JOURNAL OF COMPUTATIONAL PHYSICS 96, 339–368 (1991)

A Numerical Method for Suspension Flow*

DEBORAH SULSKY

Department of Mathematics and Statistics,
University of New Mexico, Albuquerque, New Mexico 87131

AND

J. U. BRACKBILL

Los Alamos National Laboratory, Los Alamos, New Mexico 87545

Received June 7, 1990; revised December 13, 1990

Peskin's (*J. Comput. Phys.* **25**, 220, 1977) immersed boundary technique is modified to give a new numerical method for studying a fluid with suspended elastic particles. As before, the presence of the suspended particles is transmitted to the fluid through a force density term in the fluid equations. As a result, one set of equations holds in the entire computational domain, eliminating the need to apply boundary conditions on the surface of suspended objects. The new method computes the force density by discretizing the stress–strain constitutive equations for an elastic solid on a grid, using data provided by clusters of Lagrangian points. This approach clearly specifies the material properties of the suspended objects. A simple data structure for the Lagrangian points makes it easy to model suspended solids with arbitrary shape and size. The method is validated by comparing numerical results for elastic vibrations and particle settling in viscous fluids, with theory and analysis. The capability of the method to do a wide range of problems is illustrated by qualitative results for lubrication and cavity flow problems. © 1991 Academic Press, Inc.

1. INTRODUCTION

Flow of a liquid containing suspended, solid particles is a common occurrence in natural and industrial processing environments. Examples are blood flow, transport of slurries, movement of sediments in a river bed, and ceramics and advanced composites manufacturing. Understanding the rheological properties of these materials poses a challenging theoretical and numerical problem, principally because of long-range, many-body hydrodynamic interactions between suspended particles.

Most numerical methods for treating suspensions are restricted to Stokes' flow. The review [1] summarizes use of a collocation method by Weinbaum and his coworkers. They calculate velocity and drag coefficients for a small number of rigid

* The U.S. Government's right to retain a nonexclusive royalty-free license in and to the copyright covering this paper, for governmental purposes, is acknowledged.

0021-9991/91 $3.00
Copyright © 1991 by Academic Press, Inc.
All rights of reproduction in any form reserved.

spheres suspended in a fluid. A truncated expansion in eigenfunctions is made for the velocity in 3D, or the stream funcion in 2D axisymmetric Stokes' flow, with boundary conditions imposed at 4–12 collocation points per sphere. Calculations include static determination of instantaneous velocity and drag on linear chains of spheres, and dynamic calculation of three sedimenting spheres.

Brady's Stokesian dynamics [2–8] is a molecular-dynamics-like approach to simulating the motion of hydrodynamically interacting, rigid particles at zero Reynolds number. Computation of the many-body interactions is achieved through construction of a mobility matrix that relates particle velocities to forces exerted on the fluid by the particles. The inverse matrix, the hydrodynamic resistance matrix, contains approximations to the far-field, many-body interactions. The near-field, lubrication forces are then added as contributions to this resistance matrix. Computation of the many-body interactions using the mobility matrix is an $O(N^3)$ operation, where N is the number of particles. This method has been used successfully to study a wide range of problems, including calculation of sedimentation velocity, and permeability and shear viscosity of periodic cubic arrays of spheres, over a wide range of volume fractions. A similar method is used by Ladd [9–10]; Ladd *et al.* [11] have also used the lattice–gas method to study suspensions.

Ingber [12–13] and Tran-Cong & Phan-Thien [14] use the boundary element method for suspensions of rigid particles in Stokes' flow. Since boundary elements are used to model particle geometry, particles can be any size or shape. This method is capable of providing highly accurate details of flows with suspended particles but is currently impractical for studying concentrated suspensions since it is computationally intensive. A comprehensive review which highlights the interconnections between the collocation method, boundary element method, and Stokesian dynamics is in Weinbaum [15].

A straightforward application of finite difference methods or finite element methods to suspensions is also impractical for concentrated suspensions. A direct approach would involve solution of fluid equations in the irregular region external to the suspended particles and application of boundary conditions on the surface of each particle. Generating a mesh for this region is costly and frequent regridding would be required because of the particle motion.

This paper describes a new numerical method for studying a bounded fluid containing suspended particles. It is based on Peskin's [16] treatment of the heart wall, an immersed moving boundary, in his numerical study of fluid flow in the human heart. The crucial aspect of this method is the replacement of fluid-material interfaces with suitable contributions to a force density term in the fluid equations. With this formulation, internal boundaries are eliminated and a simple, grid-based finite difference scheme is used to solve the fluid equations with an added inhomogeneous forcing term. Fogelson and Peskin [17] adapt this technique to simulate suspended particles. Particles are constructed by linking a small number of points, moving at the local fluid velocity, by elastic springs. The links between these points apply a force to cause the surrounding fluid to move as a rigid body.

We also replace the suspended particles with a force density term in the fluid

equations; however, the force density is computed by discretizing the stress–strain constitutive equations for an elastic solid. This formulation has the advantage of making material properties of the suspended particles clear. It also eliminates the need to have points linked to one another, simplifying the data structure and problem setup. Particles of any size and shape are easily defined in the code. Since computational work involved in computing the force density is linear in the number of particles, this technique has the potential to handle a large number of particles.

The next section describes the model equations, including details of the force calculation. Section 3 discusses the discretization of these equations, Section 4 stability, and Section 5 presents numerical examples. In particular, we test the accuracy and stability of the model for an elastic solid, without the suspending fluid, by comparing numerical results with theory and analysis of elastic vibrations. In Stokes' flow, the relationship between the force on a sphere and its settling velocity along the axis of a cylinder is computed and compared to theory. A similar comparison is made for a cylinder translating parallel to plane walls. The capability of the method to do a wide range of problems is illustrated by a qualitative study of lubrication forces and a cavity flow problem with suspended particles of varying shape. Although these numerical examples illustrate the method for Stokes' flow with suspended elastic solids, this treatment of the suspended particles can be added to any finite difference scheme. A Navier–Stokes solver can be substituted for the Stokes solver in the current implementation; and, in principle, the suspending fluid may be non-Newtonian. In addition, any material can be modeled provided the constitutive equations are known; for example, the formulation can be extended to study plastic deformation.

2. Model Equations

We solve Stokes' flow for an incompressible fluid:

$$0 = -\nabla p + \mu \Delta \mathbf{u} + \mathbf{f}(\mathbf{x}, t) \tag{1}$$

$$0 = \nabla \cdot \mathbf{u}. \tag{2}$$

These equations are to hold at each point $\mathbf{x}$ in the computational domain, including the interior of the suspended particles. The quantity $\mathbf{u}(\mathbf{x}, t)$ is velocity, μ the viscosity, and $p(\mathbf{x}, t)$ the pressure. The inhomogeneous term $\mathbf{f}(\mathbf{x}, t)$ is the sum of internal forces $\mathbf{f}^{\text{int}}(\mathbf{x}, t)$ and external forces $\mathbf{f}^{\text{ext}}(\mathbf{x}, t)$. The external forces are problem dependent and can include gravity, or interparticle forces such as electrical or colloidal forces. The internal forces are those present in an elastic body and act, through the mediation of the fluid, to resist deformation of the suspended bodies. This force density is how the presence of the particles is transmitted to the fluid.

The lth suspended elastic body is constructed from a collection of points labelled $\mathbf{x}_p^l(t)$. These points move at the local fluid velocity but are not constrained to

coincide with computational grid points. The local velocity is defined by averaging the fluid velocity over an appropriate neighborhood of $\mathbf{x}_p^l$,

$$\frac{d\mathbf{x}_p^l}{dt} = \int_V \mathbf{u}(\mathbf{x}, t)\, \mathscr{S}(\mathbf{x} - \mathbf{x}_p^l)\, d\mathbf{x}. \tag{3}$$

$\mathscr{S}$ is a weight function with compact support. The neighborhood is defined by the support of $\mathscr{S}$. If $\mathscr{S}$ is a delta function then the velocity of $\mathbf{x}_p^l$ is exactly the fluid velocity at that point. An average is used to define particle velocity since $\mathbf{u}$ is known numerically only at the vertices of a computational mesh. In the numerical method,

$$\mathscr{S}(\mathbf{x} - \mathbf{x}_p^l) = \sum_v S^{(1)}(\mathbf{x} - \mathbf{x}_p^l)\, \delta(\mathbf{x} - \mathbf{x}_v), \tag{4}$$

where $S^{(1)}$ is bilinear interpolation (Appendix I), and the points $\mathbf{x}_v$ are the vertices of the computational mesh. Using Eq. (4), the integral in Eq. (3) reduces to a sum over the grid points contained in the support of $S^{(1)}$. These points are the vertices of the computational cell that contains $\mathbf{x}_p^l$.

The term $\mathbf{f}^{\text{int}}(\mathbf{x}, t)$ is calculated from the elastic stress tensor

$$\mathbf{f}^{\text{int}}(\mathbf{x}, t) = \nabla \cdot \sigma(\nabla \mathbf{d} + (\nabla \mathbf{d})^{\mathrm{T}}). \tag{5}$$

Here, $\mathbf{d}(\mathbf{x}, t)$ is the displacement field. In incompressible flow $\nabla \cdot \mathbf{d} = 0$. The shear modulus $\sigma(\mathbf{x}, t)$ is a function of position because it is nonzero only within the elastic bodies. The shear modulus is treated as a particle property, that is $\mathbf{x}_p^l$ has a value σ_p^l associated with it. Values of σ at other points are then defined via interpolation

$$\sigma(\mathbf{x}, t) = \frac{1}{n} \sum_{p,l} \sigma_p^l \mathscr{S}(\mathbf{x} - \mathbf{x}_p^l), \tag{6}$$

where $n(\mathbf{x}, t)$ is the number of points p, in the support of $\mathscr{S}$, contributing to the shear modulus at $\mathbf{x}$, counted by

$$n(\mathbf{x}, t) = \sum_{p,l} \mathscr{S}(\mathbf{x} - \mathbf{x}_p^l). \tag{7}$$

From a list of current particle positions $\mathbf{x}_p^l(t)$ and equilibrium positions $\mathbf{x}_{p0}^l$, particle displacements are computed

$$\mathbf{d}_p^l(t) = \mathbf{x}_p^l(t) - \mathbf{x}_{p0}^l. \tag{8}$$

The particle displacements are then interpolated to give the center of mass displacement field,

$$\mathbf{d}(\mathbf{x}, t) = \sum_{p,l} m_p^l \mathbf{d}_p^l(t)\, \mathscr{S}(\mathbf{x} - \mathbf{x}_p^l) \Big/ \sum_{p,l} m_p^l \mathscr{S}(\mathbf{x} - \mathbf{x}_p^l), \tag{9}$$

where m_p^l is the particle mass for the lth body. Derivatives of this field are used to form $\mathbf{f}^{\text{int}}(\mathbf{x}, t)$ using Eq. (5). The next section describes the scheme for discretizing these equations.

3. Discrete Equations

The difference equations described in this section are based on the finite volume approximations originally presented in Brackbill [16] and extended in Appendix II. The method uses a staggered mesh, with velocities and forces set at vertices of the mesh and pressure at cell centers.

Equation (1) is solved by introducing an artificial time variable $\Delta t'$ to march the solutions in time to a steady state, coupled with a projection method to enforce incompressibility,

$$\frac{\mathbf{u}_v^* - \mathbf{u}_v^0}{\Delta t'} = \mu L_v(\mathbf{u}_v^*) + \mathbf{f}_v^0 \tag{10}$$

$$\frac{\mathbf{u}_v^1 - \mathbf{u}_v^*}{\Delta t'} = -\mathbf{G}_v(p_c^1) \tag{11}$$

$$D_c(\mathbf{u}_v^1) = 0. \tag{12}$$

A subscript v denotes a vertex quantity, and c a cell-centered quantity. The operators L_v, $\mathbf{G}_v$, and D_c are discrete Laplacian, gradient, and divergence operators. The superscript 0 indicates quantities evaluated at the beginning of the timestep and the superscript 1 at the end. Velocity values with an asterisk are intermediate values that do not satisfy the incompressibility constraint. Equations (11)–(12) project the intermediate velocity field onto one that is divergence free. These last two equations are solved by taking the divergence of Eq. (11) and making use of Eq. (12), to get obtain an equation for p_c^1 at cell-centers,

$$\frac{1}{\Delta t'} D_c(\mathbf{u}_v^*) = D_c(\mathbf{G}_v(p_c^1)) \equiv L_c(p_c^1); \tag{13}$$

and rewriting Eq. (11) to obtain an equation for $\mathbf{u}_v^1$ at vertices,

$$\mathbf{u}_v^1 = \mathbf{u}_v^* - \Delta t' \, \mathbf{G}_v(p_c^1). \tag{14}$$

The discrete vertex-valued Laplacian L_v appearing in Eq. (10) can also be written as $L_v \equiv D_v(\mathbf{G}_c(\cdot_v))$, i.e., in terms of a vertex-valued divergence and cell-centered gradient. Note that the discrete gradient and divergence of a vertex variable are located at cell centers. Likewise, the discrete gradient and divergence of a cell-centered quantity are defined at the vertices. Formulas for D_c, D_v, $\mathbf{G}_c$, and $\mathbf{G}_v$ are derived in Appendix II. As written, Eqs. (10)–(12) represent one step of an iteration to steady state. Values of $\mathbf{u}^0$, $\mathbf{u}^1$, $\mathbf{u}^*$, and p^1 can be updated, keeping $\mathbf{f}^0$ fixed, and

Eqs. (10)–(12) iterated, but we find in practice that one step suffices. For one step of the iteration, we can use $\Delta t' = \Delta t$, provided Δt satisfies the stability condition derived in the next section.

For the two-dimensional calculations presented in Section 5, a computational cycle involves the solution of three Poisson equations, one for each velocity component and one for the pressure. The discrete Laplace operators are, in general, 9-point operators. The resultant linear system of equations is solved with the preconditioned conjugate gradient method, the preconditioner being incomplete Cholesky decomposition. The implementation of this algorithm exploits the banded structure of the matrices and vectorization on the CRAY [19]. After $\mathbf{u}_v^1$ is determined, the particles are moved in this velocity field using Eqs. (3) and (4),

$$\frac{\mathbf{x}_p^l(t+\Delta t) - \mathbf{x}_p^l(t)}{\Delta t} = \sum_v \mathbf{u}_v^1 S^{(1)}(\mathbf{x}_v - \mathbf{x}_p^l). \tag{15}$$

Particle displacements are then determined from Eq. (8), and displacements on the grid from Eq. (9) with $\mathbf{x} = \mathbf{x}_v$. Using Eq. (5), the body force is

$$\mathbf{f}_v^{\text{int}} = D_v\{\sigma_c[\mathbf{G}_c(\mathbf{d}_v) + \mathbf{G}_c(\mathbf{d}_v)^{\mathrm{T}}]\}, \tag{16}$$

where σ_c is obtained from Eq. (6) with $\mathbf{x} = \mathbf{x}_c$. This completes the computational cycle.

The key to this method is the smoothing of interfaces between solid and fluid, simplifying the imposition of boundary conditions. One set of equations holds in the entire computational domain and the need to resolve interfaces in order to apply boundary conditions is eliminated. Smoothing is accomplished using interpolation as indicated above. Appendix I defines the spline-based interpolation functions used in the code. The numerical examples in Section 5 use a regular square grid, but the discrete operators are formulated more generally to allow for a grid of quadrilateral cells. Linear interpolation $\mathcal{S} = S^{(1)}$ is used with vertex quantities $\mathbf{u}_v$ (15), and $\mathbf{d}_v$ (9), since this is consistent with the quadrilateral cells of the computational mesh. Nearest grid point interpolation, $\mathcal{S} = S^{(0)}$, is used to calculate σ_c in (6) and (7), so that σ_c is approximated to the same order as the other terms in the stress (i.e., $\mathbf{G}_c(\mathbf{d}_v)$ and its transpose).

In Stokes' flow we expect small deformations of the elastic solid; however, displacements may be large since the body can rotate and translate in the fluid. In order to keep the displacement and displacement gradients small in each step, it is generally necessary to use body-fixed coordinates to calculate the displacement. Small displacements allow one to neglect the quadratic terms in the strain and maintain the validity of Eq. (5). The calculation for body-fixed coordinates is easily accommodated by moving the reference configuration along with the body. The reference positions $\mathbf{x}_{p0}^l$ are initially identical with the input particle positions $\mathbf{x}_p^l(0)$. For each l, the points $\mathbf{x}_{p0}^l(t)$ move together as a rigid body, with the same mean translation and rotation as the positions $\mathbf{x}_p^l(t)$. To simplify the notation, in the remainder of this section we drop the superscript l; the following calculation must be done for each l.

The update for the reference positions is separated into two parts,

$$\mathbf{x}_{p0}(t+\Delta t)=\mathbf{x}_{cm}(t+\Delta t)+\mathbf{r}_{p0}(t+\Delta t), \tag{17}$$

where $\mathbf{x}_{cm}$ is the position of the center of mass, and $\mathbf{r}_{p0}=\mathbf{x}_{p0}-\mathbf{x}_{cm}$ is the position relative to the center of mass. The first term on the right-hand side of Eq. (17) is easy to compute; the new center of mass at time $t+\Delta t$ is determined from the new particle positions

$$\mathbf{x}_{cm}(t+\Delta t)=\frac{1}{M}\sum_p m_p\mathbf{x}_p(t+\Delta t), \tag{18}$$

where $M=\sum_p m_p$ is the total mass of the body. The second term on the right-hand side of Eq. (17) is more complicated, the derivation below is for two dimensions. The vectors $\mathbf{r}_{p0}$ only undergo a rigid rotation. If this rotation is measured relative to the time-centered position $\bar{\mathbf{r}}_{p0}=\frac{1}{2}(\mathbf{r}_{p0}(t+\Delta t)+\mathbf{r}_{p0}(t))$, then the change in the length of $\mathbf{r}_{p0}$ is zero, and

$$\mathbf{r}_{p0}(t+\Delta t)=\mathbf{r}_{p0}(t)+\Omega\times\bar{\mathbf{r}}_{p0}, \tag{19}$$

where Ω is a vector directed along the axis of rotation. To obtain an explicit expression for $\mathbf{r}_{p0}(t+\Delta t)$, take the vector product of Eq. (19) with Ω and substitute $\Omega\times\mathbf{r}_{p0}(t+\Delta t)$ back into Eq. (19):

$$\mathbf{r}_{p0}(t+\Delta t)=\frac{(1-(1/4)\,\Omega^2)\,\mathbf{r}_{p0}(t)+\Omega\times\mathbf{r}_{p0}(t)}{1+(1/4)\,\Omega^2}. \tag{20}$$

The update will be complete with the specification of the mean rotation, Ω. The reference configuration rotates at the same mean rate as the particles $\mathbf{x}_p$ relative to the vector $\bar{\mathbf{r}}_p=\frac{1}{2}(\mathbf{r}_p(t+\Delta t)+\mathbf{r}_p(t))$, where $\mathbf{r}_p(t)=\mathbf{x}_p(t)-\mathbf{x}_{cm}(t)$. The mean rotation is defined by the equation

$$\Omega\sum_p m_p(\bar{\mathbf{r}}_p)^2=\sum_p m_p\Omega_p(\bar{\mathbf{r}}_p)^2. \tag{21}$$

The new quantity Ω_p is the rotation of the vector $\mathbf{r}_p(t+\Delta t)$ relative to $\bar{\mathbf{r}}_p$,

$$\Omega_p=\frac{\mathbf{r}_p(t+\Delta t)\times\bar{\mathbf{r}}_p}{(\bar{\mathbf{r}}_p)^2}. \tag{22}$$

Therefore, the mean rotation is

$$\Omega=\frac{\sum_p m_p\mathbf{r}_p(t+\Delta t)\times\bar{\mathbf{r}}_p}{\sum_p m_p(\bar{\mathbf{r}}_p)^2}. \tag{23}$$

With some algebraic manipulation, this last expression can be recast in the computationally more convenient form

$$\Omega = \frac{\sum_p m_p \mathbf{r}_p(t) \times \mathbf{x}_p(t+\Delta t)}{(1/4)\sum_p m_p[\mathbf{x}_p(t+\Delta t)+\mathbf{r}_p(t)]^2 - (1/4)\,\mathbf{x}_{cm}(t+\Delta t)^2\, M}. \tag{24}$$

If the reference positions are not updated each timestep, $\mathbf{d}_p$ in Eq. (8) is the total displacement from equilibrium; with the update, the mean motion has been subtracted and the displacements should be small.

4. Stability

To examine linear stability, consider a one-dimensional model problem based on Eqs. (10), (15), and (8). Freeze the coefficients and assume a spatial variation of the form e^{ikx},

$$\rho \frac{u^1 - u^0}{\Delta t} = -k^2 \sigma d^0 - k^2 \mu u^1 \tag{25}$$

$$\frac{d^1 - d^0}{\Delta t} = u^1. \tag{26}$$

This system of equations can be written

$$\begin{pmatrix} u^1 \\ d^1 \end{pmatrix} = \frac{1}{1+\beta} \begin{pmatrix} 1 & -\alpha \\ \Delta t & 1+\beta-\alpha\,\Delta t \end{pmatrix} \begin{pmatrix} u^0 \\ d^0 \end{pmatrix}, \tag{27}$$

with $\alpha = k^2 \sigma\, \Delta t/\rho$ and $\beta = k^2 \mu\, \Delta t/\rho$. The eigenvalues of the amplification matrix are roots of

$$\lambda^2 - \lambda(2+\beta-\alpha\,\Delta t) + 1 + \beta = 0. \tag{28}$$

We next consider several special cases. The motion of an elastic material alone, without the background fluid, corresponds to $\beta = 0$ and results in a stability condition, $\Delta t \leqslant 2\sqrt{(\rho/\sigma)}/k$. On a grid with spacing Δx, the maximum possible wave number is $k_{\max} = \pi/\Delta x$; therefore, we expect a stability condition $\sqrt{\sigma/\rho}\,\pi\,\Delta t/\Delta x \leqslant 2$. We demonstrate this condition in the next section.

With viscosity present solutions are always damped if $\sigma = 0$. If $\sigma \neq 0$, then $\Delta t = \mu/\sigma$ ($\beta = \alpha\,\Delta t$) yields a stable method for all wave numbers. This condition is also verified in practice.

Nonlinear stability is studied from an energy equation. The equation of motion for an elastic solid alone, without fluid present, is

$$\rho \frac{d\mathbf{u}}{dt} = \mathbf{f}^{\text{int}}. \tag{29}$$

Equation (5) defines $\mathbf{f}^{\text{int}}$. So the kinetic energy $\mathscr{K}$ satisfies

$$\begin{aligned}\frac{d\mathscr{K}}{dt} = \rho\mathbf{u}\cdot\frac{d\mathbf{u}}{dt} &= \mathbf{u}\cdot\mathbf{f}^{\text{int}}\\ &= \mathbf{u}\cdot\nabla\cdot\sigma(\nabla\mathbf{d}+\nabla\mathbf{d}^{\mathrm{T}})\\ &= \nabla\cdot\sigma(\nabla\mathbf{d}+\nabla\mathbf{d}^{\mathrm{T}})\cdot\mathbf{u} - \tfrac{1}{2}\sigma(\nabla\mathbf{d}+\nabla\mathbf{d}^{\mathrm{T}})\cdot(\nabla\mathbf{u}+\nabla\mathbf{u}^{\mathrm{T}})\\ &= \nabla\cdot\sigma(\nabla\mathbf{d}+\nabla\mathbf{d}^{\mathrm{T}})\cdot\mathbf{u} - \frac{d\mathscr{P}}{dt}. \end{aligned} \tag{30}$$

The quantity $\mathscr{P}$ is the elastic free energy, $\mathscr{P} = \sigma\mathbf{e}^2$, where $\mathbf{e}$ is the Cauchy strain tensor, $\mathbf{e} = \frac{1}{2}(\nabla\mathbf{d}+\nabla\mathbf{d}^{\mathrm{T}})$. In deriving the energy equation, we use $\mathbf{u} = d\mathbf{d}/dt$ and commute the time derivative and gradient operation, which is consistent with the neglect of quadratic terms in the strain.

In the absence of work being done by the boundary, Eq. (30) shows that total energy is conserved in the continuum. The overall particle algorithm is dissipative. Dissipation of kinetic energy due to interpolation between particles and the grid is discussed in [20]. There is a similar error that should cause diffusion of strain because $\mathbf{u} \neq d\mathbf{d}/dt$ in the model. The displacements are computed from (9) using the particle displacements (8). Combining Eqs. (9), (8), and (3), keeping $\mathbf{x}^l_{p0}$ fixed, and noting that $\mathscr{S}$ is a Lagrangian invariant, the difference between the velocity and time derivative of the displacement is

$$\begin{aligned}\mathbf{u}(\mathbf{x},t) - \frac{d\mathbf{d}}{dt} = \int_V \mathbf{u}(\mathbf{x}',t)\sum_{p,l} m^l_p(\delta(\mathbf{x}'-\mathbf{x})\\ - \mathscr{S}(\mathbf{x}'-\mathbf{x}^l_p))\,\mathscr{S}(\mathbf{x}-\mathbf{x}^l_p)\Big/\sum_{p,l} m^l_p\mathscr{S}(\mathbf{x}-\mathbf{x}^l_p)\,d\mathbf{x}'. \end{aligned} \tag{31}$$

By analogy with the kinetic energy analysis, we expect this difference to cause dissipation of the elasitic potential energy but we have been unable to prove this analytically. The numerical results in the next section give evidence of dissipation.

In calculations of an elastic solid suspended in a creeping fluid, there is additional viscous dissipation. Using Eqs. (1) and (2), the energy equation in the continuum is

$$\frac{d\mathscr{P}}{dt} = \nabla\cdot\sigma(\nabla\mathbf{d}+\nabla\mathbf{d}^{\mathrm{T}})\cdot\mathbf{u} - \nabla\cdot(\mathbf{u}p) - \Phi, \tag{32}$$

where Φ is the rate of viscous dissipation. In these calculations the extra dissipation due to diffusion of the strain introduced by the particle algorithm is not as important.

5. Numerical Examples

In this section results of simulations are presented. One set of simulations is qualitative, chosen to illustrate features of the numerical method. The second set of simulations provides a quantitative validation of the method. All of the calculations are two-dimensional, using either Cartesian coordinates or axisymmetric, cylindrical coordinates.

5.1. *Deformation of an Elastic Body*

To test the properties of the model for an elastic solid, several calculations are performed without a background fluid. A circular cylinder is deformed and then allowed to relax. The calculated response is examined to test linear stability, nonlinear stability, and accuracy. First, the cylinder is deformed by a uniform compression, the initial deformation $\mathbf{d} = -0.05r\check{\mathbf{r}}$, is shown in Fig. 1. The particles defining the cylinder, and the grid on which Eqs. (29), (5), and (15) are solved, are shown in Fig. 2a. The initial forces acting on the particles are shown in Fig. 3. The subsequent motion is summarized in Fig. 4, where kinetic energy elastic potential energy, and the total energy are plotted. The final particle configuration is shown in Fig. 2b. The final configuration exhibits the same regular distribution of particles as the initial configuration.

Two features of the results are noteworthy. First, the total energy, shown in Fig. 4, decreases in time, as expected from the energy analysis in the last section. (The superimposed oscillation is due to the explicit time advancement algorithm.)

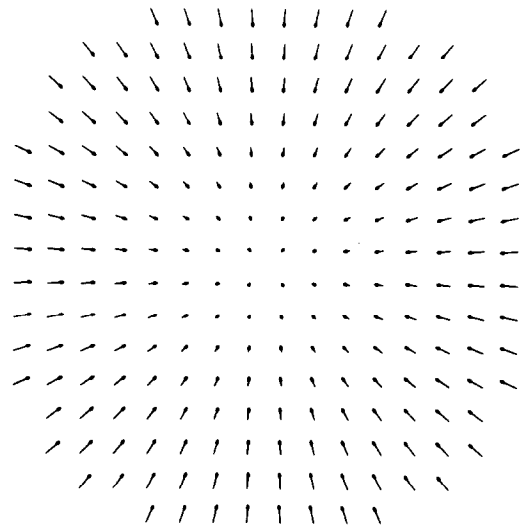

FIG. 1. The particles are displaced from equilibrium to excite an elastic vibration. The displacement vectors point from the reference positions to the displaced particle positions.

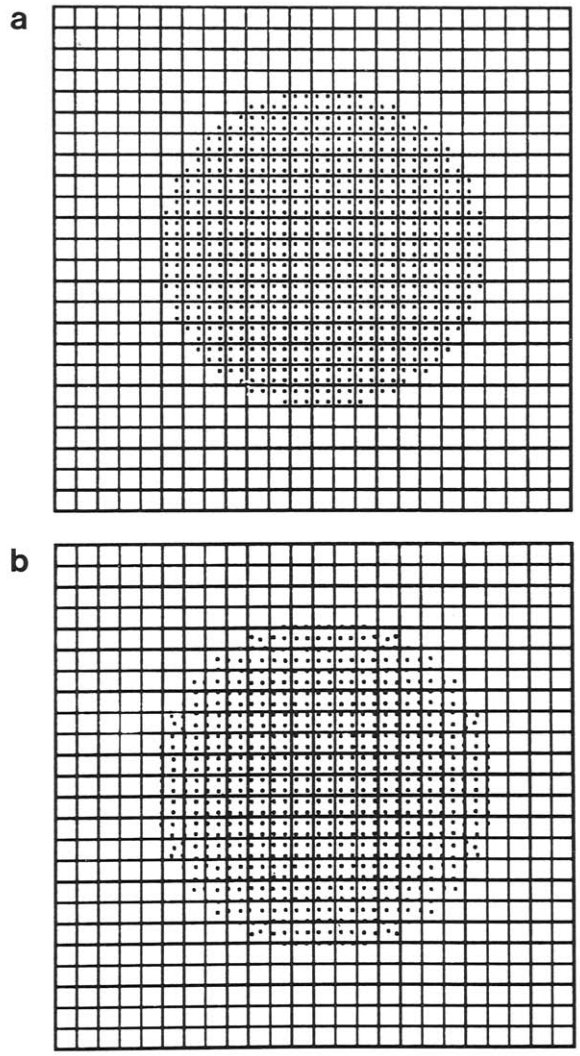

FIG. 2. Particles define the cylinder in Fig. 1, and interactions among particles are computed on a grid. The initial configuration is in (a), and the configuration after five periods is in (b).

Second, the frequency of the oscillation can be compared with an exact solution. For a cylinder with radius R, the density and elasticity are ρ_0 and σ_0 for $r \leqslant R$, and zero for $r > R$. Solving the equation of motion for $\mathbf{d}$ using a Fourier–Bessel expansion of the solution yields the dominant frequency

$$\omega = \sqrt{\frac{2\sigma_0}{\rho_0}} \frac{1.841}{R}. \tag{33}$$

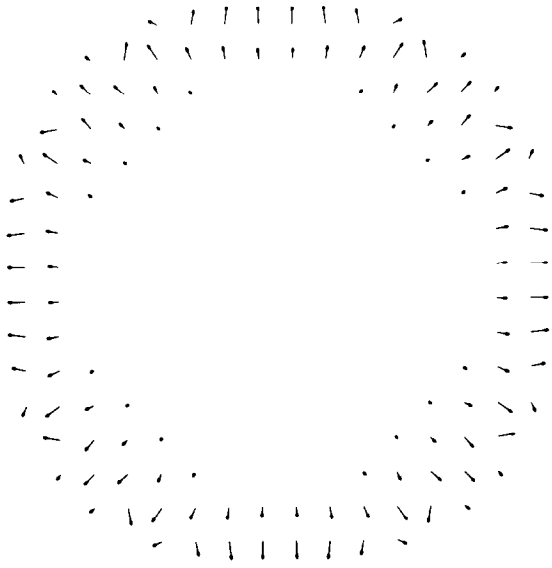

FIG. 3. The initial elastic forces acting on the compressed cylinder act to restore it to its original shape.

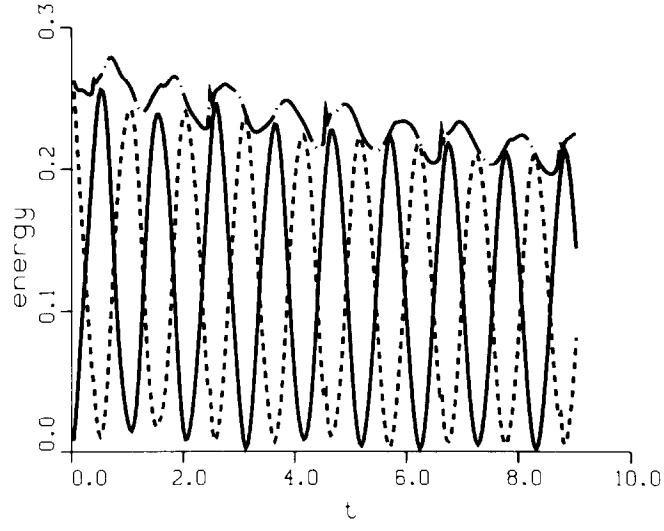

FIG. 4. The kinetic (solid curve), elastic potential (dashed curve), and total (chained curve) energy histories reflect the elastic vibration of the cylinder, and the dissipation due to discretization of the equations.

For the problem shown in Figs. 1–4, $R = 0.6$, $\sigma_0/\rho_0 = 0.5$, and $\omega = 2\pi/2.048$. The computed value is $\omega_{\text{comp}} = 2\pi/2.05$.

Stability is tested next. The timestep used above, $\Delta t = 0.04$, is less than the maximum stable step size predicted by the linear analysis, $\Delta t_{\max} = 0.072$. With $\Delta t = 0.09$, the energy increases with time and the particles become distorted as shown in Fig. 5.

If the deformation is initially incompressible, say

$$\mathbf{d}(\mathbf{x}, 0) = \nabla \times \psi(\mathbf{x})\hat{\mathbf{z}}, \tag{34}$$

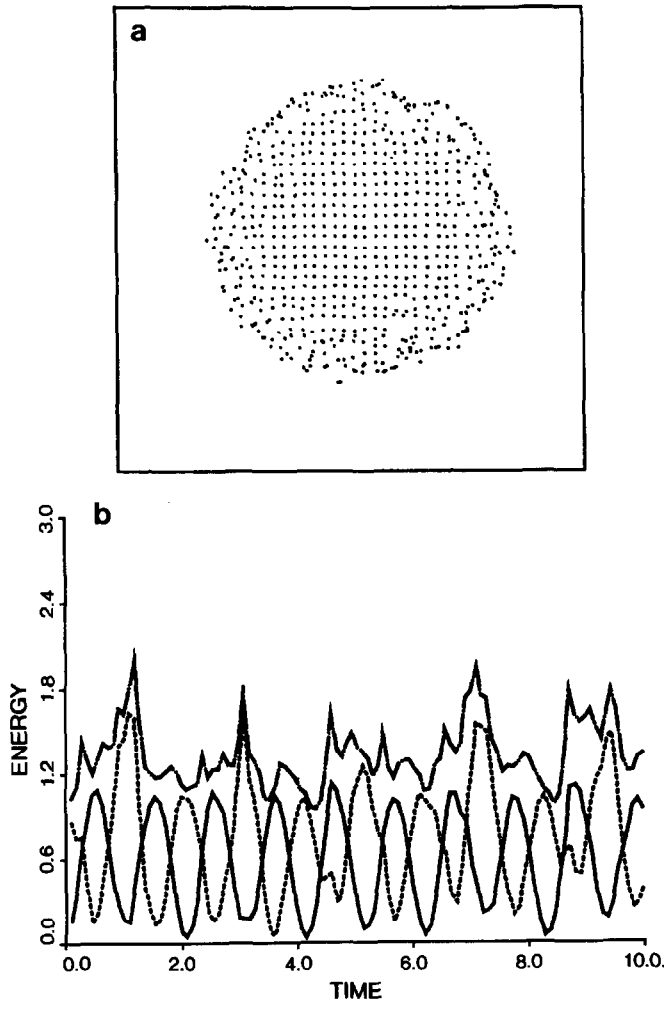

FIG. 5. In an unstable calculation, $\Delta t = 0.09$, short wavelength distortions of the particles and the total energy grow in time.

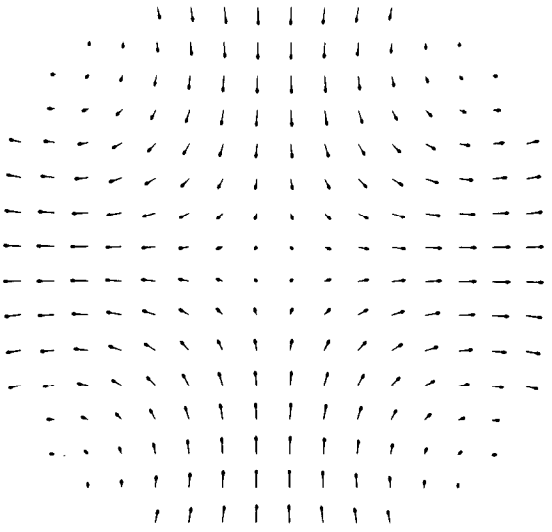

FIG. 6. The particle displacements from equilibrium excite an incompressible, elastic vibration.

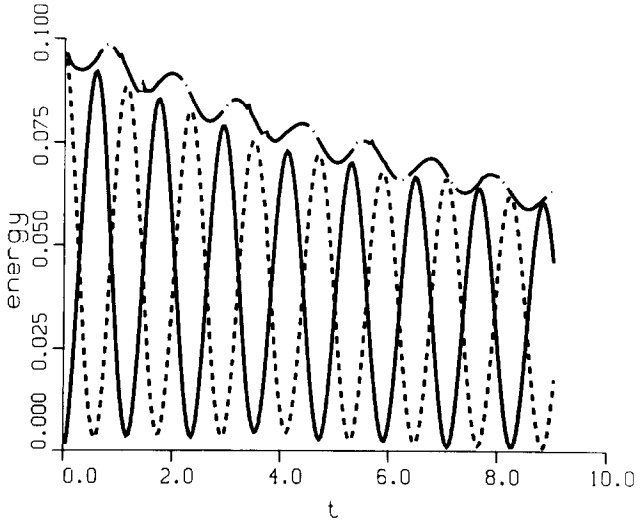

FIG. 7. The total energy (chained curve) for incompressible elastic vibrations decays due to numerical [illegible] slightly lower frequency than in Fig. 4.

with

$$\psi(\mathbf{x}) = A \sin k(x - x_c) \sin k(y - y_c), \tag{35}$$

where (x_c, y_c) is the center of the cylinder, the initial displacement shown in Fig. 6, causes an oscillation to occur at a slightly lower frequency than the compressive case. The initial displacement is small enough that the free boundary does not generate an appreciable compressive component over the duration of the calculation. The oscillation frequency remains constant, and the total energy decreases as in the compressive case, although the frequency of oscillation is slightly different, Fig. 7.

5.2. *Sedimentation*

In axisymmetric, cylindrical coordinates we calculate the motion of a sphere of radius a settling along the axis of a cylinder of radius R, under the influence of gravity, through a quiescent fluid. Figure 8 shows a sample computation for $a/R = 0.3$, $R = 1.0$, and $H = 2.0$, where H is the height of the cylinder. The phase velocity for elastic shear waves is $\sqrt{20}$ in the units used for this computation. Figures 8a and b show particle plots superimposed on streamlines for two different times, $t = 5.0$ and $t = 15$. Figure 8c shows contour lines of the shear modulus, σ, at time $t = 5.0$, illustrating the smooth interface between the fluid and the suspended solid.

Figure 8d is a plot of the position of the center of mass as a function of time. The effect of the finite sized domain is to decrease the settling velocity of the sphere near the top and bottom walls relative to its velocity in the center of the cylinder. At the end of the calculation, the cylinder has essentially come to rest, but a thin layer of fluid still remains between the sphere and the bottom wall. No-slip conditions on solid boundaries for the velocity do not allow fluid to flow through the boundary. Since the velocity field is single valued and particles move in this field, particles also do not penetrate solid boundaries.

For a sphere settling along the axis of an infinite cylinder, Faxen [21] has an asymptotic theory for small a/R that predicts the wall correction factor, K. The wall correction factor is the ratio of the drag on the sphere in the cylinder to the drag on a sphere moving with the same velocity in an unbounded fluid. Numerically, the wall correction factor is computed by applying a constant force to the sphere and calculating its settling velocity, the velocity of its center of mass. If the velocity is measured in the center of a cylinder with finite height, the effect of the top and bottom boundaries is mitigated and the computed results should agree with Faxen's theory. Table I compares computed values of the wall correction factor and Faxen's theoretically predicted values, for different sized spheres. The error is smallest for small a/R and is worst for the largest value of a/R; however, the asymptotic theory breaks down as a/R increases. The error is also smaller when the sphere is resolved better on the grid, measured by $a/\Delta x$. Using the boundary element method, Ingber [13] reports errors increasing in magnitude with a/R, up to a -5.4% error with $a/R = 0.5$.

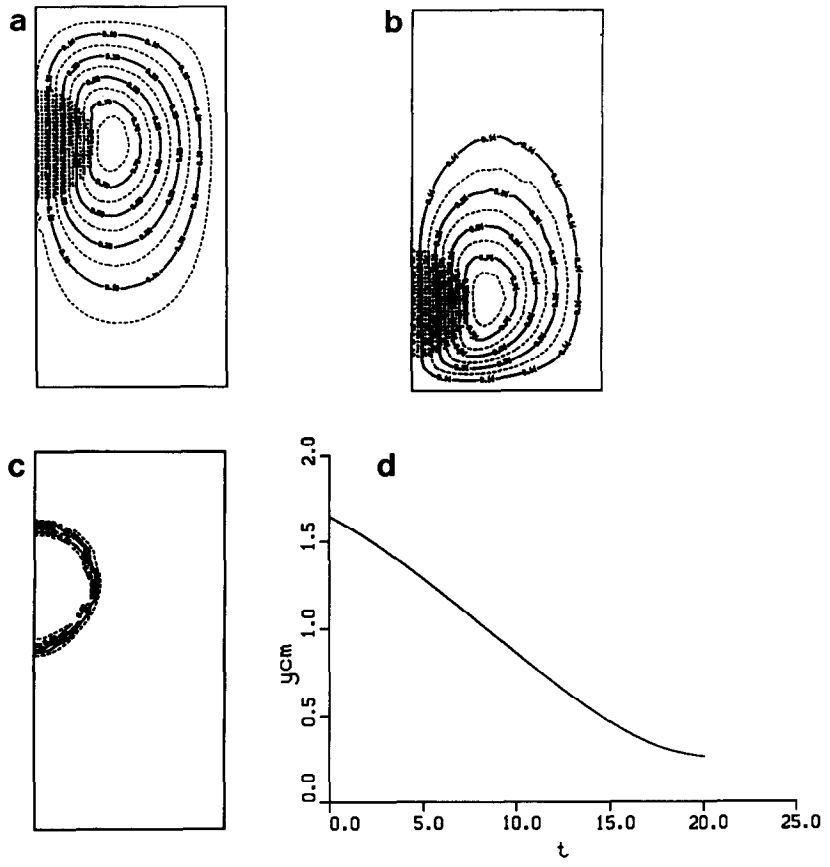

FIG. 8. The particles defining a sphere settling along an axis of symmetry in a cylinder of height 2.0 are shown at $t = 5.0$ (a) and $t = 15.0$ (b). The corresponding straemlines are superimposed. In (c) contours of the shear modulus σ show the smooth interface between the fluid and solid. The center of mass position, shown in (d), reflects the decrease in settling velocity due to the ends of the cylinder.

TABLE I

Settling Sphere on a 51×101 Mesh, $R = 2.0$, with 9 Particles/Cell

a/R	$a/\Delta x$	Number of particles	K theory	K computed	% error
0.05	2.5	86	1.12	1.17	4.4
0.10	5.0	358	1.26	1.29	2.2
0.15	7.5	788	1.45	1.53	5.3
0.20	10.0	1414	1.68	1.71	1.8
0.25	12.5	2202	1.98	2.02	2.0
0.30	15.0	3188	2.37	2.37	—
0.40	20.0	5652	3.59	3.49	−2.9
0.50	25.0	8846	5.92	5.16	−12.9

TABLE II

Settling Sphere, Convergence Study with $a/R = 0.2$, $R = 2.0$, and 9 Particles/Cell

$a/\Delta x$	Mesh size	Number of particles	K theory	K computed	% error
10.0	51×101	1414	1.68	1.71	1.8
5.0	26×51	358	1.68	1.73	2.7
2.5	13×26	90	1.68	1.87	11.4

In our model, the size of the suspended sphere cannot be determined precisely, the error is on the order of the size of a computational cell. This indeterminancy arises even though particles are input to fill a region with a given radius. The influence of the particles is felt in a larger region since the support of the interpolation function is a computational cell. The input radius is used to make the comparison with theory. As the mesh is refined, the surface of the sphere is located more precisely, consistent with the use of a finite difference method. Table II shows the computed wall correction factor as the mesh is refined, $a/\Delta x$ is increased with $a/R = 0.2$ fixed. Several computational cells must be contained within the sphere to compute accurately its settling velocity. In contrast, Table III indicates that the results are not sensitive to the number of particles used to represent the sphere.

Discrepancies between the theory and numerical calculations may be due to sources other than numerical truncation error. These include the fact that the theory pertains to an infinite cylinder and the computations are made on a finite domain and also that the sphere is not completely rigid. As a test, doubling the height of the cylinder in one case ($a/R = 0.3$) does not change the settling velocity. Since the sphere does not deform appreciably while it settles, this probably has little impact on the settling velocity.

Faxen [21] also has an asymptotic formula for the force per unit length on an infinite cylinder translating parallel to the channel formed by two infinite plates. The velocity of the cylinder is directed perpendicular to its axis. This theory is compared to calculations in Cartesian coordinates using a channel width $2L = 2.0$ and height $H = 6.0$, for cylinders with varying radius. A constant force F is applied to the cylinder, and the velocity of its center of mass, U, is computed. The dimensionless force per unit length $F/\mu U$ is compared to Faxen's theory in Table IV.

TABLE III

Variation of Settling Velocity of a Sphere with the Number of Particles, on a 26×51 Grid and $R = 2.0$

a/R	$a/\Delta x$	Number of particles	Particles per cell	K theory	K computed	% error
0.2	5	358	9	1.68	1.73	2.7
0.2	5	158	4	1.68	1.73	2.7
0.2	5	40	1	1.68	1.60	−4.7

TABLE IV

Force on a Settling Cylinder Computed on a 26×76 Mesh with 9 Particles/Cell

a/L	$a/\Delta x$	Number of particles	Faxen	$F/\mu U$	% error	Dvinsky and Popel
0.10	2.5	90	8.95	9.11	1.8	—
0.20	5.0	349	16.53	16.96	2.6	17.47
0.30	7.5	797	29.27	29.03	−0.8	30.22
0.40	10.0	1410	54.12	48.13	−10.5	53.72

Errors are small for small a/L, where the asymptotic theory applies best. The decrease in error with increased resolution of the cylinder (increased $a/\Delta x$) is again apparent. Dvinsky and Popel [22] calculate the force on a cylinder using curvilinear, boundary-conforming coordinates. Their results are also shown in Table IV for comparison.

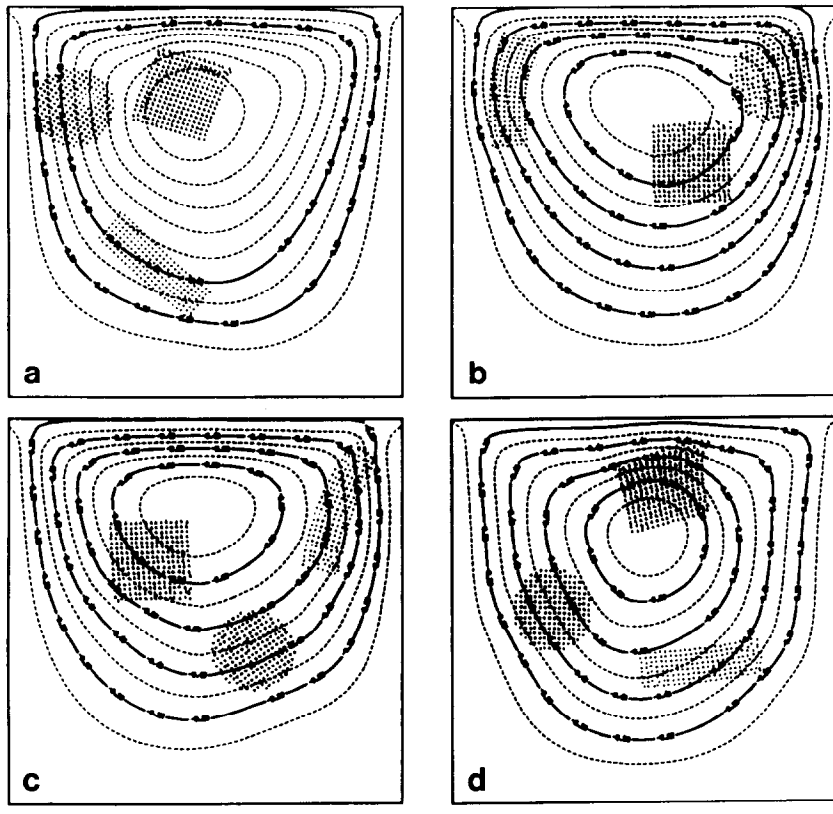

FIG. 9. Elastic objects of various shapes circulate in a driven cavity flow. They are shown in (a)–(d) with streamlines superimposed, after 2.5, 5.0, 7.5, and 10.0 fluid transit times across the top boundary. The objects deform in the high shear region near the top boundary and restore in the low shear flow in the center. The interaction of the immersed objects with the fluid alters the flow.

5.3. *Cavity Flow*

The numerical method developed here can be used to study rheological properties of suspensions made up of particles having different material properties, shapes, and sizes. As an example, Fig. 9 shows a driven cavity flow, the top boundary moving with unit velocity. The calculation is made in a square domain, with no-slip boundary conditions on all sides, using Cartesian coordinates. The phase velocity for elastic shear waves is $\sqrt{20}$. The cavity contains suspended objects with varying shape. The initial shapes are easily defined by filling regions with particles,

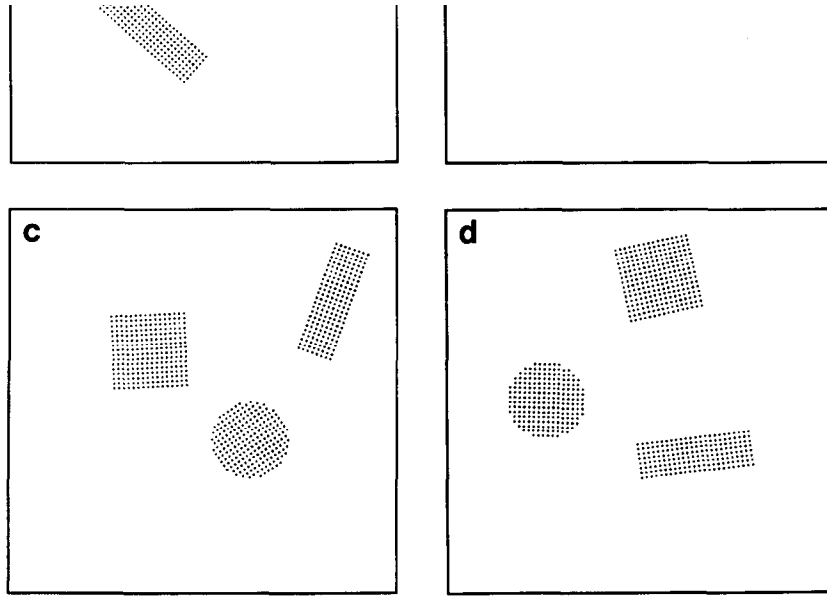

FIG. 10. The reference configurations in (a)–(d) corresponding to the simulation shown in Fig. 9. The reference configurations translate and rotate at the same mean rate as the suspended solids. The difference between the particle positions and the reference configuration is used to compute the elastic restoring force.

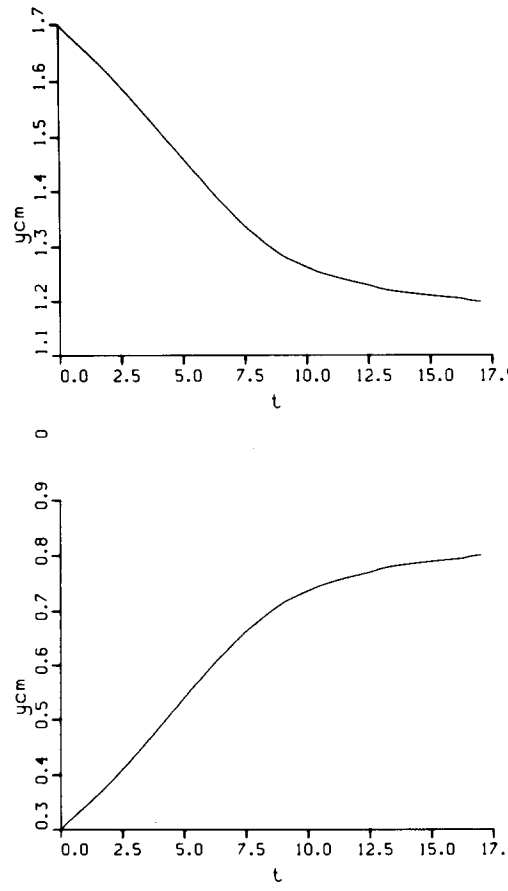

FIG. 12. Center of mass trajectories for the cylinders in the simulation shown in Fig. 11 are plotted. The decrease in velocity as the cylinders approach is in qualitative agreement with lubrication theory.

cylinders have radius 0.2 and are in a square box with side length 2.0. The motion is due to a constant force applied to each cylinder. Since the velocity field is single valued, and the particles move in this field, particles do not interpenetrate. The reference configurations (not shown) appear to touch at the end of the calculation. In contrast, the actual elastic solids have started to deform and there is still a thin layer of fluid between them. Figure 12 shows the trajectories of the center of mass of each object. The decrease in velocity as the cylinders approach is in qualitative agreement with the decrease in gap width predicted by lubrication theory.

6. DISCUSSION

An extension of Peskin's immersed boundary technique [16] is used to develop model equations for a fluid containing suspended elastic particles. The presence of

the suspended particles is transmitted to the fluid by adding an appropriately constructed inhomogeneous forcing term to the fluid equations. As a result, one set of equations holds in the entire computational domain where they are solved using a grid-based finite difference method. This formulation gives up exact knowledge of the surface location to reduce the cost of exactly applying the boundary conditions on a suspended body, a trade-off consistent with the use of a finite difference method.

The extra inhomogeneous forcing term is computed using Lagrangian fluid particles to track the position of the suspended solid. Particle displacements are easy to compute, and from them a displacement field is easily calculated by interpolating the particle data to a computational grid. The added force density term is computed from this displacement field, using the stress–strain constitutive equations for an elastic solid. The computational work involved in computing the force density is linear in the number of particles. In two dimensions, the computation time per particle, per timestep is 33 μs on a CRAY Y-MP.

The validity of the model is tested by studying the elastic equations without a background fluid, and by inserting the force calculation into a fluid code that solves Stokes' equations in two dimensions. A favorable comparison with theory is made for the elastic response to deformation of a cylinder and for the settling velocity of a particle in a viscous fluid. The capability of the method to do a wide range of problems is illustrated by a qualitative study of lubrication and cavity flow problems.

Two shortcomings of the present formulation need to be addressed. The numerical results in the last section indicate that several computational zones must be contained within each suspended object for accuracy. Improvements in both accuracy and efficiency can be realized by using an adaptive grid [23]. Increasing the grid resolution only near the suspended particles may allow sufficient savings to study highly concentrated suspensions. The general formulation of the discrete operators, in terms of a table of geometric coefficients as presented in Appendix II, sets up the future exploration of this area.

In addition, the current formulation evaluates the force density term explicitly. As in Peskin [16], we observe this explicit evaluation is unstable for large timesteps. From a linear stability analysis, and in practice, a timestep of $\Delta t = \mu/\sigma$ yields a stable scheme. An implicit scheme would eliminate this restriction. Alternatively, Peskin's [16] semi-implicit method may be sufficient to alleviate the problem and allow modelling of rigid particles.

Most problems in suspension rheology are inherently three-dimensional. The success of the two-dimensional tests presented here, shows the promise of this technique for studying fully three-dimensional phenomena. Coupled with an adaptive grid and an implicit evaluation of the force density, this method has the potential to handle concentrated suspensions.

APPENDIX I

This appendix defines the interpolation rules used in the numerical code, CLUSTER. Even though the numerical examples presented in Section 5 use a regular square grid, the numerical code is implemented to allow for a general mesh of convex quadrilateral cells, as in Brackbill and Ruppel [24]. Let $\mathbf{x}_{i,j}$, $\mathbf{x}_{i+1,j}$, $\mathbf{x}_{i+1,j+1}$, $\mathbf{x}_{i,j+1}$ denote the vertices of computational cell (i,j). If we introduce natural coordinates (ξ_1, ξ_2) which assume integer values at the vertices, then, using bilinear interpolation, any point $\mathbf{x}$ in the cell is given by

$$\mathbf{x} = \xi_1'[(1-\xi_2')\,\mathbf{x}_{i+1,j} + \xi_2'\mathbf{x}_{i+1,j+1}] + (1-\xi_1')[(1-\xi_2')\,\mathbf{x}_{i,j} + \xi_2'\mathbf{x}_{i,j+1}] \quad \text{(I.1)}$$

with $\xi_1' = \xi_1 - i$ and $\xi_2' = \xi_2 - j$ $(0 \leqslant \xi_1', \xi_2' \leqslant 1)$. This mapping from natural to physical coordinates can also be written

$$\mathbf{x} = \sum_v \mathbf{x}_v s^{(1)}(\xi_1 - i, \xi_2 - j) \quad \text{(I.2)}$$

$$s^{(1)}(\xi_1', \xi_2') = \begin{cases} (1-|\xi_1'|)(1-|\xi_2'|), & 0 \leqslant |\xi_1'|, |\xi_2'| \leqslant 1 \\ 0, & \text{otherwise} \end{cases}.$$

The function $s^{(1)}$ is positive, continuous, has compact support, and has range $[0, 1]$. It is normalized so that integrating over the domain gives

$$\int_D s^{(1)} d^2\xi = 1 \qquad \text{and} \qquad \sum_{i,j} s^{(1)}(\xi_1 - i, \xi_2 - j) = 1. \quad \text{(I.3)}$$

A corresponding interpolation function in physical coordinates is

$$S^{(1)}(\mathbf{x}(\xi_1, \xi_2)) = s^{(1)}(\xi_1 - i, \xi_2 - j), \quad \text{(I.4)}$$

with $\int_V S^{(1)} J^{-1} = 1$, where $J = \partial(x, y)/\partial(\xi_1, \xi_2)$ and V is the physical domain. On a square lattice,

$$S^{(1)}(x, y) = \begin{cases} (1-|x|/h)(1-|y|/h), & 0 \leqslant |x|, |y| \leqslant h \\ 0, & \text{otherwise} \end{cases},$$

and $J = h^2$. In general, the support of $S^{(1)}(\mathbf{x})$ depends on $\mathbf{x}$, but for a square lattice the support is h, the grid spacing.

Bilinear interpolation is used to project all particle data onto the computational grid, except the shear modulus, where nearest grid point interpolation is used. When applied to a cell-centered quantity, this interpolation rule projects a particle property onto the cell containing the particle. The logical coordinates of a cell are $(i+\frac{1}{2}, j+\frac{1}{2})$, so in logical variables

$$s^{(0)}(\xi_1 - i - \tfrac{1}{2}, \xi_2 - j - \tfrac{1}{2}) = \begin{cases} 1, & 0 \leqslant |\xi_1 - i - \frac{1}{2}|, |\xi_2 - j - \frac{1}{2}| < \frac{1}{2} \\ 0, & \text{otherwise} \end{cases}.$$

The corresponding function $S^{(0)}$ in physical variables is defined in the same way as bilinear interpolation.

The same interpolation is used for axisymmetric calculations in cylindrical coordinates, where $\mathbf{x}$ has components (r, z) instead of (x, y).

Appendix II

Formulas for discrete approximations to derivatives in the code, CLUSTER, are derived in this appendix. Since the derivation applies equally well to two- or three-dimensional grids, we present it for three dimensions. Finite volume approximations are used on a staggered grid. Velocity, displacement, and forces are associated with vertices of cells; and pressure and elasticity are stored at cell centers. In general, the grid is made up of convex cells that are the image of a unit cube under the obvious generalizaion of the map described in Appendix I. This formulation allows the flexibility of using a fixed grid, a Lagrangian grid whose vertices move with the fluid, or an adaptive grid that moves in a prescribed manner. The advantage of this formulation on a rectilinear grid is the ease with which any formula can be used to solve problems in Cartesian and cylindrical coordinates, the switch only involves generating the appropriate table of coefficients.

A derivative of a vertex variable results in a cell-centered quantity and is calculated as an average over the cell. For a velocity component u_α, the average of the derivative with respect to the coordinate x_β is

$$\left\langle \frac{\partial u_\alpha}{\partial x_\beta} \right\rangle_c \equiv \frac{1}{V_c} \int_{V_c} \frac{\partial u_\alpha}{\partial x_\beta} \, dV. \tag{II.1}$$

The cell volume is computed from

$$V_c = \int_{V_c} dV = \int_C J d^3\xi, \tag{II.2}$$

where J is the Jacobian of the map from logical to physical coordinates, and C is the unit cube. This map is defined in Appendix I for two dimensions and is easily extended to three.

To evaluate Eq. (II.1) in terms of data on the computational grid, define $\mathbf{u}$ throughout the cell from its values at the vertices

$$\mathbf{u}(\mathbf{x}, t) = \sum_v \mathbf{u}_v S^{(1)}(\mathbf{x} - \mathbf{x}_v). \tag{II.3}$$

The sum is over all vertices, but only those vertices in the support of $S^{(1)}$ make a non-zero contribution; these are the vertices of the cell containing $\mathbf{x}$. Now Eq. (II.1) becomes

$$\left\langle \frac{\partial u_\alpha}{\partial x_\beta} \right\rangle_c = \frac{1}{V_c} \int_{V_c} \sum_v (u_\alpha)_v \frac{\partial}{\partial x_\beta} S^{(1)}(\mathbf{x} - \mathbf{x}_v)\, dV$$

$$= \sum_v (u_\alpha)_v\, c^{cv}_{x_\beta} / V_c, \tag{II.4}$$

where

$$c^{cv}_{x_\beta} = \int_{V_c} \frac{\partial}{\partial x_\beta} S^{(1)}(\mathbf{x} - \mathbf{x}_v)\, dV. \tag{II.5}$$

Derivatives of the displacement field are defined analogously.

It is useful to form a vector of geometric coefficients $\mathbf{c}^{cv} = (c^{cv}_{x_\beta})$. One can now combine derivatives to form a discrete divergence operator, acting on vertex quantities and defined at a cell center,

$$V_c D_c(\mathbf{u}_v) = \int_{V_c} \nabla \cdot \mathbf{u}\, dV = \sum_v \mathbf{u}_v \cdot \mathbf{c}^{cv}. \tag{II.6}$$

In the same way, derivatives are combined to form a discrete gradient operator defined at a cell center,

$$V_c \mathbf{G}_c(u_v) = \int_{V_c} \nabla \mathbf{u}\, dV = \sum_v \mathbf{c}^{cv} u_v. \tag{II.7}$$

Note that from the expression for cell volume (II.2) it can be shown that $\mathbf{c}^{cv} = \partial V_c / \partial \mathbf{x}_v$. So, on a Lagrangian grid, the divergence satisfies the continuity equation

$$V_c D_c(\mathbf{u}_v) = \sum_v \frac{\partial V_c}{\partial \mathbf{x}_v} \cdot \mathbf{u}_v = \frac{dV_c}{dt}. \tag{II.8}$$

Formulas for the divergence and gradient, acting on cell-centered data and located at vertices, are now constructed to satisfy a discrete form of the divergence theorem

$$\int_{\delta\Omega} \phi \mathbf{w} \cdot \mathbf{n}\, dS = \int_\Omega \nabla \cdot (\phi \mathbf{w})\, dV = \int_\Omega \mathbf{w} \cdot \nabla \phi + \int_\Omega \phi \nabla \cdot \mathbf{w}\, dV. \tag{II.9}$$

This property is essential for the projection method described in Section 3. For example, to form the gradient of the pressure, which is stored at cell centers, apply Eq. (II.9) with $\phi = p$ and $\mathbf{w} = \mathbf{u}$ and let Ω be a union of computational cells. An approximation for the last term in Eq. (II.9) is already defined

$$\sum_{c \in \Omega} p_c D_c(\mathbf{u}_v)\, V_c = \sum_{c \in \Omega} p_c \sum_v \mathbf{c}^{cv} \cdot \mathbf{u}_v. \tag{II.10}$$

At this point, postulate a form for the gradient at a vertex analogous to Eq. (II.7),

$$V_v \mathbf{G}_v(p_c) = \sum_c \tilde{\mathbf{c}}^{cv} p_c, \tag{II.11}$$

for some coefficients $\tilde{\mathbf{c}}^{cv}$. The quantity V_v is a control volume centered at a vertex $\mathbf{x}_v$, $V_v = \frac{1}{8}\sum_c V_c$. The sum is over the cells that have $\mathbf{x}_v$ as a vertex. Combining Eqs. (II.9)–(II.11) gives

$$\int_{\delta\Omega} p\mathbf{u} \cdot \mathbf{n}\, dS \approx \sum_{v \in \Omega} \mathbf{u}_v \cdot \sum_c (\tilde{\mathbf{c}}^{cv} + \mathbf{c}^{cv})\, p_c. \tag{II.12}$$

The left-hand side of Eq. (II.12) is a surface integral, so the right-hand side should only involve vertices on the boundary of Ω. Since $\mathbf{u}$ and p are arbitrary, for an interior cell c we need $\tilde{\mathbf{c}}^{cv} = -\mathbf{c}^{cv}$. We now have the required formula,

$$V_v \mathbf{G}_v(p_c) = -\sum_c \mathbf{c}^{cv} p_c. \tag{II.13}$$

To obtain the formula for the divergence at a vertex, the process is similar. Start with Eq. (II.9), but now let $\mathbf{w}$ be defined at cell centers, let ϕ be a vertex quantity, and let Ω be a union of vertex-centered control volumes. Use Eq. (II.7) to define an approximation to the first term on the right-hand side of Eq. (II.9), and proceed as above to obtain

$$V_v D_v(\mathbf{w}_c) = -\sum_c \mathbf{c}^{cv} \cdot \mathbf{w}_c. \tag{II.14}$$

The discrete version of the Laplacian is defined by combining the divergence and gradient operators. There are two forms of the Laplacian, one operating on vertex data and one on cell data,

$$V_v L_v(u_v) = V_v D_v(\mathbf{G}_c(u_v)) = -\sum_c \mathbf{c}^{cv} \cdot \sum_{v'} u_{v'} \mathbf{c}^{cv'}/V_c \tag{II.15}$$

$$V_c L_c(p_c) = V_c D_c(\mathbf{G}_v(p_c)) = -\sum_v \mathbf{c}^{cv} \cdot \sum_{c'} p_{c'} \mathbf{c}^{c'v}/V_v. \tag{II.16}$$

The linear systems of equations that arise from these discretizations of the Laplacian, when multiplied by a volume, are symmetric and negative definite. On an arbitrary quadrilateral mesh the discrete Laplacians have a 9-point stencil.

All derivatives are now defined in terms of one set of geometric coefficients, $\mathbf{c}^{cv}$. The geometric coefficients are easily computed from the definition of $S^{(1)}$ (Brackbill [18]). For each cell, we store a value of the vector $\mathbf{c}^{cv}$ for each of the vertices of the cell. To illustrate these formulas, Table V gives the geometric coefficients for a square lattice in two dimensions. The discretization reduces to the box scheme for the gradient and divergence operators. The Laplacian on a square grid has a

TABLE V

Geometric Coefficients for a Square Grid in Two Dimensions

v	c_x^{cv}	c_y^{cv}
1	$h/2$	$-h/2$
2	$h/2$	$h/2$
3	$-h/2$	$h/2$
4	$-h/2$	$-h/2$

Note. For cell c, the vertices are numbered counterclockwise starting from the lower right. On a square grid, $V_c = V_v = h^2$.

5-point stencil; the central point is coupled to the four corners. Properties of the coefficients are discussed in Brackbill [25].

Some of the computations in Section 5 are made in axisymmetric cylindrical coordinates. We end the section with a description of the discretization used in this case. The vector $\mathbf{x}$ now has components (r, z) and control volumes are solids of revolution formed by rotating a quadrilateral cell about the z-axis. The quadrilateral cross-section is the image of a unit square in logical space, under the map defined in Appendix I. The volume of a cell (per radian) is now

$$V_c = \int_{V_c} dV = \int_C Jr(\xi_1, \xi_2)\, d^2\xi, \tag{II.17}$$

where $J = \partial(r, z)/\partial(\xi_1, \xi_2)$ and C is the unit square.

For a quantity ψ, located at vertices of the mesh, finite volume approximations to derivatives are defined as before,

$$\begin{aligned} V_c \left\langle \frac{\partial \psi}{\partial r} \right\rangle_c &= \int_{V_c} \frac{\partial \psi}{\partial r}\, dV = \int_{V_c} \frac{\partial \psi}{\partial r}\, r\, dr\, dz \\ &= \int_{V_c} \sum_v \psi_v \frac{\partial}{\partial r} S^{(1)}(\mathbf{x} - \mathbf{x}_v)\, r\, dr\, dz \\ &\equiv \sum_v \psi_v c_{r_1}^{cv} \end{aligned} \tag{II.18}$$

and

$$\begin{aligned} V_c \left\langle \frac{\partial \psi}{\partial z} \right\rangle_c &= \int_{V_c} \frac{\partial \psi}{\partial z}\, r\, dr\, dz = \int_{V_c} \sum_v \psi_v \frac{\partial}{\partial z} S^{(1)}(\mathbf{x} - \mathbf{x}_v)\, r\, dr\, dz \\ &\equiv \sum_v \psi_v c_z^{cv}. \end{aligned} \tag{II.19}$$

In cylindrical coordinates, we also need

$$V_c \left\langle \frac{\psi}{r} \right\rangle_c = \int_{V_c} \psi \, dr \, dz = \int_{V_c} \sum_v \psi_v S^{(1)}(\mathbf{x} - \mathbf{x}_v) \, dr \, dz$$

$$\equiv \sum_v \psi_v c_{r_2}^{cv}. \tag{II.20}$$

The geometric coefficients are

$$c_{r_1}^{cv} = \int_{V_c} \frac{\partial}{\partial r} S^{(1)}(\mathbf{x} - \mathbf{x}_v) \, r \, dr \, dz, \tag{II.21}$$

$$c_{r_2}^{cv} = \int_{V_c} S^{(1)}(\mathbf{x} - \mathbf{x}_v) \, dr \, dz, \tag{II.22}$$

$$c_z^{cv} = \int_{V_c} \frac{\partial}{\partial z} S^{(1)}(\mathbf{x} - \mathbf{x}_v) \, r \, dr \, dz. \tag{II.23}$$

The coefficient $c_{r_2}^{cv}$ results from the variation of the basis vectors with position.

It is notationally convenient to form two vectors from these geometric coefficients, $\mathbf{c}_1^{cv} = (c_{r_1}^{cv}, 0, c_z^{cv})$ and $\mathbf{c}_2^{cv} = (c_{r_1}^{cv} + c_{r_2}^{cv}, 0, c_z^{cv})$. Since

$$\nabla \cdot \mathbf{u} = \frac{1}{r} \frac{\partial ru}{\partial r} + \frac{\partial v}{\partial z} \qquad \text{and} \qquad \nabla \psi = \left(\frac{\partial \psi}{\partial r}, 0, \frac{\partial \psi}{\partial z} \right), \tag{II.24}$$

discrete divergence and gradient operators at cell centers are written as

$$V_c D_c(\mathbf{u}_v) = \sum_v \mathbf{c}_2^{cv} \cdot \mathbf{u}_v \tag{II.25}$$

$$V_c \mathbf{G}_c(\psi_v) = \sum_v \mathbf{c}_1^{cv} \psi_v. \tag{II.26}$$

Paralleling the presentation for Cartesian coordinates, the divergence and gradient at a vertex are designed to satisfy Eq. (II.9),

$$V_v D_v(\mathbf{w}_c) = -\sum_c \mathbf{c}_1^{cv} \cdot \mathbf{w}_c \tag{II.27}$$

$$V_v \mathbf{G}_v(p_c) = -\sum_c \mathbf{c}_2^{cv} p_c. \tag{II.28}$$

For scalar quantities, discrete versions of the Laplacian are defined by combining the divergence and gradient operators as before,

$$V_c L_c(p_c) = V_c D_c(\mathbf{G}_v(p_c)) = -\sum_v \mathbf{c}_2^{cv} \cdot \sum_{c'} \mathbf{c}_2^{c'v} p_{c'} / V_v \tag{II.29}$$

$$V_v L_v(\psi_v) = V_v D_v(\mathbf{G}_c(\psi_v)) = -\sum_c \mathbf{c}_1^{cv} \cdot \sum_{v'} \mathbf{c}_1^{cv'} \psi_{v'} / V_c. \tag{II.30}$$

The Laplacian of a vector must take into account the spatial variation of the basis vectors. For a vector defined at the vertices of the mesh, like $\mathbf{u}_v$, the Laplacian of the z-component satisfies Eq. (II.30). The Laplacian of the r-component of a vector is

$$\frac{\partial}{\partial r}\left(\frac{1}{r}\frac{\partial ru}{\partial r}\right)+\frac{\partial^2 u}{\partial z^2}. \tag{II.31}$$

To get an appropriate discretization of Eq. (II.31), observe that by construction, the component derivatives of a vertex quantity are given by Eqs. (II.18)–(II.20). Equations (II.27)–(II.28) show how to define component derivatives at a vertex from cell-centered data,

$$V_v\left\langle\frac{1}{r}\frac{\partial r\psi}{\partial r}\right\rangle_v=\sum_v -c_{r_1}^{cv}\psi_c \tag{II.32}$$

$$V_v\left\langle\frac{\partial\psi}{\partial z}\right\rangle_v=\sum_v -c_z^{cv}\psi_c \tag{II.33}$$

$$V_v\left\langle\frac{\partial\psi}{\partial r}\right\rangle_v=\sum_v c_{r_2}^{cv}\psi_c. \tag{II.34}$$

Note that in addition to the sign change, the roles of $\mathbf{c}_1^{cv}$ and $\mathbf{c}_2^{cv}$ are reversed; at a vertex $\mathbf{c}_1^{cv}$ contains the contribution from the spatial variation of the basis vectors. Therefore, a discretization of Eq. (II.31) is formed as

$$V_v\tilde{L}_v(u_v)=-\sum_c \mathbf{c}_2^{cv}\cdot\sum_{v'}\mathbf{c}_2^{cv'}u_{v'}/V_c. \tag{II.35}$$

All of the quantities needed for Eqs. (10)–(12) are now defined in cylindrical coordinates except $\mathbf{f}_v$. To discretize the force rewrite Eq. (5),

$$\mathbf{f}^{\text{int}}=2\nabla\cdot\sigma\,\nabla\mathbf{d}+\nabla\times\sigma\nabla\times\mathbf{d}. \tag{II.36}$$

The first term on the right-hand side of Eq. (II.36) is handled in the same way as the Laplacian of a vector

$$V_v\langle\nabla\cdot\sigma\,\nabla\mathbf{d}\rangle_v=\left(-\sum_c\mathbf{c}_2^{cv}\cdot\sigma_c\sum_{v'}\mathbf{c}_2^{cv'}d_{rv'}/V_c,\,0,-\sum_c\mathbf{c}_1^{cv}\cdot\sigma_c\sum_{v'}\mathbf{c}_1^{cv'}d_{zv'}/V_c\right), \tag{II.37}$$

where $(d_r, 0, d_z)$ are the components of $\mathbf{d}$. The second term requires a definition for the curl. From

$$\nabla\times\mathbf{d}=\left(\frac{\partial d_r}{\partial z}-\frac{\partial d_z}{\partial r}\right)\hat{\theta} \tag{II.38}$$

define

$$V_c\langle\nabla\times\mathbf{d}\rangle_c=\sum_v\mathbf{c}_1^{cv}\times\mathbf{d}_v. \tag{II.39}$$

Then, since

$$\nabla \times \sigma(\nabla \times \mathbf{d}) = -\frac{\partial}{\partial z}(\sigma \nabla \times \mathbf{d})\,\hat{\mathbf{r}} + \frac{1}{r}\frac{\partial}{\partial r}(r\sigma \nabla \times \mathbf{d})\,\hat{\mathbf{z}}, \tag{II.40}$$

the second term of Eq. (II.36) is discretized using

$$V_v \langle \nabla \times \sigma(\nabla \times \mathbf{d}) \rangle_v = -\sum_c \mathbf{c}_1^{cv} \times \sigma_c \sum_{v'} \mathbf{c}_1^{cv'} \times \mathbf{d}_{v'}/V_c. \tag{II.41}$$

This completes the description of the discretization in cylindrical coordinates.

Acknowledgments

The work of Deborah Sulsky was partially supported by Sandia-University Research Projects, Contract 54-9308, task II.

References

1. S. Weinbaum, *Lect. Math. Life Sci.* **14**, 119 (1981).
2. G. Bossis and J. Brady, *J. Chem. Phys.* **80**, 5141 (1984).
3. J. Brady and G. Bossis, *J. Fluid Mech.* **155**, 105 (1985).
4. G. Bossis and J. Brady, *J. Chem. Phys.* **87**, 5437 (1987).
5. J. Brady and G. Bossis, *Annu. Rev. Fluid Mech.* **20**, 111 (1988).
6. J. Brady and L. Durlofsky, *Phys. Fluids* **31**, 717 (1988).
7. L. Durlofsky, J. Brady, and G. Bossis, *J. Fluid Mech.* **180**, 21 (1987).
8. R. J. Phillips, J. F. Brady, and G. Bossis, *Phys. Fluids* **31**, 3462 (1988).
9. A. Ladd, *J. Chem. Phys.* **88**, 5051 (1988).
10. A. Ladd, *J. Chem. Phys.* **90**, 1149 (1989).
11. A. Ladd, M. Colvin, and D. Frenkel, *Phys. Rev. Lett.* **60**, 975 (1988).
12. M. Ingber, *Int. J. Numer. Methods Fluids* **9**, 263 (1989).
13. M. Ingber, *Int. J. Numer. Methods Fluids* **10**, 791 (1990).
14. T. Tran-Cong and N. Phan-Thien, *Phys. Fluids A* **1**, 453 (1989).
15. S. Weinbaum, P. Ganatos, and Z.-Y. Yan, *Annu. Rev. Fluid Mech.* **22**, 275 (1990).
16. C. Peskin, *J. Comput. Phys.* **25**, 220 (1977).
17. A. L. Fogelson and C. S. Peskin, *J. Comput. Phys.* **79**, 50 (1988).
18. J. U. Brackbill, *Methods Comput. Phys.* **16**, 1 (1975).
19. T. Jordan, "A Guide to Parallel Computation and Some CRAY-1 Experiences," *Parallel Computation*, edited by G. Rodrique (Academic Press, Orlando, FL, 1982).
20. J. U. Brackbill, D. B. Kothe, and H. M. Ruppel, *Comput. Phys. Commun.* **48**, 25 (1988).
21. J. Happel and H. Brenner, *Low Reynolds Number Hydrodynamics* (Martinus Nijhoff, The Hague, The Netherlands, 1983), pp. 318, 345.
22. A. S. Dvinsky and A. S. Popel, *J. Comput. Phys.* **67**, 73 (1986).
23. J. U. Brackbill and J. Saltzman, *J. Comput. Phys.* **46**, 342 (1982).
24. J. U. Brackbill and H. M. Ruppel, *J. Comput. Phys.* **65**, 314 (1986).
25. J. U. Brackbill, *Comput. Phys. Commun.* **47**, 1 (1987).