9

Discretizing Hyperbolic Transport Equations

So far in the book, we have introduced you to several basic discretizations: Section 4.4 introduced the two-point scheme for elliptic operators of the type $\nabla \cdot \mathbf{K}(\vec{x})\nabla$ and the upstream scheme for hyperbolic operators $\nabla \cdot \vec{v}$. Likewise, Section 7.2 introduced fully implicit discretizations for compressible single-phase flow, and discussed how to use a Newton–Raphson method to linearize and solve the resulting system of discrete equations. With some modifications, these techniques are all we need to describe the most widespread approach to simulate compressible, multiphase flow.

For incompressible multiphase flow, on the other hand, it is common to write the system as a pressure equation and one or more transport equations and use specialized discretization schemes for each subequations. Chapter 6 discussed consistent schemes for elliptic pressure equations. This chapter reviews some of the particular challenges that lie in discretizing transport equations. If you are not interested in this wider perspective and only wish to know the simplest possible approach to solving saturation equations, you can jump directly to Section 9.4 on page 286, which discusses the standard upstream mobility-weighting method implemented for general polyhedral grids in the `incomp` module of MRST.

9.1 A New Solution Concept: Entropy-Weak Solutions

In the first part of this chapter, we continue to discuss the homogeneous conservation law introduced in the previous chapter,

$$u_t + f(u)_x = 0, \qquad u(x,0) = u_0(x). \tag{9.1}$$

Here, u is some conserved quantity, which need not necessarily be a fluid saturation, and $f(u)$ is a generic flux function. Equation (9.1) usually arises from a more fundamental physical law on integral form,

$$\frac{d}{dt}\int_{x_1}^{x_2} u(x,t)\,dx = f\big(u(x_1,t)\big) - f\big(u(x_2,t)\big), \tag{9.2}$$

which states that the rate of change of quantity u within the interval $[x_1,x_2]$ equals the flux across the ends $x = x_1$ and $x = x_2$ of the interval.

https://doi.org/10.1017/9781108591416.013 Published online by Cambridge University Press

In the previous chapter, we saw that solutions to (9.1) may develop discontinuities in finite time, even for smooth initial data if the flux function f is nonlinear. This means that the solution of (9.1) is usually understood in the weak sense,

$$\int_0^\infty \int_{\mathbb{R}} \big(u\varphi_t + f(u)\varphi_x\big)\,dt dx = \int_{\mathbb{R}} u_0(x)\varphi(x,0)\,dx. \tag{9.3}$$

Here, $\varphi(x,t)$ is a continuous and smooth test function that has *compact support* so that it vanishes outside a bounded region in the (x,t)-plane.

Solutions defined by the weak form (9.3) are not necessarily unique. The solution concept must therefore be extended to include additional admissibility conditions to single out the correct solution among several possible candidates satisfying the weak form. A classical method to obtain uniqueness is to add a regularizing second-order term to (9.1), giving a parabolic equation

$$u_t^\epsilon + f(u^\epsilon)_x = \epsilon u_{xx}^\epsilon,$$

which has unique, smooth solutions. The unique solution of the hyperbolic equation (9.1) is then defined as the limit of $u^\epsilon(x,t)$ as ϵ tends to zero. In models from fluid dynamics, such a second-order term can be proportional to the viscosity of the fluid, and the method is therefore called the *vanishing viscosity method*. For flow in porous media, the transport equations will already have a second-order term coming from capillary forces (see (8.35)). However, capillary forces are often neglected when studying flow dominated by global pressure gradients, or the capillary function may vary with rock type and this can introduce discontinuities in the phase saturations at the interface between different rock types.

Since the vanishing viscosity solutions $u^\epsilon(x,t)$ are smooth, it is possible to use techniques from classical analysis to prove the existence, uniqueness, and stability of the solution to (9.1). This was first done in a seminal work by Kružkov [173], which paved the road for the modern theory of nonlinear partial differential equation of a hyperbolic-parabolic type. Working with limits of viscous solutions is not very practical. Instead, we impose other admissibility conditions. We have already encountered the Lax and Oleinik entropy conditions, (8.57) and (8.58), which were used to single out permissible discontinuities. An alternative approach is to introduce a (convex) entropy function $\eta(u)$ and a corresponding entropy flux $\psi(u)$, and require that an admissible weak solution u must satisfy the entropy condition

$$\eta(u)_t + \psi(u)_x \le 0, \tag{9.4}$$

which must be interpreted in the weak sense as

$$\int_0^\infty \int_{\mathbb{R}} \big(\eta(u)\varphi_t + \psi(u)\varphi_x\big)\,dt dx + \int_{\mathbb{R}} \eta(u_0(x))\varphi(x,0)\,dx \ge 0. \tag{9.5}$$

The solution $u(x,t)$ is called an *entropy weak solution* of (9.1) if it satisfies (9.5) for all $k \in \mathbb{R}$ and nonnegative test functions φ.

9.2 Conservative Finite-Volume Methods

You have already seen that using a finite-volume method made it simple to formulate a discretization on general polyhedral grids and ensured that the resulting method obeys the important property of mass conservation. Approximating the unknown function in terms of its cell averages has one additional advantage for hyperbolic conservation laws. Because the differential equation ceases to be pointwise valid in the classical sense once a discontinuity arises in $u(x,t)$, we should expect that pointwise approximations used in standard finite-difference methods will break down, since these rely on the assumption that the function is smooth in a small neighborhood where the discrete difference is taken. Instead of seeking solutions in a *pointwise* sense, the finite-volume approach seeks *globally defined solutions* of the integral form (9.2), which is more fundamental, represented in the form of discrete cell averages, as we have already seen in previous chapters.

To develop a finite-volume method for (9.1), we first define the sliding average,

$$\bar{u}(x,t) = \frac{1}{\Delta x}\int_{x-\frac{1}{2}\Delta x}^{x+\Delta x/2} u(\xi,t)\,d\xi. \tag{9.6}$$

We then associate these sliding averages with grid cells, $\{[x_{i-1/2}, x_{i+1/2}]\}$, where $x_{i\pm 1/2} = x_i \pm \frac{1}{2}\Delta x_i$, and set $u_i(t) = \bar{u}(x_i,t)$. By inserting $u_i(t)$ into the integral form of the conservation law (9.2), we obtain a semi-discrete version of (9.1),

$$\frac{d\bar{u}_i}{dt} = \frac{1}{\Delta x_i}\Big[f\big(u(x_{i-1/2},t)\big) - f\big(u(x_{i+1/2},t)\big)\Big]. \tag{9.7}$$

Alternatively, we may define $u_i^n = \bar{u}_i(t_n)$ for a set of discrete times t_n and then integrate (9.7) from t_n to t_{n+1} to derive a fully discrete version of (9.1),

$$u_i^{n+1} - u_i^n = \frac{1}{\Delta x}\int_{t_n}^{t_{n+1}} f(u(x_{i-1/2},t))\,dt - \frac{1}{\Delta x}\int_{t_n}^{t_{n+1}} f(u(x_{i+1/2},t))\,dt. \tag{9.8}$$

While (9.7) and (9.8) tell us how the unknown cell averages $u_i(t)$ and u_i^n evolve in time, they cannot be used directly *to compute* these cell averages, since we do not know the *point values* $u(x_{i\pm1/2},t)$ needed to evaluate the integrand. To obtain these, we need to make additional assumptions and approximations, e.g., as discussed in Section 4.4.3 for the linear case. Nevertheless, (9.8) suggests that a viable numerical method for (9.1) should be of the form

$$u_i^{n+1} = u_i^n - r_i^n\big(F_{i+1/2}^n - F_{i-1/2}^n\big), \qquad r_i^n = \Delta t^n/\Delta x_i, \tag{9.9}$$

where $F_{i\pm1/2}^n$ approximates the average flux over each cell interface,

$$F_{i\pm1/2}^n \approx \frac{1}{\Delta t^n}\int_{t_n}^{t_{n+1}} f\big(u(x_{i\pm1/2},t)\big)\,dt. \tag{9.10}$$

In Section 8.4 we showed that information propagates at a finite speed along so-called characteristics, which implies that the flux integral (9.10) will only depend on the solution $u(x,t_n)$ in a local neighborhood of the interface $x_{i\pm1/2}$. This means, in turn, that $F_{i\pm1/2}$ can be approximated in terms of a small collection of neighboring cell averages, i.e., $F_{i+1/2} = F(u^n_{i-p}, \dots, u^n_{i+q})$, which we alternatively will write as $F_{i\pm 1/2} = F(u^n; i \pm \frac{1}{2})$.

Any numerical method written on the form (9.9) will be *conservative*. To see this, we multiply (9.9) by Δx_i and sum over i. The flux terms will cancel in pairs and we are left with

$$\sum_{i=-M}^{N} u_i^{n+1}\Delta x_i = \sum_{i=-M}^{N} u_i^n \Delta x_i - \Delta t^n \big(F^n_{N+1/2} - F^n_{-M-1/2}\big).$$

From this, it follows that the method is conservative on a finite domain, since the accumulation inside any interval balances the sum of the flux across the interval edges. To see that the same is true in an infinite domain, we must make some additional assumptions. If the initial solution $u_0(x)$ has bounded support, it follows that the two flux terms will cancel if we choose M and N sufficiently large. Hence, the scheme (9.9) is conservative in the sense that

$$\sum_i u_i^{n+1}\Delta x_i = \sum_i u_i^n \Delta x_i = \cdots = \int u_0(x)\,dx.$$

9.3 Centered versus Upwind Schemes

To provide important insight into the numerical solution of hyperbolic equation, we briefly outline two main classes of classical schemes.

9.3.1 Centered Schemes

The simplest approach to *reconstruct* the point values $u(x_{i\pm 1/2}, t)$ needed in (9.7) and (9.9), is to assume that $u(x,t) = u_i(t)$ inside each grid cell and average the flux values on opposite sides of the grid interface, i.e., set

$$F^n_{i\pm 1/2} = \tfrac{1}{2}\big[f(u^n_{i\pm 1}) + f(u^n_i)\big]. \tag{9.11}$$

This approximation is referred to as a *centered approximation*, which unfortunately gives a notoriously unstable scheme. We can add an artificial diffusion term, $\frac{\Delta x^2}{\Delta t}\partial_x^2 u$, to stabilize, but then it is no longer possible to go back to the semi-discrete form (9.7), since the artificial diffusion will blow up in the limit $\Delta t \to 0$. Discretizing the artificial diffusion through standard centered differences gives the classical Lax–Friedrichs scheme

$$u_i^{n+1} = \frac{1}{2}\Big(u^n_{i+1} + u^n_{i-1}\Big) - \frac{1}{2}r\Big[f(u^n_{i+1}) - f(u^n_{i-1})\Big]. \tag{9.12}$$

Alternatively, the scheme can be written in conservative form (9.9) using the numerical flux

$$F(u^n; i+1/2) = \frac{1}{2r}\big(u^n_i - u^n_{i+1}\big) + \frac{1}{2}\big[f(u^n_i) + f(u^n_{i+1})\big]. \tag{9.13}$$

To ensure stability, we have to impose a restriction on the time-step through a *CFL condition*, named after Courant, Friedrichs, and Lewy, who wrote one of the first papers on finite-difference methods in 1928 [75]. The CFL condition states that the true domain of dependence for the PDE (9.1) should be contained in the domain of dependence for (9.9). For the Lax–Friedrichs scheme, this means that

$$\frac{\Delta t}{\Delta x} \max_u |f'(u)| \leq 1. \tag{9.14}$$

Under this condition, the scheme is very robust and will always converge, albeit painstakingly slow in many cases. The robustness is largely due to the added numerical diffusion, which tends to smear discontinuities. As an illustration, consider a stationary discontinuity satisfying $\partial_t u = 0$. Here, the approximate solution in each cell is defined as the arithmetic average of the cell averages in the neighboring cells, i.e., $u_i^{n+1} = \frac{1}{2}(u_{i+1}^n + u_{i-1}^n)$. From this, we see that the scheme will smear the discontinuity one cell in each direction per time step. By using a formal Taylor expansion on a single time step, it follows that the scheme has a truncation error of order two. In practice, we are more interested in the error at a fixed time, which we need to use an increasing number of time steps to reach as $\Delta x \to 0$ because of (9.14). This means that we must divide with Δt so that the error of the scheme is $\mathcal{O}(\Delta x)$ as $\Delta x \to 0$, and we say that the scheme is *formally first-order accurate*.

We can improve accuracy if we make a more accurate approximation to the integral defining $F_{i\pm 1/2}^n$. Instead of evaluating the integral at the endpoint t_n, we can evaluate it at the midpoint $t_{n+1/2} = t_n + \frac{1}{2}\Delta t$. One can show that the corresponding point values can be *predicted* with acceptable accuracy by the Lax–Friedrichs scheme on a grid with half the grid spacing. This gives a second-order, predictor-corrector scheme called the (Richtmeyer two-step) Lax–Wendroff method [178]

$$\begin{aligned} u_{i+1/2}^{n+1/2} &= \tfrac{1}{2}\big(u_i^n + u_{i+1}^n\big) - \tfrac{1}{2}r\big[f(u_{i\pm1}^n) - f(u_i^n)\big], \\ u_i^{n+1} &= u_i - r\big[f(u_{i+1/2}^{n+1/2}) - f(u_{i-1/2}^{n+1/2})\big], \end{aligned} \tag{9.15}$$

which is stable under the same CFL condition (9.14) as the Lax–Friedrichs scheme. The corresponding numerical flux reads

$$F(u^n; i+1/2) = f\Big(\tfrac{1}{2}(u_i^n + u_{i+1}^n) - \tfrac{1}{2}r\big[f(u_{i+1}^n) - f(u_i^n)\big]\Big). \tag{9.16}$$

Both the Lax–Friedrichs and the Lax–Wendroff scheme can either be interpreted as a finite-difference or as a finite-volume scheme, and by only looking at the formulas written in terms of u_i^n, it is not possible to determine which formulation is in use. However, the underlying principles are fundamentally different. Finite-difference methods evolve a discrete set of point values by using discrete differences to approximate the differential operators in (9.1). Finite-volume methods evolve *globally defined solutions* given by (9.2) and *realize* them in terms of a discrete set of cell averages. This latter perspective is the key to modern so-called high-resolution methods [183, 294], which have proved to be very successful.

Let us now see how we can implement these two schemes compactly in MATLAB. When computing on a bounded domain, one generally has to modify the stencil of the scheme in the cells next to the domain boundary to account for boundary conditions.

A widely used trick to avoid this is to pad the domain with a layer of *ghost cells*, i.e., extra cells that are not really part of the domain and whose states can be set manually to impose the desired boundary conditions.

```
function u=lxf(u0,cfl,dx,T,flux,df,boundary)
u = u0; t = 0.0;
dt = cfl*dx/max(abs(df(u0)));
i = 2:numel(u0)-1;   % do not compute on the ghost cells
while (t<T)
  if t+dt>T,  dt = T-t; end
  t=t+dt;
  u = boundary(u);   % set values in the ghost cells
  f = flux(u);
  u(i) = 0.5*(u(i+1)+u(i-1)) - 0.5*dt/dx*(f(i+1)-f(i-1));
  dt = cfl*dx/max(abs(df(u)));
end
```

Here, $u0$ gives the initial data, `cfl` and `dx` are the CFL number and the spatial discretization parameter, `T` is the desired end time, and `flux`, `df`, and `boundary` are handles to functions that compute the flux f and its derivative and set appropriate states in the ghost cells. Notice, in particular, how we use the vector `i` to avoid computing in the ghost cells that pad our domain. To get the Lax–Wendroff scheme, we introduce an auxiliary array `U` and then simply replace the second last line by

```
U(i) = 0.5*(u(i)+u(i+1)) - 0.5*r*(f(i+1)-f(i));
U = boundary(U);  f = flux(U);
u(i) = u(i) - r*(f(i)-f(i-1));
```

9.3.2 Upwind or Godunov Schemes

The centered schemes introduced in the previous subsection are black-box schemes that can be utilized without any particular knowledge about the flux function apart from an estimate of its maximum and minimum derivative (or eigenvalues for systems of equations). The reason for this is that these schemes utilize information from both sides of the grid interface when computing approximations to the flux integral in (9.10). In many cases we can do better, since we know more of how the solution behaves at the interface. If $f'(u) > 0$, for instance, we know that all characteristics are positive, and hence we can use the solution from the left side of the grid interface to evaluate the integral in (9.10). With a constant reconstruction inside each cell, this means that $F_{i+1/2} = f(u_i^n)$ is the *exact* value of the integral, provided that the solution at time t_n is constant and equal u_i^n in grid cell i. The resulting scheme,

$$u_i^{n+1} = u_i^n - r\big[f(u_i^n) - f(u_{i-1}^n)\big], \tag{9.17}$$

is called an *upwind scheme*, which you have already encountered in Chapters 4 and 7. Similarly, if $f'(u) \le 0$, the upwind scheme takes the form

$$u_i^{n+1} = u_i^n - r\big[f(u_{i+1}^n) - f(u_i^n)\big]. \tag{9.18}$$

In either case, the upwind scheme is a two-point scheme that, instead of using a centered difference, uses a one-sided difference in the *upwind* direction, i.e., in the direction from which the waves (or information) is coming. This type of upwind differencing is the underlying design principle for so-called *Godunov schemes*, named after the author of the seminal paper [118]. If we assume a constant reconstruction so that $u(x,t^n) \equiv u_i^n$ inside grid cell number i, the evolution of $u(x,t)$ can be decomposed into a set of local Riemann problems

$$\partial_t v + \partial_x f(v) = 0, \qquad v(x,0) = \begin{cases} u_i^n, & x < x_{i+1/2}, \\ u_{i+1}^n, & x \ge x_{i+1/2}, \end{cases} \tag{9.19}$$

which are of the same type as those we have already accounted in Section 8.4. Each of these Riemann problems admits a self-similar solution $v_{i+1/2}(x/t)$, which can be constructed by introducing a local convex or concave envelope[1] $f_c(u; u_i^n, u_{i+1}^n)$ of the flux function f over the interval $[u_i^n, u_{i+1}^n]$, as discussed in (8.60). The cell averages can then be correctly evolved a time step Δt forward in time by (9.9) if we use $v_{i\pm1/2}(0)$ – or a good approximation thereof – to evaluate f in the flux integral (9.10). The Riemann fan will generally expand in each direction at a constant speed, and using $v_{i\pm1/2}(0)$ to evaluate the integrand of (9.10) is only correct until the first wave from the neighboring Riemann problem reaches the interface. This implies a time-step restriction of the form

$$\max_{i,n} |f_c'(\cdot\,; u_i^n, u_{i+1}^n)| \frac{\Delta t^n}{\Delta x_i} \le 1, \tag{9.20}$$

which we recognize as a sharper version of the CFL condition (9.14). The resulting scheme is formally first-order accurate, but will generally be much less diffusive than the centered Lax–Friedrichs scheme.

Let us now go back to the general transport equation (8.35), which is hyperbolic if $P_c' \equiv 0$. Here, the fractional flow function $f(S)$ has a characteristic S-shape and its derivative is therefore always positive. This means that if gravity forces are negligible, like in the horizontal displacements we studied in Section 8.4.1, the correct Godunov scheme would be either (9.17) or (9.18), depending upon the sign of the flux $\vec{v}$. For the gravity segregation problem considered in Section 8.4.2, on the other hand, the flux function $g(S) = \lambda_w\lambda_n/(\lambda_w + \lambda_n)$ has derivatives of both signs and we would generally have to solve the Riemann problem (9.19) like we did in Figure 8.14 on page 265 to apply the Godunov scheme. The same is generally true for flow involving a combination of viscous and gravity forces, as illustrated in Figure 8.17. Having to solve a Riemann problem each time we need to evaluate the flux $F_{i\pm1/2}^n$ may seem rather cumbersome, and fortunately there is a rather elegant and physically motivated fix to this. Instead of upwinding the flux itself, we can use the phase fluxes to evaluate the relative mobilities in the upstream direction

[1] The notation $f(\cdot\,; a, b)$ signifies that the functional form of the function f depends on the parameters a and b.

$$\lambda_\alpha(S)^u\Big|_{x_{i+1/2}} = \begin{cases} \lambda_\alpha(S_i^n), & \text{if } v_\alpha > 0, \\ \lambda_\alpha(S_{i+1}^n), & \text{otherwise.} \end{cases} \tag{9.21}$$

and then use these upwind values to evaluate both the S-shaped viscous flux $f_w(S)$ and the bell-shaped gravity flux $\lambda_n(S)f_w(S)$. Here, we have used the phase flux v_α, which in the multidimensional case can be computed as $\big(\nabla p_\alpha - g\rho_\alpha\nabla z\big)\cdot\vec{n}_i$, where $\vec{n}_i$ is the normal to the interface. The resulting method, called the explicit *upstream mobility-weighting* scheme, is usually the method of choice for porous media application because of its accuracy and simplicity. In [49] it was shown that the method also satisfies the mathematical properties necessary to get a consistent and convergent scheme.

9.3.3 Comparison of Centered and Upwind Schemes

The classical schemes introduced so far in this section all have certain weaknesses and strengths that are common for many other schemes as well. In short, first-order schemes tend to smear discontinuous solutions, whereas second-order schemes introduce spurious oscillations. To highlight these numerical artifacts, we consider two different cases; you can find complete source codes for both examples in the `nummet` directory of the `book` module. The directory also contains two examples of high-resolution methods, which to a large extent cure these problems. Very cursorily explained, the idea of such a scheme is to introduce a nonlinear function that enables the scheme to switch from being high-order on smooth solutions to being first-order near discontinuities.

Example 9.3.1 *Consider linear advection on the unit interval with periodic boundary conditions*

$$u_t + u_x = 0, \qquad u(x,0) = u_0(x), \qquad u(0,t) = u(1,t).$$

The advantage of using periodic data is that we know that the exact solution at time $t = n$ for an integer number n equals $u_0(x)$. As initial data $u_0(x)$ we choose a combination of a smooth squared sine wave and a double step function,

$$u(x,0) = \sin^2\Big(\frac{x-0.1}{0.3}\pi\Big)\chi_{[0.1,0.4]}(x) + \chi_{[0.6,0.9]}(x).$$

Using the `lxf` *solver outlined in Section 9.3.1, we can set up the case as follows, where the function* `periodic` *simply sets* `u(1)=u(end-1)` *and* `u(end)=u(2)`.

```
dx = 1/100;
x  = -.5*dx:dx:1+.5*dx;
u0 = sin((x-.1)*pi/.3).^2.*double(x>=.1 & x<=.4);
u0((x<.9) & (x>.6)) = 1;

f  = @(u) u;    % flux function
df = @(u) 0*u+1;
uf = lxf(u0, .995, dx, 20, f, df, @periodic);
```

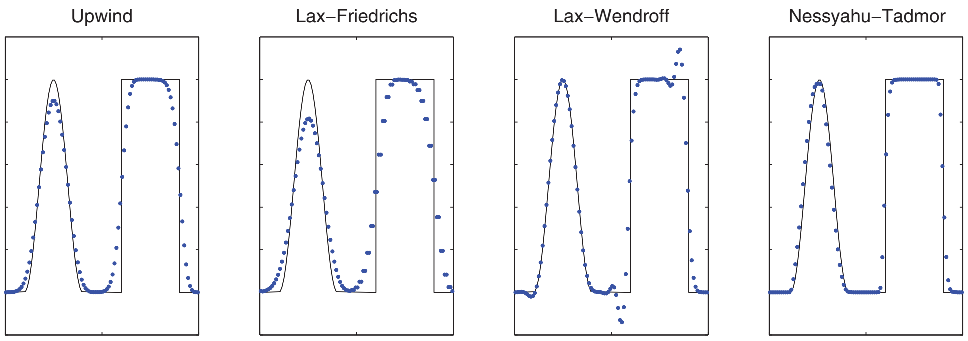

Figure 9.1 Approximate solutions at time $t = 20.0$ for the linear advection equation $u_t + u_x = 0$, with periodic boundary conditions computed by three classical schemes and a high-resolution scheme on a grid with 100 cells and CFL number 0.995.

Figure 9.1 shows approximate solutions after 20 periods ($t = 20$) computed by the upwind, Lax–Friedrichs, and Lax–Wendroff methods and the high-resolution Nessyahu–Tadmor scheme [224] on a grid with 100 cells for $\Delta t = 0.995\Delta x$. Both the centered schemes clearly give unacceptable resolution of the solution profile. The first-order Lax–Friedrichs scheme smears both the smooth and the discontinuous part of the advected profile. The second-order Lax–Wendroff scheme preserves the smooth profile quite accurately, but introduces spurious oscillations at the two discontinuities. This behavior is representative for classical schemes. The upwind method represents a reasonable compromise in the sense that it does not smear the smooth wave and the discontinuities as much as Lax–Friedrichs, and does not introduce oscillations like Lax–Wendroff. The Nessyahu–Tadmor scheme gives accurate resolution of both the smooth and the discontinuous parts of the profile, thereby combining the best of the low and high-order schemes.

The advection equation can serve as a conceptual model for more complex cases. As we saw in Figures 8.9 and 8.10, a displacement front will be self-sharpening in the sense that if the leading discontinuity is smeared, smeared states behind the displacement front will travel faster than the discontinuity (since they have characteristics pointing into it) and will hence "catch up." Likewise, smeared states ahead of the front will travel slower and be overrun by the front, since these states have characteristics that point into the front. For linear waves, the characteristics of any smeared state will run parallel to the discontinuity and hence there will be no self-sharpening to counteract the numerical dissipation inherent in any numerical scheme. Linear waves are therefore more susceptible to numerical dissipation than nonlinear shock waves.

In water-based EOR methods, for instance, active chemical or biological substances are added to modify the physical properties of the fluids or/and the porous media at the interface between oil and water. The resulting displacement processes are governed by complex interplays between the transport of chemical substances and how these substances affect the flow by changing the properties of the fluids and the surrounding rock. These

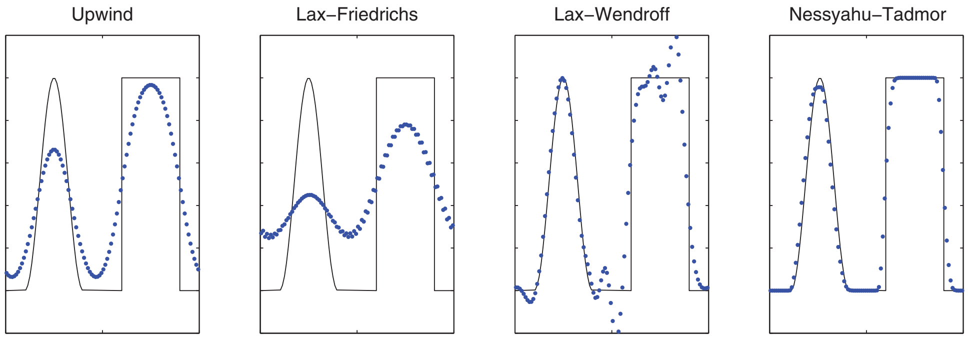

Figure 9.2 Approximate solutions at time $t = 1.0$ for the linear advection equation $u_t + u_x = 0$, with periodic boundary conditions computed by three classical schemes on a grid with 100 cells and time step of 0.5 relative to the stability limit.

property changes are often very nonlinear and highly sensitive to threshold parameters that determine sharp transitions between regions of very different behavior. The transport of chemical substances is largely linear and hence more affected by numerical diffusion. A simulation may therefore fail to resolve the local displacement process if the chemical fronts are smeared out or contain overshoots. As a result, unresolved simulation can therefore lead to misleading predictions of injectivity and recovery profiles.

Example 9.3.2 *For many systems, the linear waves travel significantly slower than the leading shock wave, and hence will be computed with a smaller effective CFL number. In Figure 9.2, we have rerun the experiment assuming that the linear waves travel at half the speed of a leading shock wave. Even after one period, the first-order schemes have introduced significantly more smearing, and the oscillations for Lax–Wendroff are much worse. The high-resolution scheme, on the other hand, maintains its resolution much better.*

The two previous examples are admittedly idealized cases compared with what we see in reservoir simulation, but illustrate very well the potential pitfalls of classical schemes. In the next example, we consider a more relevant case.

Example 9.3.3 *Let us revisit the classical Buckley–Leverett profile studied in Example 8.4.1, i.e., consider the following equation,*

$$S_t + \left(\frac{S^2}{S^2 + (1-S)^2}\right)_x = 0, \quad S(0,t) = 1, \quad S(x,0) = \begin{cases} 1, & x < 0.1, \\ 0, & x \geq 0.1. \end{cases}$$

Figure 9.3 shows the solution at time $t = 0.65$ computed by the same four schemes. The first-order schemes compute qualitatively correct approximations of both the leading shock wave and the trailing rarefaction wave. As expected, Lax–Friedrichs introduces significant smearing of both the shock and the kink. The upwind scheme delivers acceptable accuracy. Lax–Wendroff fails completely to capture the correct structure of the composite: not only does it introduce oscillations behind the displacement front, but the propagation speed is

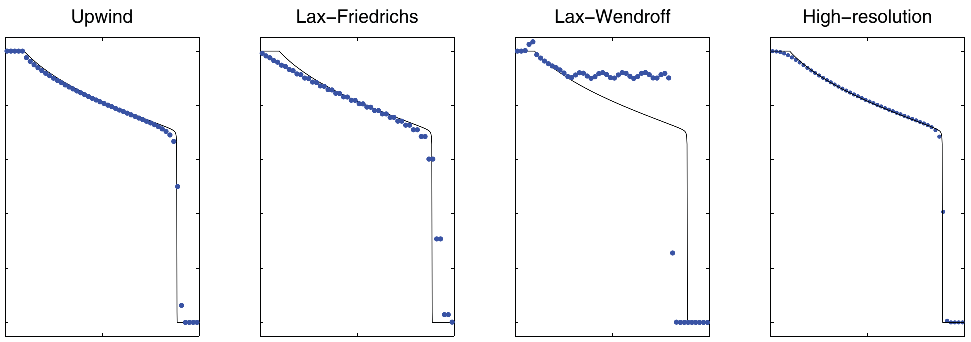

Figure 9.3 Approximate solutions at time $t = 0.65$ for the Buckley–Leverett problem on a grid with 50 cells computed by three classical schemes.

also incorrect. The reason is that once overshoots are introduced near the leading discontinuity, the only way to maintain a mass-conservative solution is if the numerically computed wave (with overshoots) travels slower than the true wave. Likewise, the scheme introduces an overshoot at the kink where the solution has discontinuous derivative. These are serious deficiencies typical of higher-order classical schemes. The high-resolution scheme computes a qualitatively correct solution and overall delivers the best accuracy. On the other hand, it is more computationally costly and would be outperformed by the upwind scheme if we compare accuracy versus computational cost.

Computer exercises

9.3.1 Try to set $\Delta x = \Delta t$ so that the CFL number equals one exactly for the periodic advection problem in Example 9.3.1. Can you explain what you observe? Why is using a CFL number identical to one not that interesting from a practical point of view?

9.3.2 Implement a Godunov solver that works correctly for strictly convex or strictly concave flux functions. Hint: try to find a formula that gives the correct self-similar Riemann solution along the line $x/t = 0$.

9.3.3 Implement the single-point upstream mobility-weighting scheme (9.21) and use it to simulate the 1D cases discussed in Examples 8.4.2–8.4.4.

9.3.4 Extend the schemes introduced in this section to the two-dimensional conservation law $u_t + f(u)_x + g(u)_y = 0$, and implement a solver that can run a 2D analogue of the periodic advection problem in Example 9.3.1. Try to make the changes in the existing solver as small as possible. Hint: In 2D, the Lax–Friedrichs scheme reads,

$$\begin{aligned} u_{ij}^{n+1} &= \tfrac{1}{4}\big(u_{i+1,j}^n + u_{i-1,j}^n + u_{i,j+1}^n + u_{i,j-1}^n\big) \\ &\quad - \frac{1}{2} r\big[f(u_{i+1,j}^n) - f(u_{i-1,j}^n)\big] - \frac{1}{2} r\big[g(u_{i,j+1}^n) - g(u_{i,j-1}^n)\big]. \end{aligned}$$

9.3.5 The `nummet` folder of the `book` module also contains another high-resolution scheme, the semi-discrete central-upwind scheme Kurganov et al. [174]. Use the supplied scripts to also familiarize yourself with this scheme.

9.3.4 Implicit Schemes

So far, we have only considered schemes based on explicit temporal discretization, since this is the most common approach in the literature on numerical methods for hyperbolic conservation laws. In principle, we could also have used *implicit* temporal discretization, as we have already encountered in Chapter 7. Going back to (9.8), we could approximate the integral using point values at t_{n+1} rather than at time t_n, which would give us a numerical method on the form,

$$u_i^{n+1} + r_i^n \big(F_{i+1/2}^{n+1} - F_{i-1/2}^{n+1}\big) = u_i^n. \tag{9.22}$$

To get a specific scheme, we must specify how the cell averages $u_{i-p}^{n+1}, \ldots, u_{i+q}^{n+1}$ are used to evaluate the numerical flux $F_{i+1/2}^{n+1}$. The standard approach in reservoir simulation is to use the upstream mobility-weighting approximation from (9.21) with S^n replaced by S^{n+1}.

In the explicit method (9.9), the flux terms were written on the right-hand side to signify that they are given in terms of the known solution u_i^n. In (9.22), the numerical flux terms appear on the left-hand side to signify that they depend on the unknown cell averages u_i^{n+1}. Equation (9.22) hence represents a coupled system of *nonlinear* discrete equations, which typically must be solved by use of a Newton–Raphson method as discussed in Section 7.1. Computing a single time step of an implicit method is typically significantly more costly than computing an explicit update, since the latter involves fewer evaluations of the flux function and its derivatives. On the other hand, the fully implicit discretization (9.22) is *unconditionally stable* in the sense that there is no stability condition (CFL condition) limiting the size of the time step. In principle, this means that the additional computational cost can be offset by using much larger time steps. In practice, however, viable time-step sizes are limited by numerical errors (as we will see shortly) and by the convergence of the numerical method used to solve the nonlinear system. For Newton–Raphson-type methods, in particular, the convergence may be quite challenging if the flux function contains inflection points/lines separating regions of different convexity; see e.g., [147, 306, 213] for a more detailed discussion.

For 1D horizontal displacements, we have already seen that the flux function has a characteristic S-shape with positive derivatives. Like in the explicit case, we can therefore use one-sided values to evaluate the flux integrals, giving the following scheme,

$$u_i^{n+1} + r_i^n \big[f\big(u_i^{n+1}\big) - f\big(u_{i-1}^{n+1}\big)\big] = u_i^n.$$

The result is a triangular nonlinear system with one nonzero band below the diagonal. For cases like the Buckley–Leverett displacement considered in Example 9.3.3, we can

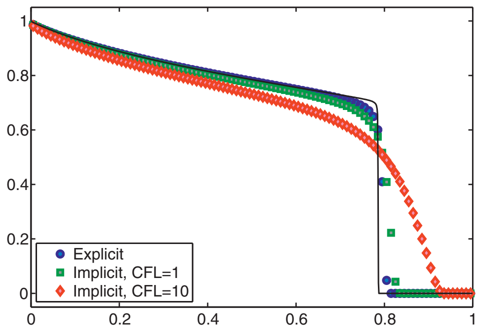

Figure 9.4 Approximate solutions at time $t = 0.65$ for the Buckley–Leverett problem on a grid with 100 cells computed by the single-point upstream mobility-weighting scheme with explicit and implicit time discretization.

therefore solve the nonlinear system very efficiently using a substitution method. That is, we start at the cell next to the inflow boundary,

$$u_1^{n+1} + r_1^n f(u_1^{n+1}) = u_1^n + r_1^n f_L,$$

where f_L is the known inflow. This single nonlinear equation can be solved robustly for an arbitrarily large Δt by use of a classical bracketing method that tracks the end points of an interval containing the unknown root. Once u_1^{n+1} has been computed, we can move to the next cell, and solve a similar scalar equation,

$$u_2^{n+1} + r_2^n f(u_2^{n+1}) = u_1^n + r_2^n f(u_1^{n+1}),$$

and so on. The same strategy can, in fact, be applied to the multidimensional case under certain assumptions guaranteeing that the system has a similar cocurrent flow property; see [220] for more details.

Example 9.3.4 *Let us revisit the setup from Example 9.3.3 and compare the explicit (9.9) and the implicit (9.22) discretizations. Instead of implementing the 1D implicit scheme, we rely on the general transport solvers from the* `incomp` *module in MRST, which will be discussed in more detail in the next chapter. Figure 9.4 shows solutions computed by the explicit scheme with a unit CFL number and by the implicit scheme with CFL numbers 1 and 10. At a unit CFL number, the explicit scheme resolves the displacement front somewhat sharper than the implicit scheme. When the CFL number is increased ten times for the implicit scheme, the numerical smearing is significant and will lead to the prediction of a (much) too early water breakthrough.*

To explain the difference in numerical smearing, let us do some simple numerical analysis. However, rather than studying the nonlinear Buckley–Leverett model, we go back to the simple advection equation, $u_t + au_x = 0$, and consider the schemes discussed earlier in this section as finite-difference schemes rather than finite-volume schemes. The classical way of analyzing numerical schemes is to first compute the truncation error, which is obtained by inserting the exact solution into the numerical scheme, assume a smooth solution, and use Taylor expansions to express the various terms around a common point (x,t). In addition, we also use the fact that $u_t = -au_x$ and that Δt and Δx are proportional. Starting with the explicit scheme,

$$\begin{aligned}
0 &= \frac{u(x,t+\Delta t) - u(x,t)}{\Delta t} + a\frac{u(x,t) - u(x-\Delta x,t)}{\Delta x} \\
&= u_t(x,t) + \tfrac{1}{2}\Delta t\, u_{tt}(x,t) + \mathcal{O}(\Delta t^2) + au_x(x,t) - \tfrac{1}{2}a\Delta x\, u_{xx}(x,t) + \mathcal{O}(\Delta x^2) \\
&= u_t + \tfrac{1}{2}\Delta t\, a^2 u_{xx} + \mathcal{O}(\Delta t^2) + au_x - \tfrac{1}{2}a\Delta x\, u_{xx} + \mathcal{O}(\Delta x^2) \\
&= u_t + au_x - \tfrac{1}{2}a\big[\Delta x - a\Delta t\big]u_{xx} + \mathcal{O}(\Delta x^2).
\end{aligned}$$

In other words, the explicit scheme will up to order $\mathcal{O}(\Delta x^2)$ solve a *parabolic* equation. This equation is only well-posed if the coefficient in front of the second-order term is negative, implying that $a\Delta t/\Delta x \leq 1$, which is exactly the same requirement as in the CFL condition. We also notice that the higher the CFL number we use, the less numerical diffusion we introduce in the approximate solution and in the special case that $a\Delta t = \Delta x$, the scheme has no numerical diffusion.

For the implicit scheme, we only need to expand the u_{i-1}^{n+1} term since we have already computed the expansion for u_i^{n+1} for the explicit scheme,

$$\begin{aligned}
u_{i-1}^{n+1} &= u(x-\Delta x, t+\Delta t) \\
&= u(x-\Delta x,t) - a\Delta t u_x(x-\Delta x,t) + \tfrac{1}{2}(a\Delta t)^2 u_{xx}(x-\Delta x,t) + \mathcal{O}(\Delta t^3) \\
&= u - \Delta x u_x + \tfrac{1}{2}\Delta x^2 u_{xx} + \mathcal{O}(\Delta x^3) - a\Delta t\partial_x\big[u - \Delta x u_x + \mathcal{O}(\Delta x^2)\big] \\
&\quad + \tfrac{1}{2}(a\Delta t)^2\partial_x^2\big[u + \mathcal{O}(\Delta x)\big] + \mathcal{O}(\Delta t^3).
\end{aligned}$$

Collecting terms, we have that

$$\begin{aligned}
0 &= \frac{u(x,t+\Delta t) - u(x,t)}{\Delta t} + a\frac{u(x,t) - u(x-\Delta x,t)}{\Delta x} \\
&= u_t + au_x - \tfrac{1}{2}a\big[\Delta x + a\Delta t\big]u_{xx} + \mathcal{O}(\Delta x^2).
\end{aligned}$$

This means that the implicit scheme also will solve a parabolic equation with accuracy $\mathcal{O}(\Delta x^2)$. In this case, the coefficient in front of the u_{xx} term is always negative and the scheme is therefore stable regardless of the choice of Δt. However, this also means that the numerical diffusion *increases* with increasing time steps.

One can also conduct a similar analysis in the nonlinear case and arrive at similar conclusions. This in turn implies an interesting observation if we look back at our discussion of linear waves on page 280. For an explicit scheme, slow waves are computed with *more* numerical diffusion than fast waves. For an implicit scheme, on the other hand, slow waves are computed with *less* numerical diffusion than fast waves. Explicit schemes are therefore best suited for systems having relatively small differences in wave speeds, whereas implicit schemes are better suited for systems with large differences in wave speeds. In a typical reservoir setting, wave speeds are large in the near-well region, of medium size in high-flow regions away from the wells, and low near or in stagnant zones. It is therefore reasonable to expect that implicit schemes are best suited for simulating real reservoir systems. However, explicit schemes are still used, mainly because of their simplicity and in connection with higher-resolution discretizations.

COMPUTER EXERCISES

9.3.1 Implement the implicit upwind scheme in 1D and verify against the solver from the `incomp` module.

9.3.2 Use the single-point upstream mobility-weighting scheme (9.21) to extend your implicit solver to also account for countercurrent flow and use it to simulate the 1D cases discussed in Examples 8.4.2–8.4.4. Compare with the corresponding explicit scheme.

9.4 Discretization on Unstructured Polyhedral Grids

So far, this chapter has introduced you to various types of numerical methods that can be used to discretize saturation equations. The methods discussed, and some of their high-resolution extensions included in the `book` module, are all reasonably simple to implement on regular Cartesian and rectilinear grids and on grids consisting entirely of simplices (triangles in 2D and tetrahedrons in 3D). However, as you have seen multiple times throughout the book, realistic grid models are rarely that simple. For the baseline solvers in MRST we have therefore chosen to go with the most robust choice of them all, namely the explicit or implicit version of the single-point upstream mobility weighting scheme. This section therefore discusses in more detail how to formulate this scheme on unstructured, polyhedral grids for a general transport equation

$$\phi \frac{\partial S}{\partial t} + \nabla \cdot \vec{H}(S) = 0. \tag{9.23}$$

Here, the flux function $\vec{H}$ includes viscous flow driven by pressure gradients, as well as gravity segregation and capillary forces,

$$\begin{aligned}\vec{H}(S) &= \frac{\lambda_w}{\lambda_w + \lambda_n}\vec{v} + \frac{\lambda_w \lambda_n \mathbf{K}}{\lambda_w + \lambda_n}\big(\Delta\rho\, \vec{g} + \nabla P_c(S)\big) \\ &= \vec{H}_f(S) + \vec{H}_g(S) + \vec{H}_c(S).\end{aligned} \tag{9.24}$$

To discretize this equation, we start from a multi-dimensional analogue of the integral form (9.8) of the conservation law,

$$S_i^{n+1} - S_i^n = \frac{1}{\phi_i |\Omega_i|} \sum_k \int_{t_n}^{t_{n+1}} \int_{\Gamma_{ik}} \vec{H}\big(S(\vec{x},t)\big) \cdot \vec{n}_{i,k}\, ds\, dt, \tag{9.25}$$

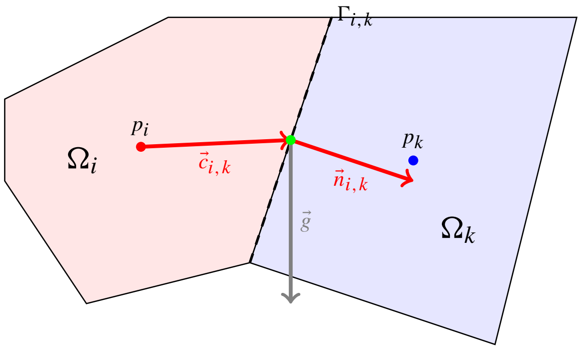

Figure 9.5 Two cells used to define the finite-volume discretization of the general two-phase transport equation (9.23). Here, the interface $\Gamma_{i,k}$ between the two cells Ω_i and Ω_k has normal vector $\vec{n}_{i,k}$ and area $A_{i,k}$.

defined over a general grid cell Ω_i as illustrated in Figure 9.5. We do not want to develop a higher-order discretization, and hence it is sufficient to evaluate the time integral at one of its end-point, so that (9.25) simplifies to

$$S_i^{n+1} - S_i^n = \frac{\Delta t}{\phi_i |\Omega_i|} \int_{\Gamma_{ik}} \vec{H}\big(S(\vec{x}, t_m)\big) \cdot \vec{n}_{i,k}\, ds, \qquad m = n, n+1. \tag{9.26}$$

Before we describe the discretization of the three flux functions H_f, H_g, and H_c, let us quickly recap how we discretize the viscous flux $\vec{v} = -\mathbf{K}\nabla p$ in the corresponding pressure equation. Referring to Figure 4.10 on page 133, the flux reads

$$\begin{aligned} v_{i,k} &\approx A_{i,k}\mathbf{K}_i \frac{(p_i - \pi_{i,k})\vec{c}_{i,k}}{|\vec{c}_{i,k}|^2} \cdot \vec{n}_{i,k} = T_{i,k}(p_i - \pi_{i,k}) \\ &= \big[T_{i,k}^{-1} + T_{k,i}^{-1}\big]^{-1}(p_i - p_k), \end{aligned}$$

where the first line gives the one-sided formulation computed entirely from quantities associated with cell i, and the second line gives the formulation that also couples to the neighboring cell k. From this expression, it follows naturally that the capillary term should be discretized as follows:

$$A_{i,k}\nabla P_c(S) \cdot \vec{n}_{i,k} \approx \big[T_{i,k}^{-1} + T_{k,i}^{-1}\big]^{-1}\big[P_c(S_i) - P_c(S_k)\big] = P_{i,k}(S). \tag{9.27}$$

Likewise, we can define a "gravity flux" g_{ik} that is independent of saturation,

$$g_{ik} = \big[g_{i,k}^{-1} + g_{k,i}^{-1}\big]^{-1}, \qquad \begin{aligned} g_{i,k} &= (\Delta\rho)|_{\Omega_i}\,(\mathbf{K}_i\vec{g}) \cdot \vec{n}_{i,k}, \\ g_{k,i} &= (\Delta\rho)|_{\Omega_k}\,(\mathbf{K}_k\vec{g}) \cdot \vec{n}_{k,i}. \end{aligned} \tag{9.28}$$

With this, all we need to do is to find the correct upstream value λ_α^u for each of the two phases, and then the overall approximation of the interface flux reads:

$$H_{ik} = \frac{\lambda_w^u}{\lambda_w^u + \lambda_n^u} v_{ik} + \frac{\lambda_w^u \lambda_n^u}{\lambda_w^u + \lambda_n^u}\big[g_{ik} + P_{ik}\big]. \tag{9.29}$$

To determine the upstream directions, we need to compare the Darcy flux v_{ik} and the gravity flux g_{ik}. (If there are capillary forces, these are henceforth assumed to be added to the gravity flux, i.e., $g_{ik} + P_{ik} \to g_{ik}$.) If v_{ik} and g_{ik} have the same signs, we know that the correct choice of the wetting mobility is from the left (right) of the interface if the fluxes are positive (negative). Likewise, when the two fluxes have different signs, we know that the correct choice of upstream value for the non-wetting fluid is independent of the actual mobility values. These cases can be summarized in the following tabular:

$\text{sign}(v_{ik})$	$\text{sign}(g_{ik})$	λ_w^u
≥ 0	≥ 0	$\lambda_w(S_L)$
≤ 0	≤ 0	$\lambda_w(S_R)$

$\text{sign}(v_{ik})$	$\text{sign}(g_{ik})$	λ_n^u
≥ 0	≤ 0	$\lambda_n(S_L)$
≤ 0	≥ 0	$\lambda_n(S_R)$

If neither of these cases are fulfilled, we need to check the sign of the phase fluxes to determine the correct upstream direction. In practice, we do not check the phase fluxes themselves, but rather the phase fluxes divided by the fractional flow functions as these quantities are less expensive to compute. That is, we check the sign of $v_{ik} + g_{ik}\lambda_n$ and $v_{ik} - g_{ik}\lambda_w$ and use this to pick the correct upstream value.

For most cases in reservoir simulation, we also have source terms that drive flow, so that (9.23) is replaced by an inhomogeneous equation

$$\phi\frac{\partial S}{\partial t} + \nabla\cdot\vec{H}(S) = \max(q,0) + \min(q,0)\,f(S).$$

In the next chapter, we describe how the resulting transport solvers are implemented in MRST, and discuss how to combine them with `incompTPFA`, or one of its consistent alternative, in a sequential solution procedure to solve incompressible, multiphase flow on general polygonal and polyhedral grids.