3

Numerical hydrodynamics

In Chapter 2 we have already discussed the Eulerian and Lagrangian formulations of the equations of fluid mechanics. Now we want to describe some of the numerical solution methods that have been developed to solve them, and in particular, in the more challenging 2-D and 3-D cases. The present discussion will focus on hydrodynamics without radiation, and the methods for treating coupled hydrodynamics and radiation will be mentioned later, in Chapter 11.

There are excellent references on this subject, among which are the classic book by Richtmyer and Morton (1967), the text by Bowers and Wilson (1991), the collection of papers by Norman and coworkers (Stone and Norman, 1992a,b; Stone, Mihalas and Norman, 1992), van Leer (1979), Colella and Woodward (1984), Caramana and coworkers (Caramana and Whalen, 1998; Caramana and Shashkov, 1998; Caramana, Shashkov, and Whalen, 1998; Caramana, Burton, Shashkov, and Whalen, 1998) and Jiang and Shu (1996).

3.1 Lagrangian methods

3.1.1 Staggered-mesh hydrodynamics for 1-D slab geometry

The mother of all numerical hydrodynamics methods has the name von Neumann–Richtmyer staggered-mesh hydrodynamics. In its simplest incarnation, for 1-D slab geometry, it is described as follows. The material of the problem is divided into N zones with fixed masses, divided by $N-1$ material interfaces. Including the outer boundaries, that makes $N+1$ interfaces in all. The basic set of unknowns is the list of z coordinates of these interfaces: z_I, $I = 1, \ldots, N+1$. Time is discretized as well, and these interface positions are to be found at a succession of times: $t_1, t_2, \ldots$. The position of interface I at time $t = t_n$ is denoted by z_I^n. The zone between interface I and interface $I+1$ carries the index $I+1/2$, and its fixed amount of mass (per unit area) is $m_{I+1/2}$. From the zone thickness and the

mass comes the mass density, $\rho^n_{I+1/2} = m_{I+1/2}/(z^n_{I+1} - z^n_I)$. There are also zone-centered values of internal energy $e^n_{I+1/2}$, temperature $T^n_{I+1/2}$ and pressure $p^n_{I+1/2}$.

The other basic set of unknowns consists of the interface velocities. These carry the same spatial indexing as the interfaces themselves, but are staggered in time by half a time step relative to the positions, so that $u_I^{n+1/2}$ represents the velocity of interface I in the time interval t_n to t_{n+1}. Thus one of the time-integration equations is

$$\frac{z_I^{n+1} - z_I^n}{t_{n+1} - t_n} = u_I^{n+1/2}. \tag{3.1}$$

In order to find the evolution of u_I we need to relate it to the pressure gradient, etc. Because of the staggering in time, the acceleration that changes $u_I^{n-1/2}$ to $u_I^{n+1/2}$ is centered in time at t_n, and therefore it can be calculated from the information available at that time:

$$2\frac{u_I^{n+1/2} - u_I^{n-1/2}}{t_{n+1} - t_{n-1}} = 2\frac{p^n_{I-1/2} - p^n_{I+1/2}}{m_{I-1/2} + m_{I+1/2}}. \tag{3.2}$$

This has assumed that the velocity $u_I^{n+1/2}$ is centered in the middle of the time interval $[t_n, t_{n+1}]$, and that the baseline for the pressure gradient is from the midpoint of zone $I - 1/2$ to the midpoint of zone $I + 1/2$. This equation has a flaw that will be mentioned shortly. The pressure $p^n_{I+1/2}$ must be calculated from the equation of state using the density $\rho^n_{I+1/2}$ and either the internal energy or temperature, and therefore the energy equation will be needed.

The time-centered internal energy equation is (from (2.4))

$$\left(e^{n+1}_{I+1/2} - e^n_{I+1/2}\right) + \frac{1}{2}\left(p^n_{I+1/2} + p^{n+1}_{I+1/2}\right)\left(\frac{1}{\rho^{n+1}_{I+1/2}} - \frac{1}{\rho^n_{I+1/2}}\right) = 0. \tag{3.3}$$

This is an implicit equation for $e^{n+1}_{I+1/2}$ since $p^{n+1}_{I+1/2}$ depends on it, or, equivalently, both depend on $T^{n+1}_{I+1/2}$. The work flow is the following. The pressure data at t_n are used to update the velocities to $t_{n+1/2}$; the new velocities are then used to update the interface positions and densities to t_{n+1}. The densities being known, the internal energy equations may be solved for the $e^{n+1}_{I+1/2}$ values. With a gamma-law equation of state $p^{n+1}_{I+1/2} = (\gamma - 1)\rho^{n+1}_{I+1/2}e^{n+1}_{I+1/2}$ this can be done directly, otherwise a Newton–Raphson procedure may be needed. Once the new internal energies are known, the new pressures follow, and the time step is complete.

In the limit of infinitesimal time steps the internal energy equation above becomes the equation for an isentrope, and in that limit the entropy of zone $I + 1/2$ would never change. This causes a problem. We know that the Euler equations can produce shocks, and that the entropy must increase across a shock. Therefore the

finite-difference equations as written so far will give the wrong answer for a flow containing a shock. Richtmyer and Morton (1967) illustrate the numerical results in this case: in what would be the postshock region there are huge zone-to-zone fluctuations of the flow velocity and pressure. The Navier–Stokes equations can give a smooth transition across what would be a shock for the Euler equations, and this is the key to the successful numerical method. A dissipative term, a pseudo-viscous pressure q, must be added to the momentum and energy equations. In those cases, the majority, for which the Euler equations are a good approximation, the pseudo-viscous pressure is much larger than the true viscous stress. It does not matter very much exactly what q is, provided that: (1) it is positive, (2) it is negligible away from shocks, (3) it is included in the momentum and energy equations in such a way that total energy is still conserved. It then acts, like true viscosity, to turn kinetic energy of fluid motion into heat. The von Neumann–Richtmyer prescription is

$$q^n_{I+1/2} = C_Q \rho^n_{I+1/2} \max\left(u_I^{n-1/2} - u_{I+1}^{n-1/2}, 0\right)^2. \tag{3.4}$$

The factor C_Q is a constant that may be 1 or 2 or whatever value seems to give good results. It may be noted that the velocity is half a time step behind the other factors in the expression for $q^n_{I+1/2}$. This is a necessity since the advanced velocity is not available when q must be computed. Lagging the velocity is found not to have a deleterious effect. The pseudo-viscosity spreads a shock over about $\pi\sqrt{C_Q}$ zones, so a larger C_Q means a smoother, less noisy calculation, while a smaller C_Q gives improved resolution.

When pseudo-viscosity is added the momentum and internal energy equations become

$$2\frac{u_I^{n+1/2} - u_I^{n-1/2}}{t_{n+1} - t_{n-1}} - 2\frac{p^n_{I-1/2} + q^n_{I-1/2} - p^n_{I+1/2} - q^n_{I+1/2}}{m_{I-1/2} + m_{I+1/2}} = 0, \tag{3.5}$$

$$\left(e^{n+1}_{I+1/2} - e^n_{I+1/2}\right) + \frac{1}{2}\left(p^n_{I+1/2} + q^n_{I+1/2} + p^{n+1}_{I+1/2} + q^{n+1}_{I+1/2}\right) \times \left(\frac{1}{\rho^{n+1}_{I+1/2}} - \frac{1}{\rho^n_{I+1/2}}\right) = 0. \tag{3.6}$$

It can be shown that with this finite-difference system the discrete total energy of the problem

$$\mathcal{E}_{\text{tot}} = \sum_I \left[\frac{1}{2}(m_{I-1/2} + m_{I+1/2})\frac{1}{2}(u_I)^2\right] + \sum_I m_{I+1/2} e_{I+1/2} \tag{3.7}$$

is precisely conserved in the limit of short time steps, provided the boundary conditions are suitable: rigid boundaries or zero-pressure boundaries.

When we speak of boundary conditions, we recall that they must, of course, be included. When the finite-difference equations would incorporate data that do not exist, since a boundary intervenes, either the missing data are generated from the specified boundary values, or that particular difference equation is replaced by a constraint. Thus at a rigid boundary the acceleration need not be computed since the velocity is forced to be zero. At a free boundary, where the pressure vanishes, the pressure of the phantom zone outside the boundary is obtained by extrapolating from the zone adjacent to the boundary on the inside and using the condition that the pressure at the boundary interface should vanish.

A very important limitation of all the methods of numerical hydrodynamics we will discuss is that the computed solution diverges wildly from the right answer if the time step is excessive in comparison with the zone sizes. This is the Courant–Friedrichs–Lewy condition, Courant or CFL for short (Courant, Friedrichs and Lewy, 1928). A linearized 1-D Lagrangian problem can help motivate the condition. Consider this idealized system:

$$\frac{z_I^{n+1} - z_I^n}{\Delta t} = u_I^{n+1/2}, \tag{3.8}$$

$$\frac{u_I^{n+1/2} - u_I^{n-1/2}}{\Delta t} = a^2 \left(\frac{1}{z_I^n - z_{I-1}^n} - \frac{1}{z_{I+1}^n - z_I^n} \right), \tag{3.9}$$

in which a is the effective speed of sound. Now we perturb around a state in which the material is at rest and the zone size is uniform and equals Δz. Denoting the perturbations to the interface positions by δz_I^n leads to these linearized equations:

$$\frac{\delta z_I^{n+1} - \delta z_I^n}{\Delta t} = u_I^{n+1/2}, \tag{3.10}$$

$$\frac{u_I^{n+1/2} - u_I^{n-1/2}}{\Delta t} = \frac{a^2}{(\Delta z)^2} \left(\delta z_{I-1}^n - 2\delta z_I^n + \delta z_{I+1}^n \right). \tag{3.11}$$

We now look for a solution of this homogeneous linear system in which both the velocities and the interface positions are proportional to $g^n \exp(i\theta I)$, where θ is a specified angle between $-\pi$ and π, and g is an amplification ratio to be found. When the factors are eliminated this quadratic equation is found for g:

$$g^2 - 2\left(1 - 2C^2 \sin^2 \frac{\theta}{2}\right) g + 1 = 0, \tag{3.12}$$

where C stands for the Courant ratio $a\Delta t/\Delta z$. This equation has a pair of complex conjugate roots with $|g| = 1$ provided $C \sin\theta/2 \leq 1$, but it has real roots, one of which has a magnitude > 1, if $C|\sin\theta/2| > 1$. Since $|\sin\theta/2|$ can be as large as 1, in the case $\theta = \pm\pi$, there will be instability if $C > 1$, the CFL result. The most

unstable mode is the one in which the fluctuations alternate in sign at consecutive zones. In practice, a smaller limit is set for C, such as 0.5. This accounts for nonlinear instability effects. Richtmyer and Morton (1967) demonstrate how the upper limit for C depends on shock strength.

The solution for the amplification factor in the continuum limit $\Delta z \to 0$, $\Delta t \to 0$ can be used as a check on the accuracy of the numerical solution. The value of g^n for the analytic solution is $\exp(i\omega t) = \exp(ikat) = \exp(ika\Delta tn)$, and therefore g should be $\exp(ika\Delta t) = \exp(iC\theta)$ when we identify θ with $k\Delta z$. In other words, for the analytic solution

$$g = 1 + iC\theta - \frac{1}{2}C^2\theta^2 - \frac{1}{6}iC^3\theta^3 + O(C^4\theta^4). \tag{3.13}$$

The expansion of the numerical amplification factor in powers of θ is

$$g = 1 + iC\theta - \frac{1}{2}C^2\theta^2 - iC\frac{3C^2+1}{24}\theta^3 + O(\theta^4). \tag{3.14}$$

The expansions agree through terms of order θ^2, and so the numerical method is second order accurate. The third order terms differ except in the case $C = 1$. In that marginally-stable circumstance the numerical method is actually exact.

This is the essence of 1-D staggered-mesh Lagrangian hydrodynamics. The variations come in slight changes in the way the time-centered factors are constructed, and variations in the pseudo-viscosity. Sometimes a linear pseudo-viscosity is used:

$$q^n_{I+1/2} = C_Q a^n_{I+1/2}\rho^n_{I+1/2}\max\left(u^{n-1/2}_I - u^{n-1/2}_{I+1}, 0\right), \tag{3.15}$$

in which the factor $a^n_{I+1/2}$ is of the order of the sound speed in the zone. The linear pseudo-viscosity is quite a bit more dissipative for weak shocks, and for no shock at all, than the quadratic one. This makes the calculation quite smooth, perhaps spuriously so. The two examples of pseudo-viscosity we have seen so far are nonzero for zones being compressed (as in a shock) and vanish for zones in expansion. A further step in the direction of smoothness is to let q for a zone in expansion be the negative of the q with the same velocity difference in compression. This is usually undesirable. We will see a type of q later that vanishes except in shocks of at least moderate strength; this is the so-called monotonic q.

3.1.2 Staggered-mesh hydrodynamics in two dimensions – quadrilateral zones

In 2-D Lagrangian hydrodynamics with quadrilateral zones the mesh consists of a more-or-less distorted map of a uniform Cartesian grid. This underlying logical

mesh has the coordinates k and ℓ, and the mesh spacing in each coordinate is unity. The nodes of the logical mesh have integer values of k and ℓ, the mesh consists of k-lines with integer values of k and variable ℓ and ℓ-lines with the reverse. The zones have half-integer values of both k and ℓ. Each zone is supposed to have a fixed mass. The fluid velocity is centered at the nodes, and staggered in time from the other quantities. The time advancement proceeds by: moving the nodes according to their current velocities, computing the new area of all the zones and thereby getting the new densities; next applying the internal energy equation to find the new internal energy and pressure in each zone; finally using the pressures in the zones surrounding a given node to compute the acceleration of the node and obtain the time-advanced nodal velocity.

The first respect in which the 2-D case is harder than the 1-D case is computing the zone area, which is needed to obtain the density. The following is one way to obtain the result. We suppose that our problem has xy geometry, i.e., it has translational invariance in the z direction. Our coordinates are x and y, and we can speak of the position vector $\mathbf{r} = (x, y)$. The zone area is given by

$$A = \iint_Z dxdy = \iint_R \left| \frac{\partial(x, y)}{\partial(k, \ell)} \right| dkd\ell, \tag{3.16}$$

in which Z represents the physical zone, and R represents the corresponding logical zone. Here we are formally using the coordinates in logical space, the indices, as real variables, and the Jacobian of the mapping from logical space to physical space appears. Of course, we do not actually know that mapping except at the integer logical mesh nodes. For definiteness let us consider the zone bounded by the mesh lines $k = 0$, $k = 1$, $\ell = 0$ and $\ell = 1$. Within this zone let us also assume a bilinear mapping

$$\mathbf{r} = \mathbf{r}_{00}(1 - k)(1 - \ell) + \mathbf{r}_{10}k(1 - \ell) + \mathbf{r}_{01}(1 - k)\ell + \mathbf{r}_{11}k\ell, \tag{3.17}$$

where the vectors $\mathbf{r}_{00}$, $\mathbf{r}_{10}$, $\mathbf{r}_{01}$, and $\mathbf{r}_{11}$ are the four corners of the quadrilateral. The bilinear mapping is consistent with the quadrilateral shape since the $k = 0$ and $k = 1$ lines and $\ell = 0$ and $\ell = 1$ lines will be straight lines connecting the corners. The Jacobian is $J = \mathbf{r}_k \times \mathbf{r}_\ell$, where the z component of the vector cross product is needed, and the subscripts k and ℓ indicate partial differentiation by that variable. But since $\mathbf{r}_k$ is a linear function of ℓ alone, and likewise $\mathbf{r}_\ell$ is a linear function of k alone, the average of J is the cross product of the average $\mathbf{r}_k$ with the

average $\mathbf{r}_\ell$. This gives

$$A = \left| \frac{1}{4}(\mathbf{r}_{10} - \mathbf{r}_{00} + \mathbf{r}_{11} - \mathbf{r}_{01}) \times (\mathbf{r}_{01} - \mathbf{r}_{00} + \mathbf{r}_{11} - \mathbf{r}_{10}) \right|$$
$$= \frac{1}{2}|(\mathbf{r}_{10} - \mathbf{r}_{01}) \times (\mathbf{r}_{11} - \mathbf{r}_{00})|. \tag{3.18}$$

This can be arranged in a variety of ways; the last expression is half the cross product of the two diagonals of the quadrilateral. There is a fallacy in this calculation, which pops up if the sign of the Jacobian changes within the zone. Such zones are called "bowtie" zones, since they look like two triangles that touch at a single point. The calculation above gives the *net* area of the two triangles, treating one as positive and the other as negative, not the sum of the areas. Lagrangian hydrodynamics codes generally crash when a bowtie zone occurs, which is not infrequent unless steps are taken to prevent it.

The next challenge is to calculate the accelerations of the nodes, which depend on a pressure gradient. One way to do this is the following. We focus our attention on a particular node and construct around it a "dual zone" made up of pieces of all the zones surrounding that node. In a quadrilateral mesh there are four surrounding zones in general. For example, we can construct a contour C that joins the midpoint of one zone edge with the center of the adjacent zone, then on to the midpoint of the nest edge around, then the next zone center, and so on back to the starting point. This contour encloses roughly one quarter of each of the surrounding zones. We call this enclosed region the dual zone and denote it by Z'. Its area and mass can be found in the same way as the area calculation above. Next we apply a form of the divergence theorem to this dual zone:

$$\iint_{Z'} \nabla p \, dA = \oint_C p\mathbf{n} \, ds = -\mathbf{e}_z \times \oint_C p \, d\mathbf{s}, \tag{3.19}$$

in which $\mathbf{n}$ is the unit outward normal vector to the zone at points along C, ds is the element of arc length along C, $\mathbf{e}_z$ is the unit vector in the z direction and $d\mathbf{s}$ is the vector path element for traversing C counterclockwise. If the last integral is broken into pieces belonging to each of the surrounding zones, and if we approximate p as constant within each of these zones, then the integral for each piece is just the pressure of the zone times the vector joining the midpoint of one edge to the midpoint of the next edge. So finally we get the integral of ∇p over the whole dual zone. The negative of this divided by the dual zone mass is the nodal acceleration.

For a uniform or very smooth mesh this approximation for ∇p is second order. Unfortunately, when the mesh is severely distorted the accuracy is not at all good.

Specifically, if a Taylor expansion of the pressure is made in the region surrounding a node, and this is used to evaluate the gradient using the line integral formula, then it is found that the second order terms introduce an error in the derived gradient unless the mesh is orthogonal and regular, where "regular" means that the zone shape and dimensions are slowly varying in the mesh; i.e., the changes of zone dimensions between adjacent zones are themselves second order. This is a limitation that many Lagrangian codes suffer with, although some improvements have been made (*viz*., the work of Caramana and coworkers (Caramana and Whalen, 1998; Caramana and Shashkov, 1998; Caramana, Shashkov, and Whalen, 1998; Caramana, Burton, Shashkov, and Whalen, 1998).

The final complicated ingredient is the pseudo-viscosity. There is considerable ambiguity involved in the centering of q in multidimensional problems. The difference between the vector velocities of the nodes (k, ℓ) and $(k+1, \ell)$, projected on the edge connecting those nodes, leads to a q that is naturally centered at the midpoint of that edge. This would have to be averaged with the similar quantity defined for the $\ell + 1$ side of the quadrilateral to obtain a zone-centered q that can be added to the normal pressure in the zone. Clearly, there are many alternatives to this prescription. It is also possible to use a different q in computing the x acceleration than when computing the y acceleration.

3.1.3 Unstructured meshes

The logic for calculating ∇p above is not critically dependent on having exactly four zones around each node, nor of each zone having exactly four edges. In an unstructured mesh code the mesh consists of an arbitrary assortment of polygons (in two dimensions). There may be any number of edges meeting at a node, and a zone will have as many neighbors as it has edges, whatever that may be. The area calculation is simple enough if the zone is first broken into triangles. The algorithm for finding ∇p works well once the list of zones surrounding the node is constructed.

In codes of this kind the mesh connectivity may be dynamic: as the mesh becomes distorted some of the mesh lines may be removed and/or created. Zones and nodes are not indexed by k and ℓ, which implies quadrilaterals, but form two long lists. Associated with each node is its current list of neighbors, and the list of zones that share the mesh line connecting this node with the neighbors. As connections are made and broken in the course of the calculation, these lists are updated. The indexing overhead and the cost of dealing with many-sided polygons make unstructured mesh codes more costly than quadrilateral-based codes, but there can be advantages in ameliorating mesh distortion.

3.2 Eulerian methods

While in Lagrangian hydrodynamics the coordinates of the mesh are dynamic variables, from which the density is calculated, in Eulerian hydrodynamics the mesh is fixed and the density is one of the primary variables. In each case the other variables are fluid velocity and internal energy or temperature.

The Eulerian methods fall into two groups: finite-difference methods and finite-volume methods. In the former group the flow variables are conceived as being samples at certain points in space and time, and from these sampled values the partial derivatives are computed that are required to obey the Euler equations. For methods of the finite-volume class the unknowns are understood to be average values over certain finite volumes – the zones – and these must obey the conservation laws in integral form. The difference is subtle and the final equations are quite similar for the different viewpoints.

3.2.1 Finite-difference methods; Lagrangian plus advection

This approach is described very well in Bowers and Wilson (1991). The Euler equations are first written in this way

$$\frac{\partial \rho}{\partial t} = -\nabla \cdot (\rho \mathbf{u}), \tag{3.20}$$

$$\frac{\partial \rho \mathbf{u}}{\partial t} = -\nabla p \ - \nabla \cdot (\rho \mathbf{u}\mathbf{u}), \tag{3.21}$$

$$\frac{\partial \rho e}{\partial t} = -p \nabla \cdot \mathbf{u} - \nabla \cdot (\rho e \mathbf{u}). \tag{3.22}$$

In the method of Bowers and Wilson a time step consists of a "Lagrangian" step followed by an advection step. The idea of operator splitting is applied here, which means that for each of these substeps just *part* of the expression for the time derivative of each variable is applied. The unknowns ρ, $\rho\mathbf{u}$ and ρe are updated using those parts of their time derivatives, and the updated values are the starting point for the next partial time step. The philosophy of operator splitting is described further in Section 11.1. For the Lagrangian step the density is left alone, the momentum density is updated using just the pressure gradient, after which a new flow velocity is calculated by division. As in staggered-mesh Lagrangian hydrodynamics, the velocity (and momentum density) components are centered in time at $t_{n+1/2}$. The new velocities are used to calculated $\nabla \cdot \mathbf{u}$ and the internal energy density is updated using just the first term on the right-hand side of (3.22). That completes the Lagrangian step.

The acceleration of the material by the pressure gradient is more easily calculated in this method than in the Lagrangian codes for two reasons. One is the

regularity of the mesh, which lends itself to a very natural expression for ∇p. The other is the frequent choice to center the velocity (and momentum density) components on cell edges. The x momentum is centered at the midpoints of the edges bounding the zones in the x direction, and likewise for the y momentum. This means, for example, that $(p_{k+1/2,\ell+1/2} - p_{k-1/2,\ell+1/2})/\Delta x$ is the natural expression for the gradient to accelerate the x velocity at point $k, \ell + 1/2$.

For the advection step these relations are used:

$$\frac{d}{dt}(\rho V) = -\iint_S \rho \mathbf{u} \cdot d\mathbf{A}, \tag{3.23}$$

$$\frac{d}{dt}(\rho \mathbf{u} V') = -\iint_{S'} \rho \mathbf{u}\mathbf{u} \cdot d\mathbf{A}, \tag{3.24}$$

$$\frac{d}{dt}(\rho e V) = -\iint_S \rho \mathbf{u} e \cdot d\mathbf{A}. \tag{3.25}$$

Here V is a zonal volume and V' is a dual zonal volume, since the velocity components are staggered in space from the quantities like density and internal energy. The surfaces S and S' represent the boundaries of V and V'. The surface integrals become sums over the edges of each zone or dual zone of the advection flux through each edge. The challenge for Eulerian hydrodynamics, to be accurate and minimize numerical diffusion, is the construction of these advection fluxes. Since the flow is in one direction or the other through each edge, there will be a donor zone and a receiver zone. The advection flux is the appropriate edge-centered normal component of velocity multiplied by an appropriately averaged value of the conserved quantity: mass density, momentum density or internal energy per unit volume. Stability requires that the edge-centered conserved density be calculated from the properties of the donor zone. If this is just set equal to the average density of the donor zone, the result is called donor-cell advection. Alas, this method, while very simple, is very diffusive.

The improvement on donor-cell advection is to notice that the contents of the donor zone within a distance $\mathbf{u} \cdot \mathbf{n} \Delta t$ will be advected out of the zone during the time step, where $\mathbf{n}$ is the unit vector normal to the zone edge. If we can construct an interpolation function representing the variation of ρ or the other conserved quantities *within* the zone, then the integral of this interpolation function over the volume that will be advected gives a much better estimate of the advection flux. This concept is now used in virtually all Eulerian codes. One very common interpolation scheme is the monotonic piecewise-linear scheme introduced by van Leer (1977). This is a 1-D interpolation method. Indeed, common practice, as described by Bowers and Wilson (1991), is to operator-split the advection by direction: first

advect in the x direction, then in the y direction. (Or the reverse. Or first one way then the other.) Constructing the piecewise-linear interpolation function in one coordinate, say x, for, let us say, ρ, comes down to finding the slope within each zone of ρ vs x. The goal of monotonic interpolation is to ensure that the zone-edge values of the interpolant do not fall outside the range of ρ between the given zone and its neighbor in that direction. If the given zone is in fact a local extremum, then the slope must be zero. The formula can be illustrated easily if the zoning is uniform. The slope $d\rho/dx$ in zone $k + 1/2$ multiplied by Δx will be $\Delta\rho$ given by $\Delta\rho = 0$ if zone $k + 1/2$ is an extremum, otherwise

$$\Delta\rho = S \min(2|\rho_{k+1/2} - \rho_{k-1/2}|, 2|\rho_{k+3/2} - \rho_{k+1/2}|, 0.5|\rho_{k+3/2} - \rho_{k-1/2}|), \tag{3.26}$$

where S is 1 if $\rho_{k+1/2} - \rho_{k-1/2} > 0$ and -1 if $\rho_{k+1/2} - \rho_{k-1/2} < 0$. The formula for variable zone sizes is only a little more complicated. The interpolation function for zone $k + 1/2$ is then the straight line that passes through $\rho_{k+1/2}$ at zone center with this slope. The advected flux during the time step is then the integral of this function over a distance $u_x \Delta t$ adjacent to the zone edge, as described above.

Staggered-mesh finite-difference Eulerian hydrodynamics, like Lagrangian hydrodynamics, cannot calculate shocks successfully unless pseudo-viscosity is included. The prescriptions for pseudo-viscosity are very much the same, apart from the differences owing to the centering of the velocity components on zone edges rather than at vertices. Van Leer interpolation lends itself to a very useful expression for the pseudo-viscosity, called the monotonic q. The x components of velocity, centered at k, $\ell + 1/2$, can be interpolated in the van Leer fashion, where now the zones become the dual zones that extend from the center of one regular zone to the center of the next. The piecewise interpolants in these dual zones will in general not be continuous at the dual zone edges, i.e., at the regular zone centers. The amount of the discontinuity will typically be very small in smooth regions of the flow, and become large where the velocity has a sharp gradient, i.e., near a shock. The monotonic q is proportional to the zone density times the square of this velocity jump. Of course, the van Leer interpolation is done one direction at a time, and this is an instance in which q definitely differs for the two directions. Thus the Lagrangian step momentum update must be split into an x update and a y update.

The CFL condition applies to Eulerian methods just as to Lagrangian methods, except that the expression for C is somewhat modified:

$$C \equiv \max\left(\frac{(a + |u_x|)\Delta t}{\Delta x}, \frac{(a + |u_y|)\Delta t}{\Delta y}\right) < 1. \tag{3.27}$$

In fact, a smaller upper limit to C may be set, such as 0.5, as mentioned earlier.

3.2.2 Godunov methods; PPM

The numerical methods described so far are quite elderly, dating in large part to the 1940s. Now we come to the methods of the 1970s and 1980s. Godunov's method (1959) and its descendants are among these. These are finite-volume methods. This means that we now consider all the flow variables to be zonal averages, and we find their evolution by determining the gains and losses of conserved quantities by each zone. The natural place to begin is with the Euler equations written in conservation law form, as in (2.60), which for our purpose becomes

$$\frac{\partial U}{\partial t} + \nabla \cdot \mathbf{F}(U) = 0. \tag{3.28}$$

The vector U contains the conserved densities ρ, ρu_x, ρu_y, and $\rho e + \rho u^2/2$. The components of the flux vector are $\rho \mathbf{u}$, $\rho \mathbf{u} u_x + p\mathbf{e}_x$, $\rho \mathbf{u} u_y + p\mathbf{e}_y$, and $\rho \mathbf{u} e + \rho \mathbf{u} u^2/2 + p\mathbf{u}$. In order to have the Euler equations in conservation-law form we must use the total energy equation (2.9), not the internal energy equation (2.7).

All our flow variables are zone-centered quantities now. Furthermore, all are specified at the same time points, t_n. The evolution equation is found by averaging (3.28) over a zone volume V and over a time interval $[t_n, t_{n+1}]$:

$$V \frac{\Delta \langle U \rangle}{\Delta t} = - \iint_S \langle \mathbf{F} \rangle \cdot \mathbf{n} \, dA. \tag{3.29}$$

The expression $\langle U \rangle$ represents the zone average of U at the beginning or end of the time step; these are our unknowns. The quantity $\langle \mathbf{F} \rangle \cdot \mathbf{n}$ represents the flux of a conserved quantity through the edge of a zone averaged over the time step. If we can construct accurate values of these edge fluxes from the beginning-of-step values of $\langle U \rangle$, then the advanced values follow immediately. From the components of U it is simple arithmetic to find $\mathbf{u}$ and e, and then the pressure can be calculated. The question remains: What are the fluxes?

Godunov's original method is described as follows. At the beginning of the time step we imagine that each zone is uniform, with its average fluid quantities ρ, $\mathbf{u}$, and p. At an interface between two zones there will thus be a discontinuity, as in a shock tube at the initial time. The name for the situation in which two regions of constant fluid properties meet at a discontinuity is a *Riemann problem;* a shock tube is a special case in which the velocities vanish. At the next instant this discontinuity will be resolved into a pair of shocks, a shock and a rarefaction wave, or two rarefaction waves, one traveling into each of the two zones. At the material interface, which at the beginning of the time step coincides with the zone boundary, there will be continuous values of pressure and velocity. Godunov's method is to use these conditions, derived from the solution of the Riemann problem, to

evaluate the edge fluxes, then to use the conservation law to update the conserved quantities to the next time step.

The general Riemann problem can be solved graphically as follows. Let the initial states be state 0 on the left of the interface and state 1 on the right. Plot the locus of points in the (u, p) diagram that can be reached from state 0 either by a shock ($p > p_0$) or a rarefaction ($p < p_0$) using these relations, assuming gamma-law gases:

$$u - u_0 = \begin{cases} \dfrac{2c_0}{\gamma_0 - 1}\left[1 - (p/p_0)^{(\gamma_0-1)/(2\gamma_0)}\right] & p < p_0, \\ -\dfrac{\sqrt{2}c_1(p/p_0 - 1)}{\sqrt{\gamma_1[\gamma_1 - 1 + (\gamma_1 + 1)p/p_0]}} & p > p_0. \end{cases} \tag{3.30}$$

Then plot the locus of points that can be reached from state 1 using

$$u - u_1 = \begin{cases} -\dfrac{2c_1}{\gamma_1 - 1}\left[1 - (p/p_1)^{(\gamma_1-1)/(2\gamma_1)}\right] & p < p_1, \\ \dfrac{\sqrt{2}c_1(p/p_1 - 1)}{\sqrt{\gamma_1[\gamma_1 - 1 + (\gamma_1 + 1)p/p_1]}} & p > p_1. \end{cases} \tag{3.31}$$

The intersection of the loci is the solution of the Riemann problem. This is illustrated in Figure 3.1 for $u_0 = 2, u_1 = 1, p_0 = 15, p_1 = 2, c_0 = 2, c_1 = 1$. This case produces a shock moving right into material 1 and a rarefaction moving left into material 0, as in a normal shock tube. The figure also shows, as a dashed curve, the release path if the relation for a shock were used instead of the correct

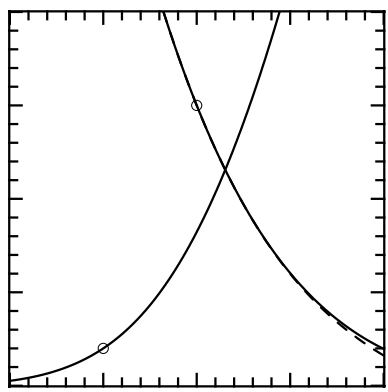

Fig. 3.1 Solid curves: states reached from left (descending curve) and right (ascending curve) initial states by a single shock or rarefaction. Circles: initial states. Dashed curve: approximation in which rarefaction is treated as a shock.

relation for a rarefaction. The locus is almost exactly the same, and this can be a useful approximation, as discussed by Colella and Glaz (1985). The solution for the Riemann problem contains a contact discontinuity, which emanates from the initial interface. The velocity of this discontinuity is u. Thus the stationary interface lies on the material 0 side of the contact if $u > 0$ and on the material 1 side of the contact if $u < 0$. This influences the correct choice of ρ at the interface, which enters the expressions for the fluxes. Efficient methods for solving the Riemann problem, including in some cases tabular equation of state data, have been discussed by van Leer (1979) and Colella and Glaz (1985).

Godunov's original method and some of its successors are actually operator-split Lagrangian-remap methods, as discussed above. The Lagrangian method still fits within the conservation-law framework, except that, in one dimension, the space coordinate is the areal mass m, the solution vector is $U = (\tau, u, E)$, with $\tau = 1/\rho$ and $E = e + u^2/2$, and the flux vector is $F = (-u, p, up)$, *viz.*,

$$\frac{\partial}{\partial t}\begin{pmatrix}\tau\\ u\\ E\end{pmatrix} + \frac{\partial}{\partial m}\begin{pmatrix}-u\\ p\\ up\end{pmatrix} = 0. \tag{3.32}$$

The reason for doing separate Lagrangian and remap steps in Godunov's method, rather than a direct Eulerian step as implied above (see also Godunov, Zabrodyn, and Prokopov (1962)), is that it is assured in the Lagrangian case that one characteristic approaches each interface from the left and the right; if the flow is supersonic this is not true for Eulerian characteristics. This is important for the higher-order Godunov methods. The remap step can make use of van Leer interpolation: the interpolant is constructed for each conserved quantity in the distorted zone that follows from the Lagrangian step, and an integration over the fixed Eulerian zone volume produces the desired Eulerian quantity. In two or three dimensions both the Lagrangian and remap steps are directionally split.

Godunov's method is first order accurate owing to its use of the zone-average quantities for the initial states of the Riemann problem. Van Leer (1979) extended it to a second order Lagrangian-remap method called MUSCL, which stands for monotonic upwind-centered scheme for conservation laws. The key element in MUSCL is the introduction of van Leer interpolants for the beginning-of-step flow variables, and obtaining left and right states at each interface for the middle of the time step by tracing right-facing and left-facing characteristics, respectively, from the interface at the middle of the time step back to the beginning of the time step. The flow variables at these points, located somewhere within the left and right zones, determine via the characteristic equations the left and right states that are used for the Riemann problem. Thus a first order accurate solution of the flow equations in characteristic form provides the data that, through the

Riemann solution, determine the fluxes that go into the conservative update for the Lagrangian step. The remap step in MUSCL is the same as just described, a van Leer interpolation on the distorted mesh followed by integration over the Eulerian zones. In two dimensions the operator-splitting follows the scheme $XYYX$, where X and Y represent Lagrangian + remap steps for the indicated directions. Doing the steps twice with the order reversed the second time, suggested by Strang (1968), eliminates the second order error associated with directional splitting. The MUSCL code is second order accurate, except near shocks, where it, and almost all others, become first order.

The next developments of Godunov-type methods were introduced by Colella and Woodward. Colella (1985) provided a second order method comparable to MUSCL, but based it on a single Eulerian step rather than Lagrangian + remap. The necessity of converting back and forth between Eulerian and Lagrangian quantities, and interpolants in mass vs space, were eliminated, and some linearizations that van Leer found necessary in the Riemann solution were also avoided. The added complication is dealing with the alternatives of two characteristics from the left, one each from left and right, and two characteristics from the right, in finding the Riemann initial states. This method is now referred to as piecewise-linear MUSCL direct Eulerian, or PLMDE, to distinguish it from the common implementation of the next method we will discuss, piecewise-parabolic MUSCL Lagrangian-remap, or PPMLR, called PPM for short.

Coming soon after Colella's second order MUSCL was the introduction of piecewise-parabolic monotonic interpolation, replacing van Leer's piecewise-linear monotonic interpolation. This makes the spatial accuracy third order, but the accuracy of the characteristic tracing and of the directional splitting in multidimensional problems reduces the overall order to second. The piecewise parabolic method (PPM) is described by Colella and Woodward (1984) and Woodward and Colella (1984). Both Lagrangian + remap and direct Eulerian versions are presented. The Lagrangian + remap version has seen considerably more use in the intervening years.

The details of PPM are rather intricate, and the reader is referred to the basic papers. The essential elements are these. Zone average data at the beginning of the time step for a specified zone, its two neighbors on the left and its two neighbors on the right, are combined to form a quadratic interpolation function in the zone that is exactly consistent with the specified zone average. The coefficients are then modified in ways described by Colella and Woodward to produce monotonicity and contact discontinuity steepening. Next the regions of influence in the left zone and the right zone for the conditions on the zone edge through the time step are found by tracing characteristics back from the end of the time step. This determines a portion of the left zone, at the beginning of the time step, that influences the

left-hand side zone edge conditions, and likewise a portion of the right-hand side. The fluid variables are averaged in these regions of influence to give the left-hand side and right-hand side conditions for the Riemann problem. The space average over the region of influence gives the correct time average along the zone edge apart from terms arising from the variation of the characteristic slopes during the time step. The fluxes in the conservative Lagrangian equations (3.32) are computed from these time-average edge conditions. In most cases no additional dissipation such as pseudo-viscosity needs to be included in the method. There are exceptions, however. If a shock is nearly stationary with respect to the mesh, oscillations will develop unless the mesh is jiggled slightly, as Woodward and Colella describe. This jiggling is comparable to artificial viscosity or artificial heat conduction in its effect, but is an order of magnitude smaller than the pseudo-viscosity usually used in the staggered-mesh methods.

The test cases shown by Woodward and Colella (1984) demonstrate the improved accuracy with PPM relative to MUSCL for calculations on the same mesh. Despite the considerable cost in CPU time per zone per cycle, PPM compares well with other methods in the cost to obtain a specified accuracy. The widths of shocks computed with PPM are about one zone, not a few zones as with pseudo-viscosity methods.

3.2.3 WENO

The next class of Eulerian hydrodynamics schemes is the essentially non-oscillatory (ENO) class and its descendant weighted essentially non-oscillatory (WENO). To explain what these are it is helpful to step back and discuss the Lax–Wendroff (1960) method. It is a method for a system of conservation laws in one space dimension, such as the Euler equations, in the form of (3.28). The two-step Lax–Wendroff method is a predictor–corrector method:

$$\begin{aligned} U_{j+1/2}^{n+1/2} &= \frac{1}{2}\left(U_{j+1}^{n} + U_{j}^{n}\right) - \frac{\Delta t}{2\Delta x}\left(F_{j+1}^{n} - F_{j}^{n}\right) \\ U_{j}^{n+1} &= U_{j}^{n} - \frac{\Delta t}{\Delta x}\left(F_{j+1/2}^{n+1/2} - F_{j-1/2}^{n+1/2}\right). \end{aligned} \tag{3.33}$$

To conform to the notation in Richtmyer and Morton (1967) and the WENO literature (Jiang and Shu, 1996) the zone centers are given integer spatial indices and the edges have half-integer indices. We see that the predictor equation finds an edge-centered solution vector at the half-time-step point using the beginning-of-step values of the fluxes evaluated at zone centers. The corrector step uses the edge-centered fluxes derived from the half-time-step unknowns to make the usual conservative update of the zone-centered unknowns.

The two-step Lax–Wendroff method is second order and the dissipation is quite small, $O(\Delta x^2 \Delta t^2)$ (Richtmyer and Morton, 1967). Sod (1978) shows the numerical results for this and a number of other methods on his now well-known shock tube problem. While the first-order method of Godunov *et al.* (1962) smears the shock and contact discontinuity slightly while avoiding ringing, the second order Lax–Wendroff method has fairly sharp jumps combined with a substantial amount of ringing. The ringing has grown to be considered unacceptable, and this was a principal motivation for developing higher order Godunov schemes in which the ringing could be minimized by monotonicity constraints and nonlinear filters.

The ENO approach is another path toward this goal which avoids the use of a Riemann solver. The conservation laws are written in a form in which space has been discretized but time is continuous:

$$\frac{dU_j}{dt} = \frac{1}{\Delta x}\left(\hat{F}_{j-1/2} - \hat{F}_{j+1/2}\right), \tag{3.34}$$

in which the fluxes $\hat{F}_{j+1/2}$ come from the application of an appropriate operator to the U_j. This operator will act on values of U in perhaps several zones surrounding the edge $j + 1/2$ in order to obtain $\hat{F}_{j+1/2}$. The next essential part of the ENO concept is the use of Runge–Kutta methods for the integration of (3.34). The predictor–corrector method is a kind of second order Runge–Kutta. The two-step Lax–Wendroff method departs from a second order Runge–Kutta solution of (3.34) by using a spatially staggered predictor step. One of the goals in the ENO and WENO schemes is to avoid the introduction of oscillations – ringing – hence the names of the methods. One aspect of this is the use of total variation diminishing (TVD) or total variation bounded (TVB) time integrators. TVD means essentially that the spatial fluctuation in the solution at the end of the time step cannot be larger than it was at the beginning of the time step. TVB means that, while the fluctuation may grow, it will always be less than some fixed bound. Some Runge–Kutta integration schemes are TVD and some are not. A popular choice, referred to as RK3, is this third order method:

$$U^{(1)} = U^n + \Delta t\, L[U^n], \tag{3.35}$$

$$U^{(2)} = \frac{3}{4}U^n + \frac{1}{4}U^{(1)} + \frac{1}{4}\Delta t\, L[U^{(1)}], \tag{3.36}$$

$$U^{n+1} = \frac{1}{3}U^n + \frac{2}{3}U^{(2)} + \frac{2}{3}\Delta t\, L[U^{(2)}], \tag{3.37}$$

in which $L[U]$ stands for the operator that yields the right-hand side of (3.34).

The heart of the ENO and WENO schemes comes in choosing the spatial difference operator. To begin with, this is an interpolation problem. Given data, in this case the $F_j^n = F(U_n^j)$, that represent the average values in the spatial zones

$[x_{j-1/2}, x_{j+1/2}]$, for a certain span of zones, find the polynomial of suitable order that fits these data, and thereby derive the edge-centered values $F^n_{j+1/2}$. Then $L[U^n]$ follows as in (3.34). For uniform spatial zoning and for stencils (the collection of zone-centers that are used for the interpolation) that consist of some number of contiguous zones, the formulae are easily written down: the single-zone stencils:

$$F_{j+1/2} = F_j, \qquad F_{j+1/2} = F_{j+1}; \tag{3.38}$$

the two-zone stencils:

$$\begin{aligned} F_{j+1/2} &= \frac{1}{2}\left(-F_{j-1} + 3F_j\right), \quad F_{j+1/2} = \frac{1}{2}\left(F_j + F_{j+1}\right), \\ F_{j+1/2} &= \frac{1}{2}\left(3F_{j+1} - F_{j+2}\right); \end{aligned} \tag{3.39}$$

and the three-zone stencils:

$$\begin{aligned} F_{j+1/2} &= \frac{1}{6}\left(2F_{j-2} - 7F_{j-1} + 11F_j\right), \\ F_{j+1/2} &= \frac{1}{6}\left(-F_{j-1} + 5F_j + 2F_{j+1}\right), \\ F_{j+1/2} &= \frac{1}{6}\left(2F_j + 5F_{j+1} - F_{j+2}\right), \\ F_{j+1/2} &= \frac{1}{6}\left(11F_{j+1} - 7F_{j+2} + 2F_{j+3}\right). \end{aligned} \tag{3.40}$$

The order of the interpolation polynomial is one less than the number of zones in the stencil in each case; the result is an approximation to $L[U]$ that is accurate to an order equal to the number of zones.

Next we discuss selecting among the stencils. As we know, stability of the method is aided by using preferentially data on the upwind side in space for the calculation of the flux. For a scalar equation the wind blows in the direction indicated by $\partial_U F(U)$: in the $+x$ direction if $\partial_U F(U) > 0$, in the $-x$ direction if $\partial_U F(U) < 0$. We thus discard one stencil from each list of candidates, the last one in each group for $\partial_U F(U) > 0$ and the first one for $\partial_U F(U) < 0$. If $F(U)$ is not a monotone function then it must be split into parts that are:

$$F(U) = F^+(U) + F^-(U), \tag{3.41}$$

with $\partial_U F^+(U) > 0$ and $\partial_U F^-(U) < 0$; there may be several ways to do this. The value of $\hat{F}$ is then calculated by summing the values of $\hat{F}^+$, calculated on stencils biased toward $-x$, with $\hat{F}^-$ calculated on stencils biased toward $+x$. For systems of conservations laws there is not just one derivative $\partial_U F(U)$ to examine, but the signs of the characteristic speeds, which are the eigenvalues of the Jacobian matrix. The eigenvectors of the Jacobian are formed into a group with positive characteristic speed and a group with negative characteristic speed. Projection

operators for the positive and negative subspaces are formed from these, and the final $\hat{F}$ is built by combining the upwind stencils for each group using these operators. This is not the only method for splitting $\hat{F}$ into $\hat{F}^+$ and $\hat{F}^-$. A simpler alternative that seems to work well is based on the Lax–Friedrich method (Lax, 1954), for which

$$F^{\pm}(U) = \frac{1}{2}\left(F(U) \pm \alpha U\right) \tag{3.42}$$

with $\alpha = \max |\partial_U F(U)|$, taken over the full sampled range of U.

Now we come to the ENO/WENO concept. Consider the three-zone stencils, of which three survive after dropping the downwind-most of the original four. A figure of merit is constructed locally for each stencil by combining the squares of estimates of the first and second spatial differences of the Fs. The preferred stencil is the one for which this figure of merit is least. Near a shock or other discontinuity in the flow, this stencil will be the one that samples the properties downwind of the jump the least. For the ENO method the result, for this flux, is taken to be the one from the preferred stencil and the others are discarded. For the WENO method the flux is given by a weighted sum of the flux from the surviving stencils, with weights inversely related to the figure of merit mentioned. These weights have the effect, in smooth regions, of canceling some of the truncation error in the individual stencils. Thus each of the three-zone stencils gives a result that is third order accurate, the weighted sum is fifth order accurate. For an r-zone stencil, each individual is rth order accurate and the weighted sum is $(2r-1)$th order accurate. The detailed expressions for the weights and the error analysis are found in Jiang and Shu (1996). This is the scheme referred to as WENO5. Balsara and Shu (2000) have given a ninth order WENO scheme that uses five different five-zone stencils. Ninth order pertains to the spatial differencing; WENO9 still uses the third order RK3 Runge–Kutta time integration scheme.

The test results using WENO5 in Jiang and Shu (1996) on Sod's problem and some of Woodward's problems (Woodward and Colella, 1984) look quite acceptable. The characteristic projection method for flux splitting has the edge in accuracy over the Lax–Friedrichs splitting. Shi, Zhang, and Shu (2003) show WENO5 and WENO9 results for Woodward's double Mach reflection problem and a Rayleigh–Taylor instability problem that contain a profusion of small-scale features for the high order methods.

3.2.4 Adaptive mesh refinement (AMR)

An Eulerian hydrodynamics variation is adaptive mesh refinement, or AMR. The concept of AMR is to provide greater grid resolution in the parts of the spatial domain that most need it any given time. While AMR might be possible in a

Lagrangian framework, the need for this capability is less since the Lagrangian zones already follow the material and tend to be where they are needed automatically. This is not completely true, since extra resolution may be needed near shocks, for example, which move through the material. The second reason for not using Lagrangian AMR is that it may be exceedingly cumbersome. The unstructured-mesh Lagrangian codes would be the best Lagrangian AMR vehicle, but we will discuss that no further.

The seminal paper describing block-structured AMR, a method that applies refinement in rectangular subdomains (patches), is Berger and Colella (1989). Details of the implementation are given by Bell, Colella, and Glaz (1989) and Bell, Berger, Saltzman, and Welcome (1994). A very nice capsule description of the method has been given by Grauer and Germaschewski (2001):

> We start by integrating the equations under consideration on a single grid with fixed resolution. Then, we constantly verify if the resolution is still sufficient by some appropriate criterion. If it is not we mark all points where the resolution is insufficient. Next, we try to cover these underresolved points by a collection of rectangles (or boxes in 3D) as effectively as possible using methods from image processing. These new rectangles need to be correctly embedded in their parent grids, otherwise they have to be modified accordingly. When the rectangles are filled with data, they are integrated using a discretization length and time step divided by certain refinement factors. In addition, all the steps described above are performed recursively. Important is the communication among the grids. The necessary boundary data are obtained from neighboring grids if present and otherwise from the parent grids. Similar treatment is applied to newly generated grids. Data are obtained from the old grids of the same level where possible, otherwise from the parent grids. This allows for effectively following moving structures. Finally, after each time step the coarse grids are updated using data from finer grids.

The AMR mesh may be thought of as hierarchical; at the top level is a uniform fairly coarse mesh, called the level 1 mesh. Portions of this are refined to produce a level 2 mesh, and so on to a maximum depth of refinement, perhaps to the third level, perhaps more. The data in any zone at a certain level are superseded by the data at the next level down if this zone is in a patch that is refined. The refinement factor may be chosen, for example, to be 4. With a refinement factor of 2 a few more levels are needed. Whatever refinement factor is used spatially, in each of the coordinate directions, the same factor is used for temporal refinement. Thus a patch that is refined four times in space is subcycled in the time integration; four of the refined times steps are computed to bring the refined patch up to the same new time level as the coarser region in which it is embedded. This is done so that each level of refinement uses the same Courant ratio. Of course, this is also done recursively when there are several levels of refinement. The maximum depth of refinement is a critical parameter in the application. If this is set too small, then important features will not be resolved. If it is set too large, then some hydrodynamics problems

will "refine themselves to death" because the criterion for accepting the current mesh may not be satisfied until the available storage is exhausted. The maximum refinement depth must be chosen by the skilled code user to meet the requirements of the problem at hand.

The criterion for refinement, like the maximum depth of refinement, must also be tuned to fit the problem, and for the same reason: insufficient refinement defeats the hoped-for accuracy, and excessive refinement stops the calculation. The general criteria, such as the Richardson's extrapolation estimate of Berger and Colella (1989) can be useful, but without adjustment can also lead to excessive refinement. Here again, the skill of the experienced code user is necessary.

Since AMR puts the mesh where it is needed to resolve features in the flow, the total number of zones, counting all the levels of refinement, is much less than for the uniform mesh that provides the same maximum resolution. That is, this is the case if the areas of refinement are relatively few and localized. There are problems, such as homogeneous hydrodynamic turbulence, for which refinement would be necessary throughout the entire problem, and for which AMR is therefore of little value. Problems of this kind benefit from PPM or the WENO methods.

One group of closely related AMR codes is described by Klein, Greenough, Howell and coworkers (Truelove *et al.*, 1997, 1998; Klein (1999); Klein *et al.*, 2003, 2004). They use block-structured AMR as described above, and employ the PLMDE Godunov method. Block-structured AMR is not the only approach. The RAGE code developed by Gittings and others (Gittings, 1992; Holmes *et al.*, 1999) uses cell-by-cell refinement. All these codes have 3-D as well as 2-D versions. Operator splitting of the spatial coordinates can as easily be applied to three dimensions as two.

3.3 ALE methods

The ALE methods will not be described in detail. The difference between the ALE equations and the Eulerian equations is slight, as we see from a comparison of (2.15)–(2.17) with (2.2), (2.3), and (2.9). The computational procedure in most of the ALE codes follows the outline of the staggered-mesh Eulerian method described earlier. One difference is that, since the mesh is distorted rather than orthogonal as is the Eulerian mesh, the computation of the pressure gradient follows the prescription used in the Lagrangian codes. The ALE modifications to the Eulerian equations are treated fully by Bowers and Wilson (1991). What is left unaddressed is the very important question of specifying the grid motion. The grid motion algorithms have the vital task of following the material as closely as possible while at the same time preventing excessive distortion of the grid. The recipes for doing this are many, derived from a great number of hydrodynamic

simulations that failed and needed a new grid motion algorithm to succeed. Unfortunately this experience is poorly documented. Some inkling as to the algorithms can be derived from the user's manual for CALE, Tipton's popular 2-D ALE code (Tipton, 1991).

3.4 3-D methods

The discussion of 3-D codes will also be very brief. The 3-D AMR codes have been mentioned. The extension from two to three dimensions of orthogonal-mesh Eulerian codes is relatively simple. There are some notable computer science issues involved with maintaining the data structures for the hierarchical mesh. The individual patches lead to box objects that carry the information about the cells they contain and also about how they interface to the boxes in which they are embedded. There are many more such items of information for the 3-D box than in two dimensions.

Besides these two codes there is PPM, which has been described already, and which exists in a 3-D version. This is the code that won the 1999 Gordon Bell award for its teraflop performance on a problem of Richtmyer–Meshkov instability (Mirin *et al.*, 1999). Another is the FLASH code developed by the ASCI Center for Thermonuclear Flashes at the University of Chicago (Fryxell *et al.*, 2000). FLASH incorporates PPM as one of its hydrodynamic options, but can also use block-structured AMR.

There are no good examples of a strictly Lagrangian 3-D code. This is for the simple reason that Lagrangian codes fail to run past the point in time when the mesh tangles, producing the bowties mentioned above. In three dimensions the opportunities to tangle are much richer, and the Lagrangian codes simply do not run long enough to be useful. An exception must be made for structural dynamics codes such as DYNA3D (Hallquist, 1982); these deal with solid materials that have strength and that therefore can resist the tangling to which fluids are susceptible.

There are 3-D ALE codes. We will mention two: the ARES code (Bazan, 1998) and the KULL code (Rathkopf *et al.*, 2000), both developed at Lawrence Livermore National Laboratory as part of the US Department of Energy's Advanced Simulation and Computing effort, which in fact supports all the 3-D codes mentioned above as well. ARES and KULL have many similarities, but ARES has a structured mesh that is almost entirely built of hexahedra, i.e., deformed cubes. The ARES mesh can have a limited number of points of irregularity where the connectivity is different from that for hexahedra. The MESH in KULL is unstructured, so in principle it is built of arbitrary polyhedra, but in practice KULL most often uses a hexahedral mesh like ARES's.

3.5 Other methods – SPH and spectral methods

Two other computational methods for hydrodynamics are current that avoid, either altogether or largely, the use of finite-difference or finite-element equations on a spatial mesh. These are smoothed particle hydrodynamics (SPH) and the spectral method. There is no mesh at all in SPH, unless self-gravity or magnetohydrodynamics are to be included. In the spectral method the mesh that is used is in Fourier (or Chebyshev) space, and normal space is used only for evaluating nonlinear products.

The SPH method was introduced by Lucy (1977) and Gingold and Monaghan (1977). Good descriptions of it are found in Monaghan (1992), Hernquist and Katz (1989), and Rasio (2000). At first blush SPH is a Monte Carlo method, since it uses a sampled set of discrete particles to represent the continuous fields of density and velocity. But the initial particles are not selected randomly, and the time advancement is completely deterministic. SPH is best thought of as a meshless Lagrangian method.

The following description of SPH is based largely on Rasio (2000). Each of the N particles in the simulation carries its particular values of mass m, velocity $\mathbf{u}$, internal energy e, and for some methods also the density ρ. The particle has an "uncertainty cloud" that is described by the function $W(\mathbf{r} - \mathbf{r}_i, h_i)$, where $\mathbf{r}_i$ is the current location of the particle and h_i is a parameter that sets its spatial extent. If the density is not an explicit attribute of the particles it can be computed from

$$\rho_i = \sum_j m_j W(\mathbf{r}_i - \mathbf{r}_j, h_j). \tag{3.43}$$

The kernel function W might be a Gaussian, or this spline function:

$$W(r,h) = \frac{1}{\pi h^3} \begin{cases} 1 - \frac{3}{2}\left(\frac{r}{h}\right)^2 + \frac{3}{4}\left(\frac{r}{h}\right)^3, & 0 \le \frac{r}{h} \le 1, \\ \frac{1}{4}\left[2 - \frac{r}{h}\right]^3, & 1 \le \frac{r}{h} \le 2, \\ 0, & \text{otherwise.} \end{cases} \tag{3.44}$$

The scale length h must be chosen to be small compared with the size of features in the solution, but large enough to encompass several of the neighbors of each particle. These are the same criteria one would use to select a zone size in a mesh-based simulation.

The evolution equations for $\mathbf{r}_i$, $\mathbf{u}_i$, and e_i are the following, for the simplest implementation:

$$\frac{d\mathbf{r}_i}{dt} = \mathbf{u}_i, \tag{3.45}$$

$$\frac{d\mathbf{u}_i}{dt} = -\sum_j m_j \left(\frac{p_i}{\rho_i^2} + \frac{p_j}{\rho_j^2} \right) \nabla_i W_{ij}, \tag{3.46}$$

$$\frac{de_i}{dt} = \frac{1}{2} \sum_j m_j \left(\frac{p_i}{\rho_i^2} + \frac{p_j}{\rho_j^2} \right) (\mathbf{u}_i - \mathbf{u}_j) \cdot \nabla_i W_{ij}, \tag{3.47}$$

where $W_{ij} = W_{ji} = W(\mathbf{r}_i - \mathbf{r}_j, h)$. (If h differs for different particles, a symmetrization procedure is required.) An important aspect of the summations over particles that appear in the dynamical equations is that, because the kernel W has a finite range, there is a very limited number of nonzero terms in the sum. The sum expression for ρ_i above introduces problems at free boundaries, and the procedure of carrying ρ explicitly on each particle may be preferred, using the following form for the equation of continuity:

$$\frac{d\rho_i}{dt} = \sum_j m_j (\mathbf{u}_i - \mathbf{u}_j) \cdot \nabla_i W_{ij}, \tag{3.48}$$

The pressure p_i is to be computed from the equation of state in terms of ρ_i and e_i.

These are the essential equations of the method. The further details involve including viscosity, constructing a dynamical equation for h_i, and other refinements. The TREESPH code described by Hernquist and Katz (1989) adds the physics of self-gravitation using the tree method. That is, a given particle exerts a force on its close neighbors that is calculated from the inverse-square law, but for remote particles the force is given by the gradient of a smoothed-out potential. A hierarchy of meshes is laid on the problem so that the cells in which the given particle lies, at all the levels of the hierarchy, form one branch of a tree. For each cell of the hierarchy a few of its multipole moments are computed. These multipole moments provide the force on the given particle from the more remote parts of the problem. If N is the total number of particles, then the number of close-neighbor interactions is $O(N)$, and, remarkably, the cost of the computation of the hierarchical multipole moments and their use to provide the remote contribution to the force is only $O(N \log N)$.

The time integration is commonly chosen to be the second order leapfrog method familiar from the staggered-mesh Lagrangian codes, *cf.*, (3.1), (3.2). There is a Courant-like limit to the time step for stability of order $0.5h/a$. The spatial accuracy is first order, i.e., the error is $O(h^2)$, and so the method is first order

accurate overall. The SPH result converges to the continuum solution if $h \to 0$ but also the number of particles N_N within the volume $4\pi h^3/3$ tends to infinity, which means, of course, that $N \to \infty$. Monaghan (1992) describes a number of tests to which SPH has been subjected, including standard gas dynamics problems such as Sod's shock tube and Woodward and Colella's flow over a step; the results are acceptable, with the errors that would be expected for a first order code. Monaghan also provides an impressive list of astrophysical applications of SPH, ranging from tidal disruption in binary star systems, through asteroid impacts on the earth and molecular cloud fragmentation to large-scale structure in the universe.

Spectral methods, or more precisely pseudo-spectral methods, have a significant presence in the world of turbulence simulations and Navier–Stokes calculations. A standard text on spectral methods in fluid dynamics is Canuto, Hussaini, Quarteroni, and Zang (1988), and another reference is Boyd (2001). The major feature of the methods is the expansion of the unknowns ρ, $\mathbf{u}$, and so forth in terms of global basis functions such as trigonometric functions (Fourier series) or orthogonal polynomials (Chebyshev series). The spatial derivatives of the expanded functions are easily evaluated using the derivatives of the basis functions, which are available analytically. The technical distinction between "spectral" and "pseudo-spectral" methods is in whether the expansion should approximate the continuous solution in the sense of integral projections (the Galerkin concept) – the spectral method – or should be accurate at a certain set of mesh points (collocation points) – the pseudo-spectral method. The equations of fluid dynamics are nonlinear, of course, and terms such as $\mathbf{u} \cdot \nabla \mathbf{u}$ and $\nabla p/\rho$ in the Euler equation are only conveniently evaluated by combining $\mathbf{u}$ and $\nabla \mathbf{u}$ or ∇p and ρ at the collocation points, but relying on Fourier or Chebyshev space for the derivative calculations.

An ideal fluid dynamic application for the pseudo-spectral method is direct numerical simulation (DNS) of homogeneous turbulence at low Mach number using the incompressible Navier–Stokes equation

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = -\frac{1}{\rho} \nabla p + \nu \nabla^2 \mathbf{u}, \tag{3.49}$$

in which the density ρ and the kinematic viscosity ν are constants, and for which the pressure is to be found from the incompressibility constraint,

$$\nabla \cdot \mathbf{u} = 0. \tag{3.50}$$

Equation (3.49) can be written like this:

$$\frac{\partial \mathbf{u}}{\partial t} = \mathbf{u} \times \boldsymbol{\omega} - \nabla \Pi + \nu \nabla^2 \mathbf{u}, \tag{3.51}$$

in which $\boldsymbol{\omega} = \nabla \times \mathbf{u}$ is the vorticity and Π is the Bernoulli "constant" $\Pi = p/\rho + u^2/2$. The divergence of (3.51) leads to this Poisson equation for Π:

$$\nabla^2 \Pi = \nabla \cdot (\mathbf{u} \times \boldsymbol{\omega}), \tag{3.52}$$

so that, in principle, Π can be eliminated from (3.51).

Poisson's equation is readily solved in Fourier space. We define the transform of a variable X by $\tilde{X} = \mathcal{F}_{\mathbf{k}}[X] = \iiint d^3\mathbf{r} \exp(i\mathbf{k} \cdot \mathbf{r}) X(\mathbf{r})$. The transform of (3.52) gives

$$-k^2 \tilde{\Pi} = -i\mathbf{k} \cdot \mathbf{s}, \tag{3.53}$$

in which the quantity $\mathbf{s}$ is the transform of the nonlinear term $\mathbf{u} \times \boldsymbol{\omega}$. The transform of the vorticity itself is given by

$$\tilde{\boldsymbol{\omega}} = -i\mathbf{k} \times \tilde{\mathbf{u}}. \tag{3.54}$$

Using (3.53), $\tilde{\Pi}$ can be eliminated from the transform of (3.51), which leads to

$$\left(\frac{d}{dt} + \nu k^2\right) \tilde{\mathbf{u}} = \left(- \frac{\mathbf{kk}}{k^2} \right) \cdot \mathbf{s}. \tag{3.55}$$

The tensor $- \mathbf{kk}/k^2$ is the projection operator perpendicular to $\mathbf{k}$, so its effect is to make only the transverse (divergence-free) component of $\mathbf{u} \times \boldsymbol{\omega}$ appear in the acceleration equation.

The integration of (3.55) is done by first choosing a uniform mesh in $\mathbf{r}$ space, to which corresponds a similarly shaped mesh in $\mathbf{k}$ space. An integration method such as Runge–Kutta is applied to follow the evolution of $\tilde{\mathbf{u}}$ on this $\mathbf{k}$ mesh. The evaluation of $d\tilde{\mathbf{u}}/dt$ at each step in the integration requires first forming $\tilde{\boldsymbol{\omega}}$ from $\tilde{\mathbf{u}}$, applying fast Fourier transforms (FFTs) to each of these to give $\mathbf{u}$ and $\boldsymbol{\omega}$, forming the cross product, then another FFT to give $\mathbf{s}$, from which $d\tilde{\mathbf{u}}/dt$ follows. An additional FFT provides Π and hence p if desired for edit purposes.

A very large-scale example of this approach to a DNS problem may be found in Yokokawa *et al.*, (2002).

3.6 Summary

Numerical hydrodynamics is a very rich field. It is a garden with many beautiful flowers, and it is not easy to compare them. Some of the methods have been developed within a certain area of application, and these methods are well suited to their area. We think of SPH and its astrophysical applications, for which a very robust Lagrangian method is highly desirable even if it is somewhat more dissipative than competing methods, and if the difficulties of adding other types of physics such as MHD or radiation transport can be overcome. Spectral methods might

be very cumbersome indeed if applied to problems with real material properties and complicated structures, but for the ultimate in accuracy, as in DNS, they are perfectly suited. The work-horse 1-D method, for everything from inertial fusion capsule implosions to pulsating stars and the big bang, is von Neumann–Richtmyer staggered-mesh Lagrangian hydrodynamics. In two and three dimensions, for the problems that do have real materials or complicated structures, there are hard choices to make. The ALE codes naturally put the resolution where it is needed in the spatial structure; their down side is a lack of robustness that is constantly being improved by more sophisticated grid-motion algorithms. The Eulerian codes are much more robust, but accuracy and freedom from numerical diffusion have traditionally been their weak points. The Godunov and ENO/WENO methods have definitely corrected that. If a sufficiently fine mesh is used to resolve all the features of interest, or with AMR to achieve that, Eulerian methods are quite attractive.

The Godunov vs ENO/WENO competition is in a state of flux at this time, and the best methods may not have been created yet. It is interesting to look at the comparisons in Liska and Wendroff (2003) and Rider, Greenough, and Kamm (2003), as well as the examples shown by Shi *et al.* (2003). Liska and Wendroff do not include PLMDE or any AMR code in the list of tested codes, but they do include a WENO3 code, with simplified flux splitting, and a WENO5 code with characteristic-based flux splitting, as well as PPM and others. The test suite consists of 17 Riemann problems, the Noh problem (Noh, 1987), a Rayleigh–Taylor bubble growth problem, an implosion problem and an explosion problem. As the authors point out, no one code is best on all the problems, although WENO5 and PPM are near the top on most of them. Interestingly, WENO5 crashed attempting to run the Noh cylindrical stagnation problem. On the Rayleigh–Taylor bubble problem the amount of fine-scale structure that develops on the edges of the bubble due to secondary Kelvin–Helmholtz instability was quite different in the various codes. The low order, rather dissipative codes showed quite smooth bubbles, while PPM had quite a large amount of such structure; WENO5 less so. This is also similar to the result in Holmes *et al.* (1999), where the AMR code RAGE was compared with PROMETHEUS, a code quite similar to PPM, and a front-tracking code FronTier (Grove *et al.*, 1993) on a Richtmyer–Meshkov instability growth problem rather similar to the Rayleigh–Taylor one. In this case RAGE and FronTier were relatively smooth and PROMETHEUS stood out for its amount of small-scale structure. RAGE also compares favorably with results of a Richtmyer–Meshkov instability experiment, as seen in Baltrusaitis *et al.* (1996). Shi *et al.* also show a Rayleigh–Taylor problem as computed by WENO5 and WENO9; the fine-scale structure in the highly-zoned WENO9 calculation is prodigious. The experiment described by Holmes *et al.* did not have sufficient spatial resolution to

select between the codes on the basis of the fine structure, so this issue remains unsettled.

Rider *et al*. (2003) focus on comparisons of WENO5 with several PLMDE and PPM variants. The test problems were the Sod shock tube, the Woodward and Colella colliding blast wave and the Shu and Osher (1989) steepening wave train. The high order accuracy of WENO5 did not result in an appreciably lower error for those problems featuring a strong shock. Neither was WENO5 much worse than the others; the error measures for the different methods spanned a surprisingly small range. An interesting comparison can be made if the accuracy results are normalized by the computing time required. That is, the codes are compared on the basis of cost to obtain a specified accuracy. Since for a given mesh WENO requires about three times the floating point operations of a Godunov method like PLMDE, and the accuracy is not very different for shock problems, the efficiency of WENO5 compares poorly with the PLM and PPM methods. It should be noted that this statement is specific to shock problems; in flows that are everywhere smooth the high order WENO performance is very good. Rider *et al*. also include in their paper a new method that adds some extremum-preserving features derived from WENO to the basic PPM scheme; that is, the slope limiting that is a normal part of PPM is relaxed somewhat to avoid clipping the peaks. This method turned out to perform exceptionally on the test problems, and to be efficient as well. Time will tell whether this new variation will fulfill its early promise.

It does seem that the high-accuracy Eulerian methods are continuing to develop and to represent a very attractive alternative to ALE methods. As the scale of computing systems expands, the ability of Eulerian codes to resolve all the relevant spatial structures will improve, and when that happens the scales will tip strongly in the Eulerian direction.