23

Finite Volume Methods on Quadrilateral Grids

Many multidimensional problems of practical interest involve complex geometry, and in general it is not sufficient to be able to solve hyperbolic equations on a uniform Cartesian grid in a rectangular domain. In Section 6.17 we considered a nonuniform grid in one space dimension and saw how hyperbolic equations can be solved on such a grid by using a uniform grid in *computational space* together with a coordinate mapping and appropriate scaling of the flux differences using *capacity form differencing*. The *capacity* of the computational cell is determined by the size of the corresponding physical cell.

In this chapter we consider nonuniform finite volume grids in two dimensions, such as those shown in Figure 23.1, and will see that similar techniques may be used. There are various ways to view the derivation of finite volume methods on general multidimensional grids. Here we will consider a direct physical interpretation in terms of fluxes normal to the cell edges. For simplicity we restrict attention to two space dimensions. For some other discussions of finite volume methods on general grids, see for example [156], [245], [475], [476].

The grids shown in Figures 23.1(a) and (b) are logically rectangular *quadrilateral grids*, and we will concentrate on this case. Each cell is a quadrilateral bounded by four linear segments. Such a grid is also often called a *curvilinear grid*. If we label the cells in a logical manner, indexing "rows" and "columns" by i and j, then cell (i, j) has the four neighbors $(i \pm 1, j)$ and $(i, j \pm 1)$. The grid can be made to wrap around the cylinder by imposing periodic boundary conditions in one direction.

The triangulation shown in Figure 23.1(c), on the other hand, gives an *unstructured grid* for which there is no simple logical structure underlying the connectivity between cells. One must keep track of the neighbors of each cell explicitly. An unstructured triangulation is often easier to generate for complicated geometries than a structured grid such as the one shown in Figure 23.1(a), but may be somewhat more difficult to work with in regard to data structures and the development of fast and accurate solvers. See [20], [324], [385], [472] for some discussions of unstructured grids.

The grids of Figure 23.1(a)–(c) are all *body-fitted grids* that conform to the geometry of the problem. Figure 23.1(d) shows a different approach in which a uniform Cartesian grid is used over most of the domain, but with some smaller irregular cells allowed where the boundary cuts through the grid. Finite volume methods of this type are often called *Cartesian-grid* or *embedded-boundary* methods. For problems with more complicated geometry this approach allows for very easy grid generation, and so it has recently become quite popular. With these

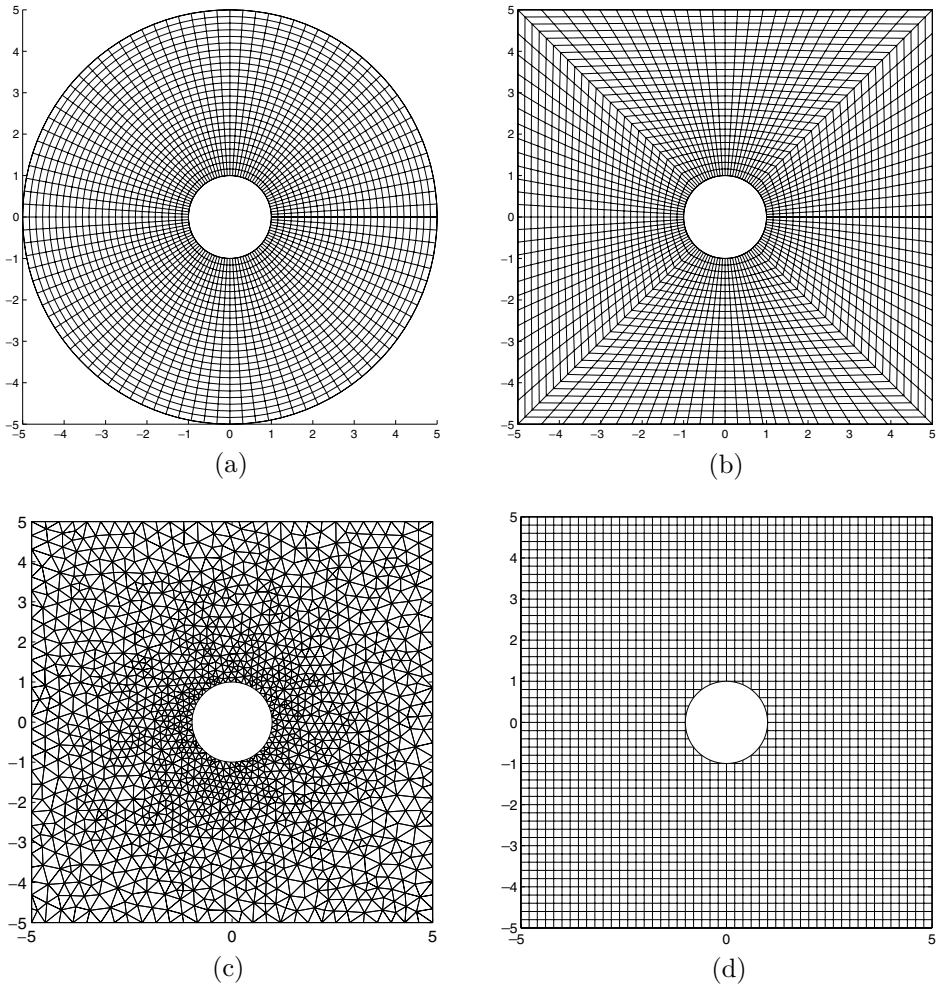

Fig. 23.1. Four possible grids for flow around a cylinder. (a) 25×100 polar grid in r–θ. (b) 25×104 quadrilateral grid interpolating between the cylinder and a square computational domain. (c) An unstructured triangulation. (d) A Cartesian grid with embedded boundary.

methods the main difficulty is in specifying fluxes at the edges of the small cut cells in such a way that good accuracy is obtained and stability is preserved with reasonable-size time steps. A variety of different approaches have been introduced; see for example [7], [31], [57], [59], [77], [104], [140], [277], [278], [358], [363], [490], [492].

23.1 Cell Averages and Interface Fluxes

With a finite volume method we view each discrete value Q as a cell average over a grid cell, which is modified due to fluxes through the edges of the cell, in the case of a conservation law, or more generally by waves moving into the cell from each edge. We start by discussing the case of a conservation law and will see how to extend this to nonconservative linear systems in Section 23.4.

The finite volume approach can be applied on any shape cell $\mathcal{C}$ using the integral form of the conservation law,

$$\frac{d}{dt}\iint_{\mathcal{C}} q(x, y, t)\, dx\, dy = -\int_{\partial\mathcal{C}} \vec{n}(s) \cdot \vec{f}(s, t)\, ds, \tag{23.1}$$

and hence is applicable on any of the grids of Figure 23.1 if a good numerical approximation to the interface flux can be determined. Here $\vec{n}(s)$ is the outward-pointing normal and

$$\vec{f}(s, t) = \begin{bmatrix} f(q(x(s), y(s), t)) \\ g(q(x(s), y(s), t)) \end{bmatrix},$$

with the boundary $(x(s), y(s))$ of $\mathcal{C}$ parameterized by the arclength s. Then

$$\breve{F}(s) = \vec{n}(s) \cdot \vec{f}(s, t) = n^x(s)\, f(q(x(s), y(s), t)) + n^y(s)\, g(q(x(s), y(s), t))$$

gives the flux per unit length per unit time in the direction $\vec{n}(s)$. Note that for a system of m equations f and g are vectors of length m and $\vec{f}$ is a vector of length $2m$, while the normal $\vec{n}(s)$ contains two scalar components n^x and n^y. (See Section 18.1 for more about the multidimensional notation used here.)

Integrating (23.1) from time t_n to t_{n+1} and dividing by $|\mathcal{C}|$, the area of the cell, gives

$$\begin{aligned}\frac{1}{|\mathcal{C}|}\iint_{\mathcal{C}} q(x, y, t_{n+1})\, dx\, dy = &\frac{1}{|\mathcal{C}|}\iint_{\mathcal{C}} q(x, y, t_n)\, dx\, dy \\ &- \frac{1}{|\mathcal{C}|}\int_{t_n}^{t_{n+1}} \int_{\partial\mathcal{C}} \vec{n}(s) \cdot \vec{f}(s, t)\, ds\, dt.\end{aligned} \tag{23.2}$$

If Q^n represents the cell average over this cell at time t_n, then this suggests the finite volume method

$$Q^{n+1} = Q^n - \frac{\Delta t}{|\mathcal{C}|}\sum_{j=1}^{N} h_j \breve{F}_j^n, \tag{23.3}$$

where $\breve{F}_j^n$ represents a numerical approximation to the average normal flux across the jth side of the cell, N is the number of sides, and h_j is the length of the jth side. The factors Δt and h_j are introduced by taking $\breve{F}_j^n$ as an approximation to the interface flux per unit length, per unit time,

$$\breve{F}_j^n \approx \frac{1}{\Delta t}\int_{t_n}^{t_{n+1}} \left(\frac{1}{h_j}\int_{\text{side } j} \vec{n} \cdot \vec{f}(s, t)\, ds\right) dt.$$

This agrees with the normalization used previously on Cartesian grids.

23.2 Logically Rectangular Grids

From now on we will consider only logically rectangular grids such as the ones shown in Figure 23.1(a) and (b). In this case we can write the finite volume method (23.3) as

$$Q_{ij}^{n+1} = Q_{ij} - \frac{\Delta t}{|\mathcal{C}_{ij}|}\big(h_{i+1/2,j}\check{F}_{i+1/2,j} - h_{i-1/2,j}\check{F}_{i-1/2,j} + h_{i,j+1/2}\check{G}_{i,j+1/2} - h_{i,j-1/2}\check{G}_{i,j-1/2}\big), \tag{23.4}$$

where

$$\begin{aligned}
|\mathcal{C}_{ij}| &= \text{area of cell } (i, j),\\
h_{i-1/2,j} &= \text{length of side between cells } (i-1, j) \text{ and } (i, j),\\
h_{i,j-1/2} &= \text{length of side between cells } (i, j-1) \text{ and } (i, j),\\
\check{F}_{i-1/2,j} &= \text{flux normal to edge between cells } (i-1, j) \text{ and } (i, j),\\
&\qquad \text{per unit time, per unit length,}\\
\check{G}_{i,j-1/2} &= \text{flux normal to edge between cells } (i, j-1) \text{ and } (i, j),\\
&\qquad \text{per unit time, per unit length.}
\end{aligned}$$

On a uniform Cartesian grid, $|\mathcal{C}_{ij}| = \Delta x\,\Delta y$, $h_{i-1/2,j} = \Delta y$, $h_{i,j-1/2} = \Delta x$, and (23.4) reduces to the standard flux-differencing formula (19.10).

Just as in one space dimension, we can put the general finite volume method (23.4) into a form where it can be viewed as capacity-form differencing on a uniform grid in computational space, which we now denote by ξ–η coordinates, as illustrated in Figure 23.2. The vertices (corners of grid cells) are mapped from the uniform computational grid to points in the physical domain by two coordinate-mapping functions $X(\xi, \eta)$ and $Y(\xi, \eta)$. We introduce the vector $\vec{h}_{i-1/2,j}$ as the vector connecting two corners of the grid cell, so that $h_{i-1/2,j}$ is the length of this vector.

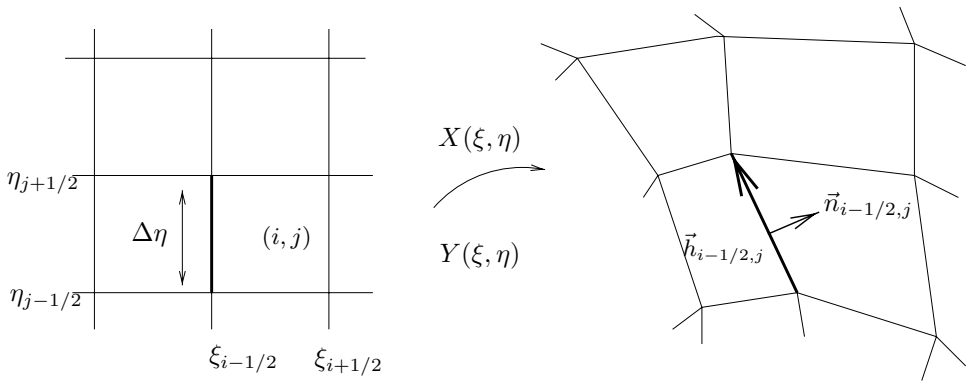

Fig. 23.2. The computational grid cells shown on the left are mapped to the physical grid cells shown on the right.

If we set

$$\begin{aligned} \kappa_{ij} &= \frac{|\mathcal{C}_{ij}|}{\Delta\xi\ \Delta\eta}, \\ F_{i-1/2,j} &= \left(\frac{h_{i-1/2,j}}{\Delta\eta}\right) \breve{F}_{i-1/2,j}, \\ G_{i,j-1/2} &= \left(\frac{h_{i,j-1/2}}{\Delta\xi}\right) \breve{G}_{i,j-1/2}, \end{aligned} \tag{23.5}$$

then the method (23.4) can be rewritten as

$$Q_{ij}^{n+1} = Q_{ij} - \frac{\Delta t}{\kappa_{ij}\ \Delta\xi}\big(F_{i+1/2,j} - F_{i-1/2,j}\big) - \frac{\Delta t}{\kappa_{ij}\ \Delta\eta}\big(G_{i,j+1/2} - G_{i,j-1/2}\big). \tag{23.6}$$

Note that $\breve{F}_{i-1/2,j}$ has units of flux per unit length in physical space (per unit time), so that multiplying by $h_{i-1/2,j}$ gives the total flux along the edge. Dividing by $\Delta\eta$ converts this into flux per unit length in computational space.

The length ratios appearing in (23.5) arise in many formulas below, and we will denote them by

$$\begin{aligned} \gamma_{i-1/2,j} &= h_{i-1/2,j}/\Delta\eta, \\ \gamma_{i,j-1/2} &= h_{i,j-1/2}/\Delta\xi. \end{aligned} \tag{23.7}$$

These quantities relate length in physical space to length in computational space. Similarly, the capacity κ_{ij} defined in (23.5) is an area ratio, between the area of the physical grid cell and the area $\Delta\xi\ \Delta\eta$ of the computational cell. If the mappings $X(\xi, \eta)$ and $Y(\xi, \eta)$ are sufficiently smooth functions, then these ratios can be related to derivatives of the mappings. However, we do not need to assume any smoothness in the mappings or the resulting grid in order to apply the finite volume methods. The accuracy may be reduced if the grid is not smooth, but the high-resolution methods typically perform quite well (see Example 23.2 in Section 23.8 below for an illustration of this on the grid shown in Figure 23.1(b)).

Note that if q represents a density function in physical space, so that its integral over the cell is the total mass, then the total mass in the (i, j) grid cell is roughly $Q_{ij}|\mathcal{C}_{ij}| = Q_{ij}\kappa_{ij}\ \Delta\xi\ \Delta\eta$. Hence we see that $q\kappa$ can be viewed as the "density function" in computational space, where the computational cell has area $\Delta\xi\ \Delta\eta$.

23.3 Godunov's Method

The fluxes $\breve{F}$ and $\breve{G}$ for Godunov's method are computed by solving a Riemann problem normal to the corresponding edge of the cell. As usual, we take the viewpoint that Q_{ij} defines the value everywhere in cell (i, j) of a piecewise constant function. Then at the $(i - 1/2, j)$ side of the cell we locally have a one-dimensional Riemann problem where there is variation only in the direction normal to this side. Let $\vec{n}_{i-1/2,j}$ be a unit normal in this direction (see Figure 23.2). Then the flux function for the one-dimensional equation in this direction is

$$\breve{F}(q) = \vec{n}_{i-1/2,j} \cdot \vec{f}(q).$$

Solving the Riemann problem with this flux function and data $Q_{i-1,j}$ and Q_{ij} gives the Godunov flux

$$\check{F}_{i-1/2,j} = \vec{n}_{i-1/2,j} \cdot \vec{f}\big(Q^{\Downarrow}_{i-1/2,j}\big), \tag{23.8}$$

where, as usual, $Q^{\Downarrow}_{i-1/2,j}$ is the solution at the interface (i.e., moving at speed zero) in the self-similar solution to the one-dimensional Riemann problem. Multiplying $\check{F}_{i-1/2,j}$ by $\gamma_{i-1/2,j}$ gives the numerical flux $F_{i-1/2,j}$. Similarly,

$$\check{G}_{i,j-1/2} = \vec{n}_{i,j-1/2} \cdot \vec{f}\big(Q^{\Downarrow}_{i,j-1/2}\big), \tag{23.9}$$

where $Q^{\Downarrow}_{i,j-1/2}$ is obtained by solving the Riemann problem normal to the $(i,\ j-1/2)$ edge with data $Q_{i,j-1}$ and Q_{ij}.

23.4 Fluctuation Form

We derived the form (23.6) for a conservation law by considering the flux through each edge of the grid cell. For a more general hyperbolic equation that may not be in conservation form, e.g.,

$$q_t + A(x, y)q_x + B(x, y)q_y = 0, \tag{23.10}$$

this approach cannot be used, and instead we wish to use a more general fluctuation updating formula in place of (23.6), of the form

$$\begin{aligned} Q^{n+1}_{ij} = Q_{ij} &- \frac{\Delta t}{\kappa_{ij}\,\Delta\xi}\big(\mathcal{A}^+\Delta Q_{i-1/2,j} + \mathcal{A}^-\Delta Q_{i+1/2,j}\big) \\ &- \frac{\Delta t}{\kappa_{ij}\,\Delta\eta}\big(\mathcal{B}^+\Delta Q_{i,j-1/2} + \mathcal{B}^-\Delta Q_{i,j+1/2}\big). \end{aligned} \tag{23.11}$$

This form can also be obtained by solving the appropriate Riemann problem normal to each edge of the cell and seeing how the resulting waves update cell averages to either side. We will see that the formulation earlier developed in Sections 6.14–6.16 and 19.3.3 can be extended to general quadrilateral grids. High-resolution correction terms can also be added to (23.11).

We assume that we know how to solve the Riemann problem normal to each edge based on piecewise constant initial data in the two neighboring cells and the direction of the normal, and that the solution consists of waves $\mathcal{W}^p$ propagating at speeds $\check{s}^p$. These waves and speeds are now used to define the fluctuations, as on a uniform Cartesian grid, but the additional scale factors γ and κ must be properly included. To see where these arise, consider the edge $(i-1/2,\ j)$. The wave $\mathcal{W}^p_{i-1/2,j}$ moves a distance $\check{s}^p_{i-1/2,j}\,\Delta t$ and has width $h_{i-1/2,j}$. If $\check{s}^p_{i-1/2,j} > 0$, for example, then this wave should update Q_{ij} by an amount

$$-\left(\frac{h_{i-1/2,j}\,\check{s}^p_{i-1/2,j}\,\Delta t}{|\mathcal{C}_{ij}|}\right)\mathcal{W}^p_{i-1/2,j}, \tag{23.12}$$

where we divide by $|\mathcal{C}_{ij}|$, the area of the cell, since Q_{ij} represents a cell average. Since we assume $\check{s}^p_{i-1/2,j} > 0$, this term should be part of the update arising from the term $-(\frac{\Delta t}{\kappa_{ij}\,\Delta\xi})\mathcal{A}^+\Delta Q_{i-1/2,j}$ in (23.11). Since $|\mathcal{C}_{ij}| = \kappa_{ij}\,\Delta\xi\,\Delta\eta$, we can rewrite (23.12) as

$$-\left(\frac{\Delta t}{\kappa_{ij}\,\Delta\xi}\right)\left(\frac{h_{i-1/2,j}}{\Delta\eta}\,\check{s}^p_{i-1/2,j}\right)\mathcal{W}^p_{i-1/2,j}. \tag{23.13}$$

We see that this wave's contribution to $\mathcal{A}^+\Delta Q$ should be $s^p_{i-1/2,j}\mathcal{W}^p_{i-1/2,j}$, where we define

$$s^p_{i-1/2,j} = \frac{h_{i-1/2,j}}{\Delta\eta}\check{s}^p_{i-1/2,j} = \gamma_{i-1/2,j}\check{s}^p_{i-1/2,j}. \tag{23.14}$$

It is this scaled speed $s^p_{i-1/2,j}$ that must also be used in high-resolution correction terms. So, after solving the Riemann problem, the wave speeds should all be scaled as in (23.14), and then the fluctuations are

$$\mathcal{A}^\pm\Delta Q_{i-1/2,j} = \sum_p \left(s^p_{i-1/2,j}\right)^\pm \mathcal{W}^p_{i-1/2,j}, \tag{23.15}$$

as usual.

Similarly, at the $(i, j-1/2)$ edge we solve the normal Riemann problem for waves $\mathcal{W}^p_{i,j-1/2}$ and speeds $\check{s}^p_{i,j-1/2}$ and then scale the speeds by $\gamma_{i,j-1/2} = h_{i,j-1/2}/\Delta\xi$:

$$s^p_{i,j-1/2} = \gamma_{i,j-1/2}\,\check{s}^p_{i,j-1/2}. \tag{23.16}$$

The fluctuations are then

$$\mathcal{B}^\pm\Delta Q_{i,j-1/2} = \sum_p \left(s^p_{i,j-1/2}\right)^\pm \mathcal{W}^p_{i,j-1/2}. \tag{23.17}$$

23.5 Advection Equations

The constant-coefficient advection equation in two dimensions is written as

$$q_t + uq_x + vq_y = 0, \tag{23.18}$$

with $\vec{u} = (u, v)$ being the velocity vector. The flux vector is

$$\vec{f}(q) = \vec{u}q = \begin{bmatrix} uq \\ vq \end{bmatrix}.$$

The flux at the $(i-1/2, j)$ edge is

$$\check{F}_{i-1/2,j} = \vec{n}_{i-1/2,j} \cdot \vec{f}\left(Q^{\Downarrow}_{i-1/2,j}\right) = \check{u}_{i-1/2,j}\,Q^{\Downarrow}_{i-1/2,j},$$

where

$$\check{u}_{i-1/2,j} = un^x_{i-1/2,j} + vn^y_{i-1/2,j}$$

is the velocity normal to the edge, and $Q^{\Downarrow}_{i-1/2,j}$ is the cell value from the upwind side of the interface,

$$Q^{\Downarrow}_{i-1/2,j} = \begin{cases} Q_{i-1,j} & \text{if } \breve{u}_{i-1/2,j} > 0, \\ Q_{ij} & \text{if } \breve{u}_{i-1/2,j} < 0. \end{cases}$$

After normalizing the flux by the ratio $\gamma_{i-1/2,j} = h_{i-1/2,j}/\Delta\eta$, we obtain

$$\begin{aligned} F_{i-1/2,j} &= \gamma_{i-1/2,j}\breve{u}_{i-1/2,j}Q^{\Downarrow}_{i-1/2,j} \\ &= \gamma_{i-1/2,j}\big(\breve{u}^{+}_{i-1/2,j}Q_{i-1,j} + \breve{u}^{-}_{i-1/2,j}Q_{ij}\big). \end{aligned} \tag{23.19}$$

Similarly, we find that

$$G_{i,j-1/2} = \gamma_{i,j-1/2}\big(\breve{v}^{+}_{i,j-1/2}Q_{i,j-1} + \breve{v}^{-}_{i,j-1/2}Q_{ij}\big), \tag{23.20}$$

where

$$\breve{v}_{i,j-1/2} = n^{x}_{i,j-1/2}u + n^{y}_{i,j-1/2}v$$

is the velocity normal to the edge $\vec{h}_{i,j-1/2}$. Using these fluxes in (23.6) gives Godunov's method (the donor-cell upwind (DCU) method).

Alternatively, we can incorporate the length ratios γ into the definitions of the velocities at cell edges, defining *edge velocities*

$$\begin{aligned} U_{i-1/2,j} &= \gamma_{i-1/2,j}\breve{u}_{i-1/2,j}, \\ V_{i,j-1/2} &= \gamma_{i,j-1/2}\breve{v}_{i,j-1/2}. \end{aligned} \tag{23.21}$$

In terms of these velocities we have fluxes

$$\begin{aligned} F_{i-1/2,j} &= U^{+}_{i-1/2,j}Q_{i-1,j} + U^{-}_{i-1/2,j}Q_{ij}, \\ G_{i,j-1/2} &= V^{+}_{i,j-1/2}Q_{i,j-1} + V^{-}_{i,j-1/2}Q_{ij}. \end{aligned} \tag{23.22}$$

This method can also be rewritten in terms of fluctuations, as shown in the next subsection. With this notation, the method looks exactly like the donor-cell method of Section 20.1 on a Cartesian grid for the variable-coefficient advection equation with velocities $U_{i-1/2,j}$ and $V_{i,j-1/2}$. Indeed, the computational ξ–η grid is Cartesian, and we see that the edge velocities give the appropriate velocity field in computational space. Note that in computational space the velocity field typically varies in space, even though we started with a constant-coefficient velocity in physical space.

23.5.1 Fluctuation Form

We have derived the DCU method for advection on a general quadrilateral grid in the form of conservative differencing (23.6), and obtained the numerical fluxes (23.22). We can rewrite this method in the fluctuation form (23.11) following the procedure of Section 23.4. The Riemann problem at $(i-1/2,\ j)$ gives one wave $\mathcal{W}_{i-1/2,j} = Q_{ij} - Q_{i-1,j}$ with speed $\breve{s}^{1}_{i-1/2,j} = \breve{u}_{i-1/2,j}$, the normal velocity at this edge. This speed must be scaled as in (23.14)

to obtain the appropriate speed to be used in computing the fluctuations and later the high-resolution corrections. Doing so results again in the *edge velocities* defined in (23.21), i.e., $s^1 = U$ and $s^2 = V$. We then have the following formulas for the fluctuations:

$$\begin{aligned} \mathcal{A}^{\pm}\Delta Q_{i-1/2,j} &= U^{\pm}_{i-1/2,j}(Q_{ij} - Q_{i-1,j}), \\ \mathcal{B}^{\pm}\Delta Q_{i,j-1/2} &= V^{\pm}_{i,j-1/2}(Q_{ij} - Q_{i,j-1}). \end{aligned} \tag{23.23}$$

We have now derived two distinct approaches to solving the original equation (23.18) on a quadrilateral grid. One approach is in conservative form using fluxes (23.22), and the other is in advective form using the fluctuations (23.23). The original equation (23.18) has constant coefficients, and on a uniform Cartesian grid these two forms would be identical. On the quadrilateral grid, however, the edge velocities are not generally constant, and so it is not clear that the different forms will be identical. In particular, one might question whether the form based on advective fluctuations will yield a conservative method. In fact it does, but this relies on a certain discrete divergence-free condition being automatically satisfied for the normal velocities at each edge,

$$h_{i+1/2,j}\breve{u}_{i+1/2,j} - h_{i-1/2,j}\breve{u}_{i-1/2,j} + h_{i,j+1/2}\breve{v}_{i,j+1/2} - h_{i,j-1/2}\breve{v}_{i,j-1/2} = 0. \tag{23.24}$$

This is zero because it is exactly equal to the integral of $\vec{n} \cdot \vec{u}$ around the boundary of the (i, j) cell and the constant velocity $\vec{u}$ is divergence-free. Using this, we can verify that

$$\Delta\xi\ \Delta\eta \sum_{i,j} \kappa_{ij} Q^{n+1}_{ij} = \Delta\xi\ \Delta\eta \sum_{i,j} \kappa_{ij} Q^{n}_{ij}, \tag{23.25}$$

as required for conservation, by multiplying (23.11) by $\kappa_{ij}\ \Delta\xi\ \Delta\eta$, summing over all grid cells, and rearranging the sums of the correction terms to collect together all terms involving a single Q_{ij}, whose coefficient is then found to be proportional to (23.24) and hence vanishes.

23.5.2 Variable-Coefficient Advection

The constant-coefficient advection equation in physical space generally becomes a variable-coefficient problem in the computational domain, since the normal velocity at an edge varies with the grid orientation. If the original advection equation has variable coefficients in physical space, i.e., a nonconstant velocity field $(u(x, y), v(x, y))$, then we must in general distinguish between the conservative equation and the advective-form color equation. Either of the approaches outlined above (conservative form or fluctuation form) can be extended to varying velocities simply by determining the average normal velocities $\breve{u}_{i-1/2,j}$ and $\breve{v}_{i,j-1/2}$ at each edge based on the varying velocity field, and then proceeding exactly as before. Ideally we would like to define these normal velocities by exactly computing the integrals

$$\breve{u}_{i-1/2,j} = \frac{1}{h_{i-1/2,j}} \int \left[n^x_{i-1/2,j} u(x(s), y(s)) + n^y_{i-1/2,j} v(x(s), y(s)) \right] ds, \tag{23.26}$$

where s is arclength along the edge of the physical grid cell. In general this integral may be difficult to compute, but in the special case where the velocity field is divergence-free and defined by a stream function $\psi(x, y)$ as described in Section 20.8.1, this reduces to a very

simple formula. The velocity is related to the stream function by $u = \psi_y$ and $v = -\psi_x$, and as a result the integrand in (23.26) becomes

$$n^x_{i-1/2,j} u(x(s), y(s)) + n^y_{i-1/2,j} v(x(s), y(s)) = n^x_{i-1/2,j} \psi_y - n^y_{i-1/2,j} \psi_x,$$

which is simply the derivative ψ_s of ψ along the edge. This is a general feature of the stream function: the velocity in any direction is obtained by differentiating the stream function in the orthogonal direction. Now the fundamental theorem of calculus can be applied to the integral in (23.26) to obtain

$$\begin{aligned} \breve{u}^1_{i-1/2,j} &= \frac{1}{h_{i-1/2,j}} \int \psi_s \, ds \\ &= \frac{\psi\left(x_{i-1/2,j+1/2}, y_{i-1/2,j+1/2}\right) - \psi\left(x_{i-1/2,j-1/2}, y_{i-1/2,j-1/2}\right)}{h_{i-1/2,j}}, \end{aligned} \tag{23.27}$$

where the corners of the grid cells are determined by the mapping functions, e.g.,

$$x_{i-1/2,j-1/2} = X\left(\xi_{i-1/2}, \eta_{j-1/2}\right), \qquad y_{i-1/2,j-1/2} = Y\left(\xi_{i-1/2}, \eta_{j-1/2}\right). \tag{23.28}$$

In computing the edge velocity we multiply (23.27) by $\gamma_{i-1/2,j}$, so that the edge length drops out and we obtain

$$U_{i-1/2,j} = \frac{\psi\left(x_{i-1/2,j+1/2}, y_{i-1/2,j+1/2}\right) - \psi\left(x_{i-1/2,j-1/2}, y_{i-1/2,j-1/2}\right)}{\Delta\eta}. \tag{23.29}$$

Similarly,

$$V_{i,j-1/2} = -\frac{\psi\left(x_{i+1/2,j-1/2}, y_{i+1/2,j-1/2}\right) - \psi\left(x_{i-1/2,j-1/2}, y_{i-1/2,j-1/2}\right)}{\Delta\xi}. \tag{23.30}$$

Note that this is the standard approach for handling a stream function on the Cartesian computational grid, as developed in Section 20.8.1. The only new feature introduced is the use of the grid mapping (23.28) in evaluating the stream function at the corners of the computational-grid cells. Differencing the stream function between two corners of this Cartesian grid and dividing by the length of the side gives the average normal velocity.

Example 23.1. We solve a problem with a constant velocity field $u = 1$, $v = 0$ (using the stream function $\psi(x, y) = y$) in a quarter annulus using polar coordinates, as illustrated in Figure 23.3. The mapping function is

$$X(\xi, \eta) = \xi \cos(\eta), \qquad Y(\xi, \eta) = \xi \sin(\eta) \tag{23.31}$$

for $0.5 \le \xi \le 2.5$ and $0 \le \eta \le \pi/2$.

The initial data is $q \equiv 0$, and the boundary data is

$$q(0, y, t) = \begin{cases} 1 & \text{if } 1 < y < 1.5, \\ 0 & \text{otherwise} \end{cases}$$

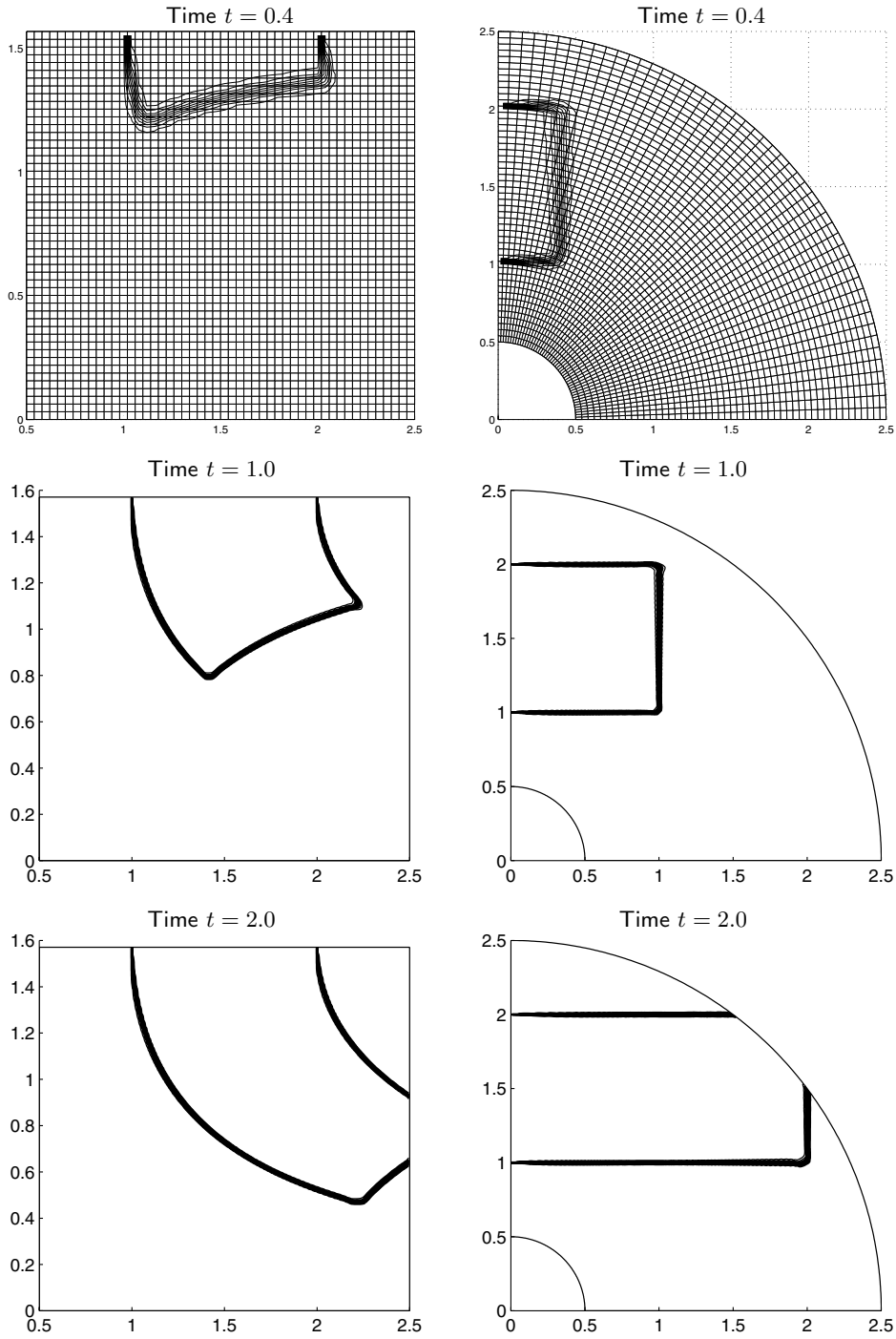

Fig. 23.3. Solution to the advection equation $q_t + q_x = 0$ in polar coordinates, as described in Example 23.1. Left column: on the computational grid. Right column: on the physical grid. Top row: on a 50×50 grid. Later times: on a 200×200 grid. Contour lines are at $q = 0.05, 0.15, \ldots, 0.95$. `[claw/book/chap23/advection/polar]`

with extrapolation at the other boundaries. The solution consists of a band of tracer moving horizontally inward from the left edge. The right-hand column shows the computed solution at three times in the physical domain. The left-hand column shows the solution in the computational domain, where the grid is uniform but the velocity field is not constant. The top row shows that solution at time $t = 0.4$ on a 50×50 grid, along with the grid in both computational and physical space. The results at later times are from a finer-grid computation, on a 200×200 grid.

Note that the top boundary ($\eta = \pi/2$) of the computational grid is mapped to the y-axis in the physical domain, and this is where the boundary conditions must be set. The superbee limiter has been used for this test, and the discontinuity in q is smeared over only two or three grid cells everywhere.

23.6 Acoustics

We now consider the constant-coefficient acoustics equations (18.24) on a quadrilateral grid. We consider the fluctuation approach outlined in Section 23.4. Solving the acoustics equations normal to each cell edge yields two acoustic waves, which update the cell averages on either side. To obtain the proper speeds we must scale the sound speed c by the edge ratios γ as described in Section 23.4. The main new complication that arises with acoustics (and also with fluid dynamic equations such as the shallow water or Euler equations) is that the solution q is no longer a scalar but rather a vector $q = (p, u, v)$, which contains the velocity as part of the solution. In each grid cell we store the Cartesian components of the velocity vector, i.e., $Q^2_{ij} = u_{ij}$ is the x-component of velocity while $Q^3_{ij} = v_{ij}$ is the y-component. To solve the acoustics equations normal to an edge we need to know the components of velocity normal and tangential to that edge in each of the two grid cells on either side. The normal velocity components, along with the pressure in each cell, determine the resulting acoustic waves. Any jump in the tangential velocity remains stationary at the cell edge. This jump constitutes the shear wave moving with speed zero that arises in two-dimensional acoustics (with background velocity $\vec{u}_0 = (u_0, v_0) = (0, 0)$ in the derivation of Section 18.2) and can be ignored in the finite volume method, since it does not affect the cell average to either side. For clarity in the discussion below and consistency of notation, we include formulas for all three waves. But in an implementation only the 1-wave and 3-wave need to be propagated. (If we solved the equations with nonzero $\vec{u}_0$, then the jump in the tangential velocity would propagate at a velocity $\vec{n} \cdot \vec{u}_0$ and a third wave with this speed would be needed, in addition to the acoustic waves propagating at speeds $(\vec{n} \cdot \vec{u}_0) \pm c$.)

23.6.1 Solving the Normal Riemann Problem

To determine the waves, speeds, and fluctuations at edge $(i - 1/2, j)$, for example, we must solve the Riemann problem for a one-dimensional system of equations normal to the interface, with data $Q_{i-1,j}$ and Q_{ij}. We will consider two possible approaches to solving the Riemann problem in the direction $\vec{n}_{i-1/2,j}$, which give equivalent results. One approach is based directly on the theory of Chapter 18: we solve the one-dimensional Riemann problem with coefficient matrix $\breve{A} = \vec{n}_{i-1/2,j} \cdot \vec{A}$ and data $Q_{i-1,j}$ and Q_{ij}. We first consider this approach.

The second approach, illustrated for acoustics in Section 23.6.3, consists of explicitly determining the normal and tangential components of velocity from $Q_{i-1,j}$ and Q_{ij}, relabeling these as the "x" and "y" components, and then solving the Riemann problem $q_t + Aq_x = 0$ in the "x" direction with this modified data. The velocity components of the resulting waves must then be recombined to obtain the proper updates in physical space. This may be conceptually easier to implement in cases where a complicated Riemann solver has already been written for Cartesian coordinates. This approach only works for isotropic equations, but applies to many physically relevant problems in fluid or solid mechanics, including the Euler and shallow water equations as well as acoustics.

For the acoustics equations we can directly solve a system of the form $q_t + \breve{A}_{i-1/2,j} q_n = 0$, where q_n denotes the derivative in the normal direction and, from (18.28),

$$\begin{aligned}\breve{A}_{i-1/2,j} &= n^x_{i-1/2,j} A + n^y_{i-1/2,j} B \\ &= \begin{bmatrix} 0 & n^x_{i-1/2,j} K_0 & n^y_{i-1/2,j} K_0 \\ n^x_{i-1/2,j}/\rho_0 & 0 & 0 \\ n^y_{i-1/2,j}/\rho_0 & 0 & 0 \end{bmatrix}. \end{aligned} \tag{23.32}$$

For clarity we drop the subscript $i - 1/2$, j from n below, and also from the eigenvalues, eigenvectors, and α-coefficients. From Section 18.5 we know that this matrix has eigenvalues and eigenvectors

$$\begin{aligned} &\breve{\lambda}^1 = -c_0, \qquad \breve{\lambda}^2 = 0, \qquad \breve{\lambda}^3 = c_0, \\ &\breve{r}^1 = \begin{bmatrix} -Z_0 \\ n^x \\ n^y \end{bmatrix}, \quad \breve{r}^2 = \begin{bmatrix} 0 \\ -n^y \\ n^x \end{bmatrix}, \quad \breve{r}^3 = \begin{bmatrix} Z_0 \\ n^x \\ n^y \end{bmatrix}. \end{aligned} \tag{23.33}$$

In fact we can just as easily solve the Riemann problem for the variable-coefficient case where the impedance Z_{ij} and sound speed c_{ij} have different values in each grid cell, and so the more general formulas are presented. In this case we wish to decompose $\delta \equiv Q_{ij} - Q_{i-1,j}$ into waves using

$$\begin{aligned} &\breve{\lambda}^1 = -c_{i-1,j}, \qquad \breve{\lambda}^2 = 0, \qquad \breve{\lambda}^3 = c_{ij}, \\ &\breve{r}^1 = \begin{bmatrix} -Z_{i-1,j} \\ n^x \\ n^y \end{bmatrix}, \quad \breve{r}^2 = \begin{bmatrix} 0 \\ -n^y \\ n^x \end{bmatrix}, \quad \breve{r}^3 = \begin{bmatrix} Z_{ij} \\ n^x \\ n^y \end{bmatrix}. \end{aligned} \tag{23.34}$$

Writing δ as a linear combination of these vectors yields the waves

$$\mathcal{W}^p = \alpha^p \breve{r}^p \quad \text{for } p = 1, 2, 3, \tag{23.35}$$

where

$$\begin{aligned}
\alpha^1 &= \frac{-\delta^1 + n^x Z_{i-1,j}\delta^2 + n^y Z_{i-1,j}\delta^3}{Z_{i-1,j} + Z_{ij}},\\
\alpha^2 &= -n^y\delta^2 + n^x\delta^3,\\
\alpha^3 &= \frac{\delta^1 + n^x Z_{ij}\delta^2 + n^y Z_{ij}\delta^3}{Z_{i-1,j} + Z_{ij}}.
\end{aligned} \tag{23.36}$$

The wave speeds are obtained by scaling $\breve{\lambda}^1$ and $\breve{\lambda}^3$ as described above:

$$\begin{aligned}
s^1 &= -\gamma_{i-1/2,j} c_{i-1,j},\\
s^2 &= 0,\\
s^3 &= \gamma_{i-1/2,j} c_{ij}.
\end{aligned} \tag{23.37}$$

The fluctuations are then given by

$$\begin{aligned}
\mathcal{A}^-\Delta Q &= s^1\mathcal{W}^1,\\
\mathcal{A}^+\Delta Q &= s^3\mathcal{W}^3,
\end{aligned} \tag{23.38}$$

since $s^1 < 0$ while $s^3 > 0$. All of these quantities should be indexed by $i - 1/2, j$. The resulting waves and speeds are also appropriate for use in the high-resolution correction terms, along with limiters in the usual form. This Riemann solver and an example may be found at `[claw/book/chap23/acoustics]`.

23.6.2 Transverse Riemann Solvers

The normal Riemann solver described above is all that is needed for a dimensionally split method. To use the unsplit methods developed in Chapter 21, we must also define transverse Riemann solvers. For the acoustics equations we start with a fluctuation, say $\mathcal{A}^+\Delta Q_{i-1/2,j}$, and must determine $\mathcal{B}^-\mathcal{A}^+\Delta Q_{i-1/2,j}$ and $\mathcal{B}^+\mathcal{A}^+\Delta Q_{i-1/2,j}$. These correspond to the portion of this fluctuation which should be transmitted through the edges "above" and "below" this cell, respectively. On a Cartesian grid we were able to determine both of these by simply splitting $\mathcal{A}^+\Delta Q$ into up-going and down-going acoustic waves in the y-direction. On a general quadrilateral grid, however, the edges "above" and "below" the cell will not be parallel, and we must solve two different transverse Riemann problems in the appropriate directions to compute the two fluctuations $\mathcal{B}^+\mathcal{A}^+\Delta Q$ and $\mathcal{B}^-\mathcal{A}^+\Delta Q$. In the case of a heterogeneous medium we had to do this anyway, since the coefficients defining the wave speeds and eigenvectors may be different in the cell "above" and "below," so this is a natural generalization of the transverse solver presented in Section 21.5.1 for a Cartesian grid.

To compute $\mathcal{B}^-\mathcal{A}^+\Delta Q$, for example, we must decompose the vector $\delta \equiv \mathcal{A}^+\Delta Q_{i-1/2,j}$ into eigenvectors corresponding to a Riemann problem in the $\vec{n}_{i,j-1/2}$ direction at the

interface $(i, j-1/2)$:

$$\delta \equiv \mathcal{A}^+ \Delta Q_{i-1/2,j} = \beta^1 \begin{bmatrix} -Z_{i,j-1} \\ n^x_{i,j-1/2} \\ n^y_{i,j-1/2} \end{bmatrix} + \beta^2 \begin{bmatrix} 0 \\ -n^y_{i,j-1/2} \\ n^x_{i,j-1/2} \end{bmatrix} + \beta^3 \begin{bmatrix} Z_{ij} \\ n^x_{i,j-1/2} \\ n^y_{i,j-1/2} \end{bmatrix}. \quad (23.39)$$

Solving this linear system gives

$$\beta^1 = \frac{-\delta^1 + \delta^2 n^x Z_{ij} + \delta^3 n^y Z_{ij}}{Z_{i,j-1} + Z_{ij}}. \quad (23.40)$$

The coefficient β^1 is the only one needed to compute the down-going fluctuation, which is obtained by multiplying the first wave in (23.39) by the physical speed $-c_{i,j-1}$ and by the geometric scaling factor $\gamma_{i,j-1/2}$:

$$\mathcal{B}^- \mathcal{A}^+ \Delta Q_{i-1/2,j} = -\gamma_{i,j-1/2}\, c_{i,j-1}\, \beta^1 \begin{bmatrix} -Z_{i,j-1} \\ n^x_{i,j-1/2} \\ n^y_{i,j-1/2} \end{bmatrix}. \quad (23.41)$$

The up-going fluctuation $\mathcal{B}^+ \mathcal{A}^+ \Delta Q_{i-1/2,j}$ and the fluctuations $\mathcal{B}^\pm \mathcal{A}^- \Delta Q_{i-1/2,j}$ are obtained by analogous decompositions.

23.6.3 Solving the Riemann Problem by Rotating the Data

As described at the beginning of Section 23.6.1, there is an alternative approach to solving the normal Riemann problem in direction $\vec{n}_{i-1/2,j}$ that may by easier to implement for some systems. Rather than computing the eigenstructure of the Jacobian matrix in an arbitrary direction, we can instead transform the data $Q_{i-1,j}$ and Q_{ij} into new data $\breve{Q}_l$ and $\breve{Q}_r$ for a Riemann problem in the "x" direction of the form $\breve{q}_t + A\breve{q}_x = 0$. For acoustics, the data $\breve{Q}_l$ and $\breve{Q}_r$ should have components

$$\begin{aligned} \breve{Q}^1_{l,r} &= \text{ pressure}, \\ \breve{Q}^2_{l,r} &= \text{ normal velocity}, \\ \breve{Q}^3_{l,r} &= \text{ transverse velocity}. \end{aligned} \quad (23.42)$$

We solve this one-dimensional Riemann problem and then must rotate the velocity components in the resulting waves and fluctuations back into the proper physical direction. This works because the acoustics equations are isotropic and so the Riemann problem has exactly the same mathematical structure in any direction. It also works for systems such as the Euler and shallow water equations; see Section 23.7.

To determine the waves, speeds, and fluctuations at edge $(i-1/2, j)$, we must do the following:

1. Determine $\check{Q}_l$ and $\check{Q}_r$ by rotating the velocity components:

$$\check{Q}_l = \mathcal{R}_{i-1/2,j} Q_{i-1,j}, \qquad \check{Q}_r = \mathcal{R}_{i-1/2,j} Q_{ij},$$

where

$$\mathcal{R}_{i-1/2,j} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & n^x_{i-1/2,j} & n^y_{i-1/2,j} \\ 0 & -n^y_{i-1/2,j} & n^x_{i-1/2,j} \end{bmatrix}. \tag{23.43}$$

This matrix leaves the pressure Q^1 unchanged and rotates the velocity components of Q into components normal and tangential to the edge.

2. Solve the one-dimensional acoustics equations with

$$A = \begin{bmatrix} 0 & K & 0 \\ 1/\rho & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

and the Riemann data $\check{Q}_l$ and $\check{Q}_r$. As in Section 23.6.1, this can be done equally well with different material parameters K and ρ (and hence impedance Z) in each grid cell. This results in the waves

$$\check{\mathcal{W}}^1_{i-1/2,j} = \alpha^1 \begin{bmatrix} -Z_{i-1,j} \\ 1 \\ 0 \end{bmatrix}, \quad \check{\mathcal{W}}^2_{i-1/2,j} = \alpha^2 \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \quad \check{\mathcal{W}}^3_{i-1/2,j} = \alpha^3 \begin{bmatrix} Z_{ij} \\ 1 \\ 0 \end{bmatrix},$$

with speeds $\check{\lambda}^1_{i-1/2,j} = -c_{i-1,j}$, $\check{\lambda}^2_{i-1/2,j} = 0$, and $\check{\lambda}^3_{i-1/2,j} = c_{ij}$.

3. Scale the wave speeds,

$$s^1_{i-1/2,j} = -\gamma_{i-1/2,j} c_{i-1,j}, \quad s^2_{i-1/2,j} = 0, \quad s^3_{i-1/2,j} = \gamma_{i-1/2,j} c_{ij}.$$

4. The waves $\check{\mathcal{W}}^p_{i-1/2,j}$ carry jumps in the normal velocity and no jump in the tangential velocity. To update the cell average Q properly, these must be converted into jumps in the x- and y-components of velocity, since this is what we store in Q. This is accomplished by setting

$$\mathcal{W}^p_{i-1/2,j} = \mathcal{R}^{-1}_{i-1/2,j} \check{\mathcal{W}}^p_{i-1/2,j}$$

for $p = 1, 2, 3$, where

$$\mathcal{R}^{-1}_{i-1/2,j} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & n^x_{i-1/2,j} & -n^y_{i-1/2,j} \\ 0 & n^y_{i-1/2,j} & n^x_{i-1/2,j} \end{bmatrix}. \tag{23.44}$$

5. Compute the fluctuations

$$\begin{aligned} \mathcal{A}^- \Delta Q_{i-1/2,j} &= s^1_{i-1/2,j} \mathcal{W}^1_{i-1/2,j}, \\ \mathcal{A}^+ \Delta Q_{i-1/2,j} &= s^3_{i-1/2,j} \mathcal{W}^3_{i-1/2,j}, \end{aligned} \tag{23.45}$$

since $s^1 < 0$ while $s^3 > 0$.

This procedure results in the same waves, speeds, and fluctuations as the approach described in Section 23.6.1.

The transverse Riemann solver of Section 23.6.2 can also be reformulated via rotation of the data. To compute $\mathcal{B}^- \mathcal{A}^+ \Delta Q$, for example, we rotate the velocity components of $\mathcal{A}^+ \Delta Q$ into directions normal and tangential to the edge $(i, j-1/2)$ by computing $\mathcal{R}_{i,j-1/2} \mathcal{A}^+ \Delta Q$. The first and second components of the resulting vector give the fluctuations in the pressure and normal velocity. These can be used to determine up-going and down-going acoustic waves, and the down-going wave is the one that is used to compute $\mathcal{B}^- \mathcal{A}^+ \Delta Q$.

We decompose

$$\mathcal{R}_{i,j-1/2} \mathcal{A}^+ \Delta Q = \beta^1 \begin{bmatrix} -Z_{i,j-1} \\ 1 \\ 0 \end{bmatrix} + \beta^2 \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} + \beta^3 \begin{bmatrix} Z_{ij} \\ 1 \\ 0 \end{bmatrix}.$$

It is the first of these three waves that is propagating downwards. On a Cartesian grid we would multiply this wave by the corresponding speed $c_{i,j-1}$ to obtain the fluctuation $\mathcal{B}^- \mathcal{A}^+ \Delta Q$. On the quadrilateral grid we must first rotate the velocity components of this wave back to x–y coordinates (multiplying by $\mathcal{R}^{-1}_{i,j-1/2}$), and then we must multiply by the *scaled* velocity $\gamma_{i,j-1/2} c_{i,j-1}$, so we obtain

$$\mathcal{B}^- \mathcal{A}^+ \Delta Q_{i-1/2,j} = \gamma_{i,j-1/2} c_{i,j-1} \mathcal{R}^{-1}_{i,j-1/2} \beta^1 \begin{bmatrix} -Z_{i,j-1} \\ 1 \\ 0 \end{bmatrix}. \tag{23.46}$$

To compute $\mathcal{B}^+ \mathcal{A}^+ \Delta Q$ we take a similar approach, but now must rotate using $\mathcal{R}_{i,j+1/2}$ and use the up-going wave, rotating back and scaling by $\gamma_{i,j+1/2}$. The left-going fluctuation $\mathcal{A}^- \Delta Q_{i-1/2,j}$ must then be handled in a similar manner.

23.7 Shallow Water and Euler Equations

The procedure described in Section 23.6 for the acoustics equations is easily extended to nonlinear systems of equations such as the two-dimensional shallow water equations. Recall that for these equations the components of q are (h, hu, hv), where h is the depth. As in the acoustics equations, we must rotate the momentum components at each cell interface using the rotation matrix (23.43). We then solve the one-dimensional Riemann problem normal to the edge just as on a Cartesian grid. An approximate Riemann solver can again be used. The waves are then rotated back into the x–y coordinate frame before using them to update Q in the grid cells, and the wave speeds are scaled by the length ratios γ.

For the unsplit algorithms of Chapter 21, we also need to define a transverse solver. As in Section 23.6.2, we need to rotate the momentum components of $\mathcal{A}^+ \Delta Q_{i-1/2,j}$ into directions normal and tangential to the edge $(i, j-1/2)$ before decomposing this vector into eigenvectors of the transverse Jacobian matrix. In addition, we need to define this Jacobian matrix using appropriate velocities relative to this edge. Just as we did for the Cartesian grid algorithm described in Section 21.7, we might use the Roe averages defined from solving the normal Riemann problem to define the transverse Jacobian, after a suitable rotation. Sample Riemann solvers for this case may be found in `[claw/book/chap23/shallow]`, and an example is presented in Example 23.2 below.

The Euler equations can be handled in exactly the same manner. The components of q are now $(\rho, \rho u, \rho v, E)$ and the energy E is a scalar, so that rotations are again applied only to the momentum components of q. Sample Riemann solvers for this case may be found in `[claw/book/chap23/euler]`.

23.8 Using CLAWPACK on Quadrilateral Grids

The sample CLAWPACK codes for this chapter all take a common form. The domain is assumed to be rectangular in the computational ξ–η plane, and it is this plane that is discretized with a uniform grid. So the parameters `dx` and `dy` in `claw2ez.data` now refer to $\Delta\xi$ and $\Delta\eta$, while the parameters `xlower`, etc., specify the range in ξ–η space.

A function `mapc2p.f` is specified that maps a computational point `(xc,yc)` $= (\xi, \eta)$ to the corresponding physical point `(xp,yp)` $= (X(\xi, \eta),\ Y(\xi, \eta))$. This mapping is used to determine the physical location of the corners of grid cells, which are needed in the `setaux.f` routine as described below. This mapping is also typically called from the `qinit.f` routine to set initial data. For each grid cell we wish to set `q(i,j)` initially to be the cell average of the initial data $\mathring{q}(x, y)$. Often one can approximate this by simply evaluating $\mathring{q}$ at the center of the grid cell, and `mapc2p` is used to map the center of the computational grid cell, `(xc,yc)` = `xlower` + `(i-0.5)*dx, ylower` + `(j-0.5)*dy)`, to the corresponding physical point `(xp,yp)`.

The `aux` array is used to store information required for each grid cell or edge. This can typically be set once at the start of the computation in the `setaux.f` routine. Exactly what information is required depends on the equations being solved, but we always require $\kappa_{ij} = |\mathcal{C}_{ij}|/(\Delta\xi\ \Delta\eta)$, which must be used as a capacity function (so `mcapa` should be set to point to this element of the `aux` array). This value can be computed from the locations of the four corners of the cell.

For the advection equation as discussed in Section 23.5.2, the only other information required is the edge velocities $U_{i-1/2,j}$ and $V_{i,j-1/2}$. If a stream function is available, then these are easily computed by differencing the stream function between the corners. See `[claw/book/chap23/advection/polar/setaux.f]` for some sample formulas.

For the acoustics, Euler, or shallow water equations, we need to store enough information to determine both the normal vector to each edge and the length ratios γ for each side. This requires a minimum of two pieces of data at each side, for example the angle of the normal and the γ-value, or else the edge vector $\vec{h}$ (see Figure 23.2), from which both the normal and the length can be computed. It is more efficient, however, to store three pieces of data

at each edge, both components of the normal vector $\vec{n}$ and also the γ value, in order to minimize the amount of computation that must be done in each time step. The CLAWPACK codes provided for this chapter use the following convention for the aux array:

$$\begin{aligned}
\texttt{aux(i, j, 1)} &= n^x_{i-1/2,j},\\
\texttt{aux(i, j, 2)} &= n^y_{i-1/2,j},\\
\texttt{aux(i, j, 3)} &= \gamma_{i-1/2,j},\\
\texttt{aux(i, j, 4)} &= n^x_{i,j-1/2},\\
\texttt{aux(i, j, 5)} &= n^y_{i,j-1/2},\\
\texttt{aux(i, j, 6)} &= \gamma_{i,j-1/2},\\
\texttt{aux(i, j, 7)} &= \kappa_{ij}.
\end{aligned} \tag{23.47}$$

For each (i, j), we store data related to the left edge and the bottom edge as well as the κ-value for this cell.

Example 23.2. As an example, we solve the two-dimensional shallow water equations around a circular cylinder and compare results obtained on two different types of grids, as illustrated in Figure 23.1(a) and (b). The simple polar coordinate grid of Figure 23.1(a) has the advantage that it is smoothly varying and orthogonal. The grid of Figure 23.1(b) was chosen to illustrate that these methods work well even on nonorthogonal grids and with abrupt changes in the orientation of grid lines.

The problem we consider consists of a planar shock wave moving towards the cylinder, with $h_r = 1$ and $u_r = 0$ ahead of the shock (so there is initially quiescent water around the cylinder). Behind the shock, $h_l = 4$, and the velocity $u_l > 0$ is determined by the Hugoniot relation (13.17). Common experience from inserting a stick vertically into a moving stream of water leads us to expect that a *bow shock* will form upstream from the cylinder once the flowing water hits the cylinder. The depth jumps to a value greater than the approaching freestream depth in this region where the flow must decelerate and go around the cylinder.

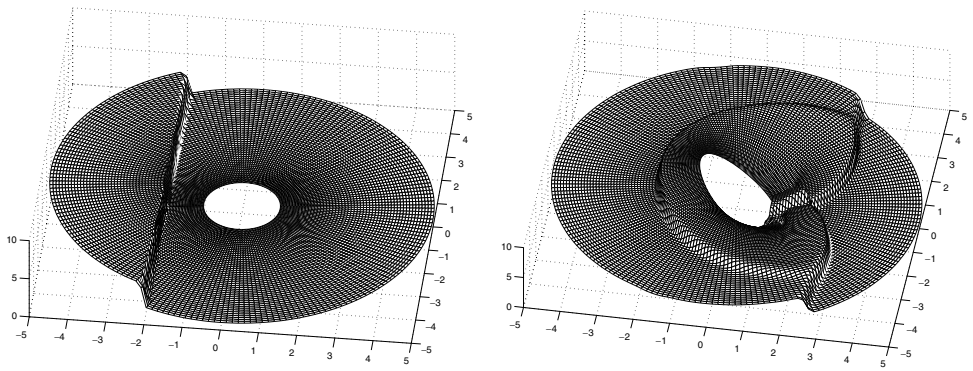

Fig. 23.4. A hydraulic jump in the shallow water equations hits a circular cylinder. The depth at two different times is shown. Left: depth at $t = 0$; right: depth at $t = 1.5$.

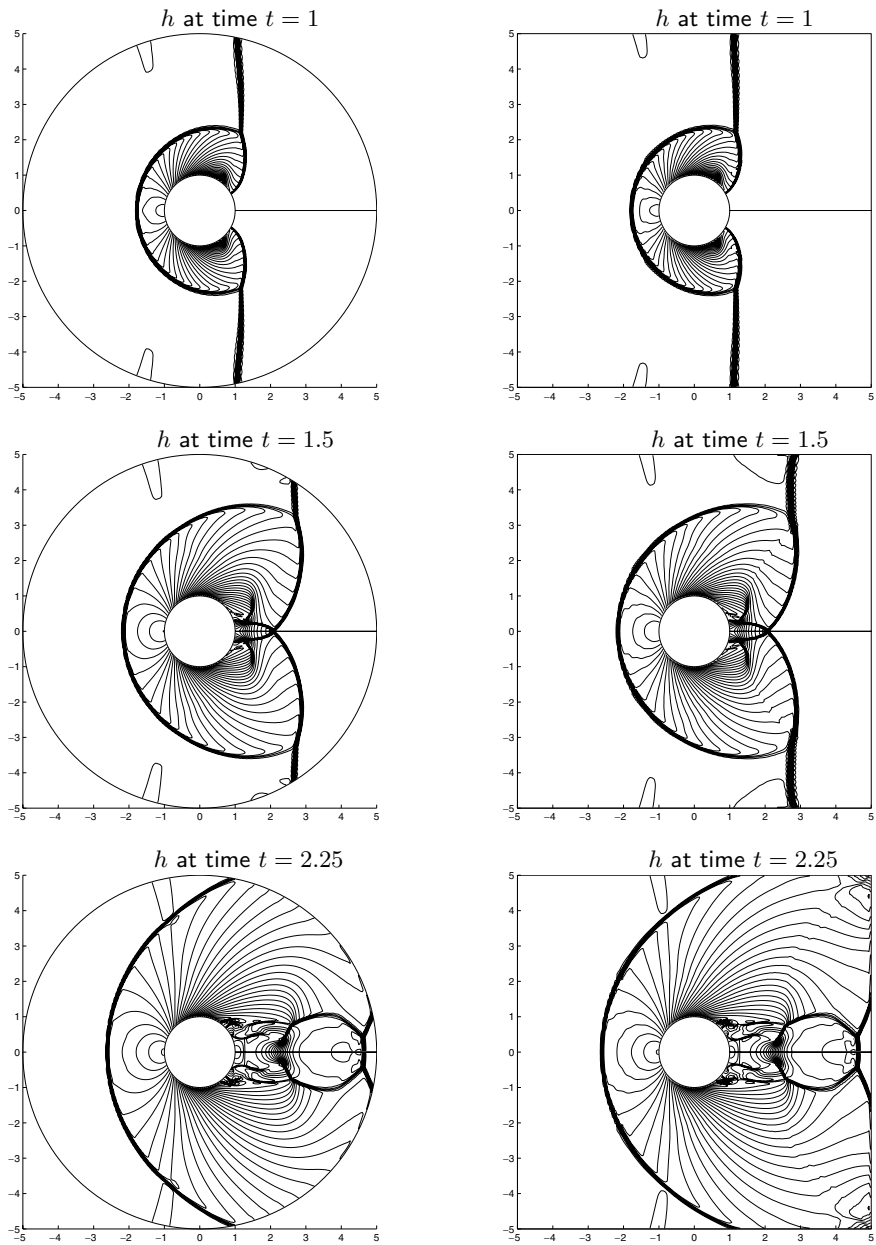

Fig. 23.5. Shallow water equations for a shock wave hitting a cylinder. On the left a 100 × 400 polar coordinate grid of the type shown in Figure 23.1(a) is used. On the right a 100 × 416 grid of the type shown in Figure 23.1(b) is used. Contour lines are at 0.05:0.2:8.05. `[claw/book/chap23/shallow/cylinder]` `[claw/book/chap23/shallow/sqcylinder]`

In the present context this can be viewed as a reflection of the incident shock from the cylinder. Figure 23.4 shows the depth initially and at a later time.

Figure 23.5 shows contour plots of computed solutions on the two grids similar to those illustrated in Figure 23.1(a) and (b), but with greater resolution. The contour lines are

slightly smoother and better resolved on the smooth polar grid, especially along the 45° lines, but the main features are captured equally well on both grids.

23.9 Boundary Conditions

Boundary conditions on quadrilateral grids can be handled in essentially the same manner as on Cartesian grids, since the computational grid is still rectangular. Ghost-cell values must be determined as described in Chapter 7, so that solving the hyperbolic problem over a slightly enlarged domain automatically leads to the proper boundary behavior. Periodic and extrapolation boundary conditions are unchanged from the Cartesian case.

Solid-wall boundary conditions must be modified to allow for the fact that the normal component of velocity should be zero at the wall. In Example 23.2 above, where flow around a cylinder is considered, the cylinder corresponds to the boundary $\xi = 0$ at cell interfaces $(1/2, j)$. To determine the ghost-cell values Q_{0j} and $Q_{-1,j}$ we must use the normal vector $\vec{n}_{1/2,j}$ to compute the normal component of velocity. Using the rotation matrix $\mathcal{R}_{1/2,j}$ defined as in (23.43), we can compute

$$\breve{Q} = \mathcal{R}_{1/2,j} Q_{ij} \quad \text{for } i = 1, 2,$$

whose components are now the depth and the velocity components normal and tangential to the boundary. If the boundary is flat, then we can compute ghost-cell values using the same approach as on the Cartesian grid, reflecting the data across the boundary and negating the normal component of velocity. We accomplish this by setting

$$\breve{Q}_{1-i,j} = \begin{bmatrix} \breve{Q}^1_{ij} \\ -\breve{Q}^2_{ij} \\ \breve{Q}^3_{ij} \end{bmatrix} \quad \text{for } i = 1, 2. \tag{23.48}$$

These are rotated back to Cartesian components of velocity,

$$Q_{1-i,j} = \mathcal{R}^{-1}_{1/2,j} \breve{Q}_{1-i,j} \quad \text{for } i = 1, 2,$$

to obtain the desired ghost-cell values. This process yields

$$\begin{aligned} Q^1_{1-i,j} &= Q_{ij}, \\ Q^2_{1-i,j} &= [(n^y)^2 - (n^x)^2] Q^2_{ij} - 2n^x n^y Q^3_{ij}, \\ Q^3_{1-i,j} &= -2n^x n^y Q^2_{ij} + [(n^y)^2 - (n^x)^2] Q^3_{ij}. \end{aligned} \tag{23.49}$$