7

Boundary Conditions and Ghost Cells

So far we have only studied methods for updating the cell average Q_i^n assuming that we have neighboring cell values Q_{i-1}^n and Q_{i+1}^n and perhaps values further away as needed in order to compute the fluxes $F_{i-1/2}^n$ and $F_{i+1/2}^n$. In practice we must always compute on some finite set of grid cells covering a bounded domain, and in the first and last cells we will not have the required neighboring information. Instead we have some set of physical boundary conditions, as discussed in Section 3.11, that must be used in updating these cell values.

One approach is to develop special formulas for use near the boundaries, which will depend both on what type of boundary conditions are specified and on what sort of method we are trying to match. However, in general it is much easier to think of extending the computational domain to include a few additional cells on either end, called *ghost cells*, whose values are set at the beginning of each time step in some manner that depends on the boundary conditions and perhaps the interior solution.

Figure 7.1 shows a grid with two ghost cells at each boundary. These values provide the neighboring-cell values needed in updating the cells near the physical domain. The updating formula is then exactly the same in all cells, and there is no need to develop a special flux-limiter method, say, that works with boundary data instead of initial data. Instead the boundary conditions must be used in deciding how to set the values of the ghost cells, but this can generally be done in a way that depends only on the boundary conditions and is decoupled entirely from the choice of numerical method that is then applied.

Suppose the problem is on the physical domain $[a, b]$, which is subdivided into cells $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_N$ with $x_1 = a$ and $x_{N+1} = b$, so that $\Delta x = (b - a)/N$. If we use a method for which $F_{i-1/2}^n$ depends only on Q_{i-1}^n and Q_i^n, then we need only one ghost cell on either end. The ghost cell $\mathcal{C}_0 = (a - \Delta x, a)$ allows us to calculate the flux $F_{1/2}$ at the left boundary while the ghost cell $\mathcal{C}_{N+1} = (b, b + \Delta x)$ is used to calculate $F_{N+1/2}^n$ at $x = b$. With a flux-limiter method of the type developed in Chapter 6, we will generally need two ghost cells at each boundary, since, for example, the jump $Q_0 - Q_{-1}$ will be needed in limiting the flux correction in $F_{1/2}$. For a method with an even wider stencil, additional ghost cells would be needed. In general we might have m_{BC} ghost cells at each side.

We will refer to the solution in the original domain $[a, b]$ as the *interior solution*; it is computed in each time step by the numerical method. At the start of each time step we have the interior values $Q_1^n, \ldots, Q_N^n$ obtained from the previous time step (or from the initial conditions if $n = 0$), and we apply a *boundary-condition procedure* to fill the ghost cells with values Q_i^n for $i = 1 - m_{\text{BC}}, \ldots, 0$ and $i = N + 1, \ldots, N + m_{\text{BC}}$ before applying the

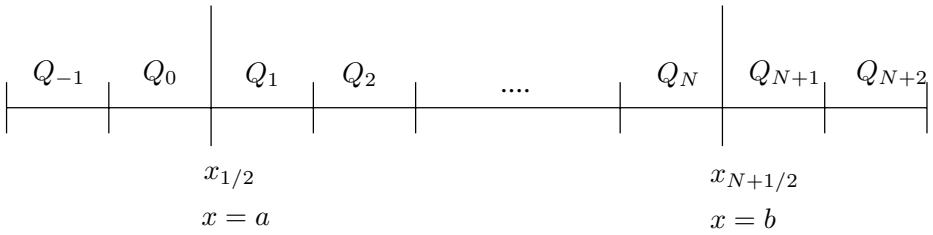

Fig. 7.1. The computational domain $[a, b]$ is extended to a set of ghost cells for specifying boundary conditions.

method on the next time step. In CLAWPACK this is done in a subroutine `bc1.f` that is called at the beginning of each time step. The default routine `claw/clawpack/1d/lib/bc1` implements most of the particular boundary conditions discussed in this chapter. We will look at several examples to see how the ghost-cell values might be set in order to implement various physical boundary conditions. In general we will discuss the boundary conditions for the case $m_{\text{BC}} = 2$, but they can easily be extended to larger values if necessary.

7.1 Periodic Boundary Conditions

Periodic boundary conditions $q(a, t) = q(b, t)$ are very easy to impose with any numerical method. In updating Q_1 we need values Q_0 to the left and Q_2 to the right (for a three-point method). By periodicity the value Q_0 should agree with the value Q_N in the last cell. One could code the formula for updating Q_1 specially to use Q_N in place of the value Q_{i-1} that would normally be used for $i > 1$, but it is simpler to use the ghost-cell approach and simply set $Q_0^n = Q_N^n$ before computing fluxes and updating the cell values, so that the same formula can then be used everywhere. With a five-point stencil we need to fill two ghost cells at each boundary, and we set

$$Q_{-1}^n = Q_{N-1}^n, \quad Q_0^n = Q_N^n, \quad Q_{N+1}^n = Q_1^n, \quad Q_{N+2}^n = Q_2^n \tag{7.1}$$

at the start of each time step. These boundary conditions are implemented in the CLAWPACK routine `claw/clawpack/1d/lib/bc1.f` and invoked by setting `mthbc(1) = 2` and `mthbc(2) = 2`.

7.2 Advection

Consider the advection equation on $[a, b]$ with $\bar{u} > 0$ and the boundary condition

$$q(a, t) = g_0(t), \tag{7.2}$$

where $g_0(t)$ is a given function. Recall from Section 3.11 that we cannot specify a boundary condition at $x = b$. We may need to specify a boundary condition at $x = b$ for the numerical method, however, if the stencil for computing Q_i^{n+1} involves points to the right of x_i.

7.2.1 Outflow Boundaries

First we consider the outflow boundary at $x = b$. If we use a one-sided method such as upwind or Beam–Warming, then we do not need any numerical boundary condition. If we

implement the method using ghost cells, then we can assign arbitrary values to the ghost cells on the right with no effect on the interior solution, since these values will never be used.

If we use a three-point method such as Lax–Wendroff that does use values to the right, then some numerical boundary conditions must be specified. We must in general take some care in how we specify these to obtain a stable and accurate method.

One possibility is to use a fully upwind method at the rightmost point, together with the Lax–Wendroff method at all other points. This works quite well. Note, however, that the Lax–Wendroff method allows information to propagate from right to left, even though the exact solution to the advection equation does not. So there is the possibility that noise generated at the right boundary by this switch in method will propagate back into the domain and contaminate the solution elsewhere, and perhaps even cause an instability. The Lax–Wendroff method is highly dissipative for left-going waves, and so this does not cause a noticeable problem, and one can prove that the resulting method is stable. Indeed, a similar change in method occurs frequently with flux-limiter methods, with only slight loss in accuracy.

In general, the theory of stability for the IBVP is much more subtle and difficult than for the Cauchy problem. See, for example, [427], [459] for an introduction to this topic.

Rather than explicitly switching to a different formula at the right boundary, we can achieve the same effect by setting ghost cell values by *extrapolation* from the interior solution. If the ghost-cell value Q^n_{N+1} is set based on Q^n_N, $Q^n_{N-1}, \ldots$, then the new value Q^{n+1}_N will effectively be computed on the basis of values to the left alone, even if the formula depends on Q^n_{N+1}, and hence this reduces to some sort of upwind method. The simplest approach is to use a *zero-order* extrapolation, meaning extrapolation by a constant function. We simply set

$$Q^n_{N+1} = Q^n_N, \qquad Q^n_{N+2} = Q^n_N \tag{7.3}$$

at the start of each time step. Then we have $\Delta Q^n_{N+1} = 0$ as the value $\delta^n_{N+1/2}$ used in the flux modification to $F^n_{N+1/2}$ in (6.32). This flux then reduces to the upwind flux,

$$F^n_{N+1/2} = \bar{u} Q^n_N,$$

since $\bar{u} > 0$. Note that the method may not reduce to the standard first-order upwind method in the last cell, however, since $\delta^n_{N-1/2}$ ($= \Delta Q^n_{N-1/2} = Q^n_N - Q^n_{N-1}$ for the Lax–Wendroff method) may be nonzero. But the resulting method behaves in roughly the same manner.

One might also consider *first-order* extrapolation, based on fitting a linear function through the interior solution. This gives

$$Q^n_{N+1} = Q^n_N + \left(Q^n_N - Q^n_{N-1}\right) = 2Q^n_N - Q^n_{N-1}. \tag{7.4}$$

In this case $\Delta Q^n_{N+1/2} = \Delta Q^n_{N-1/2}$ and the correction terms in $F^n_{N-1/2}$ and $F^n_{N+1/2}$ will cancel out (assuming the limiter function satisfies $\phi(1) = 1$). Now the update for Q^{n+1}_n does reduce to the standard first-order upwind method.

The idea of extrapolation at outflow boundaries turns out to be extremely powerful more generally, even for systems of equations where there are both incoming and outgoing characteristics. Often we have artificial computational boundaries that arise simply because we can only solve the problem on a bounded domain. At such boundaries we often want to have no

incoming signal, though there may be outgoing waves that should leave the domain cleanly without generating spurious reflections at the artificial boundary. Zero-order extrapolation, coupled with the methods we are studying that perform characteristic decomposition in the solution to each Riemann problem, is often a very effective way to accomplish this. This is discussed in Section 7.3.1 for linear acoustics and will be investigated later for nonlinear and multidimensional problems. First-order extrapolation might seem even better, but can lead to stability problems and is not recommended in general.

Zero-order extrapolation boundary conditions are implemented in CLAWPACK as one of the options that can be automatically invoked when using `claw1ez` as described in Chapter 5. The ghost-cell values are set in `claw/clawpack/1d/lib/bc1.f` and invoked by setting `mthbc(i) = 1`, where `i=1` for the left boundary or `i=2` for the right boundary. (See Section 5.4.6.)

7.2.2 Inflow Boundary Conditions

Now consider the inflow boundary at $x = a$ for the advection equation, where we specify the boundary condition (7.2). One approach would be to compute the exact flux $F^n_{1/2}$ by integrating along the boundary,

$$\begin{aligned} F^n_{1/2} &= \frac{1}{\Delta t}\int_{t_n}^{t_{n+1}} \bar{u} q(a,t)\,dt \\ &= \frac{\bar{u}}{\Delta t}\int_{t_n}^{t_{n+1}} g_0(t)\,dt, \end{aligned} \tag{7.5}$$

and use this in the flux-differencing formula for Q^{n+1}_1. Alternatively, a second-order accurate approximation such as

$$F^n_{1/2} = \bar{u} g_0(t_n + \Delta t/2) \tag{7.6}$$

could be used instead.

Note that for a five-point method we also need a special formula for the flux $F^n_{3/2}$, which might be more difficult to compute by this approach.

Again, for generality it is often simpler instead to find a way to set the ghost-cell values Q^n_0 (and perhaps Q^n_{-1}) so that the same interior method can be used at all points. For the advection equation we can easily compute sensible values using our knowledge of the exact solution. We would like to set

$$Q^n_0 = \frac{1}{\Delta x}\int_{a-\Delta x}^{a} q(x,t_n)\,dx.$$

Of course the true solution isn't even defined for $x < a$, but we can easily extend the solution past the boundary using our knowledge of characteristics, setting

$$\begin{aligned} q(x,t_n) &= q\left(a, t_n + \frac{a-x}{\bar{u}}\right) \\ &= g_0\left(t_n + \frac{a-x}{\bar{u}}\right). \end{aligned} \tag{7.7}$$

Then

$$Q_0^n = \frac{1}{\Delta x}\int_{a-\Delta x}^{a} g_0\left(t_n + \frac{a-x}{\bar{u}}\right) dx$$
$$= \frac{\bar{u}}{\Delta x}\int_{t_n}^{t_n+\Delta x/\bar{u}} g_0(\tau)\, d\tau. \tag{7.8}$$

Again we could approximate this integral by the second-order midpoint approximation, obtaining

$$Q_0^n = g_0\left(t_n + \frac{\Delta x}{2\bar{u}}\right), \tag{7.9}$$

which is also what we would get if we simply evaluated the cell-center value $q(a-\Delta x/2, t_n)$ using (7.7). Similarly, for the second ghost cell we could set

$$Q_{-1}^n = g_0\left(t_n + \frac{3\,\Delta x}{2\bar{u}}\right).$$

Specifying such boundary conditions in CLAWPACK requires modifying the `bc1` routine to set the ghost-cell values appropriately. See `[claw/book/chap7/advinflow]` for an example.

7.3 Acoustics

Similar ideas can be applied to develop boundary-condition procedures for systems of equations. The situation may be complicated by the fact that the system can have both positive and negative eigenvalues, so that each boundary will typically have both incoming and outgoing characteristics. Other sorts of physical boundary conditions, such as a solid wall where waves are expected to reflect, must also be studied. Many of these issues can be illustrated for the simple case of linear acoustics as developed in Section 2.7, a linear system of two equations with one eigenvalue of each sign. The boundary-condition procedures developed here will later be seen to extend very easily to nonlinear systems such as the full Euler equations of compressible flow.

We consider the acoustics equations from (2.52),

$$\begin{aligned} p_t + K_0 u_x &= 0 \\ \rho_0 u_t + p_x &= 0, \end{aligned} \tag{7.10}$$

with a variety of different boundary conditions. The characteristic variables for this system are

$$\begin{aligned} w^1(x,t) &= \frac{1}{2Z_0}(-p + Z_0 u), \\ w^2(x,t) &= \frac{1}{2Z_0}(p + Z_0 u), \end{aligned} \tag{7.11}$$

where Z_0 is the impedance, as derived in Section 2.8.

7.3.1 Nonreflecting Boundary Conditions

Suppose we wish to solve the *Cauchy problem* with initial data $\mathring{u}(x)$ and $\mathring{p}(x)$ that vary with x only in some region $a_1 < x < b_1$, and are constant outside this interval, say

$$(p, u) = \begin{cases} (p_L, u_L) & \text{if } x < a_1, \\ (p_R, u_R) & \text{if } x > b_1. \end{cases}$$

We know from Chapter 3 that the solution to the Cauchy problem will consist of two waves, each with fixed shape depending on $\mathring{u}$ and $\mathring{p}$. One propagates to the left with speed $-c_0$ and the other to the right with speed c_0. Eventually these waves will separate completely from one another (certainly for times $t > (b_1 - a_1)/c_0$) and leave a new constant state in between. Note that the characteristic variables (7.11) satisfy

$$\begin{aligned} w^2(a_1, t) &= \frac{1}{2Z_0}(p_L + Z_0 u_L) \qquad \text{for all } t \geq 0, \\ w^1(b_1, t) &= \frac{1}{2Z_0}(-p_R + Z_0 u_R) \quad \text{for all } t \geq 0. \end{aligned} \tag{7.12}$$

If we now want to compute this solution numerically, we must choose some finite domain $a \leq x \leq b$ on which to perform the computation, and we will suppose this domain includes the interval where the data is specified, i.e., $a < a_1 < b_1 < b$. For short times the solution should remain constant at the boundaries a and b, but eventually the acoustic waves should pass out through these boundaries. If we want to compute over longer times, then we must specify the boundary conditions in such a way that the waves leave the region cleanly, without generating any signal in the incoming characteristic variable propagating back into the computational domain.

The points $x = a$ and $x = b$ are artificial boundaries, in the sense that they are only introduced to make the computational domain finite, and do not correspond to physical boundaries. Hence we want to impose *nonreflecting boundary conditions* at these boundaries that do not introduce spurious reflections of the outgoing acoustic waves. Such boundary conditions are also called *absorbing boundary conditions*, since they are supposed to completely absorb any wave that hits them. Such boundary conditions are extremely important in many practical applications. We often wish to model what is happening in a small portion of physical space that must be truncated artificially at some point. Consider the problem of modeling seismic waves in the earth, or flow around an airplane. We cannot hope to model the entire earth or atmosphere, nor should we need to in order to understand localized phenomena. We want to cut off the domain far enough away that we will have a good model, but as close in as possible to reduce the size of the computational domain and the computing time required. Specifying good absorbing boundary conditions is often crucial in obtaining useful results. A variety of sophisticated approaches have been developed for specifying appropriate numerical boundary conditions. See Section 21.8.5 for some references and further discussion.

With Godunov-type methods that involve solving the Riemann problem at each interface, it turns out that simply using zero-order extrapolation often gives a reasonable set of absorbing boundary conditions that is extremely simple to implement. For the simple

one-dimensional acoustics problem, we can analyze this choice completely using the characteristic decomposition.

In fact, one approach to solving the problem for this linear system would be to diagonalize the equations, obtaining scalar advection equations for the characteristic variables w^1 and w^2 from (7.11). The numerical method can also be diagonalized and yields updating formulas for $W^1 = (-Q^1 + Z_0 Q^2)/2Z_0$ and $W^2 = (Q^1 + Z_0 Q^2)/2Z_0$. At the boundary $x = a$, for example, the variable w^1 is the outgoing variable and we already know that zero-order extrapolation can be used for this variable from Section 7.2.1. For the incoming variable w^2 we want to set the value as a function $g_0(t)$, following Section 7.2.2. Since we want to insure that no signal flows into the domain, regardless of what is flowing out in w^1, the correct value to set is

$$w^2(a, t) = \frac{1}{2Z_0}(p_L + Z_0 u_L)$$

for all t (recall (7.12)). Following Section 7.2.1 this would suggest setting both the ghost cell values to this value,

$$W_0^2 = \frac{1}{2Z_0}(p_L + Z_0 u_L), \qquad W_{-1}^2 = \frac{1}{2Z_0}(p_L + Z_0 u_L), \tag{7.13}$$

together with extrapolation of W^1. From these ghost-cell values for W we could then compute the required ghost-cell values for Q.

However, since W^2 is already constant near the point $x = a$, we expect that $W_1^2 = (p_L + Z_0 u_L)/2Z_0$, and so the boundary conditions (7.13) can also be obtained by simply using zero-order extrapolation for W^2 as well as for W^1. But now, if we extrapolate both of the characteristic variables and then compute Q_0 and Q_{-1}, we will find that these values are simply what would be obtained by zero-order extrapolation of Q. So we do not need to go through the diagonalization at all in setting the boundary conditions, but can simply set

$$Q_0^n = Q_1^n, \qquad Q_{-1}^n = Q_1^n$$

in each time step.

Note that by setting $Q_0 = Q_1$ we insure that the solution to the Riemann problem at the interface $x_{1/2}$ consists of no waves, or more properly that the wave strengths $\alpha_{1/2}^p$ are all zero. So in particular there are no waves generated at the boundary regardless of what is happening in the interior, as desired for nonreflecting boundary conditions.

There are also no outgoing waves generated at $x_{1/2}$, which may be incorrect if an acoustic wave in w^1 should in fact be leaving the domain. But any outgoing waves would only be used to update the solution in the ghost cell $\mathcal{C}_0$, and it is not important that we update this cell properly. In fact, the value Q_0 is not updated at all by the interior algorithm. Instead this value is reset by extrapolation again at the start of the next time step.

7.3.2 Incoming Waves

In the previous section we assumed that we wanted incoming waves to have zero strength. In some problems we may need to specify some incoming signal. For example, we might

wish to study what happens when a sinusoidal sound wave at some frequency ω hits a region with different material properties, in which case some of the energy will be reflected (see Section 9.6). Suppose the material properties vary only in a region $a_1 < x < b_1$ and we wish to compute on a larger region $a < x < b$. Then we want to impose the boundary condition

$$w^2(a, t) = \sin(\omega t) \tag{7.14}$$

as the incoming signal, together with nonreflection of any outgoing signal that has been generated within the domain. We must now apply some characteristic decomposition in the process of applying the boundary procedure in order to impose the correct boundary conditions. If we decompose Q_1 into $Q_1 = W_1^1 r^1 + W_1^2 r^2$, then the ghost cell value Q_0 should be set as

$$Q_0 = W_1^1 r^1 + \sin(\omega(t_n + \Delta x/2c_0))\, r^2.$$

Alternatively, we could first extrapolate the whole vector Q and then reset the w^2 component, setting

$$Q_0 = Q_1 + \left(\sin(\omega(t_n + \Delta x/2c_0)) - W_1^1\right) r^2.$$

See `[claw/book/chap7/acouinflow]` for an example.

For the acoustics system with only two equations, the two approaches are essentially equivalent in terms of the work required. But if we had a larger system of m equations and wanted to impose an incoming wave in only one characteristic family, e.g.,

$$w^j(a, t) = g_0(t),$$

for some j (with zero-strength signal in other incoming characteristics and nonreflection of outgoing waves), then we could set

$$Q_0 = Q_1 + \left[g_0(t_n + \Delta x/2\lambda^j) - W_1^j\right] r^j,$$

where $W_1^j = \ell^j Q_1$ is the eigen-coefficient of r^j in Q_1 and λ^j is the corresponding eigenvalue.

7.3.3 Solid Walls

Consider a tube of gas with a solid wall at one end. In this case we expect acoustic waves to reflect at this boundary in a particular way – see Section 3.11 for the relation between w^1 and w^2. Rather than working with the characteristic variables, however, in this case it is easier to return to the physical boundary condition itself. For a solid wall at $x = a$ this is

$$u(a, t) = 0. \tag{7.15}$$

The key observation is the following: Suppose we take any data $(p^0(x), u^0(x))$ defined for all $x > a$ and extend it to $x < a$ by setting

$$\begin{aligned} p^0(a-\xi) &= p^0(a+\xi), \\ u^0(a-\xi) &= -u^0(a+\xi), \end{aligned} \tag{7.16}$$

for $\xi > 0$. Then if we solve the Cauchy problem with this extended data, the solution we obtain for $x > a$ will agree exactly with the solution to the half-space problem on $x > a$ with a solid wall boundary condition (7.15) at $x = a$. This follows from symmetry: the conditions (7.16) will continue to hold for $t > 0$ and in particular $u(a) = -u(a)$ must be zero. See also Exercise 7.2.

This suggests the following formulas for ghost-cell values in each time step:

$$\begin{array}{lll} \text{for } Q_0\text{:} & p_0 = p_1, & u_0 = -u_1, \\ \text{for } Q_{-1}\text{:} & p_{-1} = p_2, & u_{-1} = -u_2. \end{array} \tag{7.17}$$

This imposes the necessary symmetry at the start of each time step.

Solid-wall boundary conditions are implemented in the CLAWPACK library routine `claw/clawpack/1d/lib/bc1.f` and invoked by setting `mthbc(i) = 3`, where `i = 1` for the left boundary or `i = 2` for the right boundary. This assumes that the solid-wall boundary condition can be set by reflecting all components of Q and then negating the second component, as in (7.17). This works for the acoustics equations and also for several other systems of equations that we will study in this book, including the shallow water equations and several forms of the gas dynamics equations. Related boundary conditions can also be used for elastic waves in solids with fixed or free boundaries; see Section 22.4.

7.3.4 Oscillating Walls

Now suppose that the solid wall at $x = a$ is oscillating with some very small amplitude, generating an acoustic wave in the gas. This is a common situation in acoustics: small-scale molecular vibrations of solid objects give rise to many familiar sounds. For very small-amplitude motions we can still use the linear acoustics equations on the fixed domain $a \le x \le b$ but with the boundary condition

$$u(a, t) = U(t) \tag{7.18}$$

to simulate the vibrating wall. For a pure-tone oscillation we might take

$$U(t) = \epsilon \sin(\omega t), \tag{7.19}$$

for example. We can implement this by setting the following ghost-cell values:

$$\begin{array}{lll} \text{for } Q_0\text{:} & p_0 = p_1, & u_0 = 2U(t_n) - u_1, \\ \text{for } Q_{-1}\text{:} & p_{-1} = p_2, & u_{-1} = 2U(t_n) - u_2. \end{array} \tag{7.20}$$

These reduce to (7.17) if $U(t) \equiv 0$. The rationale for this set of boundary conditions is explored in Exercise 7.2.

Exercises

7.1. If we use the ghost-cell value Q_0^n from (7.9) in the Lax–Wendroff method, what flux $F_{1/2}^n$ will be computed, and how does it compare with (7.6)? Note that if the Courant number is near 1, then $\Delta x/\bar{u} \approx \Delta t$.

7.2. (a) For acoustics with a solid-wall boundary, we set the ghost-cell values (7.17) and then solve a Riemann problem at $x_{1/2} = a$ with data

$$Q_0 = \begin{bmatrix} p_1 \\ -u_1 \end{bmatrix}, \quad Q_1 = \begin{bmatrix} p_1 \\ u_1 \end{bmatrix}.$$

Show that the solution to this Riemann problem has an intermediate state q^* with $u^* = 0$ along the wall, another reason why this is the sensible boundary condition to impose.

(b) Give a similar interpretation for the oscillating-wall boundary conditions (7.20).

7.3. The directory `[claw/book/chap7/standing]` models a standing wave. The acoustics equations are solved with $\rho_0 = K_0 = 1$ in a closed tube of length 1 with with initial data $\overset{\circ}{p}(x) = \cos(2\pi x)$ and $\overset{\circ}{u}(x) = 0$. Solid-wall boundary conditions are used at each end. Modify the input data in `claw1ez.data` to instead use zero-order extrapolation at the right boundary $x = 1$. Explain the resulting solution using the theory of Section 3.11.

8

Convergence, Accuracy, and Stability

Whenever we use a numerical method to solve a differential equation, we should be concerned about the accuracy and convergence properties of the method. In practice we must apply the method on some particular discrete grid with a finite number of points, and we wish to ensure that the numerical solution obtained is a sufficiently good approximation to the true solution. For real problems we generally do not have the true solution to compare against, and we must rely on some combination of the following techniques to gain confidence in our numerical results:

- *Validation on test problems*. The method (and particular implementation) should be tested on simpler problems for which the true solution is known, or on problems for which a highly accurate comparison solution can be computed by other means. In some cases experimental results may also be available for comparison.
- *Theoretical analysis of convergence and accuracy*. Ideally one would like to prove that the method being used converges to the correct solution as the grid is refined, and also obtain reasonable error estimates for the numerical error that will be observed on any particular finite grid.

In this chapter we concentrate on the theoretical analysis. Here we consider only the Cauchy problem on the unbounded spatial domain, since the introduction of boundary conditions leads to a whole new set of difficulties in analyzing the methods. We will generally assume that the initial data has *compact support*, meaning that it is nonzero only over some bounded region. Then the solution to a hyperbolic problem (which has finite propagation speeds) will have compact support for all time, and so the integrals over the whole real line that appear below really reduce to finite intervals and we don't need to worry about issues concerning the behavior at infinity.

8.1 Convergence

In order to talk about accuracy or convergence, we first need a way to quantify the error. We are trying to approximate a function of space and time, and there are many possible ways to measure the magnitude of the error. In one space dimension we have an approximation Q_i^n at each point on space–time grid, or in each grid cell when using a finite volume method. For comparison we will let q_i^n represent the exact value we are hoping to approximate well.