PARTICLE METHODS FOR HYDRODYNAMICS

J.J. MONAGHAN

Department of mathematics, Monash University, Clayton, Vic. 3168, Australia

1985

NORTH-HOLLAND – Amsterdam

Contents

Computer Physics Reports 3 (1985) 71–124
North-Holland, Amsterdam

PARTICLE METHODS FOR HYDRODYNAMICS

J.J. MONAGHAN

Department of Mathematics, Monash University, Clayton, Vic. 3168, Australia

This review examines particle methods for compressible flow. The original particle method PIC is described. It can be improved by using more efficient interpolation. Appropriate methods for interpolation from a regular grid of points, and for interpolation from disordered points are described. A general form of PIC is written down, but optimal forms for boundary conditions and the equations are not known.

The method for interpolation from disordered points leads to the particle method SPH. This is a pure Lagrangian method which does not require a grid except as an aid to more efficient computing. Particle methods for magnetohydrodynamics based on PIC and on SPH are described.

0167-7977/85/$18.90

1. Introduction

When Harlow developed the first particle method (Harlow [1,2]) the Eulerian and Lagrangian finite difference methods had serious disadvantages. The Eulerian methods could not accurately track the interface between moving materials, and they gave a poor representation of the advective terms in the equations of motion. Lagrangian methods did not have these disadvantages, but they gave low accuracy in regions of strong shear because the Lagrangian grid, moving with the fluid, became highly distorted.

Harlow's ingenious idea was to simulate advection by moving particles which carried mass, momentum and thermal energy (or entropy). The remaining, non-advective, terms in the equations of motion were calculated on an Eulerian grid. This device eliminated, at a stroke, the disadvantages of both Lagrangian and Eulerian methods. Moving interfaces could be followed easily since each material could be represented by an independent set of particles. Advection could, in principle, be treated accurately since it only required the integration of the equations of motion for particles. The problem of grid distortion was eliminated because spatial derivatives were calculated on a fixed, regular Eulerian grid. The numerical method, called PIC for particle in cell, was found to have a number of desirable properties. It was simple and robust and it could handle problems involving several different media and severe shear. Harlow and his associates (Bjork [3], Evans and Harlow [4], Harlow and Dickman [5], Evans, Harlow and Meixner [6], Amsden [7]) applied PIC to a wide variety of difficult problems including supersonic flow around projectiles, viscous flow around blunt obstacles, high explosive burning and hyper velocity impact (for a description of these simulations see Amsden [7]).

A disadvantage of PIC was that because information had to be transferred back and forth between the grid and the particles, and that transfer used, at best, only bi-linear interpolation, the method had a large implicit diffusion. It was therefore only reasonably accurate for supersonic flows, and even for such flows contact discontinuities were smeared out by diffusion. Furthermore, because it effectively used two grids (the particles are equivalent to a Lagrangian grid), it required roughly twice the storage and twice the computing time (Amsden [7]) of either an Eulerian or a Lagrangian method. These criticisms are by no means devastating because in practice it may be better to pay the price of extra time and storage in order to get a more reliable result. Harlow was, however, pessimistic about the long term prospects for PIC. He remarked (Harlow [2]) that "it is, furthermore, not likely that the PIC method will ever be applied extensively to three dimensional problems". This review will show that there is a good deal of life in the various progeny of PIC, and that particle methods are certainly competitive with, and have some advantages over, finite difference methods in many three dimensional problems.

The progeny of PIC include a generalization to special relativistic flow (Harlow et al. [8], Amsden et al. [9], Nix [10]), to magneto hydrodynamic phenomena (Leboeuf et al. [11], Brunel et al. [12]) and to astrophysical hydrodynamics (Gingold and Monaghan [13] and Lucy [14]) where, in the latter, the particle method (SPH) is truly Lagrangian and does not require a grid. A version of SPH has also been used to simulate the post Newtonian evolution of a neutron star (Thompson [66]).

2. One-dimensional PIC

2.1. Equations of motion

All the essential features of PIC appear in the simple problem of the one-dimensional motion of a non-dissipative fluid without body forces. The equations of motion are

$$\frac{\partial v}{\partial t} + v\frac{\partial v}{\partial x} = -\frac{1}{\rho}\frac{\partial P}{\partial x}, \tag{2.1}$$

$$\frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x}(\rho v) = 0, \tag{2.2}$$

and either

$$\frac{\partial u}{\partial t} + v\frac{\partial u}{\partial x} = -\frac{P}{\rho}\frac{\partial v}{\partial x}, \tag{2.3}$$

or

$$\frac{\partial s}{\partial t} + v\frac{\partial s}{\partial x} = 0, \tag{2.4}$$

where s is the entropy per unit mass, ρ is the density, P the pressure, v the velocity and u the thermal energy per unit mass. For this problem it is preferable to use (2.4) rather than (2.3) because (2.4) implies that each element of fluid caries its entropy unchanged.

In order to integrate eqs. (2.1)–(2.3) we use a one-dimensional grid and a set of particles. The particles have fixed mass and fixed entropy per unit mass. The total mass and entropy of the system therefore remains constant. We assume the separation Δx between neighbouring grid points is constant and each particle has mass m.

The equations of motion must be supplemented by initial and boundary conditions. For the present we assume the fluid extends to infinity and the initial density and entropy is uniform. The initial velocity is specified by giving each particle a velocity and a position. We denote the velocity of particle j at time step n by $v_j(n)$, its position by $x_j(n)$ and its entropy by $s_j(n)$. In the simplest PIC scheme the variables on the grid are specified at the centre of each cell by assignment from the particles. This assignment is an interpolation from the particles (which form a Lagrangian network) to the grid. For the present we assign using the nearest grid point (actually nearest cell centre) rule. Define the function $W(u)$ by

$$\text{If } |u| < \tfrac{1}{2}\Delta x \quad \text{then } W(u) = 1, \quad \text{else } W(u) = 0. \tag{2.5}$$

The density $\rho(\lambda, n)$, velocity $v(\lambda, n)$ and entropy $s(\lambda, n)$ assigned to the cell centre with coordinate x_λ at time step n are obtained by summing over particles according to

$$\rho(\lambda, n) = \frac{m}{\Delta x}\sum_j W_{\lambda j}, \tag{2.6}$$

$$\rho(\lambda, n)v(\lambda, n) = \frac{1}{\Delta x}\sum_j l_j(n)W_{\lambda j}, \tag{2.7}$$

$$\rho(\lambda, n)s(\lambda, n) = \frac{m}{\Delta x}\sum_j s_j(n)W_{\lambda j}, \tag{2.8}$$

where $W_{\lambda j} \equiv W(x_\lambda - x_j(n))$ and l_n is the momentum of particle j. Because of the definition of $W(u)$ the only particles that contribute to the properties of a given cell are those particles in the cell. The assignment (2.6) is therefore equivalent to summing the total mass in the cell and dividing by Δx. We have used the form (2.6) to allow us to make a generalization to other, more powerful, interpolation methods. A similar interpretation applies to (2.7) and (2.8). To calculate the pressure we need an equation of state in the form $P = F(\rho, s)$.

The integration from time step n to $n+1$ involves the following steps:

Step 1. Update the velocity $v(\lambda, n)$ using finite differences to calculate $\partial P/\partial x$. For example

$$\tilde{v}(\lambda, n+1) = v(\lambda, n) - \frac{\delta t(P(\lambda+1, n) - P(\lambda-1, n))}{\rho(\lambda, n)2\Delta x}, \tag{2.9}$$

where $\tilde{v}$ is a preliminary value of $v(\lambda, n+1)$. It is not the correct value because the advective term has not been included.

Step 2. Interpolate $\tilde{v}(\lambda, n+1)$ to the particles. Most PIC calculations use linear interpolation which can be written in the form

$$v_j(n+1) = \sum_\lambda \tilde{v}(\lambda, n+1)W^*_{\lambda j}, \tag{2.10}$$

where: If $|u| < \Delta x$ then $W^*(u) = 1 - |u|/\Delta x$, else $W^*(u) = 0$, and $W^*_{\lambda j} \equiv W^*(x_\lambda - x_j(n))$.

Step 3. Assign the momentum of cell λ to the particles in the cell, i.e.

$$l_j(n+1) = m\sum_\lambda \tilde{v}(\lambda, n+1)W_{\lambda j}. \tag{2.11}$$

Note that W, not W^*, is used here to ensure conservation of linear momentum (see section 2.3).

Step 4. Move particles to new positions using, for example,

$$x_j(n+1) = x_j(n) + \delta t v_j(n+1). \tag{2.12}$$

This is the advective phase. The particles carry with them their mass, momentum and entropy.

Step 5. Assign ρ, v and s to the cell centres using (2.6) to (2.8) with

$$W_{\lambda j} \equiv W(x_\lambda - x_j(n+1)).$$

These steps show the typical features of PIC: the interpolation back and forth between grid and particles, the calculation, where necessary, of derivatives by finite differences on the grid, and the moving of the particles to simulate advection. The continuity equation (2.2) is automatically solved by combination of shifting particles and the assignment rule (2.6). Eq. (2.4) is automatically solved by keeping the entropy for each particle constant.

2.2. Simple generalizations

2.2.1. More than one material

The simplest generalization is to allow several materials with a set of particles for each material. If a cell is occupied by particles for one material the pressure can be calculated from the equation of state for that material. If two materials, which we denote by A and B, occupy a cell there are various ways the pressure can be determined. Amsden [7] recommends a method based on pressure continuity across an interface. Let the masses of the respective materials in cell λ be M_A and M_B, and let the fraction of the cell occupied by material A be σ. Then pressure continuity requires that

$$P(\lambda, n) = F_A\left(\frac{M_A}{\sigma \Delta x}, s_A\right) = F_B\left(\frac{M_B}{(1-\sigma)\Delta x}, s_B\right). \tag{2.13}$$

This gives an equation for σ which can be solved by iteration. When σ is found P can be calculated. Other alternatives are discussed by Amsden [7].

2.2.2. Other boundary conditions

In our example the boundaries were taken at infinity and neglected. Boundary conditions for other cases (rigid boundary, applied pressure, periodic, inflow and continuative) are discussed by Harlow [2] and Amsden [7]. The boundary conditions depend on the interpolation scheme as is the case for standard Eulerian calculations. For the interpolation scheme used in our one-dimensional PIC the prescription of Harlow and Amsden is that

(a) If the cell boundary is a rigid wall the boundary pressure is set equal to the cell pressure and the boundary velocity is set equal to zero if it is otherwise directed outwards. If the cell boundary is that of cell k, and cells $k, k+1, \ldots,$ are occupied, then the rule above is equivalent to specifying a fictitious cell

$$k-1 \quad \text{with } P(k-1, n) = P(k, n) \quad \text{and } v(k-1, n) = -v(k, n).$$

(b) If there is an external pressure P^* acting, and the cells $k, k+1, \ldots, k+r$ are occupied, then the outer boundaries of cells k and $k+r$ are given the pressure P^*. Using linear interpolation, this implies that the cells $k-1$, $k+r+1$ have fictitious pressures $2P^* - P(k, n)$ and $2P^* - P(k+r, n)$, respectively. The condition on the velocity is that $v(k-1, n) = v(k, n)$ and $v(k+r+1, n) = v(k+r, n)$.

Amsden [7] discusses further conditions required to deal with internal cavities.

(c) If the flow is periodic in space and one period occupies the cells $k, k+1, \ldots, k+r$ then particles leaving the grid through one boundary are returned through the other.

(d) If there is a prescribed inflow through one boundary then a fictitious cell on the upstream side of the boundary is introduced. This cell has the properties of the inflow (ρ, v and s), and it provides the extra information required for the gradients at the boundary cell.
(e) If there is a continuative outflow a fictitious cell on the downstream side of the boundary is assigned the properties of the boundary cell. This ensures that all gradients vanish at the boundary.

2.3. Conservation laws for one-dimensional PIC

For the system we have been considering total mass, momentum and entropy are conserved.

The total mass of the particle system is constant. The total mass in the grid representation is also constant since

$$\sum_{\lambda} W(x_{\lambda} - x_{j}(n)) = 1. \tag{2.14}$$

and then

$$\sum_{\lambda} \Delta x \rho(\lambda, n) = m \sum_{\lambda} \sum_{j} W(x_{\lambda} - x_{j}(n)) = Nm. \tag{2.15}$$

The grid and particle representation of the total mass are therefore consistent. In the same way we find

$$\sum_{\lambda} \Delta x \rho(\lambda, n) s(\lambda, n) = m \sum_{j} s_{j}(n) = \text{constant}. \tag{2.16}$$

To establish that momentum is conserved we need to consider an isolated system without boundaries. From (2.9)

$$\Delta x \sum_{\lambda} \rho(\lambda, n) \tilde{v}(\lambda, n+1) = \Delta x \sum_{\lambda} \rho(\lambda, n) v(\lambda, n), \tag{2.17}$$

since the pressure terms vanish in pairs and the boundary condition (b) of section 2.2.2 requires that the empty cell just outside each boundary has pressure opposite in sign to that in the cell just inside the boundary (this makes the cell boundary pressure zero).

From (2.11)

$$\sum_{j} I_{j}(n+1) = \sum_{j} \sum_{\lambda} m \tilde{v}(\lambda, n+1) W_{\lambda j} = \sum_{\lambda} \tilde{v}(\lambda, n+1) \rho(\lambda, n) \Delta x. \tag{2.18}$$

This shows that the updated particle momentum equals the grid momentum. Finally from (2.7), (2.18) and (2.17) we establish that grid momentum is conserved. If there is a rigid boundary, or a finite applied pressure, the momentum need not be conserved.

2.4. A summary of one-dimensional PIC

While easy to program, our one-dimensional PIC algorithm would not be competitive with good finite difference algorithms. It requires more storage, is slower and, because of the interpolation, is strongly diffusive.

How can we improve it. The weakest aspect of PIC is the interpolation. If we could improve that we would, at a stroke, reduce the number of particles to achieve a given accuracy and greatly reduce the diffusion. In the next section we will introduce these improvements by deriving more sophisticated interpolation kernels.

3. Interpolation

3.1. Interpolation on a uniform grid

A crucial part of PIC is the interpolation. Two types of interpolation are used in PIC. When interpolating from the grid to the particles we are dealing with interpolation from equi-spaced data points. When interpolating from the particles to the grid the data points are disordered. The first type of interpolation has an abundant literature. The second type of interpolation, in the form used by particle methods, has received much less attention.

The fundamental analysis of interpolation, in the form which is of most use in particle methods, is due to Schoenberg [15,16]. He considers all the classical interpolation formula in the form

$$\tilde{f}(x, h) = \sum_{j=-\infty}^{\infty} f_j L(x - x_j, h), \tag{3.1}$$

where $f_j \equiv f(x_j)$ and $x_j = jh$. The function $L(x - x_j, h)$ is called the interpolating kernel. It is always taken to be an even function. Schoenberg distinguishes between two types of interpolation formula: ordinary interpolation, for which $\tilde{f}(x_j, h) = f_j$, and smoothing interpolation formula for which $\tilde{f}(x_j, h) \neq f_j$. The smoothing interpolation formulae were designed for interpolation from noisy data. They are, however, ideally suited to particle methods when one needs to minimize the effect of disorder.

As examples of interpolation (see also Monaghan [17]) we consider (using the variable $v = |u|/h$)

linear interpolation:

$$L(u, h) = \begin{cases} 1 - v; & 0 \leqslant v \leqslant 1, \\ 0; & \text{otherwise}, \end{cases}$$

Bessel's formula neglecting 3rd and higher differences:

$$L(u, h) = \begin{cases} (1 - v)(1 + \frac{1}{4}v); & 0 \leqslant v \leqslant 1, \\ \frac{1}{4}(1 - v)(2 - v); & 1 \leqslant v \leqslant 2, \end{cases}$$

Everett's formula neglecting 4th and higher differences:

$$L(u, h) = \begin{cases} \frac{1}{2}(2-v)(1-v^2); & 0 \leqslant v \leqslant 1, \\ \frac{1}{6}(2-v)(3-v)(1-v); & 1 \leqslant v \leqslant 2. \end{cases}$$

Schoenberg [15] has shown that the interpolating properties of (3.1) can be described in terms of the Fourier transform of the interpolating kernel. With the definition

$$g(t) := \int_{-\infty}^{\infty} L(x, h) \cos(2\pi t x) \mathrm{d}x, \tag{3.2}$$

Schoenberg's result is that (3.1) will reproduce polynomials of degree $k-1$ provided (a) $g(t)-h$ has a zero of order k at $t=0$, and (b) $g(t)$ has zeros of order k at the points $th = \pm 1, \pm 2, \ldots$.

For example, the kernel above for linear interpolation has Fourier transform

$$g(t) = h\left[\frac{\sin(\pi th)}{\pi th}\right]^2. \tag{3.3}$$

which has zeros of order 2 at $th = \pm 1, \pm 2, \ldots$ and, since $g(t) - h = -h^3\pi^2 t^2/6 + \mathcal{O}(t^4)$ for $|t| \ll 1$, it has a zero of order 2 at $t=0$. The interpolation formula will therefore reproduce linear functions.

The kernel for Bessel's formula has the Fourier transform

$$g(t) = h\left[\frac{\sin(\pi th)}{\pi th}\right]^3 \left[\cos(\pi th) + (\pi th)\sin(\pi th)\right], \tag{3.4}$$

which has zeros of order 3 at $th = \pm 1, \pm 2 \ldots$ and since $g(t) - h = \mathcal{O}(t^4)$ for $|t| \ll 1$ it has a zero of order 4 at $t=0$. The interpolation formula will therefore reproduce quadratic functions and, in general, has errors of $\mathcal{O}(h^3)$.

3.2. Basis splines

The disadvantage of the kernels discussed is their low order continuity. The kernels are continuous but their derivatives are not. This means that if they are used in a particle simulation slight errors in the particle positions may have a large effect.

Schoenberg [15] introduced a fundamental set of interpolating kernels, the central B splines, to achieve linear interpolation while allowing smoothing of the, possibly noisy, data. In his notation (3.1) becomes

$$\tilde{f}(x, h) = \sum_{j=-\infty}^{\infty} f(x_j) M_n(x - x_j, h), \tag{3.5}$$

where $M_n(u, h)$ is the nth central B spline of order n, i.e. degree $n-1$. The basis spline $M_2(u, h)$ is the function defined above for linear interpolation. This function leads to an

ordinary interpolation formula. The Fourier transform of M_n is (see Schoenberg [16])

$$h\left[\frac{\sin(\pi th)}{\pi th}\right]^n, \tag{3.6}$$

which has a zero of order n at $th = \pm 1, \pm 2$ etc., but has a zero of only order 2 at $t = 0$. The B splines can therefore only reproduce linear functions, i.e. they interpolate with, in general, errors of $\mathcal{O}(h^2)$. The Fourier transform decreases more rapidly with increasing t for larger n because the smoothness of the splines increases with n.

The $M_n(u, h)$ are given explicitly by taking the successive partial sums of the polynomial

$$\left(\frac{u}{h}+\frac{n}{2}\right)^{n-1} - \binom{n}{1}\left(\frac{u}{h}+\frac{n}{2}-1\right)^{n-1} + \cdots + (-1)^n\left(\frac{u}{h}-\frac{n}{2}\right)^{n-1}, \tag{3.7}$$

thus

$$(n-1)!M_n(u, h) = \begin{cases} 0; & \frac{u}{h} \leqslant -\frac{n}{2}, \\ \left(\frac{u}{h}+\frac{n}{2}\right)^{n-1}; & -\frac{n}{2} \leqslant \frac{u}{h} \leqslant -\frac{n}{2}+1, \\ \left(\frac{u}{h}+\frac{n}{2}\right)^{n-1} - \left(\frac{u}{h}+\frac{n}{2}-1\right)^{n-1}; & -\frac{n}{2}+1 \leqslant \frac{u}{h} \leqslant -\frac{n}{2}+2, \\ \text{and so on.} \end{cases} \tag{3.8}$$

Setting $v = |u|/h$, and noting that $M_n(u, h)$ is an even function of u we find

$$M_1(u, h) = \begin{cases} 1; & 0 \leqslant v \leqslant \frac{1}{2}, \\ 0; & \text{otherwise,} \end{cases}$$

which is the nearest grid point interpolation formula,

$$M_2(u, h) = \begin{cases} 1-v; & 0 \leqslant v \leqslant 1, \\ 0; & \text{otherwise,} \end{cases}$$

the linear interpolation formula, and

$$M_3(u, h) = \begin{cases} \frac{3}{4}-v^2; & 0 \leqslant v \leqslant \frac{1}{2}, \\ \frac{1}{2}\left(\frac{3}{2}-v\right)^2; & \frac{1}{2} \leqslant v \leqslant \frac{3}{2}, \\ 0; & \text{otherwise,} \end{cases} \tag{3.9}$$

which is the first of the basis splines to have a continuous first derivative. $M_3(u, h)$ is derived, without realizing its relation to basis splines, by Hockney and Eastwood [18] who call M_3 the

function TSC. For later reference

$$M_4(u, h) = \begin{cases} \frac{2}{3} - v^2 + \frac{1}{2}v^3; & 0 \leqslant v \leqslant 1, \\ \frac{1}{6}(2 - v)^3; & 1 \leqslant v \leqslant 2, \\ 0; & \text{otherwise.} \end{cases} \tag{3.10}$$

Hockney and Eastwood [18] show that if M_3 is used instead of M_2 in charge assignment (they were concerned with plasma problems) there is a big improvement in accuracy. This improvement is due to the fact that M_3 is less sensitive than M_2 to disorder in the interpolation points.

3.3. Other interpolation kernels

For the particle method SPH (Gingold and Monaghan [13]) interpolation is carried out using the kernel

$$L(u) = \frac{h}{a\sqrt{\pi}} e^{-u^2/a^2}, \tag{3.11}$$

for one-dimensional problems. The Fourier transform of this kernel is

$$g(t) = h\, e^{-\pi^2 t^2 a^2}. \tag{3.12}$$

From the Schoenberg rules the Gaussian cannot even interpolate a constant correctly! It turns out, however, that the errors involved in interpolating linear functions can be made quite negligible provided $a > h$. We will derive this result in section 3.5. An attractive feature of the Gaussian is its smoothness. An unattractive feature is that we need to include grid points within, say, $\pm 3a$ of the point of interest and this may double the work.

Wood [28] has used the kernel

$$L(u) = \frac{h}{2a} e^{-|u|/a}, \tag{3.13}$$

for one-dimensional problems. This kernel has

$$g(t) = \frac{h}{1 + 4\pi^2 t^2}, \tag{3.14}$$

so that, like the Gaussian, it cannot interpolate a constant correctly. This $g(t)$ does not decrease as rapidly as that in (3.12) when t increases because the kernel (3.13) is not as smooth as the Gaussian. Furthermore the kernel (3.13) does not interpolate as accurately as the Gaussian and requires more work since grid points within, say, $\pm 9a$ may need to be included.

3.4. Improving the basis splines

The basis spline M_2 produces an ordinary interpolation formula. The higher order splines give smoothing interpolation formula. In general, as we have seen, the M_n splines give errors of

$\mathcal{O}(h^2)$. In some cases it is an advantage to have a smooth interpolation formula that has higher accuracy. Schoenberg [15,16] has considered such formulae. Here we shall use a method which differs from Schoenberg's. The interpolation formulae we derive are new.

There is considerable analytical convenience in working with an integral rather than the sum (3.1). The error involved in switching from one to the other can be estimated using the Poisson summation formula (Courant and Hilbert [20]). We find

$$\sum_{j=-\infty}^{\infty} f_j L(x-x_j, h) = \int_{-\infty}^{\infty} f(uh) L(x-uh, h)\mathrm{d}u + \epsilon(h), \tag{3.15}$$

where

$$\epsilon(h) = 2\sum_{r=1}^{\infty} \int_{-\infty}^{\infty} f(uh) L(x-uh, h)\cos(2\pi r u)\mathrm{d}u. \tag{3.16}$$

The dominant contribution to $\epsilon(h)$ is from the term $r=1$. If we expand $f(uh)$ about the point $uh = x$ we find that the term with $r=1$ becomes

$$\epsilon(h) \approx \frac{2}{h}\sum_{a=0}^{\infty} \frac{f^{(a)}(x)}{a!} \int_{-\infty}^{\infty} \omega^a L(\omega, h)\cos\frac{2\pi(x+\omega)}{h}\mathrm{d}\omega. \tag{3.17}$$

The integral in (3.17) has the form

$$\cos\left(\frac{2\pi x}{h}\right)\int_{-\infty}^{\infty} \omega^a L(\omega, h)\cos\left(\frac{2\pi\omega}{h}\right)\mathrm{d}\omega - \sin\left(\frac{2\pi x}{h}\right)\int_{-\infty}^{\infty} \omega^a L(\omega, h)\sin\left(\frac{2\pi\omega}{h}\right)\mathrm{d}\omega. \tag{3.18}$$

Referring to the definition of $g(t)$ we see that the integrals in (3.18) are related to the derivatives of $g(t)$. In fact (3.18) can be written, when a is even, as

$$\cos\left(\frac{2\pi x}{h}\right)\frac{(-1)^{a/2}}{(2\pi)^a}\left[\frac{\mathrm{d}^a}{\mathrm{d}t^a} g(t)\right]_{t=1/h}, \tag{3.19}$$

and when a is odd

$$-\sin\left(\frac{2\pi x}{h}\right)\frac{(-1)^{(a-1)/2}}{(2\pi)^a}\left[\frac{\mathrm{d}^a}{\mathrm{d}t^a} g(t)\right]_{t=1/h}. \tag{3.20}$$

If L is the basis spline M_n then

$$\left[\frac{\mathrm{d}^a}{\mathrm{d}t^a} g(t)\right]_{t=1/h} = 0 \quad \text{for } a \leqslant n-1.$$

The first surviving term in (3.17) has $a=n$, so that, apart from trigonometric terms $\epsilon(h)$ is of order h^n. Provided $n \geqslant 3$ the integral in (3.15) can be used in place of the sum because the

interpolation error in the sum is of order h^2. Our interpolant is now

$$\tilde{f}(x, h) = \frac{1}{h}\int_{-\infty}^{\infty} f(x')L(x - x', h)\mathrm{d}x'. \tag{3.21}$$

By expanding $f(x')$ about the point x we find

$$\tilde{f}(x, h) = f(x) + \frac{f''(x)}{2h}\int_{-\infty}^{\infty} u^2 L(u, h)\mathrm{d}u + \ldots. \tag{3.22}$$

where we have assumed

$$\frac{1}{h}\int_{-\infty}^{\infty} L(u, h)\mathrm{d}u = 1, \tag{3.23}$$

which is necessary in order to interpolate constants correctly and, as usual, assumed $L(u, h)$ to be an even function of u.

The interpolant (3.21) has interpolation errors which are of even degree in h (because L is an even function) and the error is $\mathcal{O}(h^m)$ if the Fourier transform of the kernel has the expansion

$$g(t) - h \propto t^m, \tag{3.24}$$

when t is small, i.e. $g(t) - h$ has an mth order root at $t = 0$.

We can write (3.22) in the form

$$\tilde{f}(x, h) = f(x) + a(x)h^2 + \ldots. \tag{3.25}$$

We now try to devise a means of eliminating the term $a(x)h^2$. If we could do this the interpolant (3.21) would have errors $\mathcal{O}(h^4)$. Richardson extrapolation is a well known way of improving the accuracy of a formula and we can apply it here since, for a uniform grid with separation H,

$$\tilde{f}(x, H) = f(x) + a(x)H^2 + \ldots. \tag{3.26}$$

We can now combine (3.25) and (3.26) so as to remove the first error term. In this way we find the new interpolant

$$\int_{-\infty}^{\infty} f(x')\Phi_n(x - x', h, H)\mathrm{d}x', \tag{3.27}$$

where

$$\Phi_n(u, h, H) = \left\{\frac{H^2}{h^2}L(u, h) - \frac{h^2}{H^2}L(u, H)\right\}/(H^2 - h^2). \tag{3.28}$$

The interpolant has errors of $\mathcal{O}(h^4)$ but, as it stands, it is useless because the integration cannot be expressed accurately by a summation on either grid. To escape this difficulty we take the limit

of (3.28) as $H \to h$. We find

$$\lim_{H \to h} \Phi_n(u, h, H) = \frac{1}{2h}\left(3L(u, h) - h\frac{\partial L(u, h)}{\partial h}\right), \tag{3.29}$$

and the new interpolation kernel

$$W(u, h) := \frac{1}{2}\left[3L - h\frac{\partial L}{\partial h}\right]. \tag{3.30}$$

For the basis splines the new kernel W only exists if $n \geqslant 3$ because otherwise the derivative of M_n is not defined everywhere.

It is not difficult to show from the result

$$\int_{-\infty}^{\infty} L(u, h)u^2 \mathrm{d}u = ch^3, \tag{3.31}$$

where c is a constant, that

$$\tilde{f}(x, h) = f(x) + \mathcal{O}(h^4).$$

The integral interpolation formula is therefore of $\mathcal{O}(h^4)$ as expected. It will be clear from the previous discussion that the practical interpolation formula

$$\tilde{f}(x) = \sum_{j=-\infty}^{\infty} f_j W(x - x_j, h), \tag{3.32}$$

may not achieve the same accuracy. In fact, because W involves a derivative of L, it will not be as smooth as L and its Fourier transform will not vanish as rapidly as that of L as $t \to \infty$. Since we require $\epsilon(h)$ (see (3.17)) to be at least as small as $\mathcal{O}(h^4)$, and W is less smooth than L, we may require a very smooth L to produce $\mathcal{O}(h^4)$ interpolation.

We can make these ideas more precise by using Schoenberg's rules. Let the kernel constructed from $L \equiv M_n$ be denoted by W_n. Then the Fourier transform of W_n is (setting $\sigma = \pi th$)

$$\begin{aligned} g(t) &= \int_{-\infty}^{\infty} W_n(u, h)\cos(2\pi ut)\mathrm{d}u \\ &= h\left(\frac{\sin\sigma}{\sigma}\right)^{n-1}\left[\left(1 + \frac{n}{2}\right)\frac{\sin\sigma}{\sigma} - \frac{n}{2}\cos\sigma\right]. \end{aligned} \tag{3.33}$$

Noting that

$$g(t) - h = -\frac{h(n-1)(11n-6)}{360}(\pi th)^4, \tag{3.34}$$

we can conclude from Schoenberg's rules that if $n = 4$ the practical interpolation formula (3.32) will reproduce quadratic functions and, in general, will have errors of $\mathcal{O}(h^3)$. If $n \geqslant 5$ the formula will reproduce cubic functions and, in general, will have errors of $\mathcal{O}(h^4)$.

Referring to (3.10) and (3.30) of M_4 we find

$$W_4(u, h) = \begin{cases} 1 - \frac{5v^2}{2} + \frac{3v^3}{2}; & 0 \leqslant v \leqslant 1, \\ \frac{1}{2}(2 - v)^2(1 - v); & 1 \leqslant v \leqslant 2, \end{cases} \tag{3.35}$$

where, as before $v = |u|/h$. This kernel gives an ordinary interpolation formula (M_4 gave a smoothing interpolation formula) which reproduces polynomials of degree 2. The kernel W_4 and its first derivative are continuous so that, in its smoothness, W_4 is similar to M_3. It is also similar to the kernel (3.4) for Bessel's interpolation formula which has the same support, degree of polynomial reproduced and order of accuracy as W_4.

Because of its smoothness and accuracy W_4 is a good alternative to M_3 in particle methods. The advantages will be apparent in the numerical experiments we describe later. An apparent disadvantage is that because $W_4 < 0$ in $1 < v < 2$ the interpolation might give unphysical negative values to a variable. In practice this does not seem to be a problem except in high Mach number collisions (Lattanzio et al. [21]).

3.5. *Improving other kernels*

We will consider just one example, the Gaussian, which gives the interpolation formula

$$\tilde{f}(x) = \frac{h}{a\sqrt{\pi}} \sum_{j=-\infty}^{\infty} f_j \exp\{-(x - jh)^2/a^2\}. \tag{3.36}$$

The summation can be converted to an integration with dominant error (see (3.16))

$$\frac{2h}{a\sqrt{\pi}} \int_{-\infty}^{\infty} f(uh) \exp\{-(x - uh)^2/a^2\} \cos(2\pi u) \mathrm{d}u, \tag{3.37}$$

which is the real part of

$$\frac{2h}{a\sqrt{\pi}} \exp\left\{ -\frac{\pi^2 a^2}{h^2} + \frac{2\pi \mathrm{i} x}{h} \right\} \int_{-\infty}^{\infty} f(uh) \exp\left\{ -\frac{h^2}{a^2}\left(u - \frac{qa^2}{2h^2} \right)^2 \right\} \mathrm{d}u, \tag{3.38}$$

where

$$q = \frac{2xh}{a^2} + 2\pi \mathrm{i}. \tag{3.39}$$

By shifting the contour from the real axis to a straight line parallel to the real axis and passing through the point $qa^2/(2h^2)$, we can write (3.38) as

$$\frac{2h}{a\sqrt{\pi}} \exp\left\{ -\frac{\pi^2 a^2}{h^2} + \frac{2\pi \mathrm{i} x}{h} \right\} \int_{-\infty}^{\infty} f\left(x + \frac{2\pi a \mathrm{i}}{h} + yh \right) \mathrm{e}^{-y^2 h^2/a^2} \mathrm{d}y, \tag{3.40}$$

which can be approximated in most cases by

$$2\exp\left\{-\frac{\pi^2a^2}{h^2}+\frac{2\pi ix}{h}\right\}f\left(x+\frac{2\pi ai}{h}\right). \tag{3.41}$$

The case $x=0$ was considered by Goodwin [22] and, apart from some notational changes, the above result agrees with his for this case.

Because of the factor $\exp\{-\pi^2a^2/h^2\}$ the error can be made negligible if $a \geqslant h$. This fact has long been recognized in connection with the evaluation of the Gaussian integral by the Trapezoidal rule. Assuming then that $a \geqslant h$ and the error is negligible we can replace (3.36) by the interpolant

$$\tilde{f}(x, a)=\frac{1}{a\sqrt{\pi}}\int_{-\infty}^{\infty}f(x')\exp\{-(x-x')^2/a^2\}\,dx'. \tag{3.42}$$

By Taylor series expansion we find

$$\tilde{f}(x, a)=f(x)+\frac{a^2}{4}f''(x)+\ldots. \tag{3.43}$$

The previous arguments concerning Richardson extrapolation can be applied and we find the new kernel is

$$W(u, a)=\tfrac{1}{2}\left[3G-a\frac{\partial G}{\partial a}\right], \tag{3.44}$$

where

$$G=\frac{h}{a\sqrt{\pi}}e^{-u^2/a^2}.$$

Thus

$$W(u, a)=\frac{h}{a\sqrt{\pi}}e^{-u^2/a^2}\left(\frac{3}{2}-\frac{u^2}{a^2}\right). \tag{3.45}$$

With this kernel the interpolant has errors of $\mathcal{O}(a^4)$. The practical interpolation formula

$$\tilde{f}(x)=\frac{h}{a\sqrt{\pi}}\sum_{j=-\infty}^{\infty}f_j\exp\{-(x-jh)^2/a^2\}\left(\frac{3}{2}-\frac{(x-jh)^2}{a^2}\right), \tag{3.46}$$

interpolates with errors which are approximately of $\mathcal{O}(a^4)$, but there are also errors involved in replacing the integration by a summation. This error is larger than that involved in using the Gaussian kernel but, by taking $a>h$, this error can be made negligible.

The Fourier transform of $W(u, a)$ is

$$g(t)=h\,e^{-\pi^2t^2a^2}(1+\pi^2a^2t^2). \tag{3.47}$$

The rapid decrease with increasing t is due to the smoothness of $W(u, a)$. When t is sufficiently small

$$g(t) = h(1 - \tfrac{1}{2}\pi^4 a^4 t^4)$$

which shows, again, that the integral interpolant using $W(u, a)$ has errors of $\mathcal{O}(a^4)$.

In shock tube simulations $W(u, a)$ gives a huge improvement over the standard Gaussian (Monaghan and Gingold [23]).

3.6. Extensions to higher dimensions

We will consider the case of three-dimensional spaces. The simplest extension of our previous results is to use the interpolant

$$\tilde{f}(x, y, z, h) = \sum_j \sum_k \sum_l f_{jkl} L(x - x_j, h) L(y - y_j, h) L(z - z_j, h), \tag{3.48}$$

where we have assumed, for simplicity, that neighbouring grid points have the same separation along each coordinate axis. Not only could this assumption be changed but the assumption that the same kernel is used for each coordinate could also be changed.

The integral interpolant corresponding to (3.48) is

$$\tilde{f}(x, y, z, h) = \frac{1}{h^3} \iiint f(x', y', z') K(x - x', y - y', z - z', h) \mathrm{d}x' \mathrm{d}y' \mathrm{d}z', \tag{3.49}$$

where

$$K(u, v, w, h) = L(u, h) L(v, h) L(w, h). \tag{3.50}$$

If we expand $f(x', y', z')$ about the point x, y, z we find

$$\tilde{f}(x, y, z, h) = f(x, y, z) + ah^2 \nabla^2 f + \ldots, \tag{3.51}$$

where

$$ah^2 = \frac{1}{2h} \int_{-\infty}^{\infty} u^2 L(u, h) \mathrm{d}u.$$

Richardson extrapolation results in the new kernel

$$F(u, v, w, h, H) = \left\{ \frac{H^2}{h^3} K(u, v, w, h) - \frac{h^2}{H^3} K(u, v, w, H) \right\} / (H^2 - h^2). \tag{3.52}$$

Taking the limit $H \to h$ we find the kernel

$$W(u, v, w, h) = \tfrac{1}{2} \left[5K - h \frac{\partial K}{\partial h} \right]. \tag{3.53}$$

For the general case of n dimensions

$$W(u, v, w, h) = \tfrac{1}{2}\left[(n+2)K - h\frac{\partial K}{\partial h}\right]. \tag{3.54}$$

The simplest case is when L is a Gaussian say

$$\frac{1}{\sqrt{\pi}}\exp(-u^2/h^2),$$

where, for convenience, we have assumed a (see (3.36)) is equal to h. Substituting for L in (3.50) and (3.51) we find

$$W(u, v, w, h) = \frac{1}{\pi^{3/2}}e^{-r^2/h^2}\left[\frac{5}{2} - \frac{r^2}{h^2}\right], \tag{3.55}$$

where

$$r^2 = u^2 + v^2 + w^2.$$

The kernel defined by (3.55) is a spherically symmetric kernel which approximately interpolates with errors $\mathcal{O}(h^4)$. It differs from the kernel we would have obtained if we had simply multiplied the improved Gaussian (3.45) for each dimension. We shall see later (section 4.4) that spherically symmetric kernels have advantages for some particle simulation problems.

We could apply the formula (3.53) to the case where L is a basis spline, say M_4. However the resulting interpolation formula is complicated. It is more efficient in this case to use the improved splines as the interpolating kernel L.

Before leaving this section it is worth noting from (3.51) that the error from interpolation with the basis splines or the Gaussian is equivalent to a diffusion.

3.7. But what if the data points are disordered

The interpolation from the grid to the particles in PIC can use the interpolation formulae we have discussed. The interpolation from the particles to the grid is an entirely different matter because the particles are not on the vertices of a regular grid. We must therefore consider how accurately this latter interpolation can be implemented.

In one dimension we could use several interpolation methods e.g. Lagrangian interpolation or, better, cubic splines (de Boor [24]). However, in three dimensions, the best interpolation method is not known. The choice made for particle methods is based on the integral interpolant which we write in the form

$$\tilde{f}(\boldsymbol{r}, h) = \int f(\boldsymbol{r}')W(\boldsymbol{r} - \boldsymbol{r}', h)\mathrm{d}\boldsymbol{r}', \tag{3.56}$$

where, instead of the previous normalization (3.23), we choose

$$\int W(\boldsymbol{r}-\boldsymbol{r}', h)\mathrm{d}\boldsymbol{r}' = 1.$$

Note that h no longer refers to the grid point separation; it is just a length scale that determines the domain of influence of the kernel.

Suppose now we know the value of f at a set of points $\boldsymbol{r}_1, \boldsymbol{r}_2, \ldots, \boldsymbol{r}_N$. How can we use this information to evaluate the integral? If $n(\boldsymbol{r})$ is the number density of the points at $\boldsymbol{r}$ then an estimate of $\tilde{f}(\boldsymbol{r}, h)$ is

$$\tilde{f}(\boldsymbol{r}, h) \approx \sum_{j=1}^{N} \frac{f_j}{n_j} W(\boldsymbol{r}-\boldsymbol{r}_j, h). \tag{3.57}$$

If $n(\boldsymbol{r})$ is a constant (3.57) reduces to the summation interpolation formula. If $n(\boldsymbol{r})$ is not constant the accuracy of the approximation depends on the distribution of data points. If the points were randomly distributed, with a probability density proportional to $n(\boldsymbol{r})$, then (3.57) would be a Monte Carlo estimate of (3.56) with an error which is roughly $\propto 1/\sqrt{N}$. This error is unacceptably large for the simulation of fluid systems but, fortunately, it is inappropriate. Fluid systems do not result in fluctuations as large as the probability argument would suggest because such fluctuations require energy which is not available. Looked at another way, the probability argument assumes that some dealer deals out the disordered positions, whereas in fact the fluid must do the dealing itself.

The degree of disorder of the data points in PIC simulations is not known. We may, however, be making a reasonable estimate if we assume the points are quasi-ordered (Niedereiter [25]). In that case the estimate (3.57) has an error of order $(\ln N)^d/N$, where d is the number of dimensions. A typical example of quasi-ordered points are the points generated by the mapping

$$x_{j+1} = x_j + \alpha, \quad \mod 1, \tag{3.58}$$

where α is an irrational number, and mod 1 means the fractional part is retained. This mapping gives points uniformly dense in the interval $0 \leqslant x \leqslant 1$. Given N such points we can make an estimate of the integral

$$\int_0^1 F(x)\mathrm{d}x, \tag{3.59}$$

in the form

$$\frac{1}{N}\sum_{j=1}^{N} F(x_j). \tag{3.60}$$

The error can be estimated using the Fourier series of $F(x)$. From

$$F(x) = \sum_{n=-\infty}^{\infty} C_n \mathrm{e}^{2\pi i n x},$$

where

$$C_n = \int_0^1 e^{-2\pi i n x} F(x) dx,$$

we can write (3.60) as

$$\frac{1}{N} \sum_{j=1}^{N} \sum_{n=-\infty}^{\infty} C_n e^{2\pi i n x_j} = C_0 + \frac{1}{N} \sum_n{}' C_n \sum_{j=1}^{N} e^{2\pi i n x_j}, \tag{3.61}$$

where the term $n = 0$ is not included in $\sum_n'$. Since C_0 is just the integral (3.59), the error in the estimate (3.60) is

$$\frac{1}{N} \left| \sum_n{}' C_n \sum_{j=1}^{N} e^{2\pi i n x_j} \right|.$$

From (3.58) we can replace x_j by $j\alpha$ (there is no need to worry about the fractional part because the complex exponential automatically handles the mod 1) and we find that the error is

$$\frac{2}{N} \left| \sum_n{}' C_n \frac{\sin(\pi \alpha n (N+1))}{\sin \pi \alpha n} \right|. \tag{3.62}$$

For a detailed study of this error expression the reader should consult Davis and Rabinowitz [26]. Such a study takes account of the large contribution that might occur if αn is very close to an integer. Provided C_n falls off very rapidly, e.g. like n^{-q} with $q > 3$, we can be confident that the error varies with N as $1/N$.

In one dimension an error like $1/N$ would not be very good, but our argument generalises to three dimensions where the error is still $\approx 1/N$ and this is effectively much better because it is equivalent, for our interpolation, to an error of $\mathcal{O}(h^3)$. The same argument can be applied to the integral interpolant. The integration domain (3.56) is confined to the support of W (if W is a Gaussian we use a finite domain that results in negligible error by comparison with the infinite domain) and we can use Fourier series defined in this domain. Arguing from (3.62) we would expect that kernels with rapidly decreasing Fourier series should be used. This assertion is in agreement with practical experience of PIC methods which establishes the advantage of smooth kernels, e.g. the basis spline M_3 rather than M_2. Naturally one wants to reduce the interpolation error too. This is achieved by using, for example, the smooth high order kernels discussed in sections (3.4) and (3.5).

In particle simulations the moving interpolation points are called particles. Particle j has position $\boldsymbol{r}_j$, a mass m_j and a density ρ_j which we can take as $n_j m_j$ where, as in (3.57), n_j is the number density at the point $\boldsymbol{r}_j$. We can therefore rewrite (3.57) as

$$\tilde{f}(\boldsymbol{r}, h) = \sum_{j=1}^{N} \frac{m_j}{\rho_j} f_j W(\boldsymbol{r} - \boldsymbol{r}_j, h). \tag{3.63}$$

It is always an advantage in particle simulations to arrange to interpolate quantities in the form ρA. For example, in one-dimensional PIC, ρ, ρv and ρs were interpolated from the particles to the grid. From (3.63)

$$\tilde{\rho}(\boldsymbol{r}, h) = \sum_{j=1}^{N} m_j W(\boldsymbol{r} - \boldsymbol{r}_j, h), \tag{3.64}$$

and

$$\int \tilde{\rho}(\boldsymbol{r}, h) \mathrm{d}\boldsymbol{r} = \sum_{j=1}^{N} m_j, \tag{3.65}$$

so that total mass is always conserved. Similarly

$$\tilde{\rho}(\boldsymbol{r}, h)\tilde{\boldsymbol{v}}(\boldsymbol{r}, h) = \sum_{j=1}^{N} m_j \boldsymbol{v}_j W(\boldsymbol{r} - \boldsymbol{r}_j, h),$$

and (3.67)

$$\tilde{\rho}(r, h)\tilde{s}(\boldsymbol{r}, h) = \sum_{j=1}^{N} m_j s_j W(\boldsymbol{r} - \boldsymbol{r}_j, h).$$

We can summarize this by saying that PIC uses a moving set of interpolation points which are assigned masses and are called particles. The interpolation from the particles to the grid is achieved by using an approximation to an integral interpolant. It is the error in this interpolation which is the most troublesome to deal with. The error can be reduced by using smooth interpolation formulae.

Since it is relevant here (though we shall return to it later) we mention the error in the particle to grid interpolation can be reduced by, from time to time, regridding the particles. This simply means that the current particles will be replaced by new particles on the grid points with the mass, velocity and entropy associated with those points.

3.8. Examples of interpolation

In fig. 1a we show the effect of interpolation $\sin \pi x$ using the basis splines M_3 and M_4 with $h = 0.5$. Although both splines have errors of $\mathcal{O}(h^2)$ M_3 interpolates more accurately than M_4. In fig. 1b we show the interpolation using the improved spline W_4 with $h = 0.5$. The improvement is substantial. In fig. 1c interpolation using the Gaussian and improved Gaussian with $h = 0.5$ is shown. The improvement gained by using the improved Gaussian is again substantial.

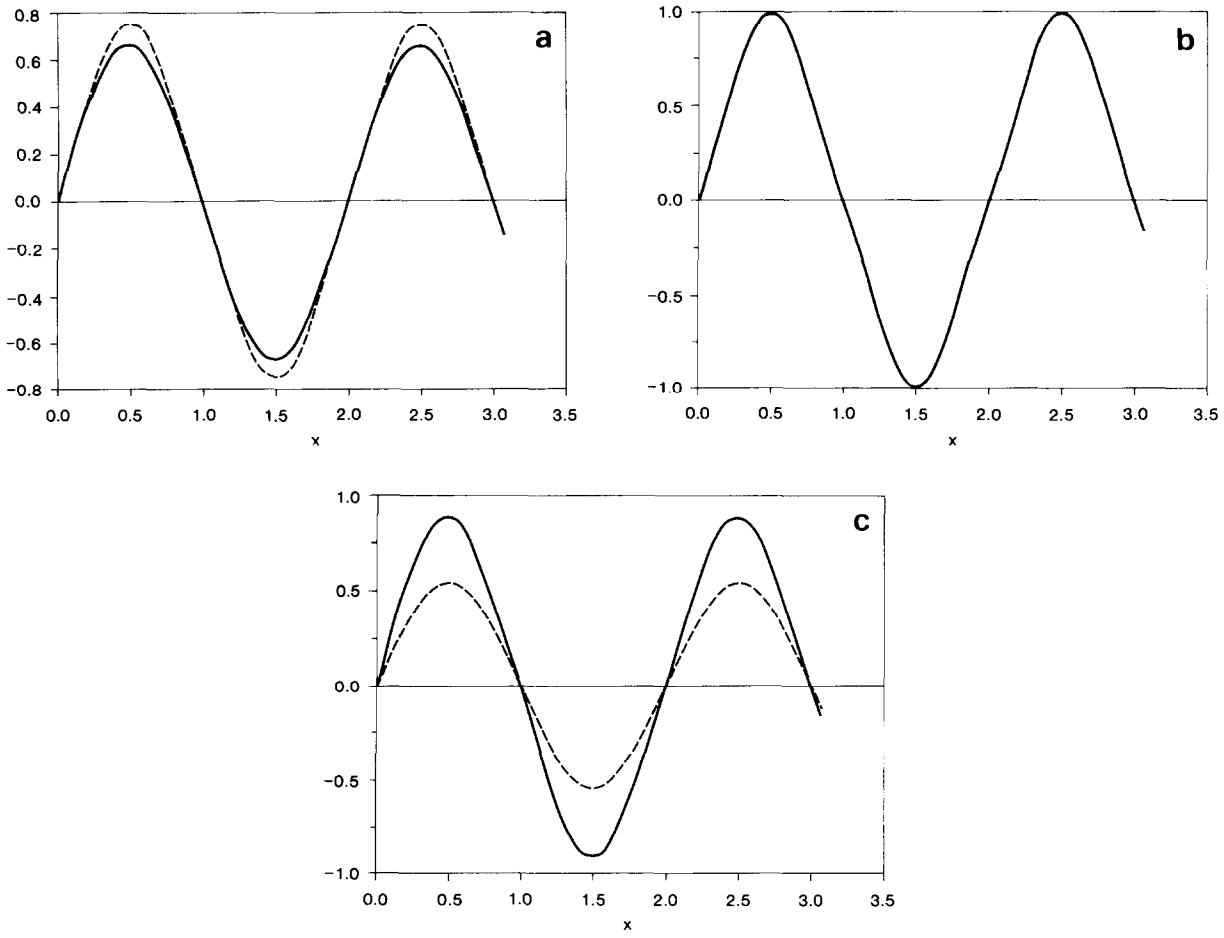

Fig. 1. Interpolations of sin πx using data at equal intervals of 0.5. (a) M_3 - - - - - -, M_4 ———; (b) W_4; (c) Gaussian - - - - - -, improved-Gaussian ———.

In figs. 2a–c the effect of interpolating a step function is shown. The improved kernels show characteristic blips on each side of the discontinuity. The price one has paid for improved interpolation of smooth functions is a loss of monotonicity when the function is discontinuous (or has very steep gradients). Problems arising from loss of monotonicity plague finite difference methods (see for example Harten [30], Van Leer [31], Boris [32]), but they seem to be less troublesome for particle methods (see section 7).

3.9. Calculating derivatives

Provided the interpolating kernels are sufficiently smooth a derivative can be estimated by differentiating the interpolation formula. For example, from

$$\tilde{f}(x, h) = \sum_j f_j M_n(x - x_j, h); \quad n \geqslant 3$$

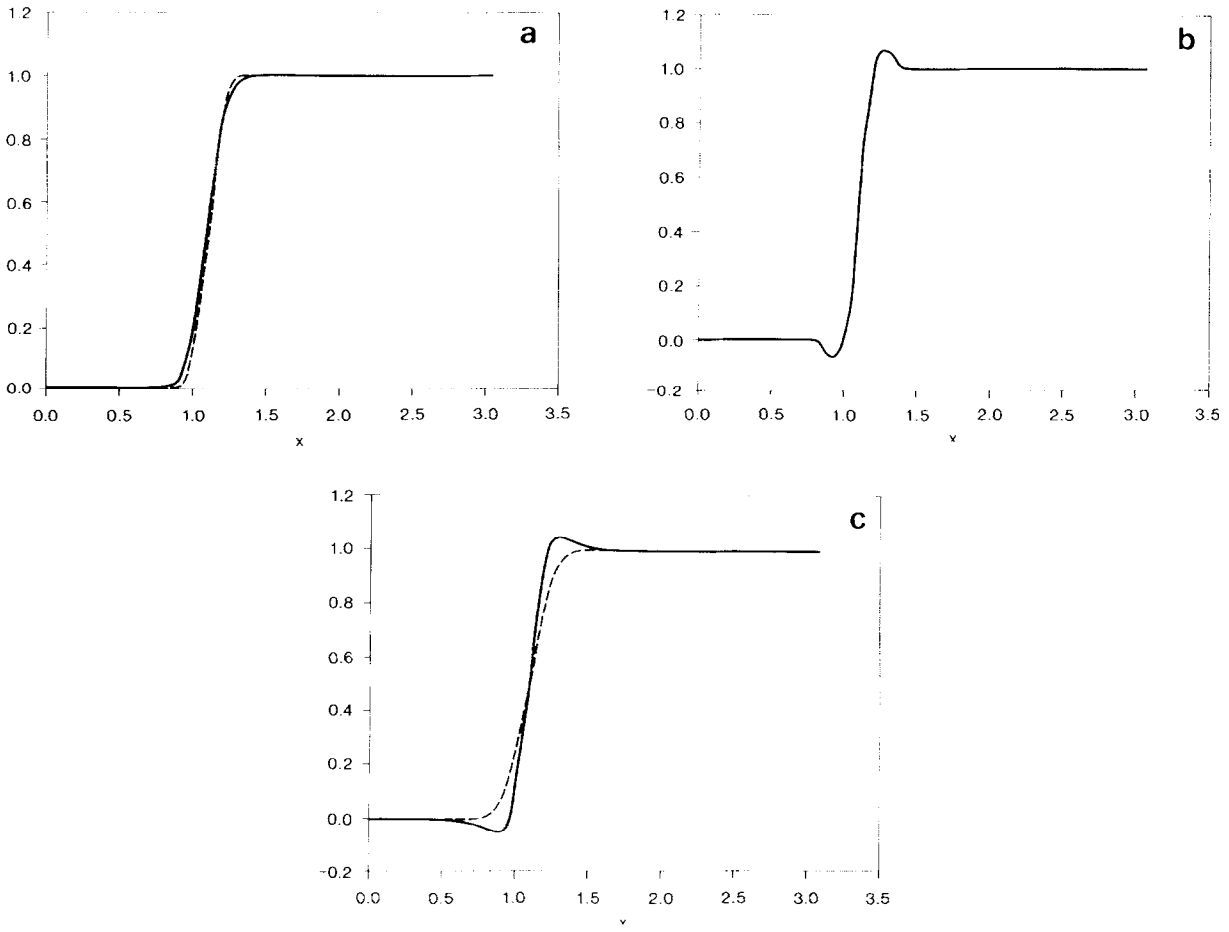

Fig. 2. Interpolation of a step function (a) M_3 - - - - - -, M_4 ———; (b) W_4; (c) Gaussian - - - - - -, improved-Gaussian ———.

we find (with $n > 2$)

$$\frac{\mathrm{d}}{\mathrm{d}x}\tilde{f}(x, h) = \sum_j f_j \frac{\mathrm{d}}{\mathrm{d}x} M_n(x - x_j, h). \tag{3.68}$$

The estimate of the derivative at data point i is then (for $n = 3$ or 4)

$$\left\{\frac{\mathrm{d}\tilde{f}}{\mathrm{d}x}(x, h)\right\}_{x = x_i} = \frac{f_{i+1} - f_{i-1}}{2h}. \tag{3.69}$$

The same result is obtained if the improved basis spline W_4 is used. If the data points are

disordered the same procedure can be used. Thus

$$\frac{\mathrm{d}\tilde{f}(x,h)}{\mathrm{d}x}=\sum_j \frac{f_j}{n_j}\frac{\mathrm{d}}{\mathrm{d}x}W(x-x_j,h). \tag{3.70}$$

3.10. How to choose h

For regular PIC h is chosen by determining how many cells are required for the accuracy desired (or by the number that can be afforded!).

For SPH the choice of h depends on similar considerations. A simple rule, which seems to work well, is to compute the average particle number density $\bar{n}$ and, for a d-dimensional calculation, take $h \propto 1/\bar{n}^{1/d}$. Note that

$$\bar{n}=\frac{1}{N}\sum_{j=1}^{N}\frac{\rho_j}{m_j}\sim \int n^2(\boldsymbol{r})\mathrm{d}\boldsymbol{r}\Big/\int n(\boldsymbol{r})\mathrm{d}\boldsymbol{r},$$

where $n(\boldsymbol{r})$ is the particle number density and $\bar{n}$ is therefore biased towards the higher density regions. The constant of proportionality in $h \propto 1/\bar{n}^{1/d}$ is fixed by the initial state.

3.11. Further possibilities: spatially varying h

We have already noted in section 3.6 that a grid could be chosen with a different h and a different kernel for each coordinate axis. In practice one would use the same kernel along each axis, but in some cases, for example highly flattened systems, it might be useful to allow h to be different for each coordinate axis.

Wood [19,28] and Miyama et al. [29] have used the exponential kernel (3.13) and the Gaussian kernel (3.11) (both generalized in an obvious way to three dimensions) with h varying in space. This is an attractive idea since it seems natural to choose a local h to make maximum use of the local concentration of interpolation points. Experiments (Gingold and Monaghan [33]) show that this procedure does often lead to better interpolation. There are, nevertheless, difficulties when a variable h is used in a hydrodynamic calculation where derivatives must be calculated.

For example, Wood [28] estimates the density at $\boldsymbol{r}$ in three dimensions from

$$\tilde{\rho}(\boldsymbol{r})=\frac{m}{8\pi h^3}\sum_j \mathrm{e}^{-|\boldsymbol{r}-\boldsymbol{r}_j|/h}, \tag{3.71}$$

where h is estimated from the conditions at $\boldsymbol{r}$ and is therefore a function $h(\boldsymbol{r})$. However now

$$\int \tilde{\rho}(\boldsymbol{r})\mathrm{d}\boldsymbol{r}\neq M, \tag{3.72}$$

where M is the total mass. Furthermore, gradients of $\tilde{\rho}(\boldsymbol{r})$ require gradients of h which may be difficult to calculate. An alternative would be to replace (3.64) by

$$\tilde{\rho}(\boldsymbol{r})=\frac{m}{8\pi}\sum_j \frac{1}{h_j^3}\mathrm{e}^{-|\boldsymbol{r}-\boldsymbol{r}_j|/h_j}, \tag{3.73}$$

since then mass is conserved and $\nabla\rho$ does not require gradients of h. The problem now is that the interpolation formula (3.73) is worse than a linear interpolation formula (except in a special case considered below) because the kernel is not a function of $(\boldsymbol{r}-\boldsymbol{r}_j)$. These difficulties can all be seen in the problem of interpolation from a set of arbitrary points in one dimension using, for example, Lagrangian methods. If we interpret such interpolation in terms of kernels, and restrict ourselves to linear interpolation, the kernel should be assigned an h which, in general, is different for the two cases $x > x_j$ and $x < x_j$.

If (3.73) is replaced by its integral interpolant it can be shown that if $h(\boldsymbol{r}) \propto 1/\sqrt{\rho(\boldsymbol{r})}$ the interpolation is accurate to $\mathcal{O}(h^2)$. However, this choice of h would not be optimal for one- or three-dimensional problems.

4. Generalized particle methods

4.1. Generalized PIC equations

The work of the previous section allows us to generalize PIC in the direction of more elaborate interpolation schemes. We assume there is a network of cells, that interpolation is from the vertices of these cells to the particles, and that assignment is from the particles to the vertices. The particles are allowed to have different (constant) masses. A particle will, in general, assign to the vertices other than those of the cell it occupies. In this section the thermal energy per unit mass u will be used instead of the entropy. The steps of PIC are then:

Step 1. Grid velocity update.

$$\tilde{\boldsymbol{v}}(\lambda, n+1) = \boldsymbol{v}(\lambda, n) - \frac{\delta t \sum_{\mu} P(\mu, n) \nabla_\lambda L_{\lambda\mu}}{\rho(\lambda, n)}, \tag{4.1}$$

where $\sum_\mu$ is over grid points, ∇_λ is the gradient taken with respect to the coordinates $\boldsymbol{r}_\lambda$, L is a suitable kernel and $L_{\lambda\mu} \equiv L(\boldsymbol{r}_\lambda - \boldsymbol{r}_\mu)$.

Step 2. Thermal energy update.

$$\boldsymbol{v}^*_\lambda = \tfrac{1}{2}(\tilde{\boldsymbol{v}}(\lambda, n+1) + \boldsymbol{v}(\lambda, n)), \tag{4.2}$$

$$\tilde{u}(\lambda, n+1) = u(\lambda, n) - \delta t \frac{P(\lambda, n)}{\rho(\lambda, n)} \sum_{\mu} \boldsymbol{v}^*_\mu \cdot \nabla_\lambda L_{\lambda\mu} \tag{4.3}$$

which is a finite difference form of the non-advective part of

$$\frac{\partial u}{\partial t} + \boldsymbol{v} \cdot \nabla u = -\frac{P}{\rho} \nabla \cdot \boldsymbol{v}. \tag{4.4}$$

Step 3. Interpolate $\tilde{v}$ to the particles using the kernel L^* (we allow for different kernels being

used for derivatives and normal interpolation)

$$v_j(n+1) = \sum_\lambda \tilde{v}(\lambda, n+1) L^*_{j\lambda}, \tag{4.5}$$

where $L^*_{j\lambda} \equiv L^*(\boldsymbol{r}_j(n) - \boldsymbol{r}_\lambda)$ and the normalization is

$$\sum_\lambda L^*_{j\lambda} = 1.$$

Step 4. Interpolate the energy per unit mass $\tilde{e}$ to the particles. Thus

$$\tilde{e}(\lambda, n+1) = \tilde{u}(\lambda, n+1) + \tfrac{1}{2}[\tilde{v}(\lambda, n+1)]^2 \tag{4.6}$$

and

$$e_j(n+1) = \sum_\lambda \tilde{e}(\lambda, n+1) L^*_{j\lambda}. \tag{4.7}$$

Step 5. Shift the particles.

$$\boldsymbol{r}_j(n+1) = \boldsymbol{r}_j(n) + \delta t\, v_j(n+1) \tag{4.8}$$

This completes the advection phase.

Step 6. Interpolate ρ, ρv and ρe back to the grid. To achieve conservation we must use the kernels L^*, and to agree with the normalization introduced for (3.56) we divide by the cell volume $\Delta\Omega$. Thus

$$\rho(\lambda, n+1) = \frac{1}{\Delta\Omega} \sum_j m_j L^*_{j\lambda}, \tag{4.9}$$

$$\rho(\lambda, n+1) \boldsymbol{v}(\lambda, n+1) = \frac{1}{\Delta\Omega} \sum_j m_j \boldsymbol{v}_j(n+1) L^*_{j\lambda}, \tag{4.10}$$

$$\rho(\lambda, n+1) e(\lambda, n+1) = \frac{1}{\Delta\Omega} \sum_j m_j e_j(n+1) L^*_{j\lambda}, \tag{4.11}$$

where now $L^*_{j\lambda} \equiv L^*(\boldsymbol{r}_j(n+1) - \boldsymbol{r}_\lambda)$.

Step 7. Calculate $P(\lambda, n+1)$ from $\rho(\lambda, n+1)$ and

$$u(\lambda, n+1) = e(\lambda, n+1) - \tfrac{1}{2}(\boldsymbol{v}(\lambda, n+1))^2. \tag{4.12}$$

4.2. Conservation laws for generalized PIC

Mass conservation follows from (4.9) and the normalization described in step 3. From (4.10)

the total grid momentum after a time step is equal to the particle momentum. From step 3 the total particle momentum is the total grid momentum calculated using $\tilde{v}$. Finally from (4.1)

$$\sum_{\lambda} \rho(\lambda, n)\tilde{v}(\lambda, n) = \sum \mathbf{v}(\lambda, n)\rho(\lambda, n) - \delta t \sum_{\lambda} \sum_{\mu} P(\mu, n)\nabla_{\lambda} L_{\lambda\mu}. \tag{4.13}$$

If the boundaries are at infinity where $P \to 0$, the double summation in (4.13) vanishes because

$$\sum_{\lambda} \nabla_{\lambda} L_{\lambda\mu} = -\nabla_{\mu} \sum_{\lambda} L_{\lambda\mu}$$

and

$$\sum_{\lambda} L_{\lambda\mu} = 1.$$

When boundary conditions must be taken into account linear momentum is not, in general, conserved. Boundary conditions for higher order interpolation have not been described in the literature. They may be derived easily for periodic, input and continuative boundaries. If the boundary is rigid the higher order schemes assign matter beyond the boundary. If the fluid forms an isolated lump the higher order schemes may assign a negative density to the cell vertices just outside the fluid. The best prescription for resolving these difficulties is not known.

An alternative form of (4.1) may have some advantages. Noting that

$$\frac{1}{\rho}\nabla P = \nabla\left(\frac{P}{\rho}\right) + \frac{P}{\rho^2}\nabla\rho, \tag{4.14}$$

and defining $g_{\mu} = P(\mu, n)/\rho^2(\mu, n)$, the grid velocity update can be written as

$$\tilde{\mathbf{v}}(\lambda, n+1) = \mathbf{v}(\lambda, n) - \delta t \sum_{\mu} \rho(\mu, n)\left[g_{\mu} + g_{\lambda}\right]\nabla_{\lambda} L_{\lambda\mu}. \tag{4.15}$$

(A similar form is often used for SPH. See section 4.4.) Then since

$$\sum_{\lambda} \sum_{\mu} \rho(\lambda, n)\rho(\mu, n)\left[g_{\mu} + g_{\lambda}\right]\nabla_{\lambda} L_{\lambda\mu}.$$

must vanish ($\nabla_{\lambda} L_{\lambda\mu}$ is anti-symmetric in λ and μ) momentum is conserved. The advantage of (4.15) over (4.1) is that when $L_{\lambda\mu}$ gives high order interpolation exact cancellation does not occur for the fluid in a finite region for (4.1) unless special rules are introduced for the pressure in a number of empty cells exterior to the fluid.

From (4.3) the thermal energy on the grid after step 2 is

$$\sum_{\lambda} \rho(\lambda, n)\tilde{u}(\lambda, n+1) = \sum_{\lambda} \rho(\lambda, n)u(\lambda, n) - \delta t \sum_{\lambda} \sum_{\mu} P(\lambda, n)\mathbf{v}_{\mu}^{*} \cdot \nabla_{\lambda} L_{\lambda\mu}. \tag{4.16}$$

Referring to (4.1) it can be seen (an interchange of labels is needed) that the second term on the r.h.s. of (4.16) is

$$-\tfrac{1}{2}\sum_{\lambda}\rho(\lambda, n)\left[(\tilde{v}(\lambda, n+1))^2-(v(\lambda, n))^2\right], \tag{4.17}$$

so that steps 1 and 2 conserve total energy. It is easy to show that the grid $\rightleftarrows$ particle interpolation conserves total energy so that, overall, total energy is conserved.

If (4.1) had been replaced by (4.15) conservation of energy would have required (4.3) to be replaced by

$$\tilde{u}(\lambda, n+1)=u(\lambda, n)-\delta t\frac{P(\lambda, n)}{\rho(\lambda, n)}\sum_{\mu}\rho(\mu, n)v^*_{\mu\lambda}\cdot\nabla_{\lambda}L_{\lambda\mu}, \tag{4.21}$$

where $v^*_{\mu\lambda}=v^*_{\mu}-v^*_{\lambda}$. Eq. (4.21) is a finite difference form of the non-advective part of (4.4) written in the form

$$\frac{\partial u}{\partial t}+v\cdot\nabla u=-\frac{P}{\rho}\left[\nabla\cdot(\rho v)-v\cdot\nabla\rho\right]. \tag{4.22}$$

Angular momentum is not conserved by PIC because the grid and the kernels are not isotropic.

4.3. Analytical discussion of the generalized PIC equations

Intuitively the PIC equations seem to be consistent, and there is ample evidence from simulations that they give good approximations to the original equations of motion. Nevertheless it would be an advantage to give a formal derivation of the equations since it might suggest alternatives. Bromberg (in the appendix to Amsden [7]) has given a derivation of the PIC equations by considering transport of mass, momentum and energy for cells. The equations can also be derived by using the basic interpolation process to reduce the original equations to the form used in the numerical calculations.

As an example, consider (4.9) which replaces the continuity equation

$$\frac{\partial\rho}{\partial t}+\nabla\cdot(\rho v)=0. \tag{4.23}$$

Multiply (4.23) by the interpolating kernel $L^*(r-r', h)$ and integrate over all space. We find

$$\frac{\partial}{\partial t}\int\rho(r, t)L^*(r-r', h)\mathrm{d}r+\nabla'\cdot\int\rho vL^*(r-r', h)\mathrm{d}r=0 \tag{4.24}$$

where ∇' is the gradient with respect to the coordinate r', and it has been assumed that there is no contribution from surface integrals.

Suppose now we want to evaluate (4.24) at the grid point λ. We find

$$\frac{\partial\rho(r_{\lambda}, t)}{\partial t}+\nabla_{\lambda}\cdot\int\rho vL^*(r-r_{\lambda}, h)\mathrm{d}r=0, \tag{4.25}$$

where, in the first term, we have neglected the interpolation error. If we want a representation of the continuity equation on the grid we replace the second term of (4.25) by

$$\nabla_\lambda \cdot \sum_\mu \rho_\mu v_\mu L^*(\boldsymbol{r}_\mu - \boldsymbol{r}_\lambda, h).$$

However, for PIC, we would like to represent the second term by a sum over particles. Referring to section 3 we can write (4.25) as

$$\frac{\partial}{\partial t}\rho(\boldsymbol{r}_\lambda, t) + \nabla_\lambda \cdot \sum_j m_j \boldsymbol{v}_j L^*(\boldsymbol{r}_j(t) - \boldsymbol{r}_\lambda, h)/\Delta\Omega = 0$$

or

$$\frac{\partial}{\partial t}\rho(\boldsymbol{r}_\lambda, t) + \sum_j m_j \boldsymbol{v}_j \cdot \nabla_\lambda L^*_{j\lambda}/\Delta\Omega = 0, \tag{4.26}$$

where $\boldsymbol{v}_j$ is the velocity of particle j at time t and, as before, $\Delta\Omega$ is the cell volume. This equation is the time derivative of the density assignment eq. (4.9) which we can write as

$$\rho(\boldsymbol{r}_\lambda, t) = \frac{1}{\Delta\Omega}\sum_j m_j L(\boldsymbol{r}_j(t) - \boldsymbol{r}_\lambda). \tag{4.27}$$

This analysis shows that (4.27) is the solution of the equation of continuity in the form (4.26) appropriate for numerical work. The errors arising from (4.27) are the errors due to interpolation and assignment.

The momentum equation can be understood in much the same way. In this case we would like to establish that, within the framework of interpolation, (4.1), (4.5) and (4.10) are equivalent to solving the original equations of motion. The momentum equation for component l of the momentum is

$$\frac{\partial}{\partial t}(\rho v^l) = -\frac{\partial P}{\partial x^l} - \frac{\partial}{\partial x^k}(\rho v^l v^k). \tag{4.28}$$

Multiplying by the kernel, and integrating as before, we find

$$\frac{\partial}{\partial t}(\rho_\lambda v^l_\lambda) = -\frac{\partial}{\partial x^l_\lambda}\int P L^*(\boldsymbol{r} - \boldsymbol{r}_\lambda, h)\mathrm{d}\boldsymbol{r} - \frac{\partial}{\partial x^k_\lambda}\int \rho v^l v^k L^*(\boldsymbol{r} - \boldsymbol{r}_\lambda, h)\mathrm{d}\boldsymbol{r}. \tag{4.29}$$

The crucial assignment equation is (4.10) which we write in the form

$$\rho(\lambda, t)\boldsymbol{v}(\lambda, t) = \frac{1}{\Delta\Omega}\sum_j m_j \boldsymbol{v}_j(t) L^*(\boldsymbol{r}_j(t) - \boldsymbol{r}_\lambda). \tag{4.30}$$

The time derivative of (4.30) is

$$\frac{\partial}{\partial t}(\rho_\lambda v_\lambda^l) = \frac{1}{\Delta\Omega}\sum_j m_j\left\{\frac{\mathrm{d}}{\mathrm{d}t}v_j^l(t)\right\}L_{j\lambda}^* + \frac{1}{\Delta\Omega}\sum_j m_j v_j^l(t)\left\{v_j^k(t)\frac{\partial}{\partial x_j^k}L_{j\lambda}^*\right\}, \tag{4.31}$$

where we have noted that the time derivative of $\rho_\lambda v_\lambda$ is at a fixed point (hence $\partial/\partial t$) whereas the time derivative of v_j is that for a moving particle (hence $\mathrm{d}/\mathrm{d}t$). The second summation on the r.h.s. of (4.31), after replacing $\partial L_{j\lambda}^*/\partial x_j^k$ by $-\partial L_{j\lambda}^*/\partial x_\lambda^k$, is the particle interpolation equivalent of the second integral in (4.29). Before launching into a detailed analysis of the 1st summation in (4.31) let us argue roughly. Since $\mathrm{d}v/\mathrm{d}t$ is the acceleration of the particle we can place it equal to the force per unit mass at the particle, i.e. $-\nabla P/\rho$. Then the summation can be identified as the particle interpolation form of

$$-\frac{\partial}{\partial x_\lambda^l}\int PL^*(\boldsymbol{r}-\boldsymbol{r}_\lambda, h)\mathrm{d}\boldsymbol{r},$$

and the time derivative of (4.30) is then perfectly consistent with (4.29). The gap in this argument is that v is found from (4.5) rather than from the equation of motion. However if (4.1) is written as

$$\tilde{\boldsymbol{v}}(\lambda, n+1) = \boldsymbol{v}(\lambda, n) + \delta t\boldsymbol{F}_\lambda,$$

and substituted into (4.5), we find

$$\boldsymbol{v}_j(n+1) = \sum_\lambda \boldsymbol{v}(\lambda, n)L_{j\lambda}^* + \delta t\sum_\lambda \boldsymbol{F}_\lambda L_{j\lambda}^*. \tag{4.32}$$

Apart from obvious interpolation errors (4.32) is equivalent to

$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{v}_j(t) = \boldsymbol{F}_j(t),$$

where $\boldsymbol{F}_j(t)$ is the pressure force per unit mass at particle j. This incidentally points to an alternative to the usual PIC. Calculate the pressure force per unit mass, and the term $P\nabla\cdot\boldsymbol{v}/\rho$ required for the energy equation, on the grid then interpolate them to the particles. The intermediate quantities $\bar{v}$ and $\bar{u}$ are then eliminated. In their place one stores the velocity and thermal energy for each particle. Marder [48] has described a form of PIC which uses the pressure forces in this way.

4.4. Smoothed particle hydrodynamics

Smoothed particle hydrodynamics (SPH) was devised for astrophysical hydrodynamics (Gingold and Monaghan [13], Lucy [14]) where, for some problems, the grids of PIC, or those of finite difference schemes, become inconveniently large. The method is based on the fact that the interpolation schemes we have discussed can be used to give smooth estimates of the nth

derivatives of a function, provided the interpolation kernels have at least nth order continuous derivatives. The grid is therefore unnecessary, and the interpolation back and forth between particles and grid, with its resulting diffusion, can be omitted. The algorithm has much less diffusion than PIC and is very robust and flexible. It has been applied to instabilities in binary star systems (Gingold and Monaghan [35,36]), to the fission of a rotating star (Gingold and Monaghan [34], Lucy [14]) to the fragmentation of gas clouds (Gingold and Monaghan [37,38], Wood [61]) and to the collision of interstellar clouds (Lattanzio et al. [41]). A derivation of the SPH equations from the hydrodynamic equations has been given by Monaghan [17].

From (3.58) the interpolated pressure is

$$\sum_j m_j \frac{P_j}{\rho_j} W(\boldsymbol{r} - \boldsymbol{r}_j, h), \tag{4.33}$$

where the normalization is

$$\int W(\boldsymbol{u}, h) \mathrm{d}\boldsymbol{u} = 1. \tag{4.34}$$

and the summation is over all particles (though only those within a few h or $\boldsymbol{r}$ contribute). The pressure gradient is then

$$\sum_j m_j \frac{P_j}{\rho_j} \nabla W(\boldsymbol{r} - \boldsymbol{r}_j, h). \tag{4.34}$$

The momentum equation for particle i can then be written

$$\frac{\mathrm{d}\boldsymbol{v}_i}{\mathrm{d}t} = -\frac{1}{\rho_i} \sum_j m_j \frac{P_j}{\rho_j} \nabla_i W_{ij}, \tag{4.36}$$

where $W_{ij} := W(\boldsymbol{r}_i - \boldsymbol{r}_j, h)$, $\mathrm{d}/\mathrm{d}t$ is the derivative following the motion, and ∇_i denotes the gradient taken with respect to the coordinates of particle i. The density ρ_i in (4.36) is the interpolated density

$$\tilde{\rho}_i = \sum_j m_j W_{ij}. \tag{4.37}$$

The calculation of the pressure requires, in general, the solution of the energy equation. Most SPH calculations have used isothermal and adiabatic approximations (the exceptions are Lucy [14] who includes radiative transfer and Monaghan and Gingold [39] who allows for viscous dissipation). The energy equation will be discussed later.

Although most SPH calculations use a leapfrog algorithm it is convenient, in order to compare equations with the standard PIC method, to use the Euler algorithm *. In the absence of

* The reader is warned that this algorithm is unstable if there is no artificial dissipation.

dissipation we can replace (4.36) by

$$\boldsymbol{v}_i(n+1) = \boldsymbol{v}_i(n) - \frac{\delta t}{\rho_i} \sum_j m_j \frac{P_j}{\rho_j} \nabla_i W_{ij}. \tag{4.38}$$

with position updated from

$$\boldsymbol{r}_i(n+1) = \boldsymbol{r}_i(n) + \delta t\, \boldsymbol{v}_i(n). \tag{4.39}$$

ρ_i is calculated from (4.37) and the pressure calculated using the fact that each particle carried its entropy unchanged or that the fluid remains isothermal.

Eq. (4.38) does not result in momentum conservation, but it can be made to do so by the device mentioned in section 4.2, i.e. by using the relation

$$\frac{1}{\rho} \nabla P = \nabla \left(\frac{P}{\rho} \right) + \frac{P}{\rho^2} \nabla \rho,$$

which results in (4.36) being replaced by

$$\frac{\mathrm{d}\boldsymbol{v}_i}{\mathrm{d}t} = -\sum_j m_j \left(\frac{P_j}{\rho_j^2} + \frac{P_i}{\rho_i^2} \right) \nabla_i W_{ij}, \tag{4.40}$$

and (4.38) being replaced by

$$\boldsymbol{v}_i(n+1) = \boldsymbol{v}_i(n) - \delta t \sum_j m_j \left(\frac{P_j}{\rho_j^2} + \frac{P_i}{\rho_i^2} \right) \nabla_i W_{ij}. \tag{4.41}$$

Provided W_{ij} has the form $K(|x_i - x_j|, |y_i - y_j|, |z_i - z_j|)$, (4.41) guarantees linear momentum conservation. If $W_{ij} = K(|\boldsymbol{r}_i - \boldsymbol{r}_j|)$ then (4.41) also gives angular momentum conservation. In this latter case the SPH method also gives excellent angular momentum transport (Gingold and Monaghan [38], Monaghan and Lattanzio [60]).

There are at least two alternatives to (4.40) which also conserve linear and angular momentum. These are

$$\frac{\mathrm{d}\boldsymbol{v}_i}{\mathrm{d}t} = -\sum_j m_j \frac{\sqrt{P_i P_j}}{\rho_i \rho_j} \nabla_i W_{ij} \tag{4.42}$$

which is based on the relation $(\nabla P)/\rho = 2\sqrt{P}\,(\nabla \sqrt{P})/\rho$ and

$$\frac{\mathrm{d}\boldsymbol{v}_i}{\mathrm{d}t} = -\sum_j m_j \frac{(P_i + P_j)}{\rho_i \rho_j} \nabla_i W_{ij} \tag{4.43}$$

which is based on the relation $(\nabla P)/\rho = (\nabla P)/\rho + (P/\rho)\nabla 1$ with

$$\nabla 1 = \nabla \int W(\boldsymbol{r} - \boldsymbol{r}', h) \mathrm{d}\boldsymbol{r}' = \sum_j \frac{m_j}{\rho_j} \nabla W(\boldsymbol{r} - \boldsymbol{r}_j, h). \tag{4.44}$$

The only published SPH results have used (4.36) or (4.40). In unpublished experiments on shock waves (4.42) and (4.43) had no advantages over (4.40).

If the gas is isothermal (4.36) had a computational advantage over (4.40) since the density and the summation can be evaluated at the same time. When the motion is adiabatic, with $P = A(s)\rho^{\gamma}$, then a useful form for $\nabla P/\rho$ is obtained by using the relation

$$\nabla(A\rho^{\gamma}) = \rho^{\gamma-1}\left[\nabla(A\rho) + (\gamma-1)A\nabla\rho\right], \tag{4.45}$$

which leads to the following momentum equation

$$\frac{\mathrm{d}\boldsymbol{v}_i}{\mathrm{d}t} = -\rho_i^{\gamma-2}\sum_j m_j\left(A_j + (\gamma-1)A_i\right)\nabla_i W_{ij}. \tag{4.46}$$

Although this equation does not give exact momentum conservation in practice (Lorimer [40]) the error is negligible.

When the motion is dissipative, or the adiabatic equation is inconvenient to obtain directly, the energy equation may be represented in various ways. There have been no studies of this equation in the literature with the exception of a calculation by Lucy [14] which, unfortunately, does not contain enough detail to assess the errors. The energy equation, in the absence of sources and sinks of energy, is

$$\frac{\mathrm{d}u}{\mathrm{d}t} = -\frac{P}{\rho}\nabla\cdot\boldsymbol{v}, \tag{4.47}$$

which can be written, in SPH form, either as

$$\frac{\mathrm{d}u_i}{\mathrm{d}t} = -\frac{P_i}{\rho_i}\sum_j m_j\frac{\boldsymbol{v}_j}{\rho_j}\cdot\nabla_i W_{ij}, \tag{4.48}$$

or as

$$\frac{\mathrm{d}u_i}{\mathrm{d}t} = -\frac{P_i}{\rho_i^2}\sum_j m_j(\boldsymbol{v}_j - \boldsymbol{v}_i)\cdot\nabla_i W_{ij}, \tag{4.49}$$

where the relation $\nabla\cdot\boldsymbol{v} = (\nabla\cdot(\rho\boldsymbol{v}) - \boldsymbol{v}\cdot\nabla\rho)/\rho$ has been used. Eq. (4.49) is computationally superior to (4.48) because the sum can be calculated when ρ is calculated. The Euler algorithm for (4.49) then gives

$$u_i(n+1) - u_i(n) = -\delta t\frac{P_i}{\rho_i^2}\sum_j m_j(\boldsymbol{v}_j - \boldsymbol{v}_i)\cdot\nabla_i W_{ij}, \tag{4.50}$$

with a similar equation replacing (4.48). Neither of these energy algorithms conserves total energy exactly. Conservation can be guaranteed by using the momentum equation or by using an equation for the total energy. Using the first method we can form an energy equation by taking

$m_i \boldsymbol{v}_i^* \cdot$ (4.41) where

$$\boldsymbol{v}_i^* = \tfrac{1}{2}(\boldsymbol{v}_i(n+1) + \boldsymbol{v}_i(n)), \tag{4.51}$$

then summing over i. We find

$$\tfrac{1}{2}\sum_i m_i\left[\boldsymbol{v}_i(n+1)^2 - \boldsymbol{v}_i(n)^2\right] = -\delta t \sum_i \sum_j m_i m_j (g_i + g_j)\boldsymbol{v}_i^* \cdot \nabla_i W_{ij}, \tag{4.52}$$

where $g_i = P_i/\rho_i^2$. The r.h.s. of (4.52) can be split into two terms, the first being

$$-\delta t \sum_i m_i g_i \boldsymbol{v}_i^* \cdot \sum_j m_j \nabla_i W_{ij}, \tag{4.53}$$

and the second, after interchanging i and j in the double sum, is

$$-\delta t \sum_i m_i g_i \sum_j m_j \boldsymbol{v}_j^* \cdot \nabla_i W_{ij}. \tag{4.54}$$

Eq. (4.52) is therefore

$$\tfrac{1}{2}\sum_i m_i\left[\boldsymbol{v}_i(n+1)^2 - \boldsymbol{v}_i(n)^2\right] = -\delta t \sum_i m_i g_i \sum_j m_j \boldsymbol{v}_{ij}^* \cdot \nabla_i W_{ij}, \tag{4.55}$$

where $\boldsymbol{v}_{ij}^* = \boldsymbol{v}_i^* - \boldsymbol{v}_j^*$. The thermal energy equation which conserves total energy is then

$$u_i(n+1) - u_i(n) = \delta t \frac{P_i}{\rho_i^2} \sum_j m_j \boldsymbol{v}_{ij}^* \cdot \nabla_i W_{ij}. \tag{4.56}$$

This equation could have been obtained by writting

$$\frac{P}{\rho}\nabla \cdot \boldsymbol{v} = \frac{P}{\rho^2}(\nabla \cdot (\rho \boldsymbol{v}) - \boldsymbol{v} \cdot \nabla \rho), \tag{4.57}$$

and using particle interpolation forms of $\nabla \cdot (\rho \boldsymbol{v})$ and $\boldsymbol{V} \cdot \nabla \rho$ and replacing $v(n)$ by $v^*(n)$ in the Euler algorithm. The computational disadvantage of (4.56) is that the calculation of the r.h.s. requires $\boldsymbol{v}_i(n+1)$ and this is only available after the density and pressure forces have been obtained.

The second way to guarantee energy conservation is to use the equation

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(u + \tfrac{1}{2}v^2\right) = -\frac{P}{\rho}\nabla \cdot \boldsymbol{v} + \boldsymbol{v} \cdot \frac{\mathrm{d}\boldsymbol{v}}{\mathrm{d}t}, \tag{4.58}$$

and replace (for the ith particle)

$$-\frac{P}{\rho}\nabla\cdot\mathbf{v} \quad \text{by} \quad -\frac{P_i}{\rho_i}\sum_j \frac{m_j}{\rho_j}\mathbf{v}_j\cdot\nabla_i W_{ij}, \tag{4.59}$$

$$\mathbf{v}\cdot\frac{\mathrm{d}\mathbf{v}}{\mathrm{d}t} \quad \text{by} \quad -\frac{\mathbf{v}_i}{\rho_i}\cdot\sum_j m_j\frac{P_j}{\rho_j}\nabla_i W_{ij}, \tag{4.60}$$

which gives the algorithm

$$u_i(n+1)+\tfrac{1}{2}\mathbf{v}_i(n+1)^2 = u_i(n)+\tfrac{1}{2}\mathbf{v}_i(n)^2 - \delta t\sum_j \frac{m_j}{\rho_i\rho_j}\left(P_i\mathbf{v}_j+\mathbf{v}_i P_j\right)\cdot\nabla_i W_{ij}. \tag{4.61}$$

It is then easy to show that for symmetric kernels energy is conserved exactly. As remarked earlier, none of the energy equations (4.50), (4.56) and (4.61) have been discussed in the literature. Harlow [2] and Amsden [7] found that for PIC it was better to use (4.47) and update specific thermal energy on the grid rather than use (4.58) and update total specific energy. One problem with the use of (4.58) is that, if the specific kinetic energy is much greater than the specific thermal energy, errors in v may contaminate the values of u_i to the extent that they become unphysical. Another form of the SPH energy equation is given by Monaghan [17].

5. Artificial viscosity

5.1. Artificial viscosity for PIC

Shock phenomena can easily be treated by particle methods. The original form of PIC (Harlow [2]) is very diffusive and it is, in fact, better suited to problems involving supersonic flow. If better interpolation methods are used in the PIC calculations then an artificial viscosity is needed to treat shock phenomena. It can be modelled on the artificial viscosities used for finite difference calculations (Roache [42]). These artificial viscosities generally take the form of a bulk viscosity which has the effect of increasing the pressure when $\nabla\cdot\mathbf{v}<0$. In one dimension the pressure P is replaced by $P+q$ where, as typical examples,

$$q = -\alpha h\rho c\frac{\partial v}{\partial x} \tag{5.1}$$

or

$$q = \alpha\rho h^2\left(\frac{\partial v}{\partial x}\right)^2, \tag{5.2}$$

provided $\partial v/\partial x<0$. If $\partial v/\partial x\geq 0$ set $q=0$. In (5.1) and (5.2) α is a constant, h is the cell width and c in a velocity (typically the local speed of sound). A combination of the two forms of q is often an advantage (Roache [42]).

The generalization of (5.1) and (5.2) to more than one dimension is not unique. A simple choice is to replace (5.1) and (5.2) by

$$-\alpha hc\nabla\cdot\boldsymbol{v} \quad \text{and} \quad \alpha\rho h^2(\nabla\cdot\boldsymbol{v})^2, \tag{5.3}$$

respectively. For some purposes other generalizations (Tscharnuter and Winkler [43]) are better.

For consistency the energy dissipated by the artificial viscosity must appear in the energy equation. If q at grid point λ is available then it is only necessary to replace (4.3) by

$$\rho_\lambda^n(\tilde{u}_\lambda^{n+1}-u_\lambda^n) = -\delta t(P_\lambda^n+q_\lambda^n)\sum_\mu \boldsymbol{v}_\mu^*\cdot\nabla_\lambda W_{\lambda\mu}. \tag{5.4}$$

5.2. *Artificial viscosity for SPH*

Because SPH is less diffusive than PIC it always needs an artificial viscosity for shock phenomena. In addition, because the velocity is a particle property, it needs an artificial viscosity to prevent streaming. For example, if a suitable artificial viscosity is not used, two gas clouds colliding at high Mach number will be simulated by two groups of particles which stream through each other. Hausman [44] found this occurring in his simulation of gas clouds using Larson's [45] particle method. Marder [48] devised a viscosity for his GAP algorithm by calculating an average cell velocity $\langle v\rangle_\lambda$ and introducing a drag term $f(\langle v\rangle_\lambda - v_i)$ into the momentum for particle i in the cell λ. A similar device was used by Brunel et al. [12] for an MHD particle method. No extensive tests of these viscosities for shock tube phenomena are available.

Viscosities like (5.1) and (5.2) have been used in SPH calculations (Lucy [14], Wood [19]). However Monaghan and Gingold [39] found that they give disappointing results for shock tube phenomena. If the viscosity coefficient is made large enough to prevent post shock oscillations the smearing of the shock front is found to be too great. Monaghan and Gingold suggested that the problem arises because the calculation of $\nabla\cdot\boldsymbol{v}$ is an average over several particles and it is insensitive to local velocity differences. They suggested and tested a momentum equation of the form

$$\frac{\mathrm{d}\boldsymbol{v}_i}{\mathrm{d}t} = -\sum_j m_j\left(\frac{P_i}{\rho_i^2}+\frac{P_j}{\rho_j^2}-\frac{\alpha h\bar{c}_{ij}}{\bar{\rho}_{ij}}\sigma_{ij}\right)\nabla_i W_{ij}, \tag{5.5}$$

where the term involving σ_{ij} provides the artificial viscosity. In (5.5) α is a constant (≈ 1), c_i is the velocity of sound at particle i and $\bar{c}_{ij}=\frac{1}{2}(c_i+c_j)$ with a similar definition for $\bar{\rho}_{ij}$. σ_{ij} is defined by

$$\sigma_{ij} = \begin{cases} \dfrac{\boldsymbol{v}_{ij}\cdot\boldsymbol{r}_{ij}}{r_{ij}^2+\epsilon h^2}; & \text{if } \boldsymbol{v}_{ij}\cdot\boldsymbol{r}_{ij}<0, \\ 0 & \text{otherwise,} \end{cases} \tag{5.6}$$

where $\boldsymbol{v}_{ij} = \boldsymbol{v}_i - \boldsymbol{v}_j$, $\boldsymbol{r}_{ij} = \boldsymbol{r}_i - \boldsymbol{r}_j$ and $\epsilon \approx 0.1$. The condition $\boldsymbol{v}_{ij} \cdot \boldsymbol{r}_{ij} < 0$ means that $\sigma_{ij} \neq 0$ only for approaching particles. This is the particle equivalent of $\nabla \cdot \boldsymbol{v} < 0$. The viscosity conserves linear momentum (and angular momentum if $W_{ij} \equiv K(|\boldsymbol{r}_i - \boldsymbol{r}_j|)$), it vanishes for rigid rotation and is Galilean invariant. In one dimension the momentum equation becomes, in the continuum limit, (with $\epsilon \to 0$)

$$\frac{\mathrm{d}v}{\mathrm{d}t} = -\frac{1}{\rho}\frac{\partial P}{\partial x} + \frac{\alpha h}{2\rho}\frac{\partial}{\partial x}\left(\rho c \frac{\partial v}{\partial x}\right), \tag{5.7}$$

showing that in one dimension it mimics a bulk viscosity. In three dimensions it mimics a Navier–Stokes viscosity with both first and second (bulk) viscosity coefficients.

Monaghan and Pongracic [46] have extensively tested the following variation of (5.5)

$$\frac{\mathrm{d}\boldsymbol{v}_i}{\mathrm{d}t} = -\sum_j m_j \left(\frac{P_i}{\rho_i^2} + \frac{P_j}{\rho_j^2}\right)\left[1 - \frac{\alpha h}{c}\sigma_{ij}\right]\nabla_i W_{ij}, \tag{5.8}$$

where c is the maximum speed of sound, and now $\alpha \approx 7$. This simplified version of (5.5) was suggested by the fact that $\bar{c}_{ij}/\bar{\rho}_{ij}$ in (5.5) provides a symmetric combination with the same dimensions as $P/(\rho^2 c)$. The viscosity has the same invariance properties as that in (5.5). In the one-dimensional continuum limit (5.8) becomes

$$\frac{\mathrm{d}v}{\mathrm{d}t} = -\frac{1}{\rho}\frac{\partial P}{\partial x} + \frac{\alpha h}{c\rho}\frac{\partial}{\partial x}\left(P\frac{\partial v}{\partial x}\right), \tag{5.9}$$

so that the artificial viscosity is a bulk viscosity. In the three-dimensional continuum limit the artificial viscosity produces a viscous force with kth component

$$\frac{\alpha h}{5\rho c}\left[\frac{\partial}{\partial x^l}\left[P\left\{\frac{\partial v^l}{\partial x^k} + \frac{\partial v^k}{\partial x^l}\right\}\right] - \tfrac{1}{3}(\nabla \cdot \boldsymbol{v})P\delta_{lk} + \tfrac{2}{3}\frac{\partial}{\partial x^k}[P\nabla \cdot \boldsymbol{v}]\right], \tag{5.10}$$

which shows that it mimics a Navier–Stokes viscosity with first and second viscosity coefficients.

While (5.8) gives excellent results for shock tube phenomena involving low ($\lesssim 5$) Mach numbers Lattanzio et al. [41] found that in high Mach number cloud collisions the artificial viscosity in (5.5) or (5.8) was inadequate to prevent penetration of one cloud by the other. The artificial viscosity in (5.8) raises the pressure in high Mach number collisions from ρc^2 to $\approx \rho c|\Delta v|$ where $|\Delta v|$ is the relative velocity. What is needed is an artificial viscosity which raises the pressure to $\approx \rho|\Delta v|^2$. This can be achieved by replacing (5.8) by

$$\frac{\mathrm{d}\boldsymbol{v}_i}{\mathrm{d}t} = -\sum_j m_j \left(\frac{P_i}{\rho_i^2} + \frac{P_j}{\rho_j^2}\right)\left[1 - \frac{\alpha h \sigma_{ij}}{c} + \beta\left(\frac{h\sigma_{ij}}{c}\right)^2\right]\nabla_i W_{ij}, \tag{5.11}$$

where α and β are two new constants. In one dimension, in the continuum limit, (5.11) becomes

$$\frac{\mathrm{d}v}{\mathrm{d}t} = -\frac{1}{\rho}\frac{\partial P}{\partial x} + \frac{\alpha h}{c\rho}\frac{\partial}{\partial x}\left(P\frac{\partial v}{\partial x}\right) - \frac{\beta}{\rho}\left(\frac{h}{c}\right)^2\frac{\partial}{\partial x}\left(P\left(\frac{\partial v}{\partial x}\right)^2\right), \tag{5.12}$$

when $\partial v/\partial x < 0$. If $\partial v/\partial x > 0$ the viscous terms do not appear. Eq. (5.12) shows that the new viscosity mimics a normal bulk viscosity plus a Von Neumann–Richtmyer bulk viscosity. The invariance properties of the new viscosity are identical to those of the previous viscosities. Lattanzio et al. [21] find that for the Gaussian kernel satisfactory results can be obtained taking $\alpha = \beta \approx 1$. For the super Gaussian one can take $\alpha = \beta \approx 0.5$.

The effect of the artificial viscosity on the energy equation can be determined by retracing the steps leading to (4.56). We find

$$u_i(n+1) - u_i(n) = \delta t \frac{P_i}{\rho_i^2} \sum_j m_j \left[1 - \frac{\alpha h}{c} \sigma_{ij} + \beta \left(\frac{h}{c} \sigma_{ij} \right)^2 \right] \boldsymbol{v}_{ij}^* \cdot \nabla_i W_{ij}. \tag{5.13}$$

If exact energy conservation is not needed then $\boldsymbol{v}_{ij}^*$ can be replaced by $\boldsymbol{v}_{ij}$. No equivalent of (4.58) has been proposed, possibly because the artificial viscosity used for SPH is not equivalent to adding a 'viscous' pressure to the gas pressure.

6. Programming considerations

A detailed discussion of efficient programming techniques is given by Hockney and Eastwood [18] in the context of particle methods for plasma physics. The following points are basic:

(1) In order to know which particles occupy a given cell the particles should be accessed through a linked list (or its equivalent) LL(j). The label of the last particle assigned to the cell j is held in the head of chain array HOC(j). The procedure is

```
set all LL(k) = 0, all HOC(j) = 0
DO j = 1, Np
  find the cell, say k, particle j is in
  LL(j) = IHOC(k)
  IHOC(k) = j
END DO
```

(2) For small computers like the Vax 11 series, page faults slow down the program unless the particles are relabelled every time step. If this is done the labels of the particles in a cell will be in serial order. Linked lists can then be replaced by the least and greatest particle labels in a cell.

(3) Even though SPH does not need cells a huge saving in computing time is achieved by using them to determine which particles contribute to the properties of any given particle.

(4) Quiet starts for astrophysical problems required a time consuming 'settling down' (Gingold and Monaghan [13]). It seems preferable to set up the initial states by placing particles on a regular grid and adjusting their mass so that the correct initial density is achieved. If the volume of a cell is $\Delta\Omega$ then the mass m_j of particle j on the cell vertex j is $\rho_j \Delta\Omega$.

(5) For serial machines the kernels should be calculated by interpolation.

7. Shock tube and collision problems

Particle methods are not as efficient as well designed finite difference methods for one-dimensional problems. They can, nevertheless, give a good description of shock tube phenomena provided the right artificial viscosity is used. Examples of such calculations for PIC are given by Harlow [67] and for SPH by Monaghan and Gingold [39].

We show in figs. 3a and b the density profile for the rarefaction wave, contact discontinuity and shock front produced by the SPH simulation of the diaphragm problem (Sod [47], Monaghan and Gingold [39]) using the viscosity discussed in section 5, eq. (5.11). For these calculations $h = 0.015$ (twice the initial particle separation), the initial density ratio is 4 : 1, the equation of state is

$$P = A\rho^{1.4} \quad \text{where, at } t = 0,\ A = 1 \quad \text{for } x < 0 \qquad A = 1.25 \quad \text{for } x > 0,$$

and the initial masses are such that those particles with $x < 0$ have mass four times those with $x > 0$.

Fig. 3a shows the profile obtained using the one-dimensional super-Gaussian kernel. Fig. 3b shows the profile obtained using the Gaussian kernel. The improvement resulting from the high order kernel is obvious. For these experiments the parameters in the artificial viscosity are $\alpha = 0.5$, $\beta = 0.5$ for the super-Gaussian, and $\alpha = 1$, $\beta = 1$ for the Gaussian.

For one-dimensional high Mach number collisions the results are similar (Lattanzio et al. [21]) but for Mach numbers $\gtrsim 10$ the super-Gaussian kernel was found to give unstable oscillations. These are due to the undershoot produced by the high order interpolation which drives the pre-shock density negative. Lattanzio et al. [21] describe a simple device which removes this problem.

Referring to the artificial viscosity in (5.11) it can be seen that the onset of problems (i.e. shock fronts which give large jumps in v, and hence ρ) is signalled by large values of $h|\sigma_{ij}|/c$. Define

$$\zeta = \frac{h|\sigma_{ij}|/c}{1 + h|\sigma_{ij}|/c}.$$

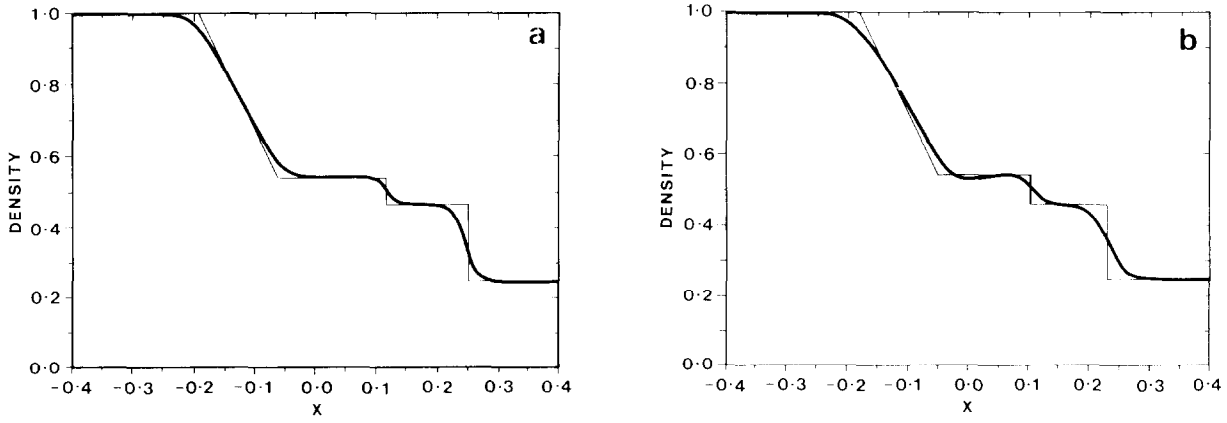

Fig. 3. Density profiles for the shock tube diaphragm problem. (a) Super-Gaussian kernel; (b) Gaussian kernel. Exact results shown ———; SPH results shown _____.

then ζ small ($\ll 1$) means the high order kernel can be used and $\zeta \approx 1$ means it cannot be used. If W^* is the super-Gaussian kernel, and W the Gaussian kernel, Lattanzio et al. replace $\nabla_i W_{ij}^*$ in (5.11) by

$$(1-\zeta)\nabla_i W_{ij}^* + \zeta \nabla_i W_{ij}.$$

It is not known if this is the best method for stabilizing the algorithm, but it works well for the diaphragm problem as well as for collision problems.

8. Particle methods for magnetohydrodynamics (MHD)

Particle methods in the MHD approximation have been described by Leboeuf et al. [11] and Brunel et al. [12] who generalize PIC, and Phillips [49] who generalizes SPH. An early MHD calculation by Butler et al. [61] used PIC in a situation where the only effect of the magnetic field was to provide an additional surface pressure.

The MHD approximation for the dynamics of a conducting fluid in a magnetic field consists of the usual fluid dynamical equations together with an additional force per unit volume

$$(\nabla \times \boldsymbol{B}) \times \boldsymbol{B}/\mu_0, \tag{8.1}$$

where μ_0 is the permeability of free space, and an equation for the time change of $\boldsymbol{B}$ which can take any of the following equivalent forms

$$\frac{\partial \boldsymbol{B}}{\partial t} = \nabla \times (\boldsymbol{v} \times \boldsymbol{B}), \tag{8.2}$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\boldsymbol{B}}{\rho}\right) = \left(\frac{\boldsymbol{B}}{\rho} \cdot \nabla\right) \boldsymbol{v}, \tag{8.3}$$

or

$$\frac{\mathrm{d}\boldsymbol{B}}{\mathrm{d}t} = (\boldsymbol{B} \cdot \nabla)\boldsymbol{v} - \boldsymbol{B}(\nabla \cdot \boldsymbol{v}), \tag{8.4}$$

in the case where the conductivity is effectively infinite. If the conductivity is finite, extra diffusive terms appear in (8.2)–(8.4), and a dissipative contribution appears in the energy equation. The resulting set of equations are considerably more difficult to deal with than the non magnetic equations because (i) there is extra computation and storage (ii) time steps are generally shorter because the magnetic field affects the signal speed and (iii) the magnetic field produces more complicated physical behaviour. As a result the application of particle methods to MHD is in its infancy.

The method proposed by Leboeuf et al. [11] is essentially PIC with (8.2) solved on the grid using an approximately time centred algorithm with a Lax damping term for stability. Particle assignment is by nearest-grid-point interpolation. As a consequence the method resulting in

excessive diffusion (particularly noticeable in simulations of the earth's geomagnetic field) and considerable noise unless a very large number of particles are used.

The cure for the noise and diffusion problems is to use better interpolation methods. Higher accuracy can be achieved by using a time centred algorithm. These improvements were included in the MHD method described by Brunel et al. [12]. They replaced nearest-grid-point assignment by area weighting (the spline M_2 being used for each coordinate) and used the two step Lax–Wendroff algorithm (Richtmeyer and Morton [57]) as a model for a time centred algorithm. Spatial derivatives were calculated by finite differences (using two meshes) or, where applicable, by Fourier methods. When compared, the Fourier and finite difference method gave similar results.

The algorithm of Brunel et al. appears to be capable of dealing with complicated MHD phenomena with reasonable accuracy. It is, however, very much more complicated than PIC algorithms for fluids as the following typical time step cycle shows:

Initial: Grid $\boldsymbol{B}(\lambda, n)$ and particle coordinates and velocities: $\boldsymbol{r}_j(n)$, $\boldsymbol{v}_j(n-\frac{1}{2})$

Step 1 Grid density: $\rho(\lambda, n) = m\sum_j W_{\lambda j}$

Grid pressure: $P(\lambda, n) = F(\rho(\lambda, n))$; barotropic

Grid forces $\boldsymbol{F}(\lambda, n)$ from finite differences:

$$\boldsymbol{F} = (\nabla \times \boldsymbol{B}) \times \boldsymbol{B}/(\rho\mu_0) - (\nabla P)/\rho.$$

Step 2 Particle force: $\boldsymbol{F}_j(n) = \sum_\lambda \boldsymbol{F}(\lambda, n) W_{\lambda j}$

Particle velocity at t^n: $\boldsymbol{v}_j(n) = \boldsymbol{v}_j(n-\frac{1}{2}) + \frac{\delta t}{2}\boldsymbol{F}_j(\mathrm{n})$

Step 3 Velocity on the grid at t^n from: $\rho(\lambda, n)\boldsymbol{v}(\lambda, n) = \sum_j m\boldsymbol{v}_j(n) W_{\lambda j}$

Step 4 Magnetic field at $t^{n+1/2}$: $\boldsymbol{B}(\lambda, \mathrm{n}+\frac{1}{2}) = \langle \boldsymbol{B}(\lambda, n)\rangle + \frac{\delta t}{2}[\nabla \times (\boldsymbol{v} \times \boldsymbol{B})]$, where $\langle \boldsymbol{B}(\lambda, n)\rangle$ denotes an average as in Lax damping, and $\nabla \times (\boldsymbol{v} \times \boldsymbol{B})$ is calculated by differences on the grid using values at t^n.

Step 5 Particle velocity at $t^{n+1/2}$: $\boldsymbol{v}_j(n+\frac{1}{2}) = \boldsymbol{v}_j(n) + \frac{\delta t}{2}\boldsymbol{F}_j(n)$, with $\boldsymbol{F}_j(n)$ from step 2.

Step 6 Particle positions at $t^{n+1/2}$: $\boldsymbol{r}_j(n+\frac{1}{2}) = \boldsymbol{r}_j(n) + \frac{\delta t}{2}\boldsymbol{v}_j(n+\frac{1}{2})$

Grid momentum at $t^{n+1/2}$: $\rho(\lambda, n+\frac{1}{2})\boldsymbol{v}(\lambda, n+\frac{1}{2}) = \sum m\boldsymbol{v}_j(n+\frac{1}{2}) W(\boldsymbol{r}_j(n+\frac{1}{2}) - \boldsymbol{r}_\lambda)$

Grid density at $t^{n+1/2}$: $\rho(\lambda, n+\frac{1}{2}) = \sum m W(\boldsymbol{r}_j(n+\frac{1}{2}) - \boldsymbol{r}_\lambda)$

Step 7 Magnetic field at t^{n+1}: $\boldsymbol{B}(\lambda, n+1) = \langle \boldsymbol{B}(\lambda, n)\rangle + \delta t[\nabla \times (\boldsymbol{v} \times \boldsymbol{B})]$, where $\boldsymbol{v} \times \boldsymbol{B}$ uses values at $t^{n+1/2}$ from step 4 and step 6.

Step 8 Particle positions at t^{n+1}: $\boldsymbol{r}_j(n+1) = \boldsymbol{r}_j(n+\frac{1}{2}) + \frac{\delta t}{2}\boldsymbol{v}_j(n+\frac{1}{2})$.

This algorithm has been applied to Alfvén and sound waves in two dimensions using four particles per cell. When the Lax–Wendroff based scheme (with area weighting assignment) is used the numerical dispersion relations are in good agreement with the exact relations down to wavelengths of ≈ 5 cell widths. When applied to the non-resistive ballooning instability the results are qualitatively correct (they may also be quantitatively correct too but no detailed

comparison between theory and experiment has been made) and they are qualitatively correct for column outflow and simulations of the magnetosphere. No description of the application of the method to problems where the energy equation must be integrated, or to problems where the boundary conditions are more complicated, has appeared in the literature.

Gingold and Monaghan [13] incorporated magnetic fields in their original version of SPH in order to determine the static structure of magnetic stars. Phillips [49] and Phillips and Monaghan [50] have generalized SPH to deal with time dependent MHD phenomena. Because SPH is a Lagrangian method either (8.3) or (8.4) is preferable to (8.2). The magnetic field is defined as a particle attribute and, for example, (8.3) becomes

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\boldsymbol{B}}{\rho}\right)_i = \sum_j m_j(\boldsymbol{v}_j - \boldsymbol{v}_i)\left(\frac{\boldsymbol{B}_i}{\rho_i}\cdot\nabla_i\right)W_{ij}. \tag{8.5}$$

This equation can be integrated forward in time using the algorithm:

If $\dfrac{\mathrm{d}y}{\mathrm{d}t} = f(y, t)$ then

$$y_{n+1} = y_n + t_n\left[(1+2q)f_n - f_{n-1}\right]/2q, \tag{8.6}$$

where $q = \delta t_{n-1}/\delta t_n$ and the subscript n denotes values at time step n.

The magnetic force term can be written in various forms some of which lead to unexpected difficulties. Phillips and Monaghan [50] consider the following expressions for the force per unit mass $(\boldsymbol{F}_M)_i$ at particle i either

(a) $(F_M)_i = (\nabla \times \boldsymbol{B})_i \times \boldsymbol{B}_i/\mu_0\rho_i$
with

$$(\nabla \times \boldsymbol{B})_i = \sum_j m_j \nabla_i W_{ij} \times \boldsymbol{B}_j/\rho_j, \tag{8.7}$$

where the combination $\boldsymbol{B}/\rho$ appears as in (8.5);
or

(b) Define the Maxwell stress tensor

$$M_{lk} = \frac{1}{\mu_0}\left(B_l B_k - \tfrac{1}{2}B^2\delta_{lk}\right) \tag{8.8}$$

so that the lth component of the force per unit mass is $M_{lk;k}/\rho$ or (recall the trick used in section 4.4)

$$\frac{1}{\rho}M_{lk;k} = (M_{lk}/\rho)_{;k} + (\rho_{;k})M_{lk}/\rho^2. \tag{8.9}$$

The lth component of the magnetic force per unit mass on particle i is then

$$F_{M,i}^{(l)} := \sum_{\gamma} m_j \left[\left(M_{lk}/\rho^2 \right)_i + \left(M_{lk}/\rho^2 \right)_b \right] \left(\nabla_i W_{ij} \right)_k , \tag{8.10}$$

where $(\nabla_i W_{ij})_k$ is the kth component of the gradient W_{ij} taken with respect to the coordinates of particle i.

This form of the magnetic force conserves linear and angular momentum when the kernel is symmetric.

The SPH equations which incorporate the magnetic field terms are only slightly more complicated than those described earlier.

Phillips and Monaghan [50] show that for one-dimensional motion (along the x axis) with B_x and B_y components of the magnetic field, the dispersion relation for small amplitude waves when (8.10) is used is

$$\omega^2 = k^2 c_s^2 \left[2Q(k) - Q^2(k) \right] - \left(B_x^2 - B_y^2 \right) k^2 \left[Q(k) - Q^2(k) \right] / \rho_0 \mu_0 + B_x^2 k^2 Q(k) / \mu_0 \rho_0 , \tag{8.11}$$

where ρ_0 is the unperturbed, uniform density and

$$Q(k) = \mathrm{e}^{-h^2 k^2/4} .$$

$Q(k)$ arises from the Fourier transform of the Gaussian kernel.

In the limit $hk \ll 1$, (8.11) is the correct dispersion relation. However if $(hk/2) > 1$, and $B_y = 0$, (8.11) becomes

$$\omega^2 = k^2 c_s^2 \left(2c^2 - \frac{B_x^2}{\rho_0 \mu_0} \right) Q(k) , \tag{8.12}$$

so that if $|B_x| > \sqrt{2\rho_0 c^2 \mu_0}$ the system is unstable. Phillips and Monaghan observed this instability in their numerical simulations. They removed the instability by replacing the stress tensor $\boldsymbol{M}$ by $\boldsymbol{M} - \boldsymbol{S}/\mu_0$ so that the factor $(B_x^2 - B_y^2)$ is replaced by $(B_x^2 - B_y^2 - S_{xx})$. If S_{xx} is chosen correctly this factor is < 0 and the motion is stable. A simple rule is to choose $\boldsymbol{S} = \mathrm{Max}(\boldsymbol{M}\mu_0)$. The error introduced is then only comparable to the errors introduced by the interpolation.

Both the force terms (a), and the stabilized form of (b), have been used to simulate Alfvén and magneto-sonic waves, static magnetic stars, the equilibrium structure of magnetic gas clouds and fragmentation of self-gravitating magnetic clouds (Phillips [49], Phillips and Monaghan [50]).

Small amplitude waves were simulated with an error $\lesssim 5\%$ for wavelengths $> 6h$. The static stars agreed with perturbation calculations to within 10% of the perturbation when 2000 particles were used. The structure of gas clouds agreed with analytical solutions to within 15%. These results show that the SPH method can be used successfully for MHD problems.

No results are available for problems involving either more complicated boundary conditions or more complicated physics.

9. Calculating the gravitational force

9.1. Introduction

When gravitational forces are important, as in astrophysical hydrodynamics, they may be calculated in a variety of ways. The simplest is direct summation which, though not efficient, has advantages for some problems (Gingold and Monaghan [34]). Direct summation takes its simplest form when the kernel W is spherically symmetric. The gravitational force $\boldsymbol{F}_i$ per unit mass on particle i is then given by

$$\boldsymbol{F}_i = -G\sum_{j=1}^{N} m_j \frac{\boldsymbol{r}_{ij}}{|\boldsymbol{r}_{ij}|^3} 4\pi \int_0^{|r_{ij}|} W(u)u^2 \mathrm{d}u, \tag{9.1}$$

where $\boldsymbol{r}_{ij} = \boldsymbol{r}_i - \boldsymbol{r}_j$. With this force linear and angular momentum are exactly conserved, a property that is important when several marginally stable orbits of a binary star are being followed.

When the simulation requires a large number of particles direct summation is too inefficient because the operation count varies with particle number N as N^2. In practice this limits its use to $N \lesssim 10^3$. A recent version of direct summation which uses hierarchical particle clusters, and refers to them by efficient data structures Appel [51], promises to breathe new life into direct summation. The operation count of Appel's method varies with N as $N \log N$ and is more efficient than direct summation for $N \gtrsim 10^3$.

The favoured alternative to direct summation is the direct solution of Poisson's equation on a grid (see for example Hockney and Eastwood [18], Hockney [53], James [62]). The universal practice has been to use an algorithm with moderate accuracy (for example the 7 point formula for three-dimensional problems Smith [52]) on a regular grid. The accuracy of these algorithms is often very poor, and very fine grids may be needed to achieve the desired accuracy. For example, if the equation

$$\nabla^2\phi = -3\pi^2 \sin \pi x \sin \pi y \sin \pi z$$

is solved in the unit cube $0 \leqslant x \leqslant 1$, $0 \leqslant y \leqslant 1$, $0 \leqslant z \leqslant 1$, with $\phi = 0$ on the boundary, the exact solution is

$$\phi = \sin \pi x \sin \pi y \sin \pi z.$$

If the 7-point formula is used the solution is

$$\tilde{\phi} = \left(1 + \frac{\pi^2 h^2}{12}\right) \sin \pi x \sin \pi y \sin \pi z, \tag{9.2}$$

which has fractional error $\pi^2 h^2/12 \approx h^2$. If this fractional error is to be 10^{-4} we require $h \approx 10^{-2}$ and the grid has $\approx 10^6$ points! We therefore immediately run into storage problems despite the fact that the problem has a solution which varies smoothly over the domain. In astrophysical problems where clustering can occur the problem is exacerbated.

9.2. High order methods

An obvious way out of this difficulty is to use an algorithm with higher accuracy. Monaghan [64] has devised a finite difference formula for Poisson's equation $\nabla^2\phi = f$ in three dimensions which uses only adjacent points and has errors of $\mathcal{O}(h^4)$. It takes the form

$$(\sum \omega\phi)_j = 6h^2\left[f + \frac{h^2}{12}\nabla^2 f\right]_j + \mathcal{O}(h^6), \tag{9.3}$$

where j runs over the grid points, $(\sum\omega\phi)_j$ is a sum over grid points adjacent to j using weights ω given below, and h is the separation of neighbouring grid points. $\nabla^2 f$ can be replaced by the 7-point formula. Formula (9.3) is a generalization of two-dimensional formulae considered by Southwell [54] and Thom and Apelt [55].

Let grid points relative to the point of interest be denoted by (l, m, n) where $l, m, n = 0, \pm 1$. If $l = m = n = 0$; $\omega = -24$. If only two of l, m, n are zero; $\omega = 2$. If only one of l, m, n is zero $\omega = 1$. Only points adjacent to the point of interest are used. The weights satisfy the necessary conditions for Jacobi, Gauss–Seidel and SOR to converge. For the problem discussed in section 9.1, the fractional error is $\pi^4 h^4/60$. To achieve a fractional error of 10^{-4} we need $h \approx 10^{-1}$ and the number of grid points is now only $\approx 10^3$.

9.3. The multi-grid algorithm

An efficient method for the solution of (9.3) is the multi-grid method pioneered by Brandt [56] (see also Fedorenko [64], Hackbusch [58], Nicolaides [59]). Rather than aim at full generality I shall describe a 3-grid algorithm. F denotes the finest grid with cell width $h_F = h$. C denotes a coarser grid with cell width $h_C = 2h$, and D is a still coarser grid with cell width $h_D = 4h$. The grids C and D fit exactly on the F grid provided the number of grid points on an edge of the F grid has the form $4n + 1$, where n is an integer.

The multi-grid method uses iteration, but it is enormously faster than the usual iteration methods because it uses coarser grids to calculate corrections when the iteration on a finer grid slows down. Consider for example (9.3) which is a finite difference representation of

$$\nabla^2\phi + \frac{h^2}{12}\nabla^4\phi = f + \frac{h^2}{12}\nabla^2 f. \tag{9.4}$$

Suppose we have, at any stage, an approximation to ϕ and denote it by ϕ^*. We can correct this ϕ^* by calculating the residual

$$r = \nabla^2\phi^* + \frac{h^2}{12}\nabla^4\phi^* - f - \frac{h^2}{12}\nabla^2 f, \tag{9.5}$$

and then calculating the correction V (on the C grid, see below) by solving

$$\nabla^2 V + \frac{h^2}{12}\nabla^4 V = r. \tag{9.6}$$

The corrected $\phi = \phi^* - V$. In the same way if, during the process of solving (9.6), we have an approximation V^* to V we can improve it by first constructing

$$q = \nabla^2 V^* + \frac{h^2}{12}\nabla^4 V^* - r, \tag{9.7}$$

then solving (on the D grid, see below)

$$\nabla^2 \zeta + \frac{h^2}{12}\nabla^4 \zeta = q, \tag{9.8}$$

after which $V = V^* - \zeta$.

The finite difference equations which replace (9.4), (9.6) and (9.8) are

$$\sum_{\mathrm{F}} W\phi/(6h_{\mathrm{F}}^2) = f + \frac{h_{\mathrm{F}}^2}{12}\nabla^2 f, \tag{9.9}$$

where the 7-point formula is used to write $\nabla^2 f$ in finite difference form.

$$\sum_{\mathrm{C}} WV/(6h_{\mathrm{C}}^2) = r, \tag{9.10}$$

$$\sum_{\mathrm{D}} W\zeta/(6h_{\mathrm{D}}^2) = q, \tag{9.11}$$

with

$$r = \frac{\sum_{\mathrm{F}} W\phi}{6h_{\mathrm{F}}^2} - f - \frac{h_{\mathrm{F}}^2}{12}\nabla^2 f, \tag{9.12}$$

and

$$q = \frac{\sum_{\mathrm{C}} WV}{6h_{\mathrm{C}}^2} - r. \tag{9.13}$$

The iteration formula based on (9.9) is

$$\phi_{\mathrm{new}} = \phi_{\mathrm{old}} + \frac{\tau}{24}\left(\sum W\phi - 6h_{\mathrm{F}}^2\left\{f + \frac{h_{\mathrm{F}}^2}{12}\nabla^2 f\right\}\right)_{\mathrm{old,new}}. \tag{9.14}$$

where the subscript old means old value, new means new value and old, new means use the new values when available. τ is a parameter (equal to 1 for Gauss–Seidel iteration) and the factor 24 arises because of the weight (-24) associated with the point of interest. The grids F, C and D are used as follows:

(1) ϕ^* is found by taking an initial approximation to ϕ and making 2 or 3 Gauss–Seidel sweeps which efficiently smooth r.

(2) Because r is smooth V can be solved on grid C. The values of r on grid C are the values on the equivalent point on the F grid, i.e. injection is used.
(3) When solving for V the initial approximation (usually zero) is corrected by a few Gauss–Seidel sweeps which efficiently smooth q.
(4) Because q is smooth ζ can be solved on grid D. The values of q on grid D are the injected values from C. This grid is usually so small that ζ can be found very quickly by successive over relaxation. In the calculations to be described we taken the SOR parameter to be 1.75.
(5) Interpolate ζ to grid C (by 4th order interpolation) and correct V.
(6) Continue Gauss–Seidel iterations on grid C provided the convergence is fast enough. If it is not, go back to grid D and calculate a new correction ζ.
(7) When V is obtained to the desired accuracy interpolate it to grid F (by 4th order interpolation) and correct ϕ^*.
(8) Continue Gauss–Seidel iterations on grid F provided the convergence is fast enough. If it is not, go back to grid C and calculate a new correction V, i.e. go back to (2).

Whether or not convergence is rapid enough on grids F and C is determined by the decrease of the residual which is defined for the F grid by the average of

$$\left| \sum_{\mathrm{F}} W\phi - 6h_{\mathrm{F}}^2 \left(f + \frac{h_{\mathrm{F}}^2}{12} \nabla^2 f \right)_{\mathrm{F}} \right| \tag{9.15}$$

on the F grid. Note that the average change to ϕ in a Gauss–Seidel sweep is $1/24$ of this residual. For grid C the residual is the average of

$$\left| \sum_{\mathrm{C}} WV - 6h_{\mathrm{C}}^2 r \right|, \tag{9.16}$$

on the C grid. Convergence on a given grid is deemed to be rapid enough if the residual after an iteration $\leqslant 0.70$ of its value before the iteration.

9.4. Boundary conditions

For the gravitational force calculation the matter occupies a finite region and the density vanishes on the surface. In order to calculate the potential this region can be surrounded by a cubical grid. It is usually convenient, for reasons which will be given later, to choose this grid so that there are at least two layers of empty cells surrounding the matter.

The potential on the surface of the cube can be estimated easily from the multipole moment expansion of the potential, i.e. if $\boldsymbol{R}$ is on the surface we take

$$\phi(\boldsymbol{R}) = -G\int \frac{\rho(\boldsymbol{r})}{|\boldsymbol{R}-\boldsymbol{r}|}\,\mathrm{d}\boldsymbol{r} = -G\sum_{n=0}^{\infty} \frac{1}{R^{n+1}} \int \rho(\boldsymbol{r}) r^n P_n(\cos\theta)\,\mathrm{d}\boldsymbol{r}, \tag{9.17}$$

where $\cos\theta = \boldsymbol{r}\cdot\boldsymbol{R}/rR$. The addition theorem for Legendre polynomials can be used to express $P_n(\cos\theta)$ in terms of associated Legendre polynomials depending on the angular coordinates of

$\boldsymbol{R}$ and the angular coordinates in the integration. The integral can be evaluated in the form

$$\int \rho(\boldsymbol{r}) Q(\boldsymbol{r}) \mathrm{d}\boldsymbol{r} = \sum_{j=1}^{N} m_j Q(r_j), \tag{9.18}$$

which means that the potential of each particle is being included in the form of a multipole expansion. Even for very highly distorted configurations, e.g. disks with imbedded binary stars, it is adequate to take $n \leqslant 6$ in the multipole moment expansion.

9.5. Density assignment and force calculation

The density can be assigned from the particles to the grid as discussed in section 3. For the 7-point difference formula it is adequate to use, say, the basis spline M_3 for each coordinate direction. Monaghan and Lattanzio [65] use a spherically symmetric form of M_3. As Hockney and Eastwood [18] show, the same kernel used for density assignment should be used for force interpolation to guarantee linear momentum conservation.

When the 7-point formula is used for Poisson's equation the force can be calculated using the central difference formula which, for one dimension, is

$$\left(\frac{\partial \phi}{\partial x}\right)_j = \frac{\phi_{j+1} - \phi_{j-1}}{2h}. \tag{9.19}$$

Provided the grid has at least two empty layers of cells surrounding the particles the force does not need to be calculated on the boundary since boundary points do not contribute to the interpolation to the particles.

When the highly accurate formula is used for Poisson's equation it is necessary, for consistency, to assign and interpolate, with kernels which are more accurate than the linearly interpolating basis splines. The kernel W_4 has been found to give a practical compromise between accuracy and excessive computation time (the latter is due to the rapidly increasing number of vertices which receive an assignment from a given particle when the accuracy and smoothness both increase). Furthermore, because the solution of Poisson's equation is not sensitive to short wave length errors in the density, the error in the potential resulting from the use of W_4 is slight.

Except for the second layer of cells the gradients can be calculated using a central difference formula with errors $\mathcal{O}(h^4)$. In one dimension

$$\left(\frac{\partial \phi}{\partial x}\right)_j = \frac{8(\phi_{j+1} - \phi_{j-1}) - (\phi_{j+2} - \phi_{j-2})}{12h}. \tag{9.20}$$

For the second layer of cells the gradients can be calculated using (9.19). Of course (9.19) is not as accurate as (9.20), but the gradients are low near the surface. Furthermore, only a small amount of matter is influenced by gradients near the surface, and the effects of using (9.19) are therefore small.

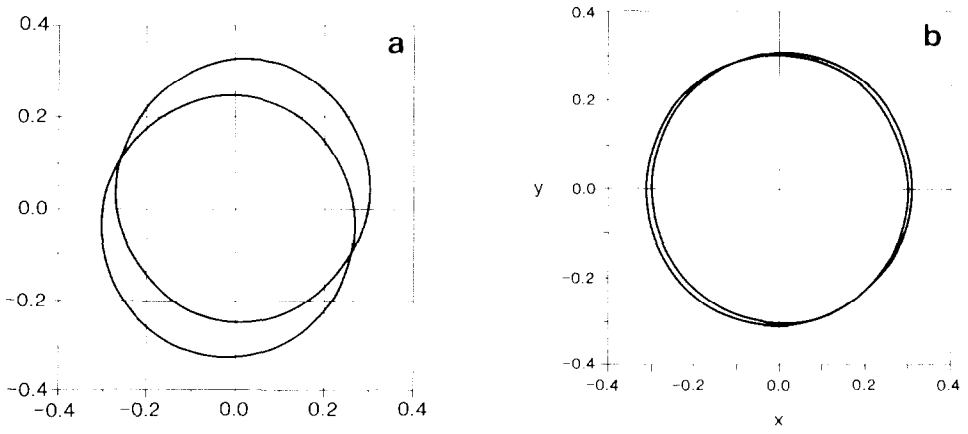

Fig. 4. Orbital motion of two particles initially separated by three cell widths. (a) Low order approximation to Poisson's equation; (b) high order approximation.

9.6. *Tests*

The most powerful test is one of the simplest. Two equal mass particles are placed on the grid and given velocities which would result in a circular orbit if the gravitational force was calculated correctly. This tests density assignment, potential calculation, force calculation and force assignment as the particles move through the grid. Because the density is concentrated at each particle the test places heavy demands on the potential solver. In fig. 4a the orbits are shown when the potential solver users the 7-point formula (Monaghan and Lattanzio [65]). The effect of the grid is to split the orbits so that each orbit is circular, but the orbit of each particle is shifted relative to the other. In fig. 4b the same orbits are computed using the algorithm discussed here. The improvement is clear.

In this test the two particles are only three cell widths apart. The total linear momentum of the system remains, in magnitude, $< 10^{-3}$ of the momentum of either particle. The angular momentum oscillates with an amplitude of 0.25%. These results are already very satisfactory, but they can be expected to be better for systems with a smoother variation of the gravitational force.

10. Relativistic particle methods

10.1. *Special relativistic PIC*

A special relativistic version of the original PIC was developed by Harlow, Amsden and Nix [8] to simulate the high-speed collision between atomic nuclei. A detailed discussion of the results and applications is given by Nix [10].

The differential equations to be solved are conveniently written in terms of quantities observed in the laboratory frame. We take these as

N: the number of nucleons per unit volume.

$\boldsymbol{M}$: the total momentum per unit volume.
E: the total energy per unit volume.
$\boldsymbol{v}$: the velocity.

In terms of n the number of nucleons per unit volume in the local rest frame of a fluid element, and ϵ the total energy per unit volume in the local rest frame,

$$N = \gamma n, \tag{10.1}$$

$$\boldsymbol{M} = \gamma^2(\epsilon + P)\boldsymbol{v}, \tag{10.2}$$

$$E = \gamma^2\epsilon + (\gamma^2 - 1)P, \tag{10.3}$$

$$\gamma = 1/\sqrt{(1 - v^2)}, \tag{10.4}$$

where we have used units such that the speed of light is 1.0. The pressure P is to be obtained from ϵ and n by means of the equation of state for nuclear matter (Harlow et al. [8], Nix [10]).

The equations of motion take the form

$$\frac{\partial N}{\partial t} + \nabla \cdot (\boldsymbol{v}N) = 0, \tag{10.5}$$

$$\frac{\partial M_k}{\partial t} + \frac{\partial}{\partial x_l}(v_l M_k) = -\frac{\partial P}{\partial x_k}, \tag{10.6}$$

$$\frac{\partial E}{\partial t} + \nabla \cdot (\boldsymbol{v}E) = -\nabla \cdot (P\boldsymbol{v}). \tag{10.7}$$

These equations are for non-dissipative motion (for problems involving the high energy collision of nuclei this is a good approximation) and they show that total nucleon number, total momentum and total energy are conserved.

The PIC steps to integrate (10.5)–(10.7) are almost identical to those discussed for non-relativistic flow. the continuity equation (10.5) is automatically satisfied (globally) by assigning ν_k nucleons to particle k and keeping ν_k fixed. The non-convective parts of (10.6) and (10.7), namely

$$\frac{\partial M_k}{\partial t} = -\frac{\partial P}{\partial x_k}, \tag{10.8}$$

$$\frac{\partial E}{\partial t} = -\nabla \cdot (P\boldsymbol{v}), \tag{10.9}$$

are integrated on the grid to give $\tilde{M}_k$ and $\tilde{E}$, the provisional estimates of these quantities. Since ϵ and n are known on the grid, a provisional grid velocity can be determined and interpolated to the particles together with the energy. Harlow et al. [8] use area weighting for this interpolation. The particles are shifted using the interpolated velocity. The momentum and energy are then

interpolated back to the grid. Harlow et al. [8] use nearest grid point assignment for this last interpolation, and before the convective step they assign $\boldsymbol{M}$ and E to each particle according to the fraction of nucleons it has been assigned relative to the total in the cell it occupies. For example, if $\bar{E}_\lambda$, $\bar{\boldsymbol{M}}_\lambda$ and $\bar{N}_\lambda$ denote total energy, momentum and nucleon number of cell λ, and particle k is in this cell, it is assigned energy

$$\Delta E_k = \left(\frac{\nu_k}{\bar{N}_\lambda} \right) \bar{E}_\lambda .$$

The final step is to calculate ϵ and n on the grid from the final $\boldsymbol{M}$ and E.

The entire algorithm could be made more accurate by using better interpolation methods.

10.2. Special relativistic SPH

The problem of simulating the collision of nuclei is ideally suited to SPH because there are no fixed boundaries. The appropriate equations have not appeared in the literature, but they can be derived easily from (10.6) and (10.7) by defining new laboratory frame quantities $\boldsymbol{g}$ (the momentum per nucleon) and e (the energy per nucleon) by

$$N\boldsymbol{g} = \boldsymbol{M} \quad \text{and} \quad Ne = E. \tag{10.10}$$

Substituting (10.10) into (10.6) and (10.7) we find

$$\frac{\partial}{\partial t}\boldsymbol{g} + (\boldsymbol{v}\cdot\nabla)\boldsymbol{g} = -\frac{1}{N}\nabla P, \tag{10.11}$$

$$\frac{\partial e}{\partial t} + (\boldsymbol{v}\cdot\nabla)e = -\frac{1}{N}\nabla\cdot(P\boldsymbol{v}), \tag{10.12}$$

or

$$\frac{\mathrm{d}\boldsymbol{g}}{\mathrm{d}t} = -\frac{1}{N}\nabla P \quad \text{and} \quad \frac{\mathrm{d}e}{\mathrm{d}t} = -\frac{1}{N}\nabla\cdot(P\boldsymbol{v}). \tag{10.13}$$

These equations are almost identical to the non-relativistic equations. Particle's positions change according to

$$\frac{\mathrm{d}\boldsymbol{r}}{\mathrm{d}t} = \boldsymbol{v} = \boldsymbol{g}n/(\gamma[\epsilon + P]). \tag{10.14}$$

The SPH momentum equation for particle i, in the momentum conserving form is

$$\frac{\mathrm{d}\boldsymbol{g}_i}{\mathrm{d}t} = -\sum_j \nu_j \left(\frac{P_j}{N_j^2} + \frac{P_i}{N_i^2} \right) \nabla_i W_{ij}. \tag{10.15}$$

The energy equation can be written in a form that automatically conserves energy by noting, as in section 4, that since

$$\frac{1}{N}\nabla\cdot(P\boldsymbol{v}) = \nabla\cdot\left(\frac{P\boldsymbol{v}}{N}\right) + \frac{P\boldsymbol{v}\cdot\nabla N}{N^2}, \tag{10.16}$$

then the second of (10.13) in SPH form becomes

$$\frac{\mathrm{d}e_i}{\mathrm{d}t} = -\sum_j v_j\left(\frac{P_j\boldsymbol{v}_j}{N_j^2} + \frac{P_i\boldsymbol{v}_i}{N_i^2}\right)\cdot\nabla_i W_{ij}. \tag{10.17}$$

Momentum and energy are conserved provided $W_{ij} = K(\boldsymbol{r}_i - \boldsymbol{r}_j)$ because then

$$\frac{\mathrm{d}}{\mathrm{d}t}\sum_i v_i\boldsymbol{g}_i = 0 \quad \text{and} \quad \frac{\mathrm{d}}{\mathrm{d}t}\sum_i v_i e_i = 0.$$

The SPH equations are more efficient and economical than PIC equations for this problem.

References

[1] F.H. Harlow, J. Assoc. Comput. Mach. 4 (1957) 137.
[2] F.H. Harlow, Meth. Comput. Phys. 3 (1964) 319.
[3] R.L. Bjork, RAND Corporation Report No. P-1662 (1958).
[4] M.W. Evans and F.H. Harlow, J. Aero. Sci. 25 (1958) 269.
[5] F.H. Harlow and D.O. Dickman, LA-2256 (1958).
[6] M.W. Evans, F.H. Harlow and B.D. Meixner, Phys. Fluids 5 (1962) 651.
[7] A.A. Amsden, LA-3466 (1966).
[8] F.H. Harlow, A.A. Amsden and J.R. Nix, J. Comput. Phys. 20 (1976) 119.
[9] A.A. Amsden, F.H. Harlow and J.R. Nix, Phys. Rev. C15 (1977) 2059.
[10] J.R. Nix, Prog. Particle Nucl. Phys. 2 (1979) 237.
[11] J.N. Leboeuf, T. Tajima and J.M. Dawson, J. Comput. Phys. 31 (1979) 379.
[12] F. Brunel, J.N. Leboeuf, T. Tajima and J.M. Dawson, J. Comput. Phys. 43 (1981) 268.
[13] R.A. Gingold and J.J. Monaghan, Monthly Notices Roy. Astron. Soc. 181 (1977) 375.
[14] L. Lucy, Astron. J.
[15] I.J. Schoenberg, Cardinal Spline Interpolation (SIAM, Philadelphia, 1973).
[16] I.J. Schoenberg, Quart. J. Appl. Math. IV (1946) 45.
[17] J.J. Monaghan, SIAM J. Sci. Statist. Comput. 3 (1982) 422.
[18] R.W. Hockney and J.W. Eastwood, Computer Simulation Using Particles (McGraw-Hill, New York, 1981).
[19] D. Wood, Monthly Notices Roy. Astron. Soc. 194 (1981) 201.
[20] R. Courant and D. Hilbert, Methods of Mathematical Physics (Interscience, New York, 1966).
[21] J.C. Lattanzio, J.J. Monaghan, H. Pongracic and M.P. Schwarz, Controlling Penetration, preprint.
[22] E.T. Goodwin, Proc. Camb. Phil. Soc. 45 (1949) 241.
[23] J.J. Monaghan and R.A. Gingold, J. Comput. Phys. 52 (1983) 374.
[24] C. de Boor, A Practical Guide to Splines (Springer Verlag, Berlin, 1978).
[25] N. Niedereiter, Bull. Am. Math. Soc. 84 (1978) 957.
[26] P.J. Davis and P. Rabinowitz, Numerical Integration (Blaisdell, Waltham, 1967).
[27] J.P. Boris and D.L. Book, Meth. Comput. Phys. 16 (1976) 85.

[28] D. Wood, Monthly Notices Roy. Astron. Soc. 199 (1982) 331.
[29] S.M. Miyama, C. Hayashi and S. Narita, Astrophys. J. 279 (1984) 621.
[30] A. Harten, J. Comput. Phys. 49 (1983) 357.
[31] B. Van Leer, J. Comput. Phys. 32 (1979) 101.
[32] J.P. Boris and D.L. Book, J. Comput. Phys. 11 (1973) 38.
[33] R.A. Gingold and J.J. Monaghan, J. Comput. Phys. 46 (1982) 429.
[34] R.A. Gingold and J.J. Monaghan, Monthly Notices Roy. Astron. Soc. 184 (1978) 481.
[35] R.A. Gingold and J.J. Monaghan, Monthly Notices Roy. Astron. Soc. 191 (1980) 897.
[36] R.A. Gingold and J.J. Monaghan, Monthly Notices Roy. Astron. Soc. 188 (1979) 45.
[37] R.A. Gingold and J.J. Monaghan, Monthly Notices Roy. Astron. Soc. 197 (1981) 461.
[38] R.A. Gingold and J.J. Monaghan, Monthly Notices Roy. Astron. Soc. 204 (1983) 715.
[39] J.J. Monaghan and R.A. Gingold, J. Comput. Phys. 52 (1983) 374.
[40] G. Lorimer, PhD Thesis, Monash University, Victoria, Australia.
[41] J.C. Lattanzio, J.J. Monaghan, H. Pongracic and M.P. Schwarz, Cloud–Cloud Collisions, preprint.
[42] P.J. Roache, Computational Fluid Dynamics (Hermosa, Albuquerque, 1975).
[43] W.M. Tscharnuter and K.-H. Winkler, Comput. PhyTs. Commun. 18 (1979) 171.
[44] M.A. HauTsman, Astrophys. J. 245 (1981) 72.
[45] R.B. Larson, J. Comput. Phys. 27 (1978) 397.
[46] J.J. Monaghan and H. Pongracic, IMACS J. Num. Math., in press.
[47] G.A. Sod, J. Comput. Phys. 27 (1978) 1.
[48] B.M. Marder, Math. Comput. 29 (1975) 434.
[49] G.A. Phillips, PhD Thesis, Monash University.
[50] G.A. Phillips and J.J. Monaghan, A Numerical Method for Three-Dimensional Simulations of Collapsing, Isothermal, Magnetic Gas Clouds, Preprint, Monash University.
[51] A.W. Appel, An Investigation of Galaxy Clustering Using an Asymptotically Fast N-body Algorithm, Princeton Univ. USA.
[52] G.D. Smith, Numerical Solution of Partial Differential Equations (Clarendon Press, Oxford, 1975).
[53] R.W. Hockney, Meth. Comput. Phys. 9 (1970) 135.
[54] R.V. Southwell, Relaxation Methods in Theoretical Physics (Clarendon Press, Oxford, 1946).
[55] A. Thom and C.J. Apelt, Field Computation in Engineering and Physics (Van Nostrand, New York, 1961).
[56] A. Brandt, Math. Comput. 31 (1977) 333.
[57] R.D. Richtmyer and K.W. Morton, Difference Methods for Initial Value Problems (Interscience, New York, 1967).
[58] W. Hackbusch, J. Inst. Math. Appl. 26 (1980) 119.
[59] R.A. Nicolaides, Math. Comput. 30 (1979) 933.
[60] J.J. Monaghan and J.C. Lattanzio, A Refined Particle Method for Astrophysical Problems, preprint (1984).
[61] T.D. Butler, I. Henins, F. Jahoda, J. Marshall and R.L. Morse, Phys. Fluids 12 (1969) 1904.
[62] R.A. James, in: Investigating the Universe, ed. F. Kahn (Reidel, Dordrecht, 1981) p. 423.
[63] J.J. Monaghan, Solving Poisson's Equation to 4th Order, preprint (1984).
[64] R.P. Fedorenko, Russ. Math. Surveys 28 (1973) 129.
[65] J.J. Monaghan and J.C. Lattanzio, A Refined Particle method for Astrophysical Problems, preprint (1984).
[66] M. Thompson, PhD Thesis, Monash University, Victoria, Australia (1984).
[67] F.H. Harlow, LA-2301 (1959).