex surfaces. The finite-element and other unstructured grid methods are quite well suited for handling flows in complex geometries. However, if one decides to use a structured grid then there are two ways in which this problem can be handled. First is the traditional approach of using a body conformal grid where one discards the simple Cartesian mesh for a curvilinear mesh that follows the shape for the complex surface. An example of a conformal grid over a simple curved body surface is shown in Fig. 31.3. The governing equations to be solved on the curvilinear mesh are usually transformed and discretized on an equispaced Cartesian mesh. For even slightly complicated configurations, nonorthogonal grids have to be used. When the governing equations are transformed from this nonorthogonal curvilinear space to the computational Cartesian space, the curvature and nonuniformity of the grid results in a number of additional terms that increase the complexity and computational expense of inverting the discretized equations. Therefore, this approach simplifies the application of the boundary conditions at the expense of complicating the governing equations.

The second technique is the so-called *immersed boundary* technique, which has seen increased use in recent years. In this technique the governing equations are discretized on a Cartesian mesh which is overlaid on the flow system (see Figure 31.3). Since the nodes of this Cartesian mesh will in general not

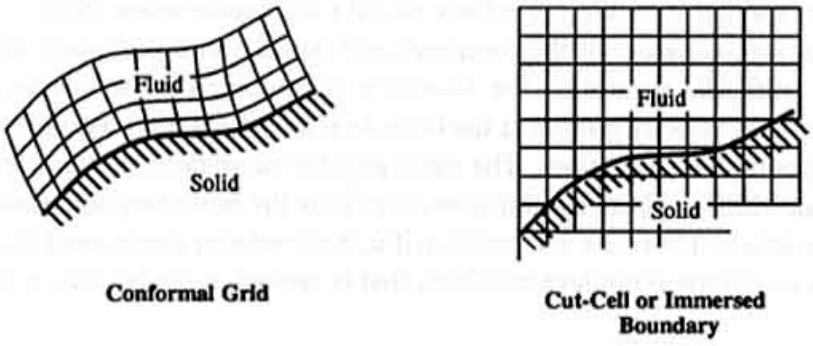

FIGURE 31.3 Different approaches for handling complex geometries.

fall on the body surface, special techniques have to be used to satisfy the boundary conditions [Shyy et al., 1996]. For example, Udaykumar et al. [1996] and Kan et al. [1996] have developed and implemented numerical techniques to track moving and/or irregular geometries on fixed Cartesian grids using a combination of Eulerian and Lagrangian components. The field equation is solved on the fixed mesh while the fluid boundaries cut through the mesh. Communication between the boundary and the underlying grid can be performed in two ways. The solid–solid boundaries are handled by the cut-cell technique using the ELAFINT method [Shyy et al., 1996; Udaykumar et al., 1996]. The two-fluid boundaries are handled using the immersed boundary technique [Peskin, 1977; Kan et al., 1996]. Different variations and applications of some of these immersed boundary approaches can be found in the literature, as reviewed by Shyy et al. [1996]. Other variations of the immersed boundary technique include the momentum forcing techniques of Goldstein et al. [1995] and M-Yusof [1996]. Although the immersed boundary technique is especially suitable for certain type of applications, there still remain a number of unresolved issues regarding the application of these methods to fluid flow problems. Using momentum forcing can result in a severe restriction on the size of the time step [Goldstein et al., 1995] and exact imposition of the boundary conditions can be difficult [M-Yusof, 1996]. In addition to this, providing adequate resolution for resolving thin boundary layers that develop on the body surface can be challenging as well. Here we will concentrate on the conventional conformal grid approach, and readers interested in the immersed body approach are referred to the above mentioned articles.

Governing Equations in Curvilinear Coordinates

The two-dimensional steady-state, incompressible, constant property, Navier–Stokes equations are used.

$$\frac{\partial}{\partial x}(\rho u)+\frac{\partial}{\partial y}(\rho v)=0 \tag{31.80}$$

$$\frac{\partial}{\partial x}(\rho uu)+\frac{\partial}{\partial y}(\rho vu)=-\frac{\partial p}{\partial x}+\frac{\partial}{\partial x}\left(\mu\frac{\partial u}{\partial x}\right)+\frac{\partial}{\partial y}\left(\mu\frac{\partial u}{\partial y}\right) \tag{31.81}$$

$$\frac{\partial}{\partial x}(\rho uv)+\frac{\partial}{\partial y}(\rho vv)=-\frac{\partial p}{\partial v}+\frac{\partial}{\partial y}\left(\mu\frac{\partial v}{\partial x}\right)+\frac{\partial}{\partial y}\left(\mu\frac{\partial v}{\partial y}\right) \tag{31.82}$$

With the introduction of coordinate transformation $\xi = \xi(x, y)$, $\eta = \eta(x, y)$, the equations above are then cast into the curvilinear coordinates [Shyy, 1994].

$$\frac{\partial}{\partial \xi}(\rho U)+\frac{\partial}{\partial \eta}(\rho V)=0 \tag{31.83}$$

$$\frac{\partial}{\partial \xi}(\rho Uu)+\frac{\partial}{\partial \eta}(\rho Vu)=-y_\eta \frac{\partial p}{\partial \xi}+y_\xi \frac{\partial p}{\partial \eta}+\frac{\partial}{\partial \xi}\left[\frac{\mu}{J}\left(q_1 u_\xi - q_2 u_\eta\right)\right]+\frac{\partial}{\partial \eta}\left[\frac{\mu}{J}\left(-q_2 u_\xi + q_3 u_\eta\right)\right] \tag{31.84}$$

$$\frac{\partial}{\partial \xi}(\rho Uv)+\frac{\partial}{\partial \eta}(\rho Vv)=+x_\eta \frac{\partial p}{\partial \xi}-x_\xi \frac{\partial p}{\partial \eta}+\frac{\partial}{\partial \xi}\left[\frac{\mu}{J}\left(q_1 v_\xi - q_2 v_\eta\right)\right]+\frac{\partial}{\partial \eta}\left[\frac{\mu}{J}\left(-q_2 v_\xi + q_3 v_\eta\right)\right] \tag{31.85}$$

where

$$U = uy_\eta - vx_\eta, \qquad V = vx_\xi - uy_\xi \tag{31.86}$$

$$q_1 = x_\eta^2 + y_\eta^2, \qquad q_3 = x_\xi^2 + y_\xi^2 \tag{31.87}$$

$$q_2 = x_\xi x_\eta + y_\xi y_\eta, \qquad J = x_\xi y_\eta - x_\eta y_\xi \tag{31.88}$$

There are several choices of the grid and variable arrangements [Rhie and Chow, 1983; Shyy and Vu, 1991; Ferziger and Peric, 1996], depending on whether the flow variables are arranged in a staggered or collocated manner. There are several issues involved in grid arrangement, including (1) boundary treatment, (2) implicitly or explicitly added numerical smoothing and impact on numerical accuracy, especially under the condition of large pressure and density variations, and (3) programming and data structures. A major issue of the collocated grid system using the non-compact stencil for pressure is the potential of the odd–even decoupling of the velocity and pressure fields [Patankar, 1980]. Recent work [Majumdar, 1988; Melaaen, 1992; Ferziger and Peric, 1996] has shown that with the incorporation of momentum interpolation [Rhie and Chow, 1983], it is possible to prevent the onset of nonphysical oscillations for such schemes. The momentum interpolation in essence introduces higher-order pressure damping terms into the estimation of the mass flux on the face of each velocity control volume [Udaykumar et al., 1997]. For flows exhibiting sharp gradients, the role of these higher order damping terms needs to be further clarified. If a compact stencil is used, then the main issue is the satisfaction of the mass continuity constraint in regions of high pressure gradients.

An important issue discussed previously is the velocity and pressure coupling in the incompressible flow computation. The extra terms arising from the nonorthogonal coordinates make the storage of the direct method several times that with Cartesian coordinates, causing the cost per linearized iteration also to be much higher. This aspect makes the direct method more expensive than the iterative method overall. While the Reynolds number and mesh skewness have noticeable effects on the convergence rate of the iterative method, interestingly, the number of iterations required by the conventional line-iterative solver, without any acceleration techniques, can be *less* sensitive than in Cartesian coordinates. This observation can be attributed to the fact that the curvilinear coordinate can follow the flow characteristics more closely; hence it is capable of improving the computational performance for flows in complex geometries.

Multi-Block Methods

Many physical problems exhibit multiple length and time scales as well as geometric complexities [Shyy et al., 1997]. To handle these characteristics, a multi-block grid method is very useful, because it allows different numbers and density of grid points to be distributed in different regions, without forcing grid lines to be continuous in the entire domain. Multi-block structured grids can be broadly classified as either patched grids [Rai, 1984] or overlapping grids [Steger, 1991]. Patched grids are individual grid

blocks of which any two neighboring blocks are joined together at a common grid line without overlap. With overlapping grids, the grid blocks can be arbitrarily superimposed on each other to cover the domain of interest. Compared to patched grids, overlapping grids are more flexible to generate and can be more easily fitted to complex geometry, but information transfer between blocks is more complicated and flux conservation is more difficult to maintain. For both grid arrangements, the issues of interface treatment regarding both conservation laws and spatial accuracy need to be addressed. For flow problems involving discontinuities or sharp gradients of flow variables, it is well known that it is often advantageous to use a numerical scheme in conservation from. The need for accurate conservative grid interfaces is illustrated by Benek et al. [1983]. Rai [1984; 1985] has developed a conservative interface treatment for patched grids and has conducted calculations demonstrating the shock capturing capability with the discontinuous grid interface. Berger [1987] has given a discussion of conservative interpolation on overlapping grids. Chesshire and Henshaw [1990; 1994] have conducted analyses on grid interface treatment and developed data structures for conservative interpolation on general overlapping grids. They solved slow- and fast-moving shock problems with both conservative and nonconservative interpolation and found that with the appropriate form of artificial viscosity, both conservative and nonconservative interface schemes can give good results.

Part-Enander and Sjogreen [1994] also compared the effects of both conservative and nonconservative interpolation on slowly moving shock problems. They found that the nonconservative interpolation could lead to large error. In their approach, the conservative interpolation alone was not stable, and the conservative flux interpolation with a characteristic decomposition and nonlinear filter were necessary to produce a satisfactory solution. Meakin [1994] investigated the spatial and temporal accuracy of an overlapping method for moving body problems and suggested that the issue with interface treatment was not necessarily one of conservative versus nonconservative, but one of grid resolution. The issue of accurate and conservative treatment of discontinuous grid has also been investigated by Kallinderis [1992]. Related work on conservative interface treatment for overlapping grids can also be found in Moon and Liou [1989] and Shyy et al. [1997]. From the literature cited above, it can be seen that several factors can affect the solution quality for flow problems involving sharp flow gradient and discontinuous grid interfaces, e.g., the order of interpolation accuracy, conservative or nonconservative interpolation, convection schemes (or forms of artificial viscosity), and grid resolution. Which factor (or combination of several different factors) has the most critical effect is still not clear. Ideally, the physical conservation laws should be satisfied in the entire domain, but when discontinuous grids are encountered, to what extent should flux be conserved is still an open question. For some problems, compromise has to be made between maintaining flux conservation and interpolation accuracy consistently in both grid interface and interior regions.

As discussed previously, in solving incompressible flow problems, because of the decoupling of thermodynamic pressure and velocity fields, the way to maintain mass conservation is crucial for designing an effective solution algorithm. For multi-block computations of incompressible flow, the maintenance of mass conservation across discontinuous grid interfaces is also an important issue. Some efforts have been made with different interface treatments for mass flux. For example, two-dimensional incompressible flow problems have been computed by employing patched grids [Perng and Street, 1991; Lai et al., 1993; Thakur et al., 1996]. In these methods, the grid interface arrangements are either continuous or consist of one coarse grid including integer number of fine grids so that the maintenance of mass conservation across the grid interface is relatively easy. Several authors [Meakin and Street, 1988; Hinatsu and Ferziger, 1991; Strikwerda and Scarbnick, 1993; Henshaw, 1994] solved the flow problems using overlapping grids and just employed direct interpolation of the dependent variables across the grid interface. The extent of satisfaction of mass conservation there depends solely on the order of the interpolation methods used. Yung et al. [1989] and Tu and Fuchs [1993] have used different mass correction methods to achieve global mass conservation across the grid interface, but the effects of those methods on solution accuracy have not been assessed. Wright and Shyy [1993] have developed a pressure-based multi-block method based on the discontinuous overlapping grid technique for solving the incompressible Navier–Stokes equations in Cartesian grid systems. A locally conservative interface scheme. with

first-order accuracy, is devised to ensure that local and global conservation of mass and momentum fluxes are maintained. For mass flux treatment, an interface scheme with global mass conservation is often not sufficient to yield accurate solution [Shyy et al., 1997]. Furthermore, for incompressible flow computations, the pressure is generally known up to an arbitrary constant. When the flow solver is applied to multi-block grids, how to couple pressure fields in different blocks and at the same time maintain mass flux conservation across interface is an important issue.

31.8 Closing

Much progress has been made in Navier–Stokes flow computations. We can now almost routinely compute flows up to moderately high Reynolds numbers in simple geometries with desirable accuracy. However, we are still far away from able to solve practical problems arising from manmade or natural environments based on rigorously derived governing laws. Geometric complexities and scaling disparities in time and in space are the major reasons, as discussed in Shyy et al. [1997]. To resolve these complexities, more computing power and algorithmic capabilities will be needed. Great strides have been made on both fronts. However, hardware improvements oftentimes challenge existing software practices, as clearly illustrated in the area of parallel computing. An integrated approach to both hardware and software is essential. Even with all these current and potential improvements, we most likely will still need to rely on imperfect physical models to close the gap between physical reality and computational capability. Hence, for practical applications, physical modeling in the context of large scale computation, i.e., computational modeling, will remain to be a most important pacing item in computational fluid dynamics.

References

Abarbanel, S. S. Don, W. S., Gottlieb, D., Rudy, D. H., and Townsend, J. C., Secondary frequencies in the wake of circular cylinder with vortex shedding, *J. Fluid Mech.*, 225, 557–570, 1991.

Ball, J. B., Collela, P., and Glaz, H. M., A second-order projection method for the incompressible Navier–Stokes equations, *J. Comput. Phys.*, 85, 257–283, 1989.

Barrette, R., Berry, M., Chan, T., Demmel, J., Donato, J., Dongarra, J., Eijkhout, V., Pozo, R., Romine, C., and Vorst, H., *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods,* SIAM, 1994.

Benek, J. A., Sterger, J. J., and Dougherty, F. C., A flexible grid embedding technique with application to the Euler equations, *AIAA–83–1944–CP,* 1983.

Boris, J. P. and Book, D. L., Flux corrected transport I, Shasta: An algorithm that works, *J. Comput. Phys.*, 11, 38–69, 1973.

Braaten, M. E., Development and evaluation of iterative and direct methods for the solution of the equations governing recirculating flows, Ph.D. thesis, University of Minnesota, Minneapolis, 1985.

Braaten, M. E. and Shyy, W., Comparison of iterative and direct method for viscous flow calculations in body-fitted coordinates, *Int. J. Numer. Meths. Fluids,* 6, 325–349, 1986.

Braaten, M. E. and Shyy, W., A study of pressure correction methods with multigrid for vicous flow calculations in non-orthogonal curvilinear coordinates, *Numer. Heat Transf.*, 11, 417–442, 1987.

Briggs, W., *A Multigrid Tutorial,* SIAM, Philadelphia, 1977.

Canuto, C., Hussaini, M. Y., Quarteroni, A., and Zang, T. A., *Spectral Methods in Fluid Dynamics,* Springer-Verlag, New York, 1987.

Chesshire, G. and Henshaw, W. D., Composite overlapping meshes for the solution of partial differential equations, *J. Comput. Phys.*, 90, 1–64, 1990.

Chesshire, G. and Henshaw, W. D., A scheme for conservative interpolation on overlapping grids, SIAM *J. Sci. Comput.*, 15(4), 819–845, 1994.

Choi, H., Moin, P., and Kim, J., Direct numerical simulation of flow over riblets, *J. Fluid Mech.*, 255, 503–539, 1993.

Choi, H. and Moin, P., Effect of computational time-step on numerical solutions of turbulent flow, *J. Comput. Phys.*, 113, 1–4, 1994.

Chorin, A. J., A numerical method for solving incompressible viscous flow problems, *J. Comput. Phys.*, 2, 12–26, 1967.

Chorin, A. J., Numerical solution of the Navier–Stokes equations, *Math. Comput.*, 22, 745–762, 1968.

Cuvelier, C., Segal, A., and Van Steenhoven, A. A., *Finite Element Methods and Navier–Stokes Equations*, Springer-Verlag, New York, 1986.

Deville, M., Klieser, L., and Montigny-Rannou, F., Pressure and time treatment for Chebyshev spectral solution of a Stokes problem, *Int. J. Numer. Meth. Fluids*, 4, 1149–1163, 1984.

Duckowicz, J. K. and Dvinsky, A. S., Approximate factorization as a high order splitting for implicit incompressible flow equations, *J. Comput. Phys.*, 102, 336–347, 1992.

Fatica, M. and Mittal, R., Progress in large-eddy simulation of an asymmetric plane diffuser, *CTR Annual Research Briefs*, Stanford University/NASA Ames, 1996.

Ferziger, J. H. and Peric, M., *Computational Methods for Fluid Dynamics*, Springer-Verlag, New York, 1996.

Goldstein, D., Handler, R., and Sirovich, L., Direct numerical simulation of turbulent flow over a modelled riblet covered surface, *J. Fluid Mech.*, 302, 333–376, 1995.

Golub, G. H. and Van Loan, C. F., *Matrix Computations*, Johns Hopkins University Press, Baltimore, MD, 1993.

Hackbusch, W., *Multi-Grid Methods and Applications*, Springer-Verlag, 1985.

Haidvogel, D. B. and Zang, T. A., The accurate solution of Poisson's equation by expansion in Chebyshev polynomials, *J. Comput. Phys.*, 30, 167–180, 1979.

Harlow and Welch, 1965.

Harten, A., High resolution schemes for hyperbolic conservation laws, *J. Comput. Phys.*, 49, 357–393, 1983.

Henshaw, W. D., A fourth-order accurate method for the incompressible Navier–Stokes equations on overlapping grids, *J. Comput. Phys.*, 113, 13–25, 1994.

Hinatsu, M. and Ferziger, J. H., Numerical computation of unsteady incompressible flow in complex geometry using a composite multigrid technique, *Int. J. Numer. Meths. Fluids*, 13, 971–997, 1991.

Hirsch, C., *Numerical Computation of Internal and External Flows*, Wiley, New York, 1990.

Horiuti, K., Comparison of conservative and rotational forms in large eddy simulation of turbulent channel flow, *J. Comput. Phys.*, 71, 343–370, 1987.

Hosangadi, A., Merkle, C. L., and Turns, A. R., Analysis of forced combusting jets, *AIAA J.*, 28, 1473–1480, 1990.

Hughes, T. J. R., Liu, W. K., and Brooks, A., Finite element analysis of incompressible viscous flow by the penalty formulation, *J. Comput. Phys.*, 30, 1–60, 1979.

Issa, R. I., Solution of the implicitly discretized fluid flow equations by operator splitting, *J. Comput. Phys.*, 62, 40–65, 1986.

Issa, R. I., Gosman, A. D., and Watkins, A. P., The computation of compressible and incompressible recirculating flows by a non-iterative scheme, *J. Comput. Phys.*, 62, 66–82, 1986.

Jameson, A., Schmidt, W., and Turkel, E., Numerical simulation of the Euler equations by finite volume methods using Runge-Kutta time stepping schemes, *AIAAA Paper No. 81–1259*, 1981.

Johnson, R. W., McHugh, P. R., and Knoll, D. A., High-order scheme implementation using Newton–Krylov solution methods, *Numer. Heat Transf.*, 31, 295-312, 1997.

Joslin, R. D., Streett, C. L., and Chanag, C-L., A comparison with linear stability and parabolic stability equation theories for boundary-layer transition on a flat plat, *NASA TP-3205*, 1992.

Kallinderis, Y., Numerical treatment of grid interfaces for viscous flows, *J. Comput. Phys.*, 98, 129–144, 1992.

Kan, H.-C., Udakumar, H. S., Shyy, W., and Tran-Son-Tay, R., Simulation of multiphase dynamics by a mixed Eulerian-Langrangian approach, *ASME Paper 96-WA/HT-32*, 1996.

Karniadakis, G. E., Israeli, M., and Orszag, S. A., High-order splitting methods for the incompressible Navier–Stokes equations, *J. Comput. Phys.*, 97, 414–443, 1991.

Karniadakis, G. E. and Triantafyllou, G. E., Three-dimensional dynamics and transition to turbulence in the wake of bluff objects, *J. Fluid Mech.*, 238, 1–30, 1992.

Kim, J. and Moin, P., Application of fractional-step method to incompressible Navier–Stokes equations, *J. Comput. Phys.*, 59, 308, 323, 1985.

Kiris, C., Rogers, S., Kwak, D., and Chang, I.-D., Computation of incompressible viscous flows through artificial heart devices with moving boundaries, in *Fluid Dynamics in Biology*, A. Y. Cheer and C. P. van Dam, Eds., American Mathematical Society, Providence, RI, 237–259, 1993.

Klieser, L. and Schumann, U., Treatment of incompressibility and boundary conditions in 3-D numerical spectral simulations of plane channel flows, *Proceedings of 3rd GAMM Conference on Numerical Methods in Fluid Mechanics*, Hirschel, E. H., Ed., 165–170, 1980.

Knoll, D. A., McHugh, P. R., and Keyes, D. E., Newton–Krylov methods for Row-Mach-number compressible combustion, *AIAA J.*, 34, 961-967, 1996.

Kravchenko, A. G. and Moin, P., On the effect of numerical errors in large eddy simulation of turbulent flows to appear in *J. Comput. Phys.*, 131, 1997.

Ku, H. C., Taylor, T. D., and Hirsh, R. S., Pseudospectral method for solution of the incompressible Navier–Stokes equations, *Comput. Fluids*, 15(2), 195–214, 1987.

Kwak, D., Chang, J. L. C., Shanks, S. P., and Chakravarthy, S. R., A three-dimensional incompressible Navier–Stokes flow solver using primitive variables, *AIAA J.*, 24, 390–396, 1986.

Lai, Y. G., Jiang, Y., and Przekwas, A. J., An implicit multi-domain approach for the solution of Navier–Stokes equations in body-fitted-coordinate grids, *AIAA-93-0541*, 1993.

Leonard, B. P., A stable and accurate convective modeling procedure based on quadratic upstream interpolation, *Comp. Meths Appl. Mech. Eng.*, 19, 59–98, 1979.

MacCormack, R. W. and Baldwin, B. S., A numerical method for solving the Navier–Stokes equations with application to shock-boundary layer interactions, *AIAA Paper No. 75–1*, 1975.

Madabhushi, R. K., Balachandar, S., and Vanka, S. P., A divergence-free Chebyshev collocation procedure for incompressible flows with two non-periodic directions, *J. Comput. Phys.*, 105, 199–207, 1993.

Majumdar, S., Role of underrelaxation in momentum interpolation for calculation of flow with nonstaggered grids, *Numer. Heat Transf.*, 13, 125–132, 1988.

Malik, M. R., Zang, T. A., and Hussaini, M. Y., A spectral collocation method for the Navier–Stokes equations, *J. Comput. Phys.*, 61(1), 64–68, 1985.

Marcus, P. S., Simulation of Taylor-Couette flow. Part I: Numerical methods and comparison with experiments, *J. Fluid Mech.*, 146, 45–64, 1984.

McHugh, P. R., Knoll, D. A., and Keyes, D. E., Schwarz-preconditioned Newton–Krylov algorithm for low speed combustion problems, *AIAA* 96-0911, 1996.

Meakin, R. L., On the spatial and temporal accuracy of overset grid methods for moving body problems, *AIAA* 94-1925, 1994.

Meakin, R. L. and Street, R. L., Simulation of environmental flow problems in geometrically complex domain, Part 2: A domain-splitting method, *Comp. Meth. Appl. Mech. Eng.*, 68, 311–331, 1988.

Melaaen, M. C., Calculation of fluid flows with staggered and nonstaggered curvilinear nonorthogonal grids — The theory, *Numer. Heat Transf.*, 21, 1–19, 1992.

Merkle, C. L., Venkateswaran, S., and Buelow, P. E. O., The relationship between pressure-based and density-based algorithms, *AIAA Paper No. 92–0425*, 1992.

Mittal, R., Study of flow past circular and elliptic cylinders using a direct numerical simulation, Ph.D. thesis, Univ. of Illinois at Urbana-Champaign, 1995.

Mittal, R., Progress in large-eddy simulation of flow past a circular cylinder, *CTR Annual Research Briefs*, Stanford University/NASA Ames, 1996.

Mittal, R. and Balachandar, S., Effect of three-dimensionality on the lift and drag of nominally two-dimensional cylinders, *Phys. Fluids*, 7, 1841–1865, 1995.

Mittal, R. and Balachandar, S., Direct numerical simulation of flow past elliptic cylinders, *J. Comput. Phys.*, 124, 351–367, 1996.

Moin, P. and Kim, J., On the numerical solution of time-dependent viscous incompressible fluid flows involving solid boundaries, *J. Comput. Phys.*, 35, 381–392, 1980.

Moon, Y. and Liou, M.-S., Conservative treatment of boundary interfaces for overlaid grids and multi-level grid adaptations, *AIAA-89-1980*, 1989.

Morinishi, Y., Conservative properties of finite difference schemes for incompresible flow, *CTR Annual Research Briefs*, Stanford University/NASA Ames, 121–132, 1995.

M-Yusof, J., Interaction of massive particles with turbulence, Ph.D. thesis, Cornell University, Ithica, NY, 1996.

Orszag, S. A. and Kells, L. C., Transition to turbulence in plane Poiseuille and plane couette flow, *J. Fluid Mech.*, 96, Part 1, 159–205, 1980.

Orszag, S. A., Israeli, M., and Deville, M.O., Boundary conditions for incompressible flow, *J. Sci. Comput.*, 1(1), 75–111, 1986.

Osher, S., Riemann solvers, the entropy condition and difference approximations, *SIAM J. Num. Anal.*, 21, 217–235, 1984.

Part-Enander, E. and Sjogreen, B., Conservative and nonconservative interpolation between overlapping grids for finite volume solutions of hyperbolic problems, *Comp. Fluids*, 23, 551–574, 1994.

Patankar, S. V., *Numerical Heat Transfer and Fluid Flow*, Hemisphere, Washington, D.C., 1980.

Perng, C. Y. and Street, R. L., A coupled multigrid-domain-splitting technique for simulating incompressible flow in geometrically complex domains, *Int. J. Numer. Meths. Fluids*, 13, 269–286, 1991.

Perot, J. B., An anlaysis of the fractional step method, *J. Comput. Phys.*, 108, 51–58, 1993.

Peskin, C. S., Numerical analysis of blood flow in the heart, *J. Comput. Phys.*, 25, 220–252, 1977.

Peyret R. and Taylor, T. D., *Computational Methods for Fluid Flow*, Springer, New York, 1983.

Press, W. H., Flannery, B. P., Teukolsky, S. A., and Vettering, W. T., *Numerical Recipes*, Cambridge Press, Cambridge, England, 1987.

Rai, M. M., A conservative treatment of zonal boundary for Euler equation calculations, *AIAA-84-0164*, 1984.

Rai, M. M., A conservative treatment of zonal boundaries for Euler equation calculations, *J. Comput. Phys.*, 62, 472-503, 1985.

Raithby, G. D. and Schneider, G. E., Numerical solution of problems in incompressible fluid flow: Treatment of velocity-pressure coupling, *Numer. Heat Transf.*, 2, 417–440, 1979.

Ramshaw, J. D. and Mousseau, V. A., Damped artificial compressibility method for steady-state low-speed flow calculations, *Comp. Fluids*, 20, 177–186, 1991.

Rhie, C. M., A pressure based Navier–Stokes solver using the multigrid method, *AIAA Paper No. 86-0207*, 1986.

Rhie, C. M. and Chow, W. L., Numerical study of the turbulent flow past an airfoil with trailing edge separation, *AIAA J.*, 21, 1525–1532.

Roe, P. L., Approximate Riemann solvers, parameter vectors and difference schemes, *J. Comput. Phys.*, 43, 357–372, 1981.

Saad, Y. and Schultz, M. H., GMRES: A generalized residual algorithm for solving non-symmetric linear systems, SIAM *J. Sci. Stat. Comput.*, 7, 856–869, 1986.

Shuen, J.-S., Chen, K.-H., and Choi, Y., A coupled implicit method for chemical non-equilibrium flows at all speeds, *J. Comput. Phys.*, 106, 306–318, 1993.

Shyy, W., A note on assessing finite difference procedures for large Peclet/Reynolds number flow calculation, in *Boundary and Interior Layers: Computational and Asymptotic Methods*, Vol. 3, J. J. H. Miller, Ed., Boole Press, Dublin, Ireland, 1984, 303–308.

Shyy, W., *Computational Modeling for Fluid Flow and Interfacial Transport*, Elsevier, Amsterdam, Netherlands, 1994.

Shyy, W. and Vu, T. C., On the adoption of velocity variable and grid system for fluid flow computation in curvilinear coordinates, *J. Comput. Phys.*, 92, 82–105, 1991.

Shyy, W., Tong, S. S., and Correa, S. M., Numerical recirculating flow calculation using a body-fitted coordinate system, *Numer. Heat Transf.*, 8, 99–113, 1985.

Shyy, W., Chen, M.-H., Mittal, R., and Udaykumar, H. S., On the suppression of numerical oscillations using a non-linear filter, *J. Comput. Phys.*, 102(1), 49–62, 1992.

Shyy, W., Udaykumar, H. S., Rao, M. M., and Smith, R. W., *Computational Fluid Dynamics with Moving Boundaries*, Taylor & Francis, Washington, DC, 1996.

Shyy, W., Thakur, S. S., Ouyang, H., Liu, J., and Blosch, E., *Computational Techniques for Complex Transport Phenomena*, Cambridge University Press, Cambridge, UK, 1997.

Steger, J. L., Thoughts on the chimera method of simulation of three-dimensional viscous flow, in *Proceedings, Computational Fluid Dynamics Symposium on Aeropropulsion*, Cleveland, OH, *NASA CP-3078*, 1–10, 1991.

Steger, J. L. and 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.

Streett, C. L. and Macareg, M., Spectral multi-domain for large scale fluid dynamic similations, *Appl. Numer. Math.*, 6, 123–139, 1989.

Streett, C. L. and Hussaini, M., A numerical simulation of the appearance of chaos in finite-length Taylor-Couette flow, *Appl. Numer. Math.*, 7, 41–71, 1991.

Strikwerda, J. C. and Scarbnick, C. D., A domain decomposition method for incompresisble viscous flow, *SIAM J. Sci. Comput.*, 14(1), 49–67, 1993.

Teman, R., *Navier–Stokes Equations*, North-Holland, Amsterdam, Netherlands, 1978.

Thakur, S. S. and Shyy, W., Development of high accuracy convection schemes for sequential solvers, *Numer. Heat Transf.*, Part B, 23, 175-199, 1993.

Thakur, S., Wright, J., Shyy, W., Liu, J., Ouyang, H., and Vu, T., Development of pressure-based composite multigrid method for complex fluid flows, *Progr. Aero. Sci.*, 32, 313–375, 1996.

Thomas, P. D. and Lombard, C. K., Geormetric conservation laws and its application to flow computations on moving grids, *AIAA J.*, 17, 339–351, 1979.

Tu, J. Y. and Fuch, L., Overlapping grids and multigrid methods for three-dimensional unsteady flow calculations in IC engines, *Int. J. Numer. Meth. Fluids*, 15, 693–714, 1992.

Udakumar, H. S., Shyy, W., and Rao, M. M., ELAFINT — A mixed Eulerian-Lagrangian method for fluid flows with complex and moving boundaries, *Int. J. Numer. Meth. Fluids*, 22, 691–704, 1996.

Udaykumar, H. S., Kan, H.-C., Shyy, W., and Tran-Son-Tay, R., Multiphase dynamics in arbitrary geometries on fixed Cartesian grids, *J. Comput. Phys.*, 137, 366-405, 1997.

Van Doormaal, J. P. and Raithby, G. D., Enhancement of the SIMPLE method for predicting incompressible fluid flows, *Numer. Heat Transfer.*, 7, 147–163, 1984.

Van Kan, J., A second-order accurate pressure-correction scheme for viscous incompressible flow, SIAM *J. Sci Stat. Comp.*, 7(3), 870–891, 1986.

Wang, J. C. and Widhopf, G. F., A high-resolution TVD finite volume scheme for the Euler equations in conservation form, *J. Comp. Phy.*, 84, 145–173, 1987.

Wright, J. A. and Shyy, W., A pressure-based composite grid method for the Navier–Stokes equations, *J. Comput. Phys.*, 107(2), 225–238, 1993.

Yanenko, N. N., *The Method of Fractional Steps for Solving Multi Dimensional Problems of Mathematical Physics in Several Variables*, Springer-Verlag, Berlin, 1971.

Yung, C., Keith, J. R. T. G., and De Witt, K. J., Numerical simulation of axisymmetric turbulent flow in combustors and diffusers, *Int. J. Numer. Meth. Fluids*, 9, 167–183, 1989.

Zang, T. A., On the rotation and skew-symmetric forms for incompresible flow simulations, *Appl. Numer. Math.*, 7, 27–40, 1991.