JOURNAL OF COMPUTATIONAL PHYSICS 100, 1–16 (1992)

Particle-Method Solution of Two-Dimensional Convection–Diffusion Equations

AARON L. FOGELSON

Department of Mathematics, University of Utah, Salt Lake City, Utah 84112

Received October 4, 1988; revised June 20, 1991

We present a new method for solving two-dimensional convection-dominated convection–diffusion equations containing spatially and temporally localized source terms. The method uses grid-free transport: particles which carry point values of the concentration gradient move by convection and undergo random-walk to simulate diffusion. No numerical diffusion is introduced. Equations for the evolution of the gradient values are solved. The concentration is recovered from the particle data by solving Poisson equations. The method is applied to problems in rotational and elongational flow fields. Numerical results demonstrating convergence of the method are presented.

1. INTRODUCTION

In this paper we present a new method for solving two-dimensional convection–diffusion equations, in particular those in which convection dominates. Such situations are often described by the statement $Pe \gg 1$, where the (nondimensional) Peclet number is defined by $Pe = UL/\sigma$, with U and L a characteristic velocity and length, respectively, and σ the diffusion coefficient.

Convection-dominated convection–diffusion equations arise in many engineering applications, but our interest is motivated primarily by the occurrence of such an equation in our model of platelet aggregation during blood clotting [9]. In this model, a platelet, which is a type of blood cell, may be stimulated to secrete the chemical adenosine diphosphate (ADP) into the surrounding blood plasma. The secreted ADP acts as a signal to other platelets: A sufficiently high plasma concentration of it near a platelet which has not yet secreted ADP induces that platelet to secrete its store of ADP, as well as to become "sticky" and thus able to cohere with other platelets to form an aggregate. The transport of ADP, following its secretion, is described by a convection–diffusion equation, and in typical physiological situations, convection is strongly dominant over diffusion. The secretion of ADP by a platelet implies the existence of a source of ADP. The facts that the diameter of a platelet is much smaller than that of the blood vessels we study (the diameter ratio is about 1:25) and that we model each platelet as a discrete entity, mean that each platelet's contribution to this source is highly localized in space. The timecourse of secretion is such that the source is also highly localized in time. At any instant during the aggregation calculations, several platelets may be secreting ADP, so that the ADP source is a sum of spatially localized terms. Since the locations of the secreting platelets depends on their past motion (part of which is random), the spatial distribution of the ADP source is not known a priori. If we let $c(\mathbf{x}, t)$ denote the concentration of the chemical, then we are interested in equations of the form

$$\frac{\partial c}{\partial t} + \mathbf{u} \cdot \nabla c = Pe^{-1} \Delta c + s(\mathbf{x}, t), \tag{1}$$

where $Pe \gg 1$ and $s(\mathbf{x}, t)$ is a sum of terms each of which is highly localized both in space and time. We assume that the velocity field $\mathbf{u}$ is known, perhaps from another portion of the computation.

Finite difference methods and spectral methods are not well suited to this type of problem. There are two major reasons for this: the small nondimensional diffusion coefficient Pe^{-1} and the presence of multiple randomly-situated highly localized sources. Standard finite-difference schemes for convection-dominated convection–diffusion problems either suffer from nonphysical oscillations or introduce numerical diffusion which overwhelms the small physical diffusion (see, for example, [13]). More sophisticated finite-difference schemes can reduce numerical diffusion (and increase resolution) through the use of adaptive grids which concentrate computational effort in regions in which the solution's gradient is large. This is not practical in the aggregation calculations because the nature of the sources would compel frequent and extensive regridding. Spectral methods are well suited for smooth problems, or for problems which involve a steep gradient at a known location at which computational nodes can be concentrated. Our problem is not of one of these types. Furthermore,

0021-9991/92 $5.00

spectral methods for convection—diffusion problems have an error which becomes unbounded, at least linearly, as $Pe \to \infty$ [2].

By contrast, the particle method we present introduces no numerical diffusion and automatically concentrates computational effort near steep gradients. In our method, discrete particles are used to track the evolution of the concentration field. These particles are convected at the local fluid velocity and, in addition, undergo a random walk motion to simulate the diffusion. Each particle carries with it point values of the *gradient* of the concentration field. At any instant of time, the concentration field may be recovered from the particle locations and gradient values by integration. Since the random walk treatment of diffusion is the same as the treatment of viscosity in the random vortex method [6] and of diffusion in other related grid-free methods (reviewed in [10]), our method shares features of these methods: Particle transport reflects the true physical diffusivity, for it does not introduce additional numerical diffusion. Although the method introduces a statistical error which can be large for the concentration gradient field, the fact that the concentration c is obtained through integration makes the statistical fluctuations in c itself much smaller. Furthermore, the statistical fluctuations decrease towards zero as $Pe \to \infty$. The integration step, which is carried out on a lattice, introduces a small amount of smoothing, but this smoothing is *not* cumulative. We note that our method also has the potential to allow study of Eq. (1) in irregular domains. This will be pursued in future work.

2. CONTINUUM EQUATIONS

We wish to find the solution $c(\mathbf{x}, t)$ to the (nondimensional) convection–diffusion equation (1). We assume that the velocity field $\mathbf{u}(\mathbf{x}, t)$ and the source term $s(\mathbf{x}, t)$ are known, and we further assume that the flow is incompressible so that $\nabla \cdot \mathbf{u} = 0$. Instead of working directly with Eq. (1), we consider the system of equations

$$\frac{\partial}{\partial t}\begin{pmatrix} c_x \\ c_y \end{pmatrix} + \mathbf{u} \cdot \nabla \begin{pmatrix} c_x \\ c_y \end{pmatrix} = \left(\frac{1}{Pe}\right) \Delta \begin{pmatrix} c_x \\ c_y \end{pmatrix} - \begin{pmatrix} u_x & v_x \\ u_y & v_y \end{pmatrix} \begin{pmatrix} c_x \\ c_y \end{pmatrix} + \begin{pmatrix} s_x \\ s_y \end{pmatrix}, \quad (2)$$

obtained by differentiating each term in Eq. (1) with respect to x and y. The process of obtaining $c(\mathbf{x}, t)$ from Eq. (2) consists of two stages: one stage, which we call "transport," involves use of a Lagrangian analogue of Eq. (2) to move particles and to evolve "gradient vectors" associated with each particle. The second stage of the process, which we term "reconstruction," begins with the definition of Eulerian fields c_x and c_y from the instantaneous positions of the particles and values of the gradient vectors. The concentration $c(\mathbf{x}, t)$ is then determined from the data (c_x, c_y). We will describe the solution process for a square domain $R = [0, 1] \times [0, 1]$, but, we believe that the method will also work in irregular domains (see below).

We begin our discussion with the reconstruction stage of the process. Suppose that at time t, values of the smooth functions c_x and c_y are known at all points of R. (Since the time t does not enter explicitly into the reconstruction process, we drop reference to it during this portion of the exposition.) Introduce functions ϕ and ψ which satisfy

$$\begin{aligned} \Delta\phi &= c_x, && \text{in } R \\ \phi_y &= 0, && \text{for } y = 0 \text{ and } y = 1 \quad (3) \\ \phi &= 0, && \text{for } x = 0 \text{ and } x = 1 \\ \Delta\psi &= c_y, && \text{in } R \\ \psi_x &= 0, && \text{for } x = 0 \text{ and } x = 1 \quad (4) \\ \psi &= 0, && \text{for } y = 0 \text{ and } y = 1. \end{aligned}$$

Define $f(x, y) = c(x, y) - \phi_x(x, y) - \psi_y(x, y)$. Note that $\Delta f = (c_x - \Delta\phi)_x + (c_y - \Delta\psi)_y = 0$ in R; that is, f is harmonic. Further, $f_x(x, y) = c_x - \phi_{xx} - \psi_{xy} = \phi_{yy} - \psi_{xy}$. Since, $\phi(0, y) = 0$, it follows that $\phi_{yy}(0, y) = 0$. Also, the equation $\psi_x(0, y) = 0$ implies that $\psi_{xy}(0, y) = 0$. Hence, $\lim_{x \to 0} f_x(x, y) = 0$. Similarly, the normal derivative of f vanishes on the other edges of R. So, f is a constant function (at each time), and therefore, at time t,

$$c = \phi_x + \psi_y + \text{const.} \quad (5)$$

The constant in Eq. (5) may be determined from the value of c at a boundary point (e.g., c may be zero at the upstream inlet to the domain), or from the total amount of chemical in domain R. (It follows upon integrating Eq. (5) over R and using the boundary values of ϕ and ψ that the constant equals the total amount of chemical in R divided by the area of R.) We use the latter approach in this paper. We note that we can offer no proof that the function c defined by Eq. (5) is non-negative. However, in all of the computational tests we performed, this was always the case. In summary, c can be reconstructed from c_x and c_y by solving the two Poisson equations (3) and (4) and then using Eq. (5).

Note that the boundary conditions on c do *not* enter into the reconstruction process; they influence only the transport of (c_x, c_y) (see below). Thus, it should be possible to carry out the reconstruction for transport in an irregular region by embedding that region in a rectangle and imposing boundary conditions on ϕ and ψ (as in Eqs. (3) and (4)) along the edges of the rectangle.

The transport of (c_x, c_y) is accomplished by tracking discrete particles $\mathbf{x}_k = (x_k, y_k)$, with each of which is associated a "gradient vector" $\mathbf{g}_k = (gx_k, gy_k)$. Each such

particle–gradient pair contributes to the Eulerian concentration–gradient field in such a way that this field is given by the expression

$$\begin{pmatrix} c_x \\ c_y \end{pmatrix}(x, y, t) = \sum_k \mathbf{g}_k(t)\, \delta_s(x - x_k(t), y - y_k(t)). \quad (6)$$

Here, δ_s is an approximation to the two-dimensional Dirac delta function; it has support in a region of diameter $O(s)$ and has integral 1. Since $\delta_s(x, y)$ is a smooth (albeit highly localized) function, Eq. (6) defines smooth functions c_x and c_y from the particle data $(\mathbf{x}_k, \mathbf{g}_k)$. We note that because δ_s has units of reciprocal length squared, $\mathbf{g}_k$ has units of concentration times length.

New particles $\mathbf{x}_k$ and gradient vectors $\mathbf{g}_k$ are introduced whenever a localized source of chemical is present. If such a source is currently centered at a point $\xi = (\xi_1, \xi_2)$ and produces an amount of chemical A_0 during a short time interval dt, we imagine that this chemical is instantaneously distributed uniformly in a small disk of radius r_0 about ξ. The incremental concentration field due to this secretion is zero outside of this disk and $c_0 = A_0/\pi r_0^2$ within it. The (singular) gradient field associated with this concentration field may be expressed as

$$\begin{pmatrix} c_x \\ c_y \end{pmatrix}(x, y) = c_0 \int_{\|\mathbf{X} - \xi\| = r_0} \delta(x - X)\, \delta(y - Y)\, \frac{\xi - \mathbf{X}}{\|\xi - \mathbf{X}\|}\, dS \quad (7)$$

or

$$\begin{aligned} \begin{pmatrix} c_x \\ c_y \end{pmatrix}(x, y) = -c_0 \int_0^1 & \delta(x - \xi_1 - r_0 \cos(2\pi\alpha)) \\ & \times \delta(y - \xi_2 - r_0 \sin(2\pi\alpha)) \\ & \times (\cos(2\pi\alpha), \sin(2\pi\alpha))\, 2\pi r_0\, d\alpha. \end{aligned} \quad (8)$$

Formally, this is approximated by the expression

$$\begin{aligned} \begin{pmatrix} c_x \\ c_y \end{pmatrix}(x, y) = -c_0 \lim_{m \to \infty} \sum_{k=1}^{m} & \delta_s(x - \xi_1 - r_0 \\ & \times \cos(2\pi\alpha_k),\ y - \xi_2 - r_0 \sin(2\pi\alpha_k)) \\ & \times (\cos(2\pi\alpha_k), \sin(2\pi\alpha_k)) \frac{2\pi r_0}{m}, \end{aligned} \quad (9)$$

where $\alpha_k = k/m$. For finite m, Eq. (9) motivates our introduction of m new particles,

$$\mathbf{x}_k = (\xi_1 + r_0 \cos(2\pi\alpha_k),\ \xi_2 + r_0 \sin(2\pi\alpha_k)), \quad (10)$$

along with corresponding gradient vectors,

$$\mathbf{g}_k = -2\pi r_0 c_0/m\ (\cos(2\pi\alpha_k), \sin(2\pi\alpha_k)). \quad (11)$$

The vector $\mathbf{g}_k$ should be thought of as originating at the point $\mathbf{x}_k$. As $m \to \infty$, more and more particles are introduced and the strength $2\pi r_0 c_0/m$ (but not the form) of each gradient vector diminishes accordingly. New particles are introduced in this way as long as the source remains active.

It will simplify the subsequent discussion if we assume that the total production of chemical occurs at time 0 and results from a single source centered at ξ; in effect, the source is replaced by an *initial* distribution of chemical and corresponding particle–gradient pairs. (The corresponding simplification in Eq. (2) is the omission of the (s_x, s_y) term.) There is no loss of generality in making this simplification. The linearity of the equations with which we are concerned and the method by which new particles are introduced, allow us to regard the solution to a problem which involves prolonged secretion from one or more sources as a superposition of solutions from problems of the simpler type we will consider. Henceforth, we will consider a collection of particles and gradient-vectors whose *initial* values are given by Eqs. (10)–(11).

The motion of the particles $\mathbf{x}_k$ and the evolution of the gradient vectors $\mathbf{g}_k$ are governed by the equations

$$d\mathbf{x}_k(t) = \mathbf{u}(\mathbf{x}_k(t), t)\, dt + (2/Pe)^{1/2}\, d\mathbf{w}_t \quad (12)$$

and

$$\frac{d}{dt}\begin{pmatrix} gx_k \\ gy_k \end{pmatrix} = -\begin{pmatrix} u_x & v_x \\ u_y & v_y \end{pmatrix}\begin{pmatrix} gx_k \\ gy_k \end{pmatrix}, \quad (13)$$

respectively. In Eq. (13), the derivatives of u and v are evaluated at $\mathbf{x}_k(t)$. In Eq. (12), $\mathbf{w}_t$ is a Brownian motion path, so that for the jth component $w_{j,t}$ of $\mathbf{w}_t$, the difference $w_{j,t} - w_{j,s}$ is a Gaussian-distributed random variable with mean 0 and variance $t - s$ (for $t > s$).

Equations (12) and (13) deserve further discussion. In the absence of diffusion, these equations are simply the Lagrangian form, or characteristic equations, of Eq. (2), provided we identify $\mathbf{g}$ with (c_x, c_y). Equation (12) also includes a random-walk term to simulate the diffusion term in Eq. (2); this is identical to the way diffusion is modelled in the random vortex method for solving the Navier–Stokes' equations (see [6]) and is motivated by the following considerations: According to Eq. (2), if $\mathbf{u} = 0$, then c_x and c_y each satisfy a simple diffusion equation with diffusion coefficient $\nu = Pe^{-1}$. The Green's function for this equation (for an unbounded domain) is

$$\begin{aligned} G(\mathbf{x}, \mathbf{x}', t) &= \frac{1}{4\pi t\nu} \exp\left\{\frac{-\|\mathbf{x} - \mathbf{x}'\|^2}{4t\nu}\right\} \\ &= \frac{1}{(4\pi t\nu)^{1/2}} \exp\left\{\frac{-(x - x')^2}{4t\nu}\right\} \\ &\quad \times \frac{1}{(4\pi t\nu)^{1/2}} \exp\left\{\frac{-(y - y')^2}{4t\nu}\right\}. \end{aligned} \quad (14)$$

Equation (14) can be given a probabilistic interpretation. At time 0, place n particles $\mathbf{x}_k = (x_k^0, y_k^0)$ at the point (x', y') and assign to each particle a strength $1/n$. Divide the time interval $[0, t]$ into m increments of size Δt, and during each increment, let $\mathbf{x}_k$ undergo a random walk; that is, define

$$\begin{aligned} x_k^{l+1} &= x_k^l + (2\,\Delta tv)^{1/2}\,\eta_{1,k}^l \\ y_k^{l+1} &= y_k^l + (2\,\Delta tv)^{1/2}\,\eta_{2,k}^l, \end{aligned} \tag{15}$$

for $l = 0, ..., m-1$, where $\{\eta_{1,k}^l, \eta_{2,k}^l\}$ are independent Gaussian random variables, each with mean 0 and variance 1. The displacement at time t in the x-direction is a Gaussian random variable with mean 0 and variance $2m\,\Delta tv = 2tv$. A similar statement holds for the y-displacement. These displacements are independent, so the probability that a particle will fall in the box $B = [x, x+\Delta x] \times [y, y+\Delta y]$ is $(4\pi tv)^{-1/2} \exp\{-(x-x')^2/4tv\}(4\pi tv)^{-1/2} \exp\{-(y-y')^2/4tv\}\,\Delta x\,\Delta y = G(\mathbf{x}, \mathbf{x}', t)\,\Delta x\,\Delta y$ and $\lim_{n\to\infty}$ (Total strength in $B)/\Delta x\,\Delta y = G(\mathbf{x}, \mathbf{x}', t)$. If, as in the present application of the random-walk model, the initial distribution is $c_x(\mathbf{x}, 0) = f(\mathbf{x})$, then the solution at time t is

$$c_x(\mathbf{x}, t) = \iint_{R^2} G(\mathbf{x}, \mathbf{x}', t)\, f(\mathbf{x}')\, d\mathbf{x}'. \tag{16}$$

Suppose that n particles $\mathbf{x}_k$ are distributed randomly in the support of f and that with particle $\mathbf{x}_k$ is associated a strength $f(\mathbf{x}_k)/n$. Suppose also that each particle undergoes the random-walk motion given by Eq. (15) as before. Then, at time t, the quantity (Total strength in $B)/\Delta x\,\Delta y$ approaches the expression on the right-hand side of Eq. (16) as $n \to \infty$. (For further discussion, see [7].)

Equations (12)–(13) describe the transport of (c_x, c_y) when $\mathbf{x}_k$ is away from the material boundary of the flow domain. If $\mathbf{x}_k$ contacts the boundary, then the boundary conditions on c determine how $\mathbf{g}_k$ is treated. For example, suppose we want $c_y = 0$ along $y = 0$. Then, $(\partial/\partial y)(c_x) = 0$ as well. So, this boundary is an absorptive boundary for c_y and a reflective boundary for c_x. If $\mathbf{x}_k$ hits this boundary, it is reflected back into the domain and (gx_k, gy_k) is reflected as $(gx_k, -gy_k)$. The boundary condition should therefore be satisfied in a statistical sense. Note that this procedure is not limited to a boundary parallel to a coordinate axis; the appropriate reflection for $\mathbf{g}_k$ is readily determined at any boundary point at which the normal direction is known.

In summary, the solution of Eq. (1) is carried out in two stages: The Lagrangian transport stage begins with the introduction of new particles $\mathbf{x}_k$ and gradient vectors $\mathbf{g}_k$ according to equations like Eqs. (10)–(11) when a source is active. The subsequent motion of $\mathbf{x}_k$ by convection and diffusion, and the evolution of $\mathbf{g}_k$ are governed by Eqs. (12) and (13), respectively (with appropriate modifications at boundary points). The Eulerian reconstruction stage involves definition of gradient fields c_x and c_y from the particle data $\{\mathbf{x}_k, \mathbf{g}_k\}$ by means of Eq. (6). The Poisson equations (3) and (4) are then solved, and c is obtained from Eq. (5). Note that the information transfer between the Lagrangian and Eulerian stages is one-way, from Lagrangian to Eulerian; hence, errors made in computing approximate solutions to the Eulerian equations do *not* feed back to corrupt the Lagrangian particle data.

It is interesting to consider the behavior of a system like that described by Eqs. (8), (10)–(13), but in the absence of diffusion. Suppose that the small secretion disk is centered at the origin and that a continuum of particles $\mathbf{x}(\alpha, 0)$ and gradient vectors $\mathbf{g}(\alpha, 0)$ is introduced along its boundary curve Γ_0. Equation (8) can be rewritten

$$\begin{aligned} \begin{pmatrix} c_x \\ c_y \end{pmatrix}(x, y, 0) = -c_0 \int_0^1 \delta(x - x(\alpha, 0)) \\ \times \delta(y - y(\alpha, 0))\, \mathbf{n}(\alpha, 0)\, d\alpha, \end{aligned} \tag{17}$$

where $\mathbf{n}(\alpha, 0) = (y_\alpha(\alpha, 0), -x_\alpha(\alpha, 0))$ is an outward normal to the initial curve Γ_0. Corresponding to Eqs. (10)–(11), we now have

$$\begin{aligned} &\mathbf{x}(\alpha, 0) = (x(\alpha, 0), y(\alpha, 0)) = (r_0 \cos(2\pi\alpha), r_0 \sin(2\pi\alpha)) \\ &\mathbf{g}(\alpha, 0) = -c_0 \mathbf{n}(\alpha, 0) \end{aligned} \tag{18}$$

for $0 \leqslant \alpha < 1$. We suppose that for $t > 0$, the vectors $\mathbf{x}(\alpha, t)$ and $\mathbf{g}(\alpha, t)$ evolve according to

$$\mathbf{x}_t = \mathbf{u}(\mathbf{x}(\alpha, t), t) \tag{19}$$

$$\mathbf{g}_t = -\begin{pmatrix} u_x & v_x \\ u_y & v_y \end{pmatrix} \mathbf{g}, \tag{20}$$

respectively, with $\mathbf{x}(\alpha, 0)$, $\mathbf{g}(\alpha, 0)$ given as above. We define the vector field $(\tilde{c}_x, \tilde{c}_y)$ from this data using, by analogy with Eq. (8), the equation

$$\begin{aligned} \begin{pmatrix} \tilde{c}_x \\ \tilde{c}_y \end{pmatrix}(x, y, t) = \int_0^1 \mathbf{g}(\alpha, t)\, \delta(x - x(\alpha, t)) \\ \times \delta(y - y(\alpha, t))\, d\alpha. \end{aligned} \tag{21}$$

At $t = 0$, the curve Γ_0 separates a region of area a_0 with concentration c_0 from a region with concentration 0. The curve Γ_0 is a material curve of the fluid, and so it evolves in time t to a curve Γ_t, which, because of the incompressibility of the flow, encloses a region of area a_0. Hence, since Eq. (1) implies that in the absence of diffusion there is no flux across the material curve Γ_t, Γ_t also separates a region of concentration c_0 from a region of concentration 0. The gradient field associated with the situation at time t is therefore

$$\begin{aligned} \begin{pmatrix} c_x \\ c_y \end{pmatrix}(x, y, t) = -c_0 \int_0^1 \mathbf{n}(\alpha, t)\, \delta(x - x(\alpha, t)) \\ \times \delta(y - y(\alpha, t))\, d\alpha, \end{aligned} \tag{22}$$

since the curve Γ_t is defined by the points $(x(\alpha, t), y(\alpha, t))$, $0 \leqslant \alpha < 1$, which move according to Eq. (19). This equation is identical to Eq. (21) except that the vector $\mathbf{g}(\alpha, t)$ in Eq. (21) is replaced by the vector $-c_0\mathbf{n}(\alpha, t)$. Hence, the system Eqs. (18)–(21) will give the correct gradient field provided $\mathbf{g}(\alpha, t) = -c_0\mathbf{n}(\alpha, t)$ for all $t \geqslant 0$. This equation holds for $t = 0$, and so, by the uniqueness theory for linear ordinary differential equations, it will hold for all $t > 0$, provided $\mathbf{n}(\alpha, t)$ satisfies Eq. (20). Differentiating $\mathbf{n}(\alpha, t)$ with respect to t, we find

$$\begin{aligned}\mathbf{n}_t(\alpha, t) &= (y_{t\alpha}, -x_{t\alpha})^{\mathrm{T}} \\ &= (v_\alpha(x, y), -u_\alpha(x, y))^{\mathrm{T}} \\ &= \begin{pmatrix} v_y & -v_x \\ -u_y & u_x \end{pmatrix}\begin{pmatrix} y_\alpha \\ -x_\alpha \end{pmatrix} \\ &= -\begin{pmatrix} u_x & v_x \\ u_y & v_y \end{pmatrix}\mathbf{n}(\alpha, t).\end{aligned}$$

The last step uses the fact that $u_x + v_y = 0$. Hence, $\mathbf{g}$ evolves so as to remain normal to Γ_t, and, since $\|\mathbf{g}\| = c_0\|\mathbf{n}\| = c_0((x_\alpha)^2 + (y_\alpha)^2)^{1/2}$, $\|\mathbf{g}\|$ grows in response to local stretching of the curve Γ_t.

3. DISCRETE EQUATIONS

In this section, we discuss the discrete version of the reconstruction and transport process that we use to find an approximate solution to Eq. (1). The method that we use is a hybrid in the sense that a computational lattice is used during the reconstruction process to find ϕ and ψ, while the points $\mathbf{x}_k$ are introduced and tracked independently of this lattice. We will again assume that our computational domain is the square $R = [0, 1] \times [0, 1]$, and we place a uniform lattice with lattice-spacing h over R. The value of a lattice function f at a point (ih, jh) of the lattice is denoted $f_{i,j}$. Time is discretized into increments of length Δt and a superscript n on a variable denotes its value at time $t = n\,\Delta t$.

We begin by considering the analogue of Eqs. (3)–(5) and for now assume that values of the functions c_x and c_y are known at all points of the lattice. Equations (3) and (4) are each replaced with a discrete Poisson equation with the lattice functions c_x and c_y as their respective inhomogeneous terms. In these equations, we use the standard second-order discrete Laplacian operator based on a five-point stencil. The system of linear equations which is defined by each discrete Poisson equation is solved by a combination of Fourier transform and tridiagonal solvers (see, for example, [15]). More specifically, for the discrete analogue of Eq. (3), a fast one-dimensional cosine transform (with respect to y) is performed on c_x along each line $x = j_1 h$ of the lattice. For each Fourier mode, an independent tridiagonal system, which corresponds to differencing in the x-direction, results. These are easily solved, and a set of one-dimensional inverse fast cosine transforms yields values of ϕ at all lattice points. This fast solver has the virtue that it can easily be implemented in such a way as to vectorize on a CRAY computer. During the Fourier transform step, we perform one-dimensional transforms on many independent data sets; vectorization is achieved by replacing each scalar assignment statement in a standard one-dimensional fast cosine transform subroutine by a DO-loop whose index indicates the successive datasets. A similar technique is used to achieve vectorization in the solution of the tridiagonal systems. The discrete version of Eq. (4) is solved in a similar fashion, except that the roles of x and y are reversed. Once lattice values of ϕ and ψ are known, lattice values of c are calculated using second-order difference approximations to the derivatives in Eq. (5). Earlier we remarked that we believe that our method can be used to solve Eq. (1) in an irregular domain by embedding that domain in a rectangle for the reconstruction stage. In that case, the use of fast Poisson solvers (such as that just described) would still be available to us.

We turn next to the solution of the stochastic differential equations, Eq. (12), which govern the motion of the particles $\mathbf{x}_k$. For convenience, we will ignore the subscript k. The usual approach to approximating the solution of an equation of this type is based on fractional steps (see, for example, [6, 10]). To advance the solution from time $n\,\Delta t$ to time $(n+1)\,\Delta t$, a standard ordinary differential equation solver is used to find an approximate solution $\mathbf{x}^*$ to the equation

$$\frac{d\mathbf{x}}{dt} = \mathbf{u}(\mathbf{x}, t); \qquad \mathbf{x}(n\,\Delta t) = \mathbf{x}^n \tag{23}$$

and then an independent increment $\boldsymbol{\eta}$ of the Brownian motion (with mean 0 and variance 1) is added to give

$$\mathbf{x}^{n+1} = \mathbf{x}^* + (2\,\Delta t/Pe)^{1/2}\,\boldsymbol{\eta}. \tag{24}$$

However, Chang [3] has shown that no matter how accurately Eq. (23) is solved, Eq. (24) gives, at best, a first-order accurate solution to Eq. (12) (see [3] for precise definitions of the order of an approximation to a stochastic differential equation like Eq. (12)). In the same paper, Chang proposed a new method of Runge–Kutta type for solving Eq. (12) and gave numerical evidence that this scheme yields a second-order accurate approximation to the solution of Eq. (12). Recently, Long [12] proved that the scheme is indeed second-order accurate. Chang's scheme, which is based on the mid-point rule, is as follows:

$$\mathbf{q} = \mathbf{x}^n + 1/2\,\Delta t\mathbf{u}(\mathbf{x}^n, n\,\Delta t) \tag{25}$$

$$\mathbf{q}^* = \mathbf{x}^n + 1/2\,\Delta t\mathbf{u}(\mathbf{x}^n, n\,\Delta t) + (2\,\Delta t/Pe)^{1/2}\,\boldsymbol{\eta}^n \tag{26}$$

$$\begin{aligned}\mathbf{x}^{n+1} = \mathbf{x}^n + 1/2\{&\mathbf{u}(\mathbf{q}, (n+1/2)\,\Delta t) \\ &+ \mathbf{u}(\mathbf{q}^*, (n+1/2)\,\Delta t)\}\,\Delta t + (2\,\Delta t/Pe)^{1/2}\,\boldsymbol{\eta}^n.\end{aligned} \tag{27}$$

Here, $\boldsymbol{\eta}^n = (\eta_1^n, \eta_2^n)$ is a pair of independent Gaussian random variables each with mean 0 and variance 1. In practice, these numbers are generated using the Box–Muller transformation on two uniformly-distributed random variables [8]. The random steps $\boldsymbol{\eta}^n$ from different timesteps are also chosen independently of one another. Note that in Eqs. (25)–(27), the deterministic part $\mathbf{u}(\mathbf{x}, t)$ and the random step are inter-twined. This is essential for achieving a higher order of accuracy than the fractional-step schemes can provide [3].

The gradient vectors $\mathbf{g}_k$ evolve according to Eq. (13). Let S denote the matrix of velocity derivatives which appears in that equation. Then, the exact solution to this equation over the time interval $(n\,\Delta t, (n+1)\,\Delta t)$ is, of course, given by the expression (again we suppress the subscript k)

$$\mathbf{g}((n+1)\,\Delta t) = \exp\left\{-\int_{n\,\Delta t}^{(n+1)\,\Delta t} S(\mathbf{x}(\tau), \tau)\, d\tau\right\} \mathbf{g}(n\,\Delta t), \tag{28}$$

where $\mathbf{x}(\tau)$ is the (exact) position of the corresponding particle. We approximate the integral in this expression using the trapezoidal rule. Let

$$Q = \tfrac{1}{2}\{S(\mathbf{x}^n, n\,\Delta t) + S(\mathbf{x}^{n+1}, (n+1)\,\Delta t)\}. \tag{29}$$

Then, we define

$$\mathbf{g}^{n+1} = \exp\{-\Delta t Q\}\, \mathbf{g}^n. \tag{30}$$

It follows from the incompressibility of the flow field $(u_x + v_y = 0)$, that $S(\mathbf{x}(\tau), \tau)$ and, hence, also Q have zero trace. Thus, the exponential $\exp\{-\Delta t Q\}$ has a series expansion which collapses to two terms:

$$\exp\{-\Delta t Q\} = I\cosh(\gamma^n\,\Delta t) - Q\,\frac{\sinh(\gamma^n\,\Delta t)}{\gamma^n}. \tag{31}$$

Here, I is the 2×2 identity matrix and $\gamma^n = (\overline{u_x}^2 + \overline{u_y v_x})^{1/2}$, where $\overline{u_x} = \frac{1}{2}\{u_x(x^n, n\,\Delta t) + u_x(x^{n+1}, (n+1)\,\Delta t)\}$ and similar expressions define $\overline{u_y}$ and $\overline{v_x}$. Our formula for updating $\mathbf{g}$ is, therefore,

$$\mathbf{g}^{n+1} = \left\{I\cosh(\gamma^n\,\Delta t) - Q\,\frac{\sinh(\gamma^n\,\Delta t)}{\gamma^n}\right\} \mathbf{g}^n. \tag{32}$$

Equations (25)–(27) and (32) comprise the particle/gradient-vector update step of our calculations. We have tacitly assumed in the discussion of these equations that analytic expressions for the velocity $\mathbf{u}$ and its derivatives are available for evaluation in Eqs. (25)–(27) and (32). In practice, the velocity is more likely to be known only at discrete times $(n\,\Delta t)$ and then only at points of a lattice. Hence, in general, it is necessary to interpolate these lattice values to the particle positions for use in Eqs. (25)–(27). Further, Eq. (27), as written, calls for the evaluation of $\mathbf{u}$ at time $t = (n+\frac{1}{2})\,\Delta t$. If this data is unavailable, we replace $\mathbf{u}(\cdot, (n+\frac{1}{2})\,\Delta t)$ by $\mathbf{u}(\cdot, n\,\Delta t)$ or $\frac{1}{2}(\mathbf{u}(\cdot, n\,\Delta t) + \mathbf{u}(\cdot, (n+1)\,\Delta t))$, depending on whether $\mathbf{u}(\cdot, (n+1)\,\Delta t)$ is currently known. If the velocity field is smooth and slowly varying in time, then these modifications of Eqs. (25)–(27) should cause little loss in accuracy. Similar considerations apply to the velocity gradients required by Eq. (32). Equations (25)–(27) and (32) can easily be implemented in a vectorizable fashion by making the index of innermost DO-loops run over the index k of the particles. If it is necessary to interpolate the velocity and its derivatives from a lattice to the particle positions, then preliminary use of the CRAY GATHER utility to collect the necessary lattice values will facilitate vectorization of the update steps.

It remains to specify a discrete version of Eq. (6) by means of which the particle data $\{\mathbf{x}_k, \mathbf{g}_k\}$ define the lattice functions c_x^n and c_y^n that appear in the discrete Poisson equations for ϕ and ψ. This is accomplished by the equation

$$(c_x^n, c_y^n)_{i,j} = \sum_{k=1}^{N} \mathbf{g}_k^n H_{i,j}(\mathbf{x}_k^n) \tag{33}$$

where N is the total number of particles in the domain and $H_{i,j}(x, y)$ is the piecewise-linear "hat" function given by $H_{i,j}(x, y) = h^{-2}\max\{0, 1 - h^{-1}|x - ih|\}\max\{0, 1 - h^{-1}|y - jh|\}$. $H_{i,j}(x, y)$ has support consisting of the four lattice points nearest to the point (x, y) and $\sum_{i,j} h^2 H_{i,j}(x, y) = 1$ for any (x, y). The second property has the important consequence that the process of transferring data from the particles to the lattice is conservative; that is, if $c_{x,k}$ is the lattice function generated by the x-component of $\mathbf{g}_k$, then $\sum_{i,j} h^2 (c_{x,k})_{i,j} = g x_k$. A similar relation holds for the y-component of the gradient.

The computational complexity of our method is as follows: In the transport stage, each particle is moved, the associated gradient vector is updated, and values are transferred to the lattice. Since this is done independently for each particle, the work required is linearly proportional to the total number N of particles. The fast solvers used in the reconstruction stage require $O((NG)\log(NG))$ operations, where NG is the total number of lattice points.

In his study of (inviscid) flows with slight density variations, Anderson [1] used particles to transport point values of the density gradient (without diffusion) and used a representation of the gradient field similar to our Eq. (6). His method of recovering the density from the density gradients is based on the formula (we use c for density)

$$c = G_x * c_x + G_y * c_y, \tag{34}$$

where G is the fundamental solution to Laplace's equation in the plane and $*$ denotes convolution. This formula is derived from the identity $c = G * \Delta c$ using integration by parts, under the assumption that initially c has support in a bounded region. We note that our Eqs. (3)–(5) yield an identical expression when we let the rectangle R grow to fill the plane. So, at the continuum equation level, the two approaches are almost the same. However, at the level of their discrete implementation, they differ substantially. Anderson substitutes Eq. (6) into Eq. (34) to obtain

$$c(\mathbf{x}, t) = \sum_{k=1}^{N} \{(G_x * \delta_s)(\mathbf{x} - \mathbf{x}_k(t))\, gx_k(t) + (G_y * \delta_s)(\mathbf{x} - \mathbf{x}_k(t))\, gy_k(t)\}. \tag{35}$$

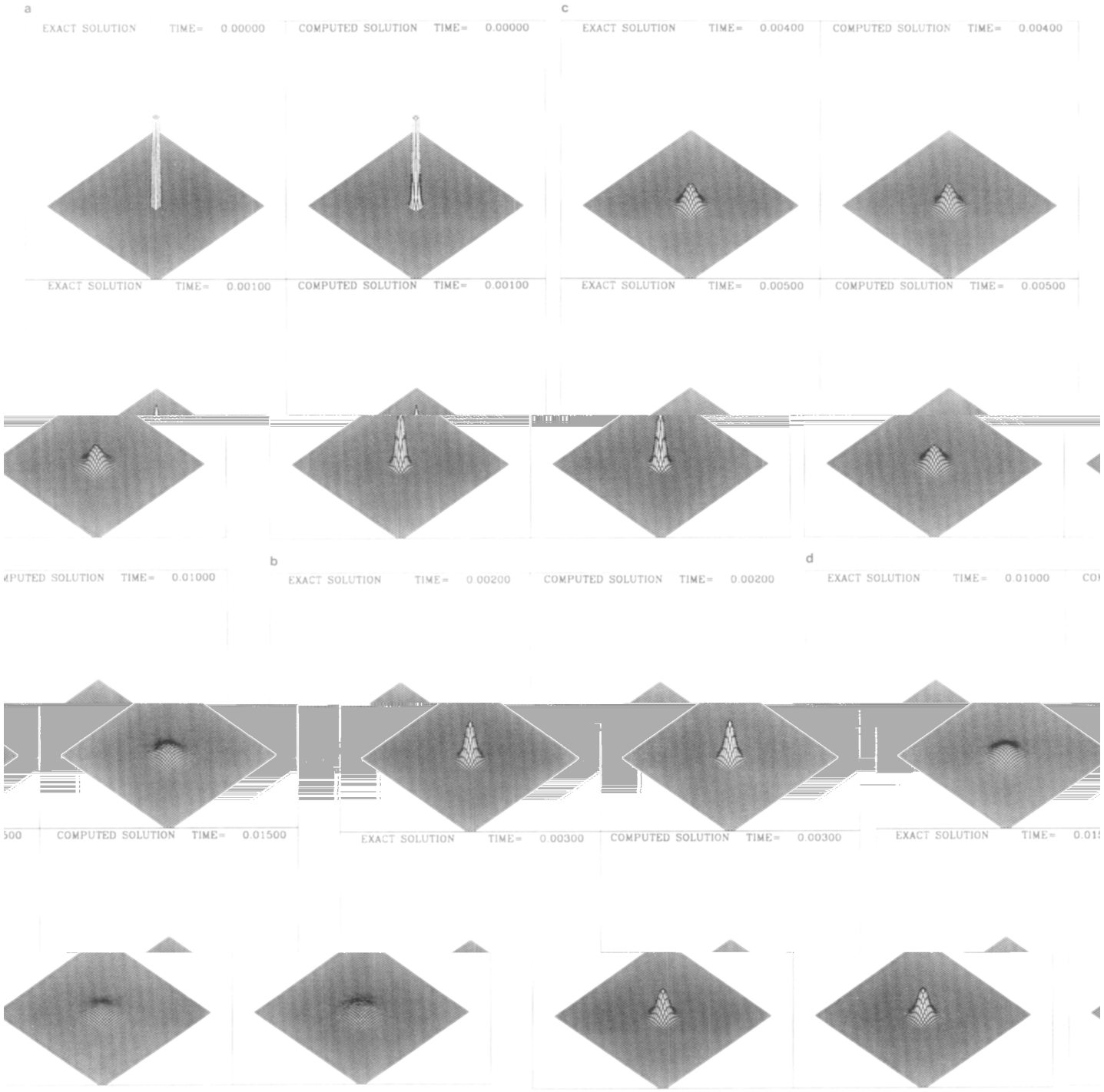

FIG. 1. Exact and computed concentrations at selected times. Flow is rigid rotation: $Pe = 4.5$, $N = 16{,}000$, $\Delta t = 0.05\tau_0$. Time is measured in seconds.

We solve discrete analogues of Eqs. (3)–(4). Both approaches introduce smoothing through the functions δ_s, and the major difference between them is in the complexity. To evaluate c at m points requires $O(mN)$ operations with Eq. (35), and $O((NG)\log(NG))$ operations with our approach. For large m and N, the lattice-based approach is

4. RESULTS

In this section, we report results of tests of the above method on problems involving three different flow fields. Since our aim in this project is to develop a method for solving the ADP transport equation that arises in our

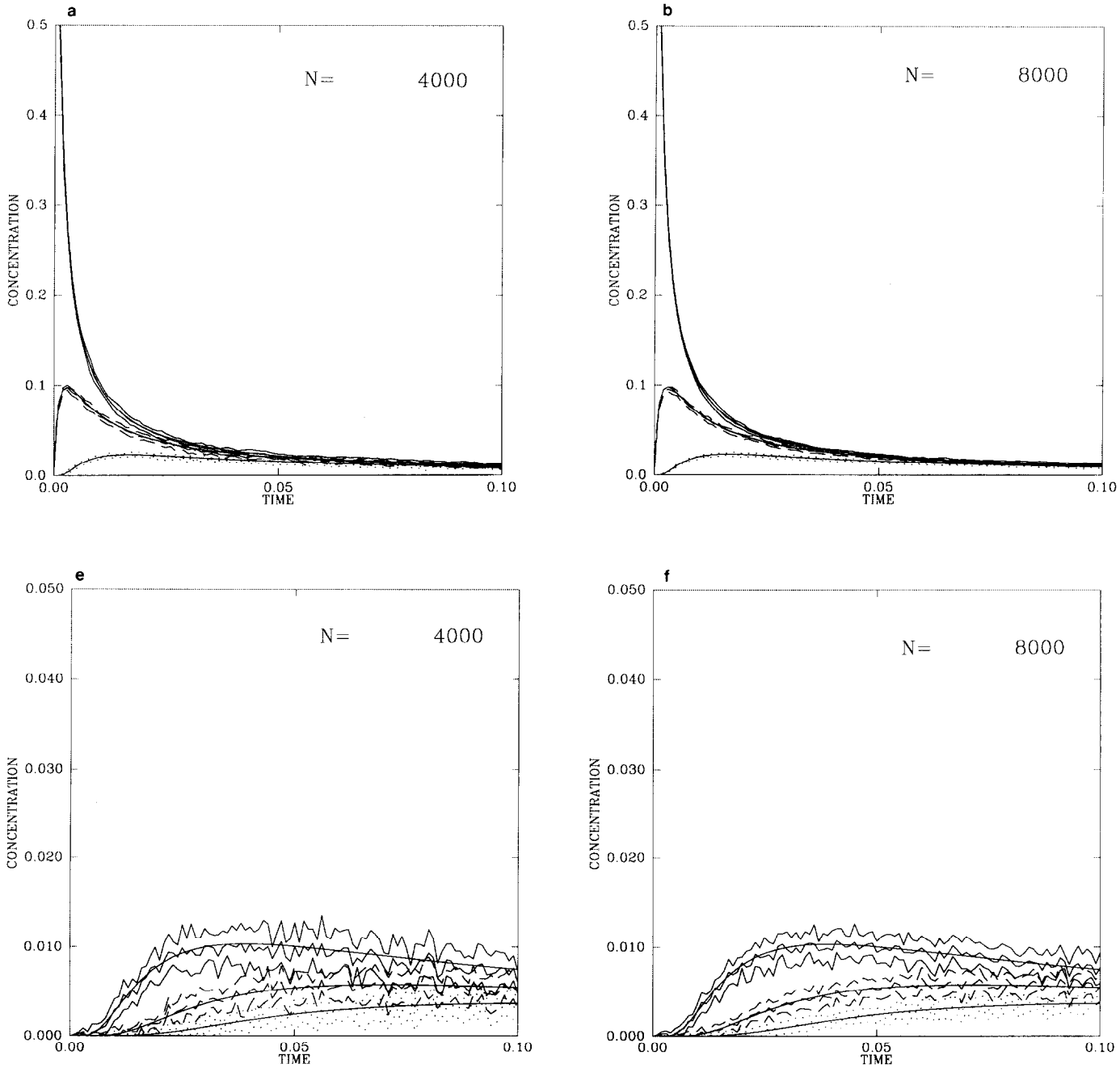

FIG. 2. Time course of exact concentration, ensemble average of computed concentrations, and ensemble average +/− standard deviation at six target points $(x_0 + jd_0, y_0)$, $j = 0, 1, ..., 5$. Top (bottom) panels show concentrations for points 0, 1, 2 (3, 4, 5). Note the difference in vertical scales. $N = 4000$ for the first top panels and increases to $N = 32{,}000$ for final bottom panels. Time is measured in seconds.

problems that would indicate how the method would work in that context. In each test, the initial state is a uniform concentration $c_0 = 1$ inside of a disk D_0 of radius r_0 (and diameter d_0), centered at $\mathbf{x}_0 = (x_0, y_0)$, and concentration 0 outside of this disk. There is no source of chemicals at later times and, hence, $s(\mathbf{x}, t) = 0$ in Eq. (1). Each test problem involves three physical parameters: the radius of the disk r_0, the convection time scale τ_0, and the diffusion coefficient σ_0. These are specified in CGS units with values relevant to the aggregation calculations.

In the first set of tests, the flow field is that of a rigid rotation about the point $\mathbf{x}_0$. In physical units, $\mathbf{u} = (u(x, y), v(x, y)) = \tau_0^{-1}(-(y - y_0), (x - x_0))$. The virtue of this flow field is that it preserves radial symmetry. Therefore

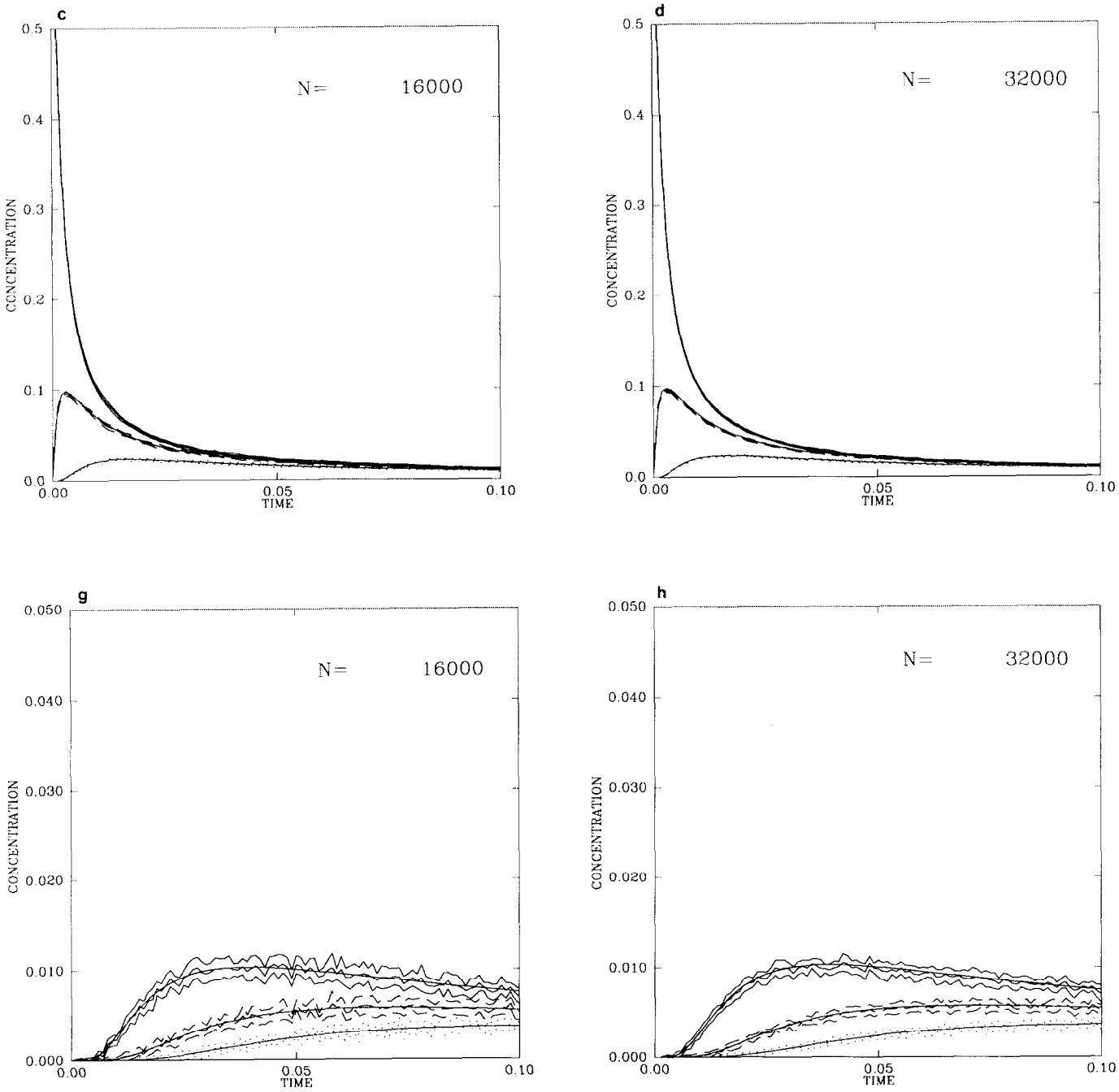

FIGURE 2—*Continued*

this problem has the same solution as the simple diffusion problem, namely, the convolution of the Green's function given in Eq. (14) (with $\nu = \sigma_0$) with the indicator function for the disk D_0. We denote this solution by $ce(\mathbf{x}, t)$. We used the radius of a platelet $r_0 = 1.5 \times (10)^{-4}$ cm, the ADP diffusion coefficient $\sigma_0 = 5 \times (10)^{-6}$ cm^2/s, and, since the shear rate in a typical small blood vessel is about 1000 s^{-1}, we use $\tau_0 = 1 \times (10)^{-3}$ s. These parameter values give a Peclet number based on platelet size of $Pe = r_0^2/\tau_0\sigma_0 = 4.5$. Each experiment simulated a time period of 0.1 s or $100\tau_0$. Compare this with the convection time scale τ_0 for rotation through one radian of the arc and the diffusion time scale $\sigma_0/r_0^2 = 4.5\tau_0$.

We performed a series of experiments in which we varied the number of particles N used to represent the initial jump in concentration, and the timestep Δt. For each value of N and Δt, $m = 10$ experiments were performed with different realizations of the particles' random walks. The lattice spacing h used for the reconstruction process was chosen, based *not* on the radius of the disk D_0 (i.e., the size of a platelet), but instead based on the radius of the blood vessel used in the aggregation calculations, $l_0 = 25 \times (10)^{-4}$cm. The domain used for the reconstruction was a $2l_0 \times 2l_0$ square concentric with the disk D_0. For the results described below, $h = 1/64l_0$, so $r_0 = 3.84h$. In this sense, the initial concentration field represented secretion from a spatially localized source.

In Fig. 1, we show surface plots of the exact concentration $ce(\mathbf{x}, t)$ and the computed concentration $c(\mathbf{x}, t)$ at selected times. The latter is from an experiment with $N = 16{,}000$ and $\Delta t = 0.05\tau_0$. The top panels show that the reconstruction process smears the initial step in concentration, but is successful overall in reconstructing the initial localized concentration field. Later panels show computed surfaces which look much like the corresponding exact solution surfaces.

In each pair of panels of Fig. 2, we follow the timecourse of the concentration at six "target" points $(x_0 + jd_0, y_0)$, $j = 0, 1, ..., 5$, lying on or to the "east" of the disk D_0. At each time and for each target point, we computed the local-area average of the computed and exact concentration by averaging their respective values at the lattice points contained within a platelet-sized disk (radius r_0) centered on the target point. (The motivation for this averaging is that platelets respond to distributed sensing of their environment.) We plot the ensemble average (i.e., the average of the results of the 10 experiments with identical values of N and Δt), as well as this ensemble average plus/minus one standard deviation. Also plotted is the local-area average of the exact solution. Results are shown for $\Delta t = 0.05\tau_0$ and $N = 4000$, 8000, 16,000, and 32,000. The ensemble averages all track the exact solution. The standard deviations are at most a fraction of 1% of the peak concentration 1.0. For example, at target point $(x_0 + 4d_0, y_0)$, the maximum standard deviation in time is 0.0031 with $N = 4000$, and 0.0008 with $N = 32{,}000$. For each target point, the time average of the standard deviation decreases as $N^{-1/2}$. Similar calculations with $\Delta t = 0.025\tau_0$ (not shown) are almost identical to those shown.

For $t > 5\tau_0$, the maximum error at each time over the entire computational lattice is about 1% of the peak concentration when $N = 4000$, and decreases as $N^{-1/2}$ for larger N. For earlier times, the errors are somewhat larger and appear to be dominated by the inability of the reconstruction process to capture completely the steep initial gradients.

It would be useful to know how much of the error in these calculations is due to the chance statistical fluctuations that arise because we choose particular random steps for each of our N particles and how much is a systematic bias which results from the use of finite N, Δt, and h. Towards this end,

TABLE I

Decomposition of Mean Square Error

Time	GAPSME	GAPVAR	GAPSB
	$N = 4000$		
0.00	7.29E(−5)	0.00E(−0)	7.29E(−5)
1.00	1.20E(−5)	2.62E(−6)	9.36E(−6)
5.00	5.40E(−6)	4.29E(−6)	1.11E(−6)
10.00	4.78E(−6)	4.33E(−6)	4.47E(−7)
20.00	6.19E(−6)	5.49E(−6)	6.97E(−7)
50.00	8.21E(−6)	7.24E(−6)	9.76E(−7)
100.00	8.12E(−6)	7.04E(−6)	1.07E(−6)
	$N = 8000$		
0.00	7.29E(−5)	0.00E(−0)	7.29E(−5)
1.00	7.85E(−6)	1.48E(−6)	6.38E(−6)
5.00	2.98E(−6)	2.19E(−6)	7.85E(−7)
10.00	2.69E(−6)	2.40E(−6)	2.95E(−7)
20.00	3.50E(−6)	3.06E(−6)	4.37E(−7)
50.00	3.49E(−6)	3.59E(−6)	3.22E(−7)
100.00	4.42E(−6)	4.06E(−6)	3.63E(−7)
	$N = 16{,}000$		
0.00	7.29E(−5)	0.00E(−0)	7.29E(−5)
1.00	7.55E(−6)	5.65E(−7)	6.96E(−6)
5.00	1.56E(−6)	1.01E(−6)	5.57E(−7)
10.00	1.48E(−6)	1.20E(−6)	2.70E(−7)
20.00	1.57E(−6)	1.36E(−6)	2.07E(−7)
50.00	1.98E(−6)	1.69E(−6)	2.90E(−7)
100.00	1.95E(−6)	1.70E(−6)	2.46E(−7)
	$N = 32{,}000$		
0.00	7.29E(−5)	0.00E(−0)	7.29E(−5)
1.00	7.53E(−6)	3.06E(−7)	7.22E(−6)
5.00	1.26E(−6)	6.34E(−7)	6.24E(−7)
10.00	8.32E(−7)	5.94E(−7)	2.37E(−7)
20.00	8.40E(−7)	6.87E(−7)	1.53E(−7)
50.00	9.94E(−7)	8.63E(−7)	1.32E(−7)
100.00	1.03E(−6)	8.92E(−7)	1.34E(−7)

we proceed as follows: For fixed N, Δt, and h, let $c^{(r)}(\cdot, t)$ denote the computed solution at time t in the rth of m experiments. Define $\bar{c}^m = (1/m)\sum_{r=1}^m c^{(r)}$; this is an estimated ensemble average, the true ensemble average is defined as $\bar{c}^\infty = \lim_{m\to\infty} \bar{c}^m$. From the definition of $\bar{c}^m$, it follows easily that

$$\frac{1}{m}\sum_{r=1}^{m}(c^{(r)}-ce)^2 = \frac{1}{m}\sum_{r=1}^{m}(c^{(r)}-\bar{c}^m)^2 + (\bar{c}^m - ce)^2. \quad (36)$$

We take the average of each side of Eq. (37) over the computational lattice to obtain

$$\left\|\frac{1}{m}\sum_{r=1}^{m}(c^{(r)}-ce)^2\right\|_1 = \left\|\frac{1}{m}\sum_{r=1}^{m}(c^{(r)}-\bar{c}^m)^2\right\|_1 + \|(\bar{c}^m - ce)^2\|_1. \quad (37)$$

(Note that the first term may also be written $(1/m)\sum_{r=1}^m \|c^{(r)} - ce\|_2^2$.) The first term on the right side of Eq. (37) gives a measure of the fluctuations from one run to the next about the mean computed solution; we call it the grid-averaged pointwise variance (GAPVAR). The last term in Eq. (37) gives an estimate of the squared bias, the error which remains after the fluctuations are averaged out; we call it the grid-averaged pointwise squared bias (GAPSB). We call the term on the left in Eq. (37) the grid-averaged pointwise mean square error (GAPSME). In Table I, we display these quantities at selected times for experiments with $N = 4000$, 8000, 16,000, and 32,000 and $\Delta t = 0.05\tau_0$. For each N, $m = 10$ experiments were performed. For all times $t > \tau_0$, GAPVAR $\gg$ GAPSB. For $t \leqslant \tau_0$, the principal component of the error is due to the finite resolution of the reconstruction process. At later times, the error is dominated by statistical noise. As N increases, GAPVAR decreases as N^{-1}. GAPSB also generally decreases with increasing N. In a linear problem such as ours, the limit $m \to \infty$, with fixed N, is the same as the limit $N \to \infty$, because the samples may be pooled. Hence, if $\bar{c}^\infty$ were used to compute GAPSB, the latter should display no N dependence. Thus, the computed GAPSB, calculated using the estimated ensemble average $\bar{c}^m$, contains residual variance. Still, we conclude that the true bias is less than the displayed values of GAPSB and that the total error is predominantly due to noise.

In the platelet aggregation calculations, a Peclet number based on the blood-vessel radius is more appropriate than one based on platelet size. The former typically has a magnitude of approximately 1000. The remaining tests involve a Peclet number of about 1000 and also involve flow fields whose action is to elongate the initial distribution of chemical, in opposition to the action of diffusion.

For the second set of tests, we use $r_0 = 12.5 \times 10^{-4}$ cm,

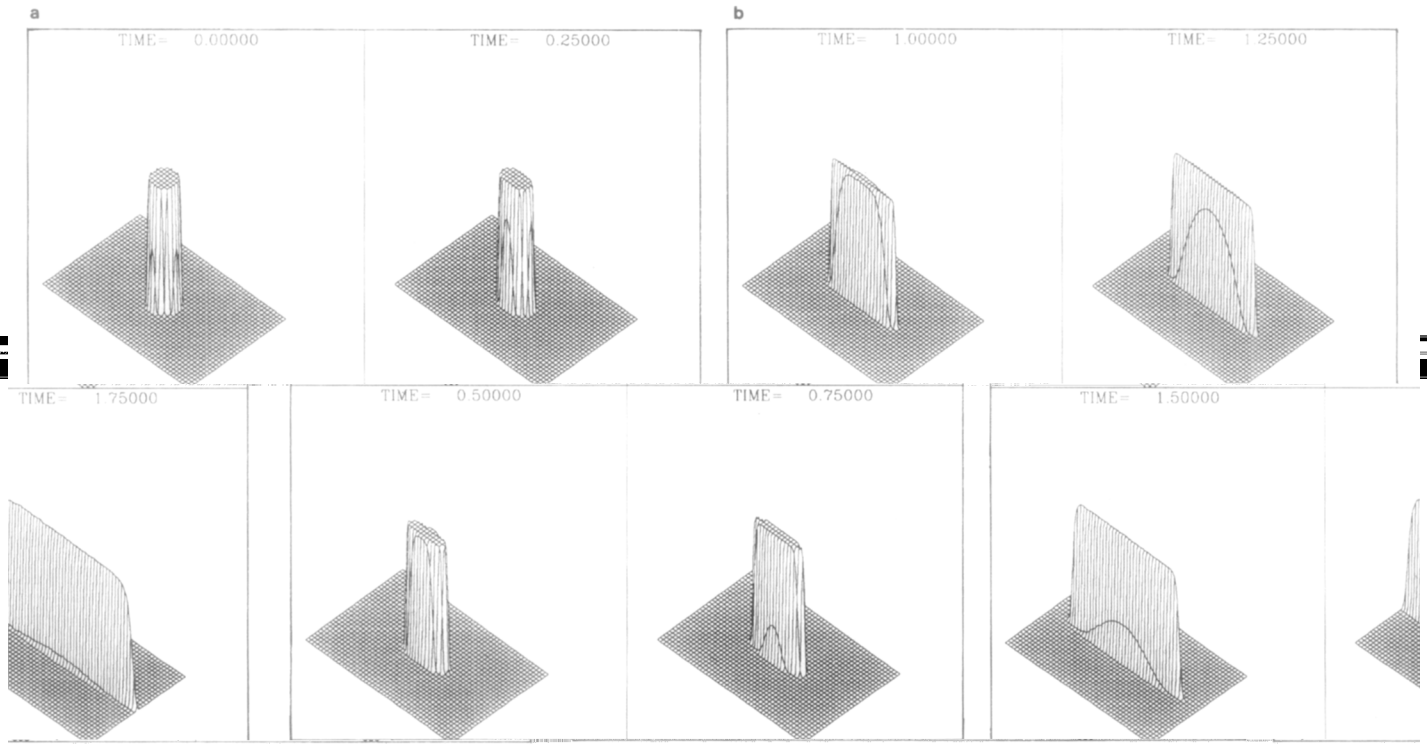

FIG. 3. "Gold standard" computed concentration at selected times. Flow is stagnation point flow. $Pe \approx 977$. Ensemble average of 20 runs with $N = 32{,}000$ and $\Delta t = 0.0125\tau_0$. Time is measured in seconds.

$\sigma_0 = 4 \times 10^{-7}$ cm^2/s, and the stagnation-point flow field $\mathbf{u}(x, y) = \tau_0^{-1}((x - x_0), -(y - y_0))$, with $\tau_0 = 1 \times 10^{-3}$ s. This gives a Peclet number $Pe = r_0^2/\tau_0\sigma_0 \approx 1000$. For these experiments, the reconstruction of c is carried out in a $4l_0 \times 2l_0$ rectangle centered on (x_0, y_0). The stagnation-point flow is assumed to obtain in the entire plane, and the duration of the experiments is limited so that no particles are carried close to the boundary of the rectangle used in the reconstruction. We know of no exact solution for this problem, so, for purposes of comparison, we define a "gold standard" approximate solution by forming the ensemble average of 20 experiments each with $N = 32{,}000$, $\Delta t = 0.0125\tau_0$, and $h = \frac{1}{64}l_0 = \frac{1}{16}r_0$.

In Fig. 3 we show surface plots of the gold-standard solution at time intervals of $0.25\tau_0$. In these calculations, the characteristic diffusion distance $(\sigma_0\tau_0)^{1/2}$ is less than h, and the computed results show more smoothing than this. However, the extra smoothing is *not* cumulative; it has extent $O(h)$, begins at timestep 0, and does not get worse with time. This contrasts strongly with finite-difference methods which introduce artificial diffusion whose effect accumulates with time. The ability of the current method to tightly constrain the extra smoothing comes about because the concentration field is constructed from the instantaneous positions for the particles $\{\mathbf{x}_k\}$, and the motion of these particles reflects the true diffusion coefficient. In fact, the method can be run down to zero diffusion coefficient and then gives a filtered view of the concentration field. These statements are illustrated in Fig. 4 which shows the position of the particles at selected times for $\sigma_0 = 0$ (left

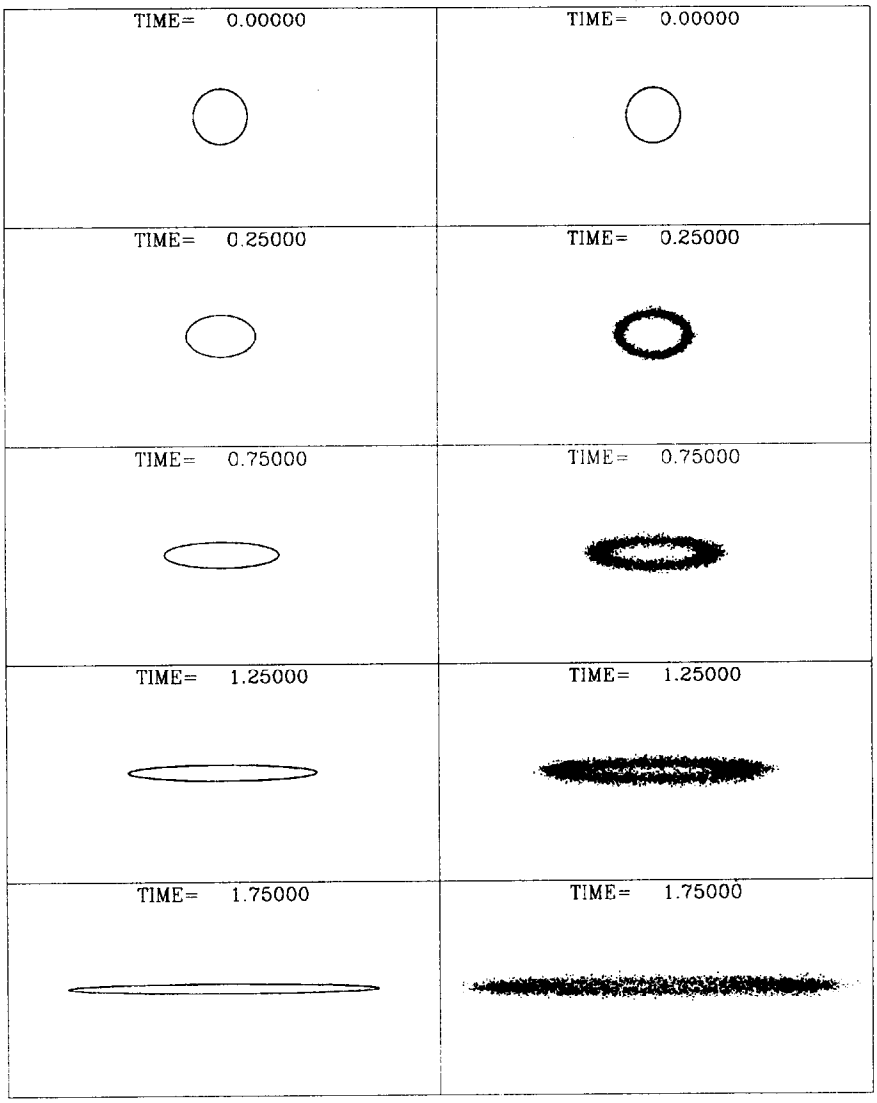

FIG. 4. Particle locations at selected times. Flow is stagnation point flow. Left panels depict case of no diffusion ($Pe = \infty$). Right panels show the case $Pe \approx 977$, $N = 16{,}000$, $\Delta t = 0.0125\tau_0$. Time is measured in seconds.

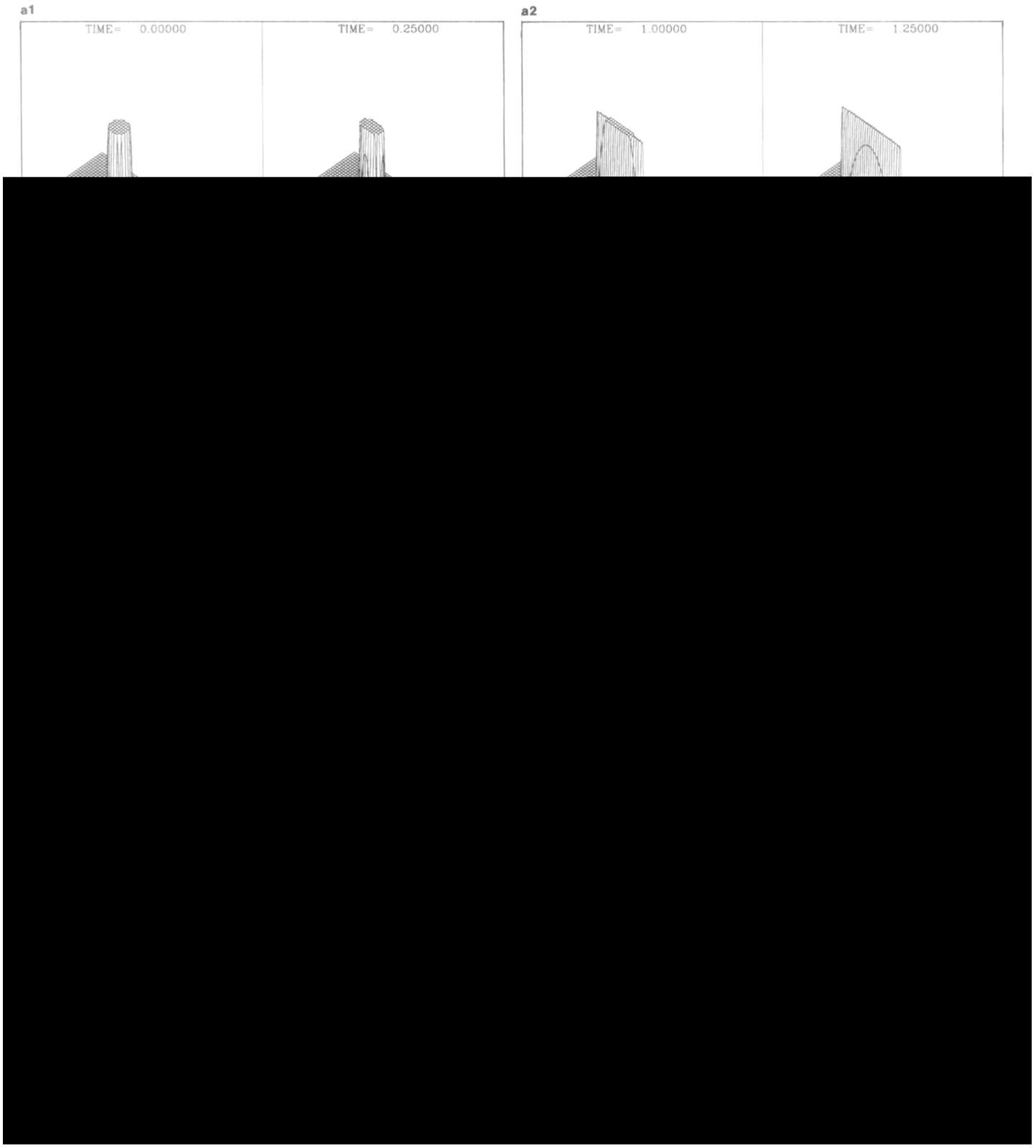

FIG. 5. (a) Computed concentration at selected times. Flow is stagnation point flow: $Pe = \infty$, $N = 16{,}000$, $\Delta t = 0.0125\tau_0$. Concentrations correspond to particle positions in left panels of Fig. 4. (b) Computed concentration at selected times. Flow is stagnation point flow: $Pe \approx 977$, $N = 16{,}000$, $\Delta t = 0.0125\tau_0$. Concentrations correspond to particle positions in right panels of Fig. 4.

panels) and $\sigma_0 = 4 \times 10^{-7}$ as above (right panels). Each of these experiments involves one run with 16,000 particles. In Fig. 5 we show surface plots of the respective reconstructed concentration fields from these two experiments. The increased width of the surfaces shown in the last pictures of Figs. 3 and 5b compared with that in Fig. 5a is a consequence of nonzero diffusion. The somewhat fuzzy character of the plots in Fig. 5b compared with the "gold-standard" surface shown in Fig. 3 reflects the different number of particles used in the two calculations (16,000 vs 640,000).

We have used the gold-standard solution to compute estimated errors in runs using various values of N and Δt. We note that these are estimates only of the errors due to N and Δt, not to h, because all of the calculations used the same value of h. Breakdown of the error into variance (GAPVAR) and estimated squared-bias (GAPSB) shows that the error is dominated by the variance and decreases as $N^{-1/2}$. The results show little sensitivity to variations in Δt.

In Fig. 6, we have plotted the l_1, l_2, and l_∞ norms of the estimated errors as functions of time, for an experiment with $N = 16{,}000$ and $\Delta t = 0.0125\tau_0$. Error curves for other values of N are very similar to those shown except for a vertical shift whose size reflects the $N^{-1/2}$ behavior of the errors. As seen from Fig. 6, the error increases approximately exponentially in time (but not with timestep at fixed t), a rate constant close to that governing the error growth (see Fig. 6). Recall from Section 2 that the norm of a gradient vector $\mathbf{g}_k$ grows in response to local stretching of the fluid in the direction perpendicular to the vector. Based on this, it would be natural to suppose that this growth magnifies the noise inherent in our use of a random walk and that this is the origin of the exponential growth in time of the variance seen in these calculations. However, closer examination suggests that this is *not* the case.

We modified the algorithm described above in the following way to limit growth in the size of $\|\mathbf{g}_k\|$. Let g_0 denote the maximum of $\|\mathbf{g}_k\|$ at $t = 0$, and let β denote a positive number. At the end of each timestep, compute $\|\mathbf{g}_k\|$ for each k. For any k for which $\|\mathbf{g}_k\| > \beta g_0$, let $p = \text{int}(\|\mathbf{g}_k\|/(\beta g_0)) + 1$, and partition the existing particle k into p particles each located at $\mathbf{x}_k$ and with gradient vector $\mathbf{g}_k/p$. By thus partitioning the particles, $\|\mathbf{g}_k\|$ is kept below βg_0 for all time. The new particles created when particle k is partitioned are initially located at the same point, but in subsequent timesteps their different random steps lead to their dispersal. In Fig. 7, we show the l_2 error for an experiment using the modified algorithm. For this experiment, we begin with $N = 16{,}000$ and set $\beta = 1$, so that the size of a gradient vector never exceeds its initial size (the sizes are initially uniform). By the time of the first point on the error curve, N has grown to approximately 34,000 and by the end of the

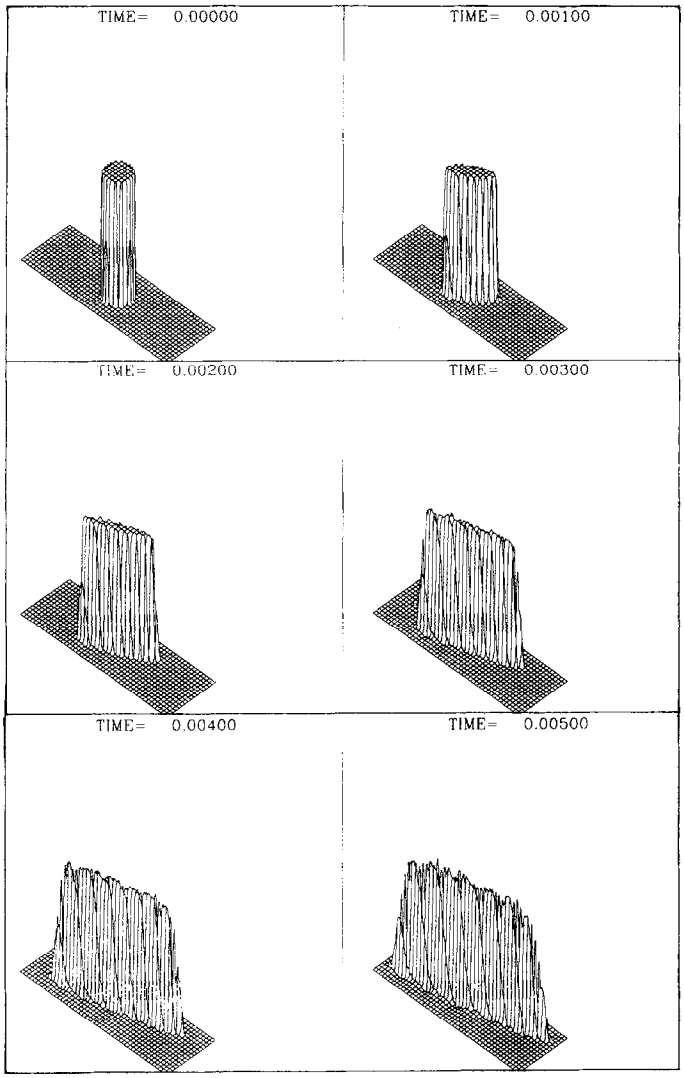

FIG. 10. Computed concentration at selected times. Flow is linear shear flow: $Pe \approx 977$, $N = 16{,}000$, $\Delta t = 0.0125\tau_0$. Time is measured in seconds.

5. CONCLUSIONS AND FUTURE DIRECTIONS

We believe that the numerical results presented above demonstrate that our method solves the two-dimensional convection-diffusion equation (1). The special strength of the method is its ability to accurately track convective transport and to simulate diffusive transport at the true diffusivity even when this diffusivity is very small. The method gives computed concentrations of moderate accuracy and precision for a moderate amount of computational work. The computational work rises rapidly as higher accuracy and precision for a moderate amount of computational work. The computational work rises rapidly as higher accuracy and precision are required. This is reflected in the $O(N^{-1/2})$ dependence of the method's errors and is similar to the behavior of other Monte Carlo methods (see, for example, [14, 16, 17]). We will explore the possibility of adapting to our method variance reduction techniques such as those introduced by Chorin [5] and extended to random-vortex calculations by Chang [4]. For a prescribed level of error, these techniques may reduce substantially the method's computational costs. We will also explore the possibility of using a grid-free method to solve the Poisson equations (3)–(4) in order to reduce the small amount of smoothing currently associated with the reconstruction process. In this regard, Greengard and Rokhlin's [11] multipole-expansion algorithm offers much promise; it provides for very efficient calculation of the potential and/or its derivatives due to a large collection of point charges. We may use this algorithm to find the functions ϕ_x and ψ_y needed in Eq. (5) without first having to determine ϕ and ψ, and without using difference approximations to their derivatives.

ACKNOWLEDGMENTS

The author is grateful to Professors Charles Peskin, Alexandre Chorin, and James Keener for helpful discussions on the subject of this paper. This work was supported, in part by the National Science Foundation under Grants DMS86-02166 and DMS88-03482, in part by an Alfred P. Sloan Foundation Fellowship, and in part by a grant from the University of Utah Research Fund. Computations were performed at the San Diego Supercomputing Center and the University of Utah's Center for Scientific Computing.

REFERENCES

1. C. R. Anderson, *J. Comput. Phys.* **61**, 417 (1985).
2. N. Bressan and A. Quarteroni, *SIAM J. Numer. Anal.* **23**, 1138 (1986).
3. C. C. Chang, *Math. Comput.* **49**, 523 (1987).
4. C. C. Chang, *J. Comput. Phys.* **76**, 281 (1988).
5. A. J. Chorin, *J. Comput. Phys.* **8**, 472 (1971).
6. A. J. Chorin, *J. Fluid Mech.* **57**, 785 (1973).
7. A. J. Chorin and J. E. Marsden, *A Mathematical Introduction to Fluid Mechanics* (Springer Verlag, New York, 1979), p. 110.
8. G. Dahlquist and A. Bjorck, *Numerical Methods* (Prentice–Hall, Englewood Cliffs, NJ, 1974), p. 453.
9. A. L. Fogelson, *J. Comput. Phys.* **56**, 111 (1984).
10. A. F. Ghoniem and F. S. Sherman, *J. Comput. Phys.* **61**, 1 (1985).
11. L. Greengard and V. Rohklin, *J. Comput. Phys.* **73**, 325 (1987).
12. D. G. Long, unpublished.
13. E. L. Oran and J. P. Boris, *Numerical Simulation of Reactive Flow* (Elsevier, New York, 1987), p. 105.
14. S. Roberts, *J. Comput. Phys.* **58**, 29 (1985).
15. P. N. Swartzrauber, *SIAM Rev.* **19**, 490 (1977).
16. A. S. Sherman and C. S. Peskin, *SIAM J. Sci. Stat. Comput.* **7**, 1360 (1986).
17. A. S. Sherman and C. S. Peskin, *SIAM J. Sci. Stat. Comput.* **9**, 170 (1988).