25
Fluid Dynamics

We have already introduced some fluid dynamics in the previous chapter's discussion of shallow-water solitons. This chapter confronts the more general equations of computational fluid dynamics (CFD) and their solutions.[1] The mathematical description of the motion of fluids, although not a new subject, remains a challenging one. Equations are complicated and nonlinear, there are many degrees of freedom, the nonlinearities may lead to instabilities, analytic solutions are rare, and the boundary conditions for realistic geometries (like airplanes) are not intuitive. These difficulties may explain why fluid dynamics is often absent from undergraduate and even graduate physics curricula. Nonetheless, as an essential element of the physical world that also has tremendous practical importance, we encourage its study. We recommend Fetter and Walecka (1980); Landau and Lifshitz (1987) for those interested in the derivations, and Shaw (1992) for more details about the computations.

25.1
River Hydrodynamics

Problem In order to give migrating salmon a place to rest during their arduous upstream journey, the Oregon Department of Environment is placing objects in several deep, wide, fast-flowing streams. One such object is a long beam of rectangular cross section (Figure 25.1a), and another is a set of plates (Figure 19.4b). The objects are to be placed far enough below the water's surface so as not to disturb the surface flow, and far enough from the bottom of the stream so as not to disturb the flow there either. Your *problem* is to determine the spatial dependence of the stream's velocity when the objects are in place, and, specifically, whether the wake of the object will be large enough to provide a resting place for a meter-long salmon.

[1] We acknowledge some helpful reading of by Satoru S. Kano.

Computational Physics, 3rd edition. Rubin H. Landau, Manuel J. Páez, Cristian C. Bordeianu.

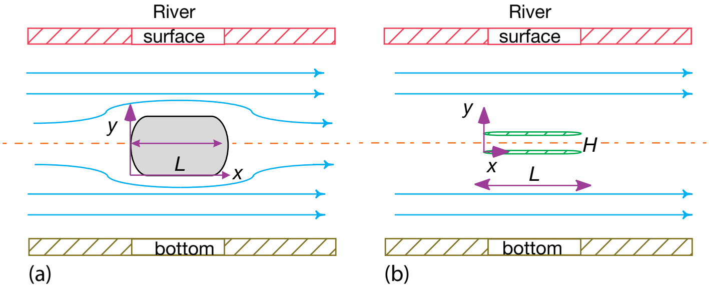

Figure 25.1 Side view of the flow of a stream around a submerged beam (a) and around two parallel plates (b). Both beam and plates have length L along the direction of flow. The flow is seen to be symmetric about the centerline and to be unaffected at the bottom and at the surface by the submerged object.

25.2 Navier–Stokes Equation (Theory)

As with our study of shallow-water waves, we assume that water is *incompressible* and thus that its density ρ is constant. We also simplify the theory by looking only at steady-state situations, that is, ones in which the velocity is not a function of time. However, to understand how water flows around objects, like our beam, it is essential to include the complication of frictional forces (*viscosity*).

For the sake of completeness, we repeat here the first equation of hydrodynamics, the continuity equation (24.2):

$$\frac{\partial \rho(\boldsymbol{x}, t)}{\partial t} + \boldsymbol{\nabla} \cdot \boldsymbol{j} = 0 \,, \qquad \boldsymbol{j} \overset{\text{def}}{=} \rho \boldsymbol{v}(\boldsymbol{x}, t) \,. \tag{25.1}$$

Before proceeding to the second equation, we introduce a special time derivative, the *hydrodynamic* derivative $D\boldsymbol{v}/Dt$, which is appropriate for a quantity contained in a moving fluid (Fetter and Walecka, 1980):

$$\frac{D\boldsymbol{v}}{Dt} \overset{\text{def}}{=} (\boldsymbol{v} \cdot \boldsymbol{\nabla})\boldsymbol{v} + \frac{\partial \boldsymbol{v}}{\partial t} \,. \tag{25.2}$$

This derivative gives the rate of change, as viewed from a stationary frame, of the velocity of material in *an element of flowing fluid* and so incorporates changes as a result of the motion of the fluid (first term) as well as any explicit time dependence of the velocity. Of particular interest is that $D\boldsymbol{v}/Dt$ is of second order in the velocity, and so its occurrence introduces nonlinearities into the theory. You may think of these nonlinearities as related to the fictitious (inertial) forces that occur when we describe the motion in the fluid's rest frame (an accelerating frame to us).

The material derivative is the leading term in the *Navier–Stokes equation*,

$$\frac{D\boldsymbol{v}}{Dt} = \nu\nabla^2\boldsymbol{v} - \frac{1}{\rho}\boldsymbol{\nabla}P(\rho, T, x) \ , \tag{25.3}$$

$$\begin{aligned}
\frac{\partial v_x}{\partial t} + \sum_{j=x}^{z} v_j \frac{\partial v_x}{\partial x_j} &= \nu \sum_{j=x}^{z} \frac{\partial^2 v_x}{\partial x_j^2} - \frac{1}{\rho}\frac{\partial P}{\partial x} \ , \\
\frac{\partial v_y}{\partial t} + \sum_{j=x}^{z} v_j \frac{\partial v_y}{\partial x_j} &= \nu \sum_{j=x}^{z} \frac{\partial^2 v_y}{\partial x_j^2} - \frac{1}{\rho}\frac{\partial P}{\partial y} \ , \\
\frac{\partial v_z}{\partial t} + \sum_{j=x}^{z} v_j \frac{\partial v_z}{\partial x_j} &= \nu \sum_{j=x}^{z} \frac{\partial^2 v_z}{\partial x_j^2} - \frac{1}{\rho}\frac{\partial P}{\partial z} \ .
\end{aligned} \tag{25.4}$$

Here, ν is the kinematic viscosity, P is the pressure, and (25.4) gives the derivatives in (25.3) in Cartesian coordinates. This equation describes transfer of the momentum of the fluid within some region of space as a result of forces and flow (think $\mathrm{d}\boldsymbol{p}/\,\mathrm{d}t = \boldsymbol{F}$). There is a simultaneous equation for each of the three velocity components. The $\boldsymbol{v}\cdot\nabla\boldsymbol{v}$ term in $D\boldsymbol{v}/Dt$ describes the transport of momentum in some region of space resulting from the fluid's flow and is often called the *convection* or *advection* term.[2] The $\boldsymbol{\nabla}P$ term describes the velocity change as a result of pressure changes, and the $\nu\nabla^2\boldsymbol{v}$ term describes the velocity change resulting from viscous forces (which tend to dampen the flow).

The explicit functional dependence of the pressure on the fluid's density and temperature $P(\rho, T, x)$ is known as the *equation of state of the fluid*, and would have to be known before trying to solve the Navier–Stokes equation. To keep our problem simple, we assume that the pressure is independent of density and temperature. This leaves the four simultaneous partial differential equations to solve, (25.1) and (25.3). Because we are interested in *steady-state* flow around an object, we assume that all time derivatives of the velocity vanish. Because we assume that the fluid is incompressible, the time derivative of the density also vanishes, and (25.1) and (25.3) become

$$\boldsymbol{\nabla}\cdot\boldsymbol{v} \equiv \sum_i \frac{\partial v_i}{\partial x_i} = 0 \ , \tag{25.5}$$

$$(\boldsymbol{v}\cdot\boldsymbol{\nabla})\boldsymbol{v} = \nu\nabla^2\boldsymbol{v} - \frac{1}{\rho}\boldsymbol{\nabla}P \ . \tag{25.6}$$

The first equation expresses the equality of inflow and outflow and is known as the *condition of incompressibility*. In as much as the stream in our problem is much wider than the width (z direction) of the beam, and because we are staying away

[2] We discuss pure advection in Section 24.1. In oceanology or meteorology, convection implies the transfer of mass in the vertical direction where it overcomes gravity, whereas advection refers to transfer in the horizontal direction.

from the banks, we will ignore the z dependence of the velocity. The explicit PDEs we need to solve then reduce to

$$\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} = 0 \,, \tag{25.7}$$

$$\nu \left(\frac{\partial^2 v_x}{\partial x^2} + \frac{\partial^2 v_x}{\partial y^2} \right) = v_x \frac{\partial v_x}{\partial x} + v_y \frac{\partial v_x}{\partial y} + \frac{1}{\rho} \frac{\partial P}{\partial x} \,, \tag{25.8}$$

$$\nu \left(\frac{\partial^2 v_y}{\partial x^2} + \frac{\partial^2 v_y}{\partial y^2} \right) = v_x \frac{\partial v_y}{\partial x} + v_y \frac{\partial v_y}{\partial y} + \frac{1}{\rho} \frac{\partial P}{\partial y} \,. \tag{25.9}$$

25.2.1 Boundary Conditions for Parallel Plates

The plate problem is relatively easy to solve analytically, and so we will do it! This will give us some experience with the equations as well as a check for our numerical solution. To find a unique solution to the PDEs (25.7)–(25.9), we need to specify boundary conditions. As far as we can tell, picking boundary conditions is somewhat of an acquired skill.

We assume that the submerged parallel plates are placed in a stream that is flowing with a constant velocity V_0 in the horizontal direction (Figure 25.1b). If the velocity V_0 is not too high or the kinematic viscosity ν is sufficiently large, then the flow should be smooth and without turbulence. We call such flow *laminar*. Typically, a fluid undergoing laminar flow moves in smooth paths that do not close on themselves, like the smooth flow of water from a faucet. If we imagine attaching a vector to each element of the fluid, then the path swept out by that vector is called a *streamline* or *line of motion* of the fluid. These streamlines can be visualized experimentally by adding colored dye to the stream. We assume that the plates are so thin that the flow remains laminar as it passes around and through them.

If the plates are thin, then the flow upstream of them is not affected, and we can limit our solution space to the rectangular region in Figure 25.2. We assume that the length L and separation H of the plates are small compared to the size of the stream, so the flow returns to uniform as we get far downstream from the plates. As shown in Figure 25.2, there are boundary conditions at the *inlet* where the fluid enters the solution space, at the *outlet* where it leaves, and at the stationary plates. In addition, because the plates are far from the stream's bottom and surface, we assume that the dotted-dashed centerline is a plane of symmetry, with identical flow above and below the plane. We thus have four different types of boundary conditions to impose on our solution:

Solid plates Because there is friction (viscosity) between the fluid and the plate surface, the only way to have laminar flow is to have the fluid's velocity equal to the plate's velocity, which means both are zero:

$$v_x = v_y = 0 \,. \tag{25.10}$$

Such being the case, we have smooth flow in which the negligibly thin plates lie along streamlines of the fluid (like a “streamlined” vehicle).

Inlet The fluid enters the integration domain at the inlet with a horizontal velocity V_0. Because the inlet is far upstream from the plates, we assume that the fluid velocity at the inlet is unchanged by the presence of the plates:

$$v_x = V_0 \,, \quad v_y = 0 \,. \tag{25.11}$$

Outlet Fluid leaves the integration domain at the outlet. While it is totally reasonable to assume that the fluid returns to its unperturbed state there, we are not told what that is. So, instead, we assume that there is a physical outlet at the end with the water just shooting out of it. Consequently, we assume that the water pressure equals zero at the outlet (as at the end of a garden hose) and that the velocity does not change in a direction normal to the outlet:

$$P = 0 \,, \quad \frac{\partial v_x}{\partial x} = \frac{\partial v_y}{\partial x} = 0 \,. \tag{25.12}$$

Symmetry plane If the flow is symmetric about the $y = 0$ plane, then there cannot be flow through the plane, and the spatial derivatives of the velocity components normal to the plane must vanish:

$$v_y = 0 \,, \quad \frac{\partial v_y}{\partial y} = 0 \,. \tag{25.13}$$

This condition follows from the assumption that the plates are along streamlines and that they are negligibly thin. It means that all the streamlines are parallel to the plates as well as to the water surface, and so it must be that $v_y = 0$ everywhere. The fluid enters in the horizontal direction, the plates do not change the vertical y component of the velocity, and the flow remains symmetric about the centerline. There is a retardation of the flow around the plates because of the viscous nature of the flow and that of the $\boldsymbol{v} = 0$ boundary layers formed on the plates, but there are no actual v_y components.

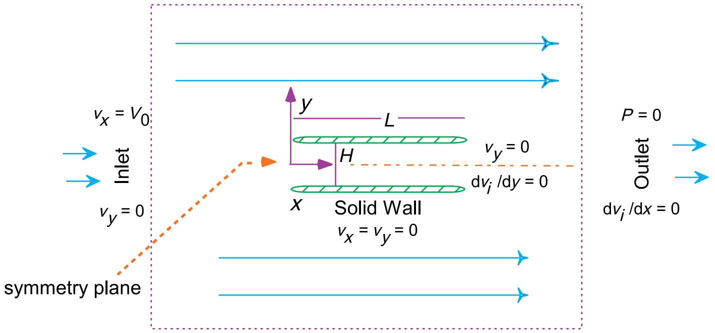

Figure 25.2 The boundary conditions for two thin submerged plates. The surrounding box is the integration volume within which we solve the PDEs and upon whose surface we impose the boundary conditions. In practice, the box is much larger than L and H.

25.2.2 Finite-Difference Algorithm and Overrelaxation

Now we develop an algorithm for solution of the Navier–Stokes and continuity PDEs using successive over-relaxation. This is a variation of the method used in Chapter 19, to solve Poisson's equation. We divide space into a rectangular grid with the spacing h in both the x and y directions:

$$x = ih \ , \quad i = 0, \dots, N_x \ ; \quad y = jh \ , \quad j = 0, \dots, N_y \ .$$

We next express the derivatives in (25.7)–(25.9) as finite differences of the values of the velocities at the grid points using central-difference approximations. For $\nu = 1\,\mathrm{m}^2/\mathrm{s}$ and $\rho = 1\,\mathrm{kg/m}^3$, this yields

$$v^x_{i+1,j} - v^x_{i-1,j} + v^y_{i,j+1} - v^y_{i,j-1} = 0 \ , \tag{25.14}$$

$$\begin{aligned} v^x_{i+1,j} + v^x_{i-1,j} + v^x_{i,j+1} + v^x_{i,j-1} - 4v^x_{i,j} &= \frac{h}{2} v^x_{i,j} \left[v^x_{i+1,j} - v^x_{i-1,j} \right] \\ &+ \frac{h}{2} v^y_{i,j} \left[v^x_{i,j+1} - v^x_{i,j-1} \right] + \frac{h}{2} \left[P_{i+1,j} - P_{i-1,j} \right] \ , \end{aligned} \tag{25.15}$$

$$\begin{aligned} v^y_{i+1,j} + v^y_{i-1,j} + v^y_{i,j+1} + v^y_{i,j-1} - 4v^y_{i,j} &= \frac{h}{2} v^x_{i,j} \left[v^y_{i+1,j} - v^y_{i-1,j} \right] \\ &+ \frac{h}{2} v^y_{i,j} \left[v^y_{i,j+1} - v^y_{i,j-1} \right] + \frac{h}{2} \left[P_{i,j+1} - P_{i,j-1} \right] \ . \end{aligned} \tag{25.16}$$

Because $v^y \equiv 0$ for this problem, we rearrange terms to obtain for v^x:

$$\begin{aligned} 4v^x_{i,j} &= v^x_{i+1,j} + v^x_{i-1,j} + v^x_{i,j+1} + v^x_{i,j-1} - \frac{h}{2} v^x_{i,j} \left[v^x_{i+1,j} - v^x_{i-1,j} \right] \\ &- \frac{h}{2} v^y_{i,j} \left[v^x_{i,j+1} - v^x_{i,j-1} \right] - \frac{h}{2} \left[P_{i+1,j} - P_{i-1,j} \right] \ . \end{aligned} \tag{25.17}$$

We recognize in (25.17) an algorithm similar to the one we used in solving Laplace's equation by relaxation. Indeed, as we did there, we can accelerate the convergence by writing the algorithm with the new value of v^x given as the old value plus a correction (residual):

$$v^x_{i,j} = v^x_{i,j} + r_{i,j} \ , \quad r \overset{\text{def}}{=} v^{x(\text{new})}_{i,j} - v^{x(\text{old})}_{i,j} \tag{25.18}$$

$$\begin{aligned} \Rightarrow \quad r = \frac{1}{4} \Bigg\{ & v^x_{i+1,j} + v^x_{i-1,j} + v^x_{i,j+1} + v^x_{i,j-1} - \frac{h}{2} v^x_{i,j} \left[v^x_{i+1,j} - v^x_{i-1,j} \right] \\ & - \frac{h}{2} v^y_{i,j} \left[v^x_{i,j+1} - v^x_{i,j-1} \right] - \frac{h}{2} \left[P_{i+1,j} - P_{i-1,j} \right] \Bigg\} - v^x_{i,j} \ . \end{aligned} \tag{25.19}$$

As performed with the Poisson equation algorithm, successive iterations sweep the interior of the grid, continuously adding in the residual (25.18) until the change becomes smaller than some set level of tolerance, $|r_{i,j}| < \varepsilon$.

A variation of this method, *successive over-relaxation*, increases the speed at which the residuals approach zero via an amplifying factor ω:

$$v^x_{i,j} = v^x_{i,j} + \omega r_{i,j} \quad \text{(SOR)} \; . \tag{25.20}$$

The standard relaxation algorithm (25.18) is obtained with $\omega = 1$, an accelerated convergence (*over-relaxation*) is obtained with $\omega \geq 1$, and *underrelaxation* occurs for $\omega < 1$. Values $\omega > 2$ are found to lead to numerical instabilities. Although a detailed analysis of the algorithm is necessary to predict the optimal value for ω, we suggest that you test different values for ω to see which one provides the fastest convergence for your problem.

25.2.3 Successive Overrelaxation Implementation

1. Modify the program `Beam.py`, or write your own, to solve the Navier–Stokes equation for the velocity of a fluid in a 2D flow. Represent the x and y components of the velocity by the arrays `vx[Nx,Ny]` and `vy[Nx,Ny]`.
2. Specialize your solution to the rectangular domain and boundary conditions indicated in Figure 25.2.
3. Use of the following parameter values,

$$\begin{aligned} \nu &= 1\,\text{m}^2/\text{s}\,, \quad \rho = 10^3\,\text{kg/m}^3\,, \quad \text{(flow parameters)}\,, \\ N_x &= 400\,, \quad N_y = 40\,, \quad h = 1\,, \quad \text{(grid parameters)}\,, \end{aligned}$$

   leads to the analytic solution

$$\frac{\partial P}{\partial x} = -12, \quad \frac{\partial P}{\partial y} = 0\,, \quad v^x = \frac{3j}{20}\left(1 - \frac{j}{40}\right)\,, \quad v^y = 0\,. \tag{25.21}$$

4. For the relaxation method, output the iteration number and the computed v^x and then compare the analytic and numeric results.
5. Repeat the calculation and see if SOR speeds up the convergence.

25.3 2D Flow over a Beam

Now that the comparison with an analytic solution has shown that our CFD simulation works, we return to determining if the beam in Figure 25.1 might produce a good resting place for salmon. While we have no analytic solution with which to compare, our canoeing and fishing adventures have taught us that *standing waves* with fish in them are often formed behind rocks in streams, and so we expect there will be a standing wave formed behind the beam.

25.4 Theory: Vorticity Form of Navier–Stokes Equation

We have seen how to solve numerically the hydrodynamics equations

$$\boldsymbol{\nabla} \cdot \boldsymbol{v} = 0 \,, \tag{25.22}$$

$$(\boldsymbol{v} \cdot \boldsymbol{\nabla})\boldsymbol{v} = -\frac{1}{\rho}\boldsymbol{\nabla} P + \nu \nabla^2 \boldsymbol{v} \,. \tag{25.23}$$

These equations determine the components of a fluid's velocity, pressure, and density as functions of position. In analogy to electrostatics, where one usually solves for the simpler scalar potential and then takes its gradient to determine the more complicated vector field, we now recast the hydrodynamic equations into forms that permit us to solve two simpler equations for simpler functions, from which the velocity is obtained via a gradient operation.[3]

We introduce the *stream function* $\boldsymbol{u}(\boldsymbol{x})$ from which the velocity is determined by the curl operator:

$$\boldsymbol{v} \overset{\text{def}}{=} \boldsymbol{\nabla} \times \boldsymbol{u}(\boldsymbol{x}) = \hat{\epsilon}_x \left(\frac{\partial u_z}{\partial y} - \frac{\partial u_y}{\partial z} \right) + \hat{\epsilon}_y \left(\frac{\partial u_x}{\partial z} - \frac{\partial u_z}{\partial x} \right) \,. \tag{25.24}$$

Note the absence of the z component of velocity $\boldsymbol{v}$ for our problem. Because $\boldsymbol{\nabla} \cdot (\boldsymbol{\nabla} \times \boldsymbol{u}) \equiv 0$, we see that any $\boldsymbol{v}$ that can be written as the curl of $\boldsymbol{u}$ automatically satisfies the continuity equation $\boldsymbol{\nabla} \cdot \boldsymbol{v} = 0$. Further, because $\boldsymbol{v}$ for our problem has only x and y components, $\boldsymbol{u}(\boldsymbol{x})$ needs have only a z component:

$$u_z \equiv u \quad \Rightarrow \quad v_x = \frac{\partial u}{\partial y} \,, \quad v_y = -\frac{\partial u}{\partial x} \,. \tag{25.25}$$

It is worth noting that in 2D flows, the contour lines $u =$ constant are the *streamlines*.

The second simplifying function is the *vorticity* field $\boldsymbol{w}(\boldsymbol{x})$, which is related physically and alphabetically to the angular velocity $\boldsymbol{\omega}$ of the fluid. Vorticity is defined as the curl of the velocity (sometimes with a − sign):

$$\boldsymbol{w} \overset{\text{def}}{=} \boldsymbol{\nabla} \times \boldsymbol{v}(\boldsymbol{x}) \,. \tag{25.26}$$

Because the velocity in our problem does not change in the z direction, we have

$$w_z = \left(\frac{\partial v_y}{\partial x} - \frac{\partial v_x}{\partial y} \right) \,. \tag{25.27}$$

Physically, we see that the vorticity is a measure of how much the fluid's velocity curls or rotates, with the direction of the vorticity determined by the right-hand

[3] If we had to solve only the simpler problem of *irrotational flow* (no turbulence), then we would be able to use a scalar velocity potential, in close analogy to electrostatics (Lamb, 1993). For the more general *rotational flow*, two vector potentials are required.

rule for rotations. In fact, if we could pluck a small element of the fluid into space (so it would not feel the internal strain of the fluid), we would find that it is rotating like a solid with angular velocity $\boldsymbol{\omega} \propto \boldsymbol{w}$ (Lamb, 1993). That being the case, it is useful to think of the vorticity as giving the local value of the fluid's angular velocity vector. If $\boldsymbol{w} = 0$, we have *irrotational* flow.

The field lines of w are continuous and move as if they are attached to the particles of the fluid. A uniformly flowing fluid has vanishing curl, while a nonzero vorticity indicates that the current curls back on itself or rotates. From the definition of the stream function (25.24), we see that the vorticity $\boldsymbol{w}$ is related to it by

$$\boldsymbol{w} = \boldsymbol{\nabla} \times \boldsymbol{v} = \boldsymbol{\nabla} \times (\boldsymbol{\nabla} \times \boldsymbol{u}) = \boldsymbol{\nabla}(\boldsymbol{\nabla} \cdot \boldsymbol{u}) - \nabla^2 \boldsymbol{u} \, , \tag{25.28}$$

where we have used a vector identity for $\boldsymbol{\nabla} \times (\boldsymbol{\nabla} \times \boldsymbol{u})$. Yet the divergence $\boldsymbol{\nabla} \cdot \boldsymbol{u} = 0$ because $\boldsymbol{u}$ has a component only in the z-direction and that component is independent of z (or because there is no source for $\boldsymbol{u}$). We have now obtained the basic relation between the stream function $\boldsymbol{u}$ and the vorticity $\boldsymbol{w}$:

$$\boldsymbol{\nabla}^2 \boldsymbol{u} = -\boldsymbol{w} \, . \tag{25.29}$$

Equation 25.29 is analogous to Poisson's equation of electrostatics, $\nabla^2 \phi = -4\pi\rho$, only now each component of vorticity $\boldsymbol{w}$ is a source for the corresponding component of the stream function $\boldsymbol{u}$. If the flow is irrotational, that is, if $\boldsymbol{w} = 0$, then we need only to solve Laplace's equation for each component of u. Rotational flow, with its coupled nonlinearities equations, leads to more interesting behavior.

As is to be expected from the definition of $\boldsymbol{w}$, the vorticity form of the Navier–Stokes equation is obtained by taking the curl of the velocity form, that is, by operating on both sides with $\boldsymbol{\nabla}\times$. After significant manipulations one obtains

$$\nu \nabla^2 \boldsymbol{w} = [(\boldsymbol{\nabla} \times \boldsymbol{u}) \cdot \boldsymbol{\nabla}] \boldsymbol{w} \, . \tag{25.30}$$

This and (25.29) are the two simultaneous PDEs that we need to solve. In 2D, with $\boldsymbol{u}$ and $\boldsymbol{w}$ having only z components, they are

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = -w \, , \tag{25.31}$$

$$\nu \left(\frac{\partial^2 w}{\partial x^2} + \frac{\partial^2 w}{\partial y^2} \right) = \frac{\partial u}{\partial y} \frac{\partial w}{\partial x} - \frac{\partial u}{\partial x} \frac{\partial w}{\partial y} \, . \tag{25.32}$$

So after all that work, we end up with two simultaneous, nonlinear, elliptic PDEs that look like a mixture of Poisson's equation with the wave equation. The equation for u is Poisson's equation with source w, and must be solved simultaneously with the second. It is this second equation that contains mixed products of the derivatives of u and w and thus introduces the nonlinearity.

25.4.1 Finite Differences and the SOR Algorithm

We solve (25.31) and (25.32) on an $N_x \times N_y$ grid of uniform spacing h with

$$x = i\Delta x = ih\,, \quad i = 0, \dots, N_x\,, \quad y = j\Delta y = jh\,, \quad j = 0, \dots, N_y\,. \quad (25.33)$$

Because the beam is symmetric about its centerline (Figure 25.1a), we need the solution only in the upper half-plane. We apply the now familiar central-difference approximation to the Laplacians of u and w to obtain the difference equation

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \simeq \frac{u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{i,j}}{h^2}\,. \quad (25.34)$$

Likewise, for the product of the first derivatives,

$$\frac{\partial u}{\partial y}\frac{\partial w}{\partial x} \simeq \frac{u_{i,j+1} - u_{i,j-1}}{2h}\frac{w_{i+1,j} - w_{i-1,j}}{2h}\,. \quad (25.35)$$

The difference form of the vorticity of the Navier–Stokes equation (25.31) becomes

$$u_{i,j} = \frac{1}{4}\left(u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} + h^2 w_{i,j}\right)\,, \quad (25.36)$$

$$\begin{aligned} w_{i,j} = {} & \frac{1}{4}(w_{i+1,j} + w_{i-1,j} + w_{i,j+1} + w_{i,j-1}) - \frac{R}{16}\left\{\left[u_{i,j+1} - u_{i,j-1}\right]\right. \\ & \left. \times \left[w_{i+1,j} - w_{i-1,j}\right] - \left[u_{i+1,j} - u_{i-1,j}\right]\left[w_{i,j+1} - w_{i,j-1}\right]\right\}\,, \end{aligned} \quad (25.37)$$

$$R = \frac{1}{\nu} = \frac{V_0 h}{\nu}(\text{in normal units})\,. \quad (25.38)$$

Note that we have placed $u_{i,j}$ and $w_{i,j}$ on the LHS of the equations in order to obtain an algorithm appropriate to a solution by relaxation.

The parameter R in (25.38) is related to the *Reynolds number*. When we solve the problem in natural units, we measure distances in units of grid spacing h, velocities in units of initial velocity V_0, stream functions in units of $V_0 h$, and vorticity in units of V_0/h. The second form is in regular units and is dimensionless. This R is known as the *grid Reynolds number* and differs from the physical R, which has a pipe diameter in place of the grid spacing h.

The grid Reynolds number is a measure of the strength of the coupling of the nonlinear terms in the equation. When the physical R is small, the viscosity acts as a frictional force that damps out fluctuations and keeps the flow smooth. When R is large ($R \simeq 2000$), physical fluids undergo phase transitions from laminar to turbulent flow in which turbulence occurs at a cascading set of smaller and smaller space scales (Reynolds, 1883). However, simulations that produce the onset of turbulence have been a research problem because Reynolds first experiments in 1883 (Falkovich and Sreenivasan, 2006), possibly because the laminar flow is stable against small perturbations and some large-scale “kick” may be needed to change laminar to turbulent flow. Recent research along these lines have been able to find unstable, traveling-wave solutions to the Navier–Stokes equations, and the hope is that these may lead to a turbulent transition (Fitzgerald, 2004).

As discussed in Section 25.2.2, the finite-difference algorithm can have its convergence accelerated by the use of successive over-relaxation (25.36):

$$u_{i,j} = u_{i,j} + \omega r^{(1)}_{i,j} , \quad w_{i,j} = w_{i,j} + \omega r^{(2)}_{i,j} \quad \text{(SOR)} . \tag{25.39}$$

Here, ω is the over-relaxation parameter and should lie in the range $0 < \omega < 2$ for stability. The residuals are just the changes in a single step, $r^{(1)} = u^{\text{new}} - u^{\text{old}}$ and $r^{(2)} = w^{\text{new}} - w^{\text{old}}$:

$$\begin{aligned} r^{(1)}_{i,j} &= \frac{1}{4}\left(u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} + w_{i,j}\right) - u_{i,j} , \\ r^{(2)}_{i,j} &= \frac{1}{4}\Big(w_{i+1,j} + w_{i-1,j} + w_{i,j+1} + w_{i,j-1} - \frac{R}{4}\big\{\left[u_{i,j+1} - u_{i,j-1}\right] \\ &\quad \times \left[w_{i+1,j} - w_{i-1,j}\right] - \left[u_{i+1,j} - u_{i-1,j}\right]\left[w_{i,j+1} - w_{i,j-1}\right]\big\}\Big) - w_{i,j} . \end{aligned} \tag{25.40}$$

25.4.2 Boundary Conditions for a Beam

A well-defined solution of these elliptic PDEs requires a combination of (less than obvious) boundary conditions on u and w. Consider Figure 25.3, based on the analysis of Koonin (1986). We assume that the inlet, outlet, and surface are far from the beam, which may not be evident from the not-to-scale figure.

Freeflow If there were no beam in the stream, then we would have free flow with the entire fluid possessing the inlet velocity:

$$v_x \equiv V_0 , \quad v_y = 0 , \quad \Rightarrow \quad u = V_0 y , \quad w = 0 . \tag{25.41}$$

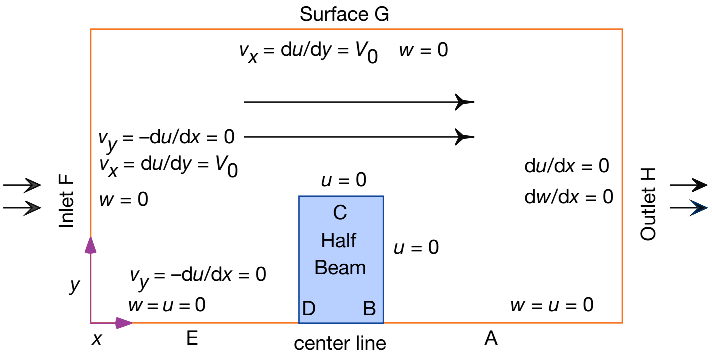

Figure 25.3 Boundary conditions for flow around the beam in Figure 25.1. The flow is symmetric about the centerline, and the beam has length L in the x direction (along flow).

(Recall that we can think of $w = 0$ as indicating no fluid rotation.) The centerline divides the system along a symmetry plane with identical flow above and below it. If the velocity is symmetric about the centerline, then its y component must vanish there:

$$v_y = 0\,, \quad \Rightarrow \quad \frac{\partial u}{\partial x} = 0 \quad \text{(centerline AE)}\,. \tag{25.42}$$

Centerline The centerline is a streamline with $u =$ constant because there is no velocity component perpendicular to it. We set $u = 0$ according to (25.41). Because there cannot be any fluid flowing into or out of the beam, the normal component of velocity must vanish along the beam surfaces. Consequently, the streamline $u = 0$ is the entire lower part of Figure 25.3, that is, the centerline and the beam surfaces. Likewise, the symmetry of the problem permits us to set the vorticity $w = 0$ along the centerline.

Inlet At the inlet, the fluid flow is horizontal with uniform x component V_0 at all heights and with no rotation:

$$v_y = -\frac{\partial u}{\partial x} = 0\,, \quad w = 0 \quad \text{(inlet F)}\,, \quad v_x = \frac{\partial u}{\partial y} = V_0\,. \tag{25.43}$$

Surface We are told that the beam is sufficiently submerged so as not to disturb the flow on the surface of the stream. Accordingly, we have free-flow conditions on the surface:

$$v_x = \frac{\partial u}{\partial y} = V_0\,, \quad w = 0 \quad \text{(surface G)}\,. \tag{25.44}$$

Outlet Unless something truly drastic happens, the conditions on the far downstream outlet have little effect on the far upstream flow. A convenient choice is to require the stream function and vorticity to be constant:

$$\frac{\partial u}{\partial x} = \frac{\partial w}{\partial x} = 0 \quad \text{(outlet H)}\,. \tag{25.45}$$

Beamsides We have already noted that the normal component of velocity v_x and stream function u vanish along the beam surfaces. In addition, because the flow is viscous, it is also true that the fluid "sticks" to the beam somewhat and so the tangential velocity also vanishes along the beam's surfaces. While these may all be true conclusions regarding the flow, specifying them as boundary conditions would over-restrict the solution (see Table 19.2 for elliptic equations) to the point where no solution may exist. Accordingly, we simply impose the *no-slip* boundary condition on the vorticity w. Consider a grid point (x, y) on the upper surface of the beam. The stream function u at a point $(x, y + h)$ above it can be related via a Taylor series in y:

$$u(x, y + h) = u(x, y) + \frac{\partial u}{\partial y}(x, y)h + \frac{\partial^2 u}{\partial y^2}(x, y)\frac{h^2}{2} + \cdots \tag{25.46}$$

Because $\boldsymbol{w}$ has only a z component, it has a simple relation to $\boldsymbol{\nabla} \times \boldsymbol{v}$:

$$w \equiv w_z = \frac{\partial v_y}{\partial x} - \frac{\partial v_x}{\partial y}\,. \tag{25.47}$$

Because of the fluid's viscosity, the velocity is stationary along the beam top:

$$v_x = \frac{\partial u}{\partial y} = 0 \quad \text{(beam top)} . \tag{25.48}$$

Because the current flows smoothly along the top of the beam, v_y must also vanish. In addition, because there is no x variation, we have

$$\frac{\partial v_y}{\partial x} = 0 \quad \Rightarrow \quad w = -\frac{\partial v_x}{\partial y} = -\frac{\partial^2 u}{\partial y^2} . \tag{25.49}$$

After substituting these relations into the Taylor series (25.46), we can solve for w and obtain the finite-difference version of the top boundary condition:

$$w \simeq -2\frac{u(x, y+h) - u(x, y)}{h^2} \quad \Rightarrow \quad w_{i,j} = -2\frac{u_{i,j+1} - u_{i,j}}{h^2} \quad \text{(top)} . \tag{25.50}$$

Similar treatments applied to other surfaces yield the boundary conditions.

$$\begin{array}{lll}
u = 0 & w = 0 & \text{Centerline EA} \\
u = 0 & w_{i,j} = -2(u_{i+1,j} - u_{i,j})/h^2 & \text{Beam back B} \\
u = 0 & w_{i,j} = -2(u_{i,j+1} - u_{i,j})/h^2 & \text{Beam top C} \\
u = 0 & w_{i,j} = -2(u_{i-1,j} - u_{i,j})/h^2 & \text{Beam front D} \\
\partial u/\partial x = 0 & w = 0 & \text{Inlet F} \\
\partial u/\partial y = V_0 & w = 0 & \text{Surface G} \\
\partial u/\partial x = 0 & \partial w/\partial x = 0 & \text{Outlet H}
\end{array} \tag{25.51}$$

25.4.3 SOR on a Grid

`Beam.py` in Listing 25.1 is our program for the solution of the vorticity form of the Navier–Stokes equation. You will notice that while the relaxation algorithm is rather simple, some care is needed in implementing many boundary conditions. Relaxation of the stream function and of the vorticity are performed by separate functions.

Listing 25.1 Beam.py solves the Navier–Stokes equation for the flow over a plate.

```
# Beam.py: solves Navier--Stokes equation for flow around beam

import matplotlib.pylab as p;
from mpl_toolkits.mplot3d import Axes3D;
from numpy import *;

print("Working, wait for the figure after 100 iterations")

Nxmax = 70;   Nymax = 20;   IL = 10;   H = 8;   T = 8;   h = 1.
u = zeros((Nxmax+1, Nymax+1), float)                       # Stream
w = zeros((Nxmax+1, Nymax+1), float)                       # Vorticity
V0 = 1.0;    omega = 0.1;   nu = 1.;   iter = 0;   R = V0 * h/nu
```

```
def borders():
    for i in range(0, Nxmax+1):                          # Init stream
       for j in range(0, Nymax+1):                       # Init vorticity
            w[i, j] = 0.
            u[i, j] = j * V0
    for i in range(0, Nxmax+1 ):                         # Fluid surface
      u[i, Nymax] = u[i, Nymax-1] + V0*h
      w[i, Nymax-1] = 0.
    for j in range(0, Nymax+1):
        u[1, j] = u[0, j]
        w[0, j] = 0.                                     # Inlet
    for  i in range(0, Nxmax+1):                         # Centerline
       if i <=  IL and i>= IL+T:
           u[i, 0] = 0.
           w[i, 0] = 0.
    for j in range(1, Nymax ):                           # Outlet
        w[Nxmax, j] = w[Nxmax-1, j]
        u[Nxmax, j] = u[Nxmax-1, j]

def beam():                                               # BC for beam
   for  j in range (0, H+1):                              # Sides
      w[IL, j] =  - 2 * u[IL-1, j]/(h*h)                  # Front
      w[IL+T, j] =  - 2 * u[IL  +  T  +  1, j]/(h*h)      # Back
   for  i in range(IL, IL+T + 1): w[i, H - 1] =  - 2 * u[i, H]/(h*h);
   for i in range(IL, IL+T+1):
     for j in range(0, H+1):
         u[IL, j] = 0.                                   # Front
         u[IL+T, j] = 0.                                 # Back
         u[i, H] = 0;                                    # Top

def relax():                                             # Relax stream
    beam()                                               # Reset conditions
    for i in range(1, Nxmax):                            # Relax stream
      for  j in range (1, Nymax):
        r1 = omega*((u[i+1,j]+u[i-1,j]+u[i,j+1]+u[i,j-1] +
             h*h*w[i,j])/4-u[i,j])
        u[i, j] +=  r1
    for  i in range(1, Nxmax):                           # Relax vorticity
      for j in range(1, Nymax):
          a1 = w[i+1, j]  +  w[i-1,j]  +  w[i,j+1] + w[i,j-1]
          a2 = (u[i,j+1] - u[i,j-1])*(w[i+1,j] - w[i - 1, j])
          a3 = (u[i+1,j] - u[i-1,j])*(w[i,j+1] - w[i, j - 1])
          r2 = omega *(  (a1 - (R/4.)*(a2 - a3) )/4. - w[i,j])
          w[i, j] +=  r2

borders()
while (iter <=  100):
    iter +=  1
    if iter%10 ==  0: print (iter)
    relax()
for i in range (0, Nxmax+1):
    for  j in range(0, Nymax+ 1):  u[i,j] = u[i,j]/V0/h     # V0h units
x = range(0, Nxmax-1);         y = range(0, Nymax-1)
X, Y = p.meshgrid(x, y)

def functz(u):                                           # Stream flow
    z = u[X, Y]
    return z

Z = functz(u)
fig = p.figure()
ax = Axes3D(fig)
ax.plot_wireframe(X, Y, Z, color = 'r')
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Stream Function')
p.show()
```

25.4.4 Flow Assessment

1. Use `Beam.py` as a basis for your solution for the stream function u and the vorticity w using the finite-differences algorithm (25.36).
2. A good place to start your simulation is with a beam of size $L = 8h$, $H = h$, Reynolds number $R = 0.1$, and intake velocity $V_0 = 1$. Keep your grid small during debugging, say, $N_x = 24$ and $N_y = 70$.
3. Explore the *convergence* of the algorithm.
   a) Print out the iteration number and u values upstream from, above, and downstream from the beam.
   b) Determine the number of iterations necessary to obtain three-place convergence for successive relaxation ($\omega = 0$).
   c) Determine the number of iterations necessary to obtain three-place convergence for successive over-relaxation ($\omega \simeq 0.3$). Use this number for future calculations.
4. Change the beam's horizontal placement so that you can see the undisturbed current entering on the left and then developing into a standing wave. Note that you may need to increase the size of your simulation volume to see the effect of all the boundary conditions.
5. Make surface plots including contours of the stream function u and the vorticity w. Explain the behavior seen.
6. Is there a region where a big fish can rest behind the beam?
7. The results of the simulation (Figure 25.4) are for the one-component stream function u. Make several visualizations showing the fluid velocity throughout

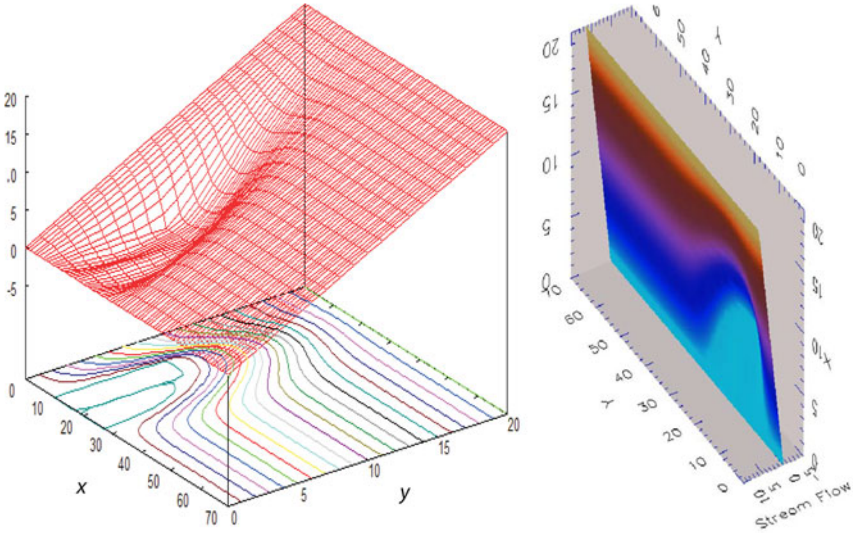

Figure 25.4 Two visualizations of the stream function u for Reynold's number $R = 5$. The one on the left uses contours, while the one on the right uses color.

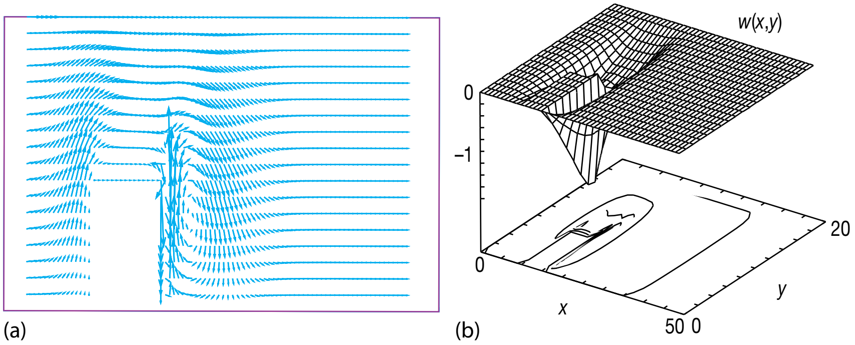

Figure 25.5 (a) The velocity field around the beam as represented by vectors. (b) The vorticity as a function of x and y for the same velocity field. Rotation is seen to be largest behind the beam.

the simulation region. Note that the velocity is a vector with two components (or a magnitude and direction), and both degrees of freedom are interesting to visualize. A plot of vectors would work well here.

8. Explore how increasing the Reynolds number R changes the flow pattern. Start at $R = 0$ and gradually increase R while watching for numeric instabilities. To overcome the instabilities, reduce the size of the relaxation parameter ω and continue to larger R values.
9. Verify that the flow around the beam is smooth for small R values, but that it separates from the back edge for large R, at which point a small vortex develops, as can be seen in Figure 25.5.

25.4.5 Exploration

1. Determine the flow behind a circular rock in the stream.
2. The boundary condition at an outlet far downstream should not have much effect on the simulation. Explore the use of other boundary conditions there.
3. Determine the pressure variation around the beam.