directly available, but must be derived from the equation of motion subject to specified boundary conditions for the velocity or traction. We will see that the accurate implementation of derived pressure boundary conditions requires special attention.

We will begin this chapter by discussing a class of methods for computing the structure of a steady flow and the evolution of an unsteady flow based on the vorticity transport equation. The numerical procedure involves computing the evolution of the vorticity field, while simultaneously obtaining the simultaneous evolution of the velocity field by inverting the definition of the vorticity, $\boldsymbol{\omega} = \nabla \times \mathbf{u}$, subject to the continuity equation. One advantage of this approach is that the pressure field does not need to considered. One disadvantage is the need to derive boundary conditions for the vorticity.

Methods based on the vorticity transport equation can be regarded as generalizations of the vortex methods for inviscid or weakly viscous flows discussed in Chapter 11. What distinguishes vortex methods from other methods based on vorticity transport is that the velocity field is obtained from the vorticity field efficiently using the Biot–Savart integral or a related contour integral. In the case of viscous flow, because the support of the vorticity is not necessarily compact, and it is more expedient to recover the velocity from the vorticity field by finite-difference or other domain discretization methods.

A variety of finite-difference procedures are available for solving the equations of steady and unsteady incompressible Newtonian flow, and a choice must be made according to the tolerated level of programming complexity and available computational resources. In this chapter, we outline the fundamental principles of several alternative formulations and discuss the basic steps involved their numerical implementation. Extensions and discussion of particular issues and specialized methods can be found in the references cited as well as in general reviews and monographs on finite-difference methods in fluid dynamics dynamics (e.g., [9, 69, 155, 185, 286, 294, 346]). Numerical methods for free-surface and interfacial flow are reviewed in Reference [131].

13.1 Vorticity–stream function formulation for two-dimensional flow

We begin the discussion of finite-difference methods by presenting a classical formulation based on the vorticity transport equation for two-dimensional flow, known as the stream function–vorticity formulation. In Section 13.3, we address the more general case of three-dimensional flow.

13.1.1 Governing equations

In the case of two-dimensional flow, solving for the velocity in terms of the vorticity is done with the least amount of computational effort by introducing the stream function, ψ. The two components of the velocity in the x and y directions are $u_x = \partial\psi/\partial y$ and $u_y = -\partial\psi/\partial x$, and the z component of the vorticity is

$$\omega_z = -\nabla^2\psi, \tag{13.1.1}$$

where ∇^2 is the Laplacian operator in the xy plane. The computation proceeds according to the two fundamental steps of the vortex methods discussed in Chapter 11.

In the first step, the rate of change of the vorticity is computed using the simplified vorticity transport equation for two-dimensional flow written in the vorticity–stream function form

$$\frac{\partial \omega_z}{\partial t} + \frac{\partial \psi}{\partial y}\frac{\partial \omega_z}{\partial x} - \frac{\partial \psi}{\partial x}\frac{\partial \omega_z}{\partial y} = \nu \nabla^2 \omega_z, \tag{13.1.2}$$

where ν is the kinematic viscosity of the fluid. The sum of the second and third terms on the left-hand side of (13.1.2) is sometimes designated as the Jacobian, $\mathcal{J}(\omega_z, \psi)$. In the second step, the simultaneous evolution of the stream function is obtained by solving the Poisson equation (13.1.1) for ψ in terms of ω_z. Boundary conditions are requiring in both the integration of (13.1.2) and the inversion of (13.1.1). It is instructive to note that the absence of an explicit evolution equation for the pressure in the original system of governing equations is reflected in the absence of an explicit evolution equation for the stream function.

Pressure field

One important feature of the vorticity–stream function formulation is that computing the pressure is not required. If the instantaneous pressure field is desired, it can be computed *a posteriori* by solving a Poisson equation that emerges by taking the divergence of the Navier–Stokes equation and using the continuity equation to obtain

$$\nabla^2 p = 2\rho \left[\frac{\partial^2 \psi}{\partial x^2}\frac{\partial^2 \psi}{\partial y^2} - \left(\frac{\partial^2 \psi}{\partial x \partial y}\right)^2 \right]. \tag{13.1.3}$$

Boundary conditions for the pressure arise by applying the equation of motion at the boundaries, enforcing the specified boundary conditions, and then projecting the result onto the normal or tangential unit vector, as discussed in Section 13.3 in the context of the velocity–pressure formulation.

13.1.2 Flow in a rectangular cavity

To illustrate the implementation of the finite-difference method, we consider the classical problem of flow in a rectangular cavity driven by a lid that translates parallel to itself with a generally time-dependent velocity, $V(t)$, as illustrated in Figure 13.1.1. The no-penetration condition requires that the normal velocity component is zero at each wall. In terms of the stream function,

$$\psi = c \quad \text{over all walls,} \tag{13.1.4}$$

where c is an arbitrary constant set for simplicity to zero. The no-slip boundary condition requires that the tangential component of the velocity is zero over the bottom, left, and right walls, whereas the tangential velocity at the upper wall is equal to the wall velocity, $V(t)$. In terms of the stream function, we obtain the equivalent statement

$$\frac{\partial \psi}{\partial y} = 0 \quad \text{at the bottom,} \qquad \frac{\partial \psi}{\partial x} = 0 \quad \text{at the sides,} \qquad \frac{\partial \psi}{\partial y} = V \quad \text{at the lid.} \tag{13.1.5}$$

Given these boundary conditions for the velocity, we derive simplified expressions for the boundary values of the vorticity in terms of the stream function. Beginning with (13.1.1) and noting that, for

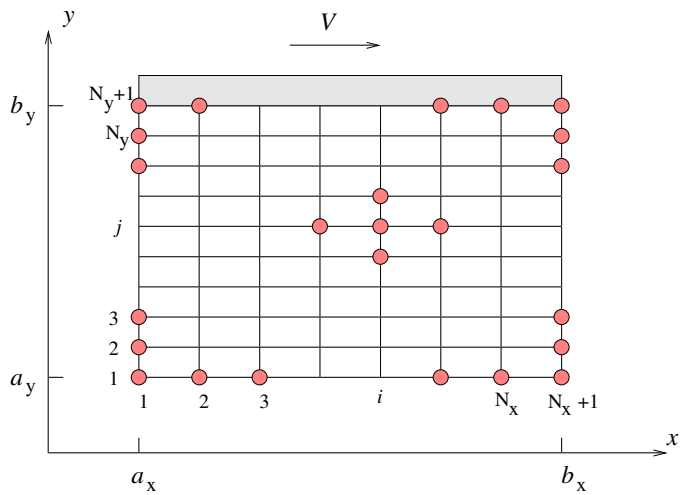

FIGURE 13.1.1 A finite-difference grid for computing two-dimensional flow in a rectangular cavity driven by a moving lid. The pressure and the two components of the velocity are defined at the same grid nodes.

example, $\partial^2\psi/\partial x^2 = -\partial u_y/\partial x = 0$ at the top wall because of the no-penetration condition, we find that

$$\omega_z = -\frac{\partial^2\psi}{\partial y^2} \text{ at the top and bottom walls,} \qquad \omega_z = -\frac{\partial^2\psi}{\partial x^2} \text{ at the side walls.} \tag{13.1.6}$$

To implement a finite-difference method, we introduce a two-dimensional grid with $N_x \times N_y$ divisions, as illustrated in Figure 13.1.1. For simplicity, we have assumed that the grid lines are evenly spaced, which means that the grid spacings, Δx and Δy, are uniform throughout the domain of flow. The finite-difference formulation involves assigning discrete values to the stream function and vorticity at all internal and boundary grid points, and replacing the governing differential equations (13.1.1) and (13.1.2) with difference equations, as discussed in Chapter 12. The specific strategy of computation depends on whether we wish to compute a steady or an unsteady flow.

Steady flow

Two distinct but somewhat related approaches are available for computing a steady flow. The first class of methods involves solving the equations of steady flow using an iterative scheme. The second class of methods involves computing the solution of a fictitious transient flow problem governed by a modified set of differential equations from a given initial condition up to the steady state. The solution of the modified problem at steady state satisfies the equations of steady two-dimensional incompressible Newtonian flow.

13.1.3 Direct computation of a steady flow

In one version of the direct approach, the governing equations (13.1.1) and (13.1.2) are regarded as a coupled nonlinear system of Poisson equations for ψ and ω_z,

$$\nabla^2 \psi = -\omega_z, \tag{13.1.7}$$

and

$$\nabla^2 \omega_z = \frac{1}{\nu} \left(u_x \frac{\partial \omega_z}{\partial x} + u_y \frac{\partial \omega_z}{\partial y} \right). \tag{13.1.8}$$

In the special case of Stokes flow, the right-hand side of (13.1.8) vanishes, yielding Laplace's equation for the vorticity, $\nabla^2 \omega_z = 0$. Equation (13.1.7) then shows that the stream function satisfies an inhomogeneous biharmonic equation, $\nabla^4 \psi = -\omega_z$. The computational algorithm in the general case of nonzero Reynolds-number flow involves the following steps:

Step 1: *Guess the vorticity distribution.*

Step 2: *Solve the Poisson equation (13.1.7) for the stream function.*

For boundary conditions, we have the choice between the Dirichlet boundary condition that specifies the boundary distribution of the stream function, and the Neumann boundary condition that specifies the boundary distribution of the normal derivative of the stream function, which is equal to the tangential component of the velocity. A combination of the Dirichlet and Neumann boundary conditions at different boundaries can also be employed. If we use the Neumann condition over all boundaries, the Poisson equation will have a solution only if the following compatibility condition is fulfilled,

$$\oint_{Walls} \mathbf{n} \cdot \nabla \psi \, \mathrm{d}l = \iint_{Flow} \omega_z \, \mathrm{d}x \, \mathrm{d}y, \tag{13.1.9}$$

where $\mathbf{n}$ is the normal unit vector pointing into the flow. Even when (13.1.9) is fulfilled by a fortuitous guess of the vorticity distribution in Step 1, the singular nature of the linear system of equations that arises from the finite-difference discretization of (13.1.7) introduces additional complications. For these reasons, the Dirichlet condition expressed by (13.1.4) is preferred around all boundaries.

Step 3: *Compute the right-hand side of (13.1.8) and the boundary values of the vorticity using the specified boundary conditions for the velocity.*

Step 4: *Solve the Poisson equation (13.1.8) for the vorticity.*

Step 5: *Check whether the computed vorticity field agrees with that guessed in Step 1 at all grid points. If it does not agree within a specified tolerance, replace the guessed with the computed vorticity and return to Step 2.*

Implementation

The details of the numerical implementation will be discussed with reference to flow in a cavity driven by a moving lid, as illustrated in Figure 13.1.1.

Step 1: *Assign initial values for the stream function to all internal and boundary* $(N_x+1)\times(N_y+1)$ *grid points.*

A simple choice consistent with the no-penetration boundary condition is to set the initial stream function to zero, corresponding to a quiescent fluid.

Step 2: *Assign values for the vorticity to all internal* $N_x \times N_y$ *grid points.*

A simple choice is to set all initial vorticity gird values to zero, corresponding to a quiescent fluid. Note that this choice disregards the motion of the fluid due to the translation of the upper wall.

Step 3: *Solve the Poisson equation (13.1.7) for* ψ *subject to the Dirichlet boundary condition (13.1.4) at all four walls.*

Since the vorticity on the right-hand side is only an approximation to the exact solution, high accuracy is not necessary. To reduce the computational effort, we solve the Poisson equation iteratively and carry out only a small number of iterations. To perform the iterations, we introduce a fictitious unsteady diffusion–reaction problem governed by the equation

$$\frac{\partial \psi}{\partial t} = \kappa\,(\nabla^2\psi + \omega_z), \tag{13.1.10}$$

where κ is a diffusivity. The solution of this equation at steady state satisfies (13.1.7). For simplicity, we denote $\omega = \omega_z$. Implementing the forward time centered space (FTCS) discretization discussed in Section 12.3.7, we find that

$$\psi_{ij}^{(l+1)} = \psi_{ij}^{(l)} + \alpha_x\, R_{ij}^{(l)}, \tag{13.1.11}$$

where $\alpha_x = \kappa\Delta t/\Delta x^2$ is the diffusion number in the x direction, the superscript (l) denotes the lth iteration level,

$$R_{i,j} \equiv \psi_{i+1,j} - 2\,(1+\beta)\psi_{i,j} + \psi_{i-1,j} + \beta\psi_{i,j+1} + \beta\psi_{i,j-1} + \Delta x^2\omega_{i,j} \tag{13.1.12}$$

is the residual, and $\beta = \Delta x^2/\Delta y^2$. The transient evolution is numerically stable if $\alpha_x(1+\beta) < \frac{1}{2}$. The iterative method involves computing a time-like sequence of grid values parametrized by the index l, computed using the formula

$$\psi_{i,j}^{(l+1)} = \psi_{i,j}^{(l)} + \frac{\varrho}{2(1+\beta)} R_{i,j}^{(l)}, \tag{13.1.13}$$

for $l = 1, 2, \ldots$, where ϱ is a relaxation factor used to control the iterations. When the iterations are executed for the first time, the initial guess $\psi^{(0)}$ is set equal to that assigned in Step 1. When the iterations converge, the solution is second-order accurate with respect to Δx and Δy.

Step 4: *Use the simplified expressions (13.1.6) to compute the vorticity at the boundary grid points by one-sided finite differences.*

To compute the vorticity at a grid point along the upper wall, we expand the stream function in a Taylor series with respect to y about a grid point that lies at the lid and evaluate the series at the

N_y level immediately below to obtain

$$\psi_{i,N_y} \simeq \psi_{i,N_y+1} - \Delta y \left(\frac{\partial \psi}{\partial y}\right)_{i,N_y+1} + \frac{1}{2}\,\Delta y^2 \left(\frac{\partial^2 \psi}{\partial y^2}\right)_{i,N_y+1}. \tag{13.1.14}$$

The no-slip boundary condition (13.1.5) requires that $(\partial\psi/\partial y)_{i,N_y+1} = V$, and the first equation in (13.1.6) requires that $\omega_{i,N_y+1} = -(\partial^2\psi/\partial y^2)_{i,N_y+1}$. Substituting these expressions into (13.1.14) and solving for ω_{i,N_y+1}, we obtain

$$\omega_{i,N_y+1} = 2\,\frac{\psi_{i,N_y+1} - \psi_{i,N_y}}{\Delta y^2} - 2\,\frac{V}{\Delta y}, \tag{13.1.15}$$

which is first-order accurate in Δy. Working in a similar fashion for the bottom and side walls, we derive the corresponding expressions

$$\omega_{i,1} = 2\,\frac{\psi_{i,1} - \psi_{i,2}}{\Delta y^2}, \qquad \omega_{1,j} = 2\,\frac{\psi_{1,j} - \psi_{2,j}}{\Delta y^2}, \qquad \omega_{N_x+1,j} = 2\,\frac{\psi_{N_x+1,j} - \psi_{N_x,j}}{\Delta y^2}. \tag{13.1.16}$$

To increase the accuracy of the method to second order, we expand the stream function in a Taylor series about a grid point at the upper wall, evaluate the series at two layers immediately below the upper wall, and retain terms up to third order to find

$$\psi_{i,N_y} \simeq \psi_{i,N_y+1} - \Delta y \left(\frac{\partial \psi}{\partial y}\right)_{i,N_y+1} + \frac{1}{2}\,\Delta y^2 \left(\frac{\partial^2 \psi}{\partial y^2}\right)_{i,N_y+1} + \frac{1}{6}\,\Delta y^3 \left(\frac{\partial^3 \psi}{\partial y^3}\right)_{i,N_y+1} \tag{13.1.17}$$

and

$$\psi_{i,N_y-1} \simeq \psi_{i,N_y+1} - 2\Delta y \left(\frac{\partial \psi}{\partial y}\right)_{i,N_y+1} + 2\,\Delta y^2 \left(\frac{\partial^2 \psi}{\partial y^2}\right)_{i,N_y+1} - \frac{4}{3}\,\Delta y^3 \left(\frac{\partial^3 \psi}{\partial y^3}\right)_{i,N_y+1}. \tag{13.1.18}$$

Combining these equations to eliminate the third derivative of the stream function, solving for the second derivative, using the boundary condition (13.1.5), and taking into account (13.1.6), we find that

$$\omega_{i,N_y+1} = \frac{7\,\psi_{i,N_y+1} - 8\,\psi_{i,N_y} + \psi_{i,N_y-1}}{2\Delta y^2} - 3\,\frac{V}{\Delta y}. \tag{13.1.19}$$

Working in a similar fashion for the bottom and side walls, we find that

$$\omega_{i,1} = \frac{7\,\psi_{i,1} - 8\,\psi_{i,2} + \psi_{i,3}}{2\Delta y^2}, \qquad \omega_{1,j} = \frac{7\,\psi_{1,j} - 8\,\psi_{2,j} + \psi_{3,j}}{2\Delta x^2}, \qquad \omega_{N_x+1,j} = \frac{7\,\psi_{N_x+1,j} - 8\,\psi_{N_x,j} + \psi_{N_x,j}}{2\Delta x^2}. \tag{13.1.20}$$

Step 5: *Differentiate the stream function to compute the velocity at the internal grid points subject to the boundary values (13.1.4).*

Step 6: *Differentiate the vorticity to compute the right-hand side of (13.1.8) at the internal grid points subject to the boundary values computed from (13.1.15) and (13.1.16) or from (13.1.19) and (13.1.20).*

Step 7: *Solve the Poisson equation (13.1.8) for the vorticity subject to the Dirichlet boundary conditions for the vorticity derived in Step 4.*

This is done by iteration using, for example, the forward time–centered space (FTCS) algorithm discussed in Step 3, carrying out only a small number of iterations. Grid values of the forcing function on the right-hand side at the internal nodes are available from Step 6.

Step 8: *If the vorticity computed in Step 7 does not agree with that previously available within a specified tolerance, we return to Step 2 and repeat the computations with the new values of the vorticity.*

This outer iteration is terminated when the absolute value of the difference in the vorticity between two successive iterations at each grid point becomes less than a present threshold, ϵ, or when the sum of the absolute values of the differences in the vorticity over all internal $N_x \times N_y$ grid points becomes less than $N_x \times N_y \times \epsilon$.

One noteworthy feature of the algorithm is that the corner grid points do not enter the computation, which means that the numerical scheme is oblivious to the velocity discontinuity at the two upper corner points. The local jump may cause local oscillations and slow down the convergence of the iterations, but does not have a deleterious effect on the overall numerical method.

Structure of the flow

Vorticity and stream function contour plots for flow in a cavity with aspect ratio $L_x/L_y = 2$, and Reynolds number $\mathrm{Re} = VL_x/\nu = 1$ and 100 are shown in Figure 13.1.2. In the notation of Figure 13.1.1, $L_x = b_x - a_x$ and $L_y = b_y - a_y$. Stream function contours are both streamlines and particle paths in a two-dimensional flow.

When $\mathrm{Re} = 1$, we obtain a nearly creeping flow whose streamlines are symmetric with respect to the midplane due to reversibility of Stokes flow, as discussed in Section 6.1.7. Small regions of recirculating flow are present at the bottom two corners, requiring increased spatial resolution. The vorticity is singular at the upper two cavity corners due to the discontinuous boundary velocity. A local analysis shows that the shear stress diverges at these corners, and an infinite force is required to slide the lid as a result of the sharp-corner idealization, as discussed in Section 6.2. However, these physical singularities do not deter the performance of the numerical method.

As the Reynolds number increases, the center of the central eddy is shifted toward the upper right corner due to the fluid inertia and the flow becomes unsymmetric. To compute flow at even higher Reynolds numbers, it is helpful to perform parameter continuation where the initial guesses for the stream function and vorticity are identified with the corresponding converged values at a lower Reynolds number. In fact, the Reynolds number can be increased gradually up to a targeted value in the course of the iterations.

Improvements and extensions

The computational procedure discussed in this section can be improved in several ways (e.g., [159]). In one improvement, instead of iterating on the Poisson equation for the vorticity in Step 7, we

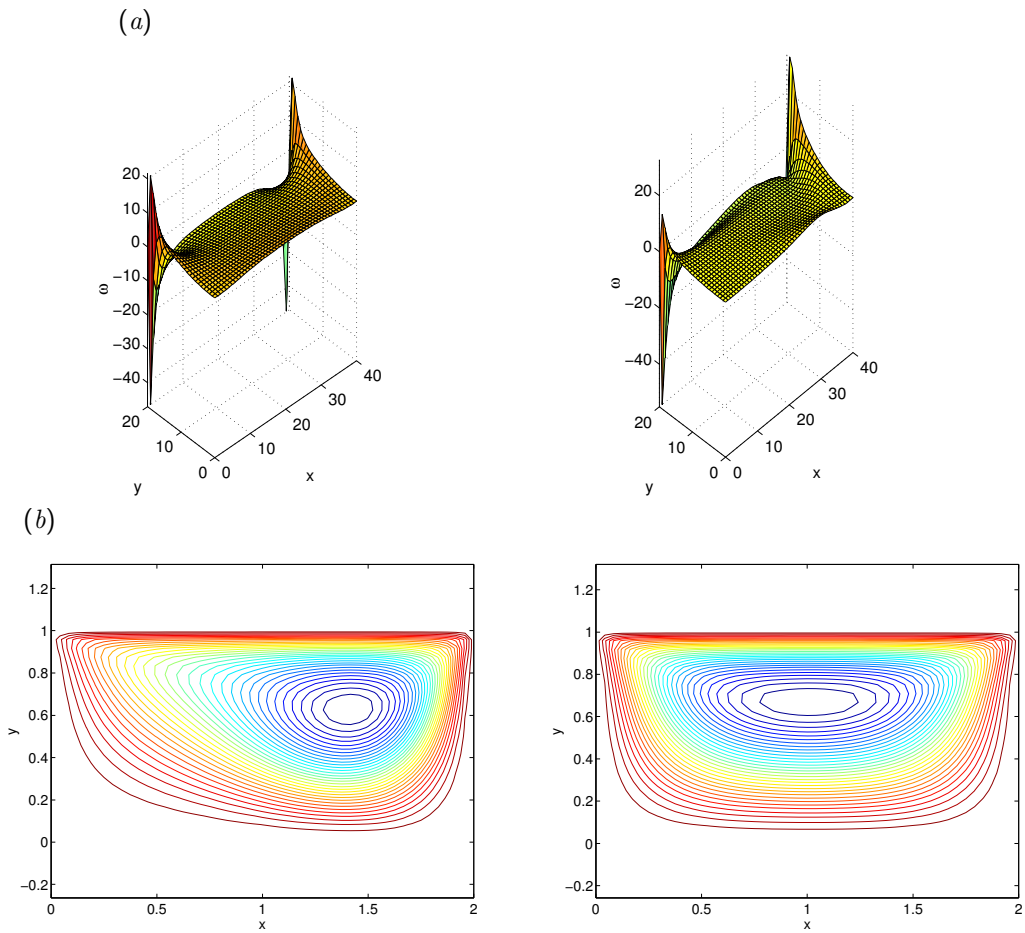

FIGURE 13.1.2 (*a*) Vorticity and (*b*) stream function contour plots for flow in a rectangular cavity with aspect ratio $L_x/L_y = 2$, at Reynolds number $\mathrm{Re} = VL_x/\nu = 1$ (left) and 100 (right).

iterate on the full diffusion–convection equation (13.1.8). This requires that the nonlinear term on the right-hand side is recomputed after each iteration using the updated vorticity. The iterations can be carried out using an explicit or implicit finite-difference method for the convection–diffusion equation in two dimensions discussed in Section 12.7. Because the flow near the center of the cavity is dominated by convection at high Reynolds numbers, using upwind differencing helps improve numerical stability. The rate of convergence of the inner and outer iterations depends on the details of the particular implementation [198].

The method for computing the boundary values of the vorticity described in Step 3 has been the subject of criticism [155]. It has been argued that it is not appropriate to explicitly impose the boundary values of the vorticity. Instead, the boundary distribution of the vorticity should arise

implicitly as part of the solution using the natural boundary conditions for the velocity or traction. This issue will be discussed further in Section 13.2 in the more general context of three-dimensional flow.

13.1.4 Modified dynamics for steady flow

A distinguishing feature of the direct approach described in Section 13.1.3 is the solution of a Poisson equation for the stream function, reflecting the elliptic nature of the equations governing the structure of a steady flow. Physically, the velocity at a point in a flow affects the velocity at every other point. We saw that one way to perform the associated inner iterations is to introduce a fictitious unsteady diffusion–reaction problem and then implement the explicit FTCS discretization.

This observation suggests reformulating the problem by retaining the vorticity transport equation (13.1.1) and replacing (13.1.2) with an evolution equation equation for the stream function expressed by (13.1.10), where the diffusivity κ is a free parameter of the numerical method. The idea is to compute the evolution of the flow from a certain initial condition subject to (13.1.1) and (13.1.10) until a steady state has been established. The steady-state solution will satisfy the original equations (13.1.7) and (13.1.8). An advantage of this approach is that all governing equations are parabolic in time. Time-marching schemes similar to those developed in Chapter 12 for convection–diffusion may then be employed.

For the problem of flow in a cavity illustrated in Figure 13.1.1, the method of modified dynamics is implemented according to the following steps:

1. Assign initial values to the stream function and vorticity at all internal and boundary grid points. A simple choice is to set them both to zero.
2. Differentiate the stream function to compute the two components of the velocity at all internal grid points.
3. Compute the vorticity at the boundary grid points.
4. Advance the vorticity at all internal grid points on the basis of (13.1.2) using, for example, the ADI method for the convection–diffusion equation described in Section 12.7, while keeping the vorticity at the boundary grid points constant.
5. Advance the stream function at all internal grid points on the basis of (13.1.10) using, for example, the ADI method for the convection–diffusion equation described in Section 12.7.
6. Return to Step 3 and repeat the computations for another time step.

13.1.5 Unsteady flow

To compute the evolution of an unsteady flow, we combine features of the direct method with features of the method of modified dynamics for steady flow. The algorithm involves computing the evolution of the vorticity field using (13.1.2), while simultaneously describing the evolution of the velocity field in terms of the stream function using (13.1.1). To simplify the notation, we denote $u = u_x$, $v = u_y$, and $\omega = \omega_z$. A simple strategy for computing the evolution of flow in a cavity when the lid starts translating suddenly with constant velocity V involves the following steps: