SIAM J. SCI. COMPUT.
Vol. 15, No. 6, pp. 1280–1293, November 1994

A GRADIENT RANDOM WALK METHOD FOR TWO-DIMENSIONAL REACTION-DIFFUSION EQUATIONS*

ARTHUR SHERMAN† AND MICHAEL MASCAGNI†‡

Abstract. An extension to two space dimensions of the gradient random walk algorithm for reaction-diffusion equations is presented. This family of algorithms is related closely to the random vortex method of computational fluid dynamics. Although the computational cost is high, the method has the desirable features of being grid free and of automatically adapting to the solution by concentrating elements where the gradient is large. In addition, the method can be extended easily to more than two space dimensions. A key feature of the method is discretization in terms of the dependent, rather than independent, variable, giving it features in common with Lagrangian particle methods. The method is derived here and its application to some simple reaction-diffusion wave propagation problems is illustrated.

Key words. Monte Carlo method, reaction-diffusion equation, gradient transport, grid-free, adaptive, N-body problem

AMS subject classifications. 65C05, 35K57, 65M99, 31C20

1. Introduction.

We are interested in numerical methods for solving reaction-diffusion equations:

$$u_t = \Delta u + f(u), \tag{1.1}$$

$$u = u(\mathbf{x}, t), \qquad \mathbf{x} \in \mathbb{R}^d.$$

We will focus mostly on the initial-value problem,

$$u(\mathbf{x}, 0) = u_0(\mathbf{x}),$$

in two space dimensions. Our goal is a method that will work well on problems that are difficult for finite-difference methods, such as cases where the solution has sharp gradients. For this reason, we consider a particle method in which computational elements representing the gradient of the solution move by diffusion and are modeled by random walks. The method is grid free and automatically adapts to the solution. We derive the method as the natural extension of a one-dimensional (1-D) Monte Carlo method [27]. Here we report preliminary computational studies on some simple model problems in order to outline the main features of the method and gain computational experience to guide future work.

An early time-dependent Monte Carlo method was proposed for the heat equation and was based on a stochastic interpretation of the explicit finite-difference equations [8]. This method used random walks on spatial grids, but the extension to grid-free walks was easy because the fundamental solution of the heat equation is Gaussian. These ideas remained mostly of theoretical interest due to the large variance of the methods. A step toward improving the accuracy was to have the density of diffusing elements represent the *gradient* of the solution, rather than the solution itself. Integrating to obtain the solution reduces the variance considerably. This lead to the coining of the term "gradient random walk" (GRW) [11].

The first of the GRW methods was Chorin's random vortex method (RVM) for incompressible two-dimensional (2-D) fluid flow [5]. Subsequently, Chorin proposed the "random

*Received by the editors February 15, 1993; accepted for publication (in revised form) September 14, 1993.

†National Institutes of Health, National Institute of Diabetes and Digestive and Kidney Diseases, Mathematical Research Branch, Bethesda, Maryland 20892 (sherman@helix.nih.gov).

‡Supercomputing Research Center, Institute for Defense Analyses, Bowie, Maryland 20715-4300 (mascagni@super.org, na.mascagni@na-net.ornl.gov).

element method" [7] for 1-D reaction-diffusion equations, which models diffusion via random walk and reaction via deterministic particle growth and decay. This method was studied further by Ghoniem and colleagues [21], [11], both in its own right and as a model for the vortex sheet algorithm [6]. Hald [14], [15] proved convergence for several simplified versions of the Chorin–Ghoniem algorithm, and Puckett [22] proved convergence of the full method for the Fisher/Kolmogorov–Petrovskii–Piskunov equation. Whereas Puckett's estimate of the error due to the Monte Carlo particle-based discretization was $O(N^{-1/4})$, where N is the number of particles, he presented numerical evidence that the error is actually $O(N^{-1/2})$. Sherman and Peskin [27] developed a variant of the Chorin–Ghoniem method, which was purely stochastic, and applied it to a simplification of the Hodgkin–Huxley nerve conduction equations [28]. Chauvin and Rouault [4] have proved the convergence of this variant.

The extension of the GRW method to higher dimensions began when Anderson [1] showed how to recover a scalar field in two dimensions from a δ-function representation of its gradient in a pure convection problem. Anderson's method relied on representing the solution as the Laplacian of the solution convolved with the Green function. Integration by parts gave an algorithm for the recovery of the solution at discrete points from this point-gradient representation. Sherman and Peskin [28] sketched how to apply Anderson's approach to reaction-diffusion equations. Fogelson [10] used ideas similar to Anderson's on a convection-diffusion problem, but instead of point recovery he used a fast Poisson solver to recover the function values.

In related work, Russo [24]–[26], Raviart [23], and others [16], [9] proposed particle methods for collisional equations. Their methods are deterministic, but share the Lagrangian feature of the GRW discretization.

Although the problem of large variance for the Monte Carlo method has not yet been solved, and these methods have not displaced finite-difference methods, they have stimulated much theoretical work.

A major motivation for studying Monte Carlo methods in this context is the deep connection between reaction-diffusion equations and stochastic processes. It is well known that certain functionals of Brownian motion have expected values that solve reaction-diffusion equations [18]. Moreover, the microscopic phenomenology of the chemical processes described by reaction-diffusion equations are based on the Brownian motion of the reacting species. Thus, it is intellectually appealing to search for numerical methods that share features in common with these fundamental viewpoints.

In addition to the connection between probability theory and partial differential equations (PDEs), the possibility of grid-free methods, especially for multidimensional problems, remains alluring. In this spirit we present the extension of the GRW algorithm for reaction-diffusion equations to two (or more) space dimensions and describe some of its computational properties.

In §2 we briefly review the 1-D GRW methods and derive their extension to two dimensions. In §3 we apply the method to track a traveling wavefront generated by Nagumo's equation without recovery. We illustrate two initial-value problems in the plane and an initial-boundary-value problem in a half-space. In §4 we summarize the results and discuss open problems and possible directions for future work.

2. Derivation. We first review the GRW method [7], [11], [27] for (1.1) with $d = 1$:

$$u_t = u_{xx} + f(u), \qquad u(x,0) = u_0(x). \tag{2.1}$$

We assume $u(-\infty, t) = 0$, $u(+\infty, t) = 1$. The strategy is to represent $v = u_x$ by diffusing particles. Differentiating (2.1) with respect to x,

$$v_t = v_{xx} + f'(u)v, \qquad v(x,0) = u_0'(x), \tag{2.2}$$

and discretizing v as a sum of δ-functions with strength m_j, we have

$$v(x,t) = \sum_{j=1}^{N} m_j \delta\left(x - X_j(t)\right), \tag{2.3}$$

where $X_j(t)$, $j = 1, \ldots, N$ represent the location of N particles. We use capital X to indicate that the positions are random variables. The density of the particles determines the value of v, and heuristically one thinks of m_j as the "mass" of the jth particle.

We recover u from v by $u(x,t) = \int_{-\infty}^{x} v(x',t)\,dx'$:

$$u(x,t) = \sum_{j=1}^{N} m_j H\left(x - X_j(t)\right). \tag{2.4}$$

Thus u is represented as a step function with jumps of size m_j at X_j. If all the particles have the same mass, m, the value of u at x is m times the number of particles that lie to the left of x. The boundary condition at $-\infty$ is automatically satisfied, and the condition at $+\infty$ is satisfied on average with fluctuations [27]. This corresponds to conservation of mass.

As described in [11], there is much less noise in the computed value of u than of v because all the particles contribute to the value at any point x. Moreover, if the particles have equal mass, their density is large precisely where u has large gradients.

Once the initial data is discretized, the GRW method evolves the particle positions and masses such that u satisfies (2.1). This is done by a fractional step iteration in which the diffusion term is modeled by a random walk and the reaction term is modeled as exponential growth or decay of the particle masses. For each timestep the sequence is as follows.

- Gaussian random walk step:

$$X_j(t + \Delta t) = X_j(t) + \sigma_j, \tag{2.5}$$

  where the σ_j are independent $N(0, 2\Delta t)$ random variables.
- Evaluate $u_j = u(X_j(t + \Delta t))$, $j = 1, \ldots, N$ using (2.4).
- Kill or replicate particles with probability $|f'(u_j)|\Delta t$:
  1. Kill particle if $f' < 0$;
  2. Create a new particle at X_j if $f' > 0$.

Note that the only way the particles interact with each other, and hence the only way that the nonlinearity of the equations is manifest in the algorithm, is that the local value of u depends on the positions of all the particles. Not surprisingly, this is the step that is most difficult computationally. Naively, computing (2.4) for all the X_j's requires $O(N^2)$ work, but if the particles are sorted, it requires only $O(N)$ work plus $O(N \log N)$ for the sorting.

The Chorin–Ghoniem method [7], [21], [11] is essentially the same except that instead of killing and replicating particles, the masses are increased or decreased according to the ordinary differential equation

$$\frac{dm_j}{dt} = f'(u_j). \tag{2.6}$$

In the mean and to $O(\Delta t)$ the two procedures are equivalent.

Both methods have been used to solve traveling-wave problems in one dimension. The mechanism of wave propagation is transfer of mass from behind the front to ahead of it. With randomized particle death and replication, particles are killed off behind the wavefront, where f' is mostly negative, and replicated ahead of the front, where f' is mostly positive. With

deterministic particle growth and decay, there is a graded transmission of mass from neighbor to neighbor.

We now generalize the method to two space dimensions. The key issue is how to recover u from its gradient, represented by particles. For this we follow a suggestion of Anderson [1] to employ Poisson's formula [17]:

$$u(\mathbf{x}, t) = \int G(\mathbf{x} - \mathbf{x}')\Delta u(\mathbf{x}', t)\, d\mathbf{x}', \tag{2.7}$$

where G is the fundamental solution of the Laplacian in $\mathbb{R}^2$:

$$G(\mathbf{x}) = \frac{1}{2\pi} \log |\mathbf{x}|. \tag{2.8}$$

Integrating by parts,

$$u(\mathbf{x}, t) = \int \nabla G(\mathbf{x} - \mathbf{x}') \cdot \nabla u(\mathbf{x}', t)\, d\mathbf{x}'. \tag{2.9}$$

As in one dimension, we represent ∇u as a sum of δ-functions:

$$\nabla u(x, y, t) = \sum_{j=1}^{N} m_j \delta\left(x - X_j(t), y - Y_j(t)\right) \mathbf{n}_j. \tag{2.10}$$

Now, in addition to position and mass, each particle has a unit vector, $\mathbf{n}_j = (\xi_j, \eta_j)$, representing the direction of ∇u at (X_j, Y_j).

Substituting (2.10) in (2.9) we obtain

$$u(x, y, t) = \frac{1}{2\pi} \sum_{j=1}^{N} \frac{\mathbf{r}_j \cdot m_j \mathbf{n}_j}{|\mathbf{r}_j|^2}, \tag{2.11}$$

where $\mathbf{r}_j = (x - X_j, y - Y_j)$. The sum (2.11) is very similar to that found in the RVM [5]. As in the RVM, if $(x, y) = (X_k, Y_k)$, the kth term is excluded from the sum, and a smoothing procedure is needed to avoid numerical instability when $(x, y) \approx (X_k, Y_k)$. See §3.

Although this method of recovering u from its gradient appears to be very different from the 1-D method, it is actually a natural generalization. In $\mathbb{R}^1$, with $G_{xx}(x - x') = -\delta(x - x')$, then $G_x(x - x') = H(x' - x)$, so substituting (2.3) into the 1-D equivalent of (2.9) gives exactly (2.4). From this point of view, one can reinterpret the positive and negative masses used to represent nonmonotonic functions in [28] as particles with orientation $+1$ for up-jumps and -1 for down-jumps.

We can extend the analogy a little further. In two dimensions one can represent u as a step function by constructing a contour plot and replacing u by a function that jumps in value whenever a contour is crossed. Thus we need to be able to represent a function that is 0 outside a closed curve Γ and 1 inside. This is accomplished by placing N particles along Γ, oriented along the outward normal with mass

$$m = \frac{\text{length}(\Gamma)}{N}. \tag{2.12}$$

Then (2.11) is an approximation to the line integral

$$\frac{1}{2\pi} \oint_{\Gamma} \nabla G \cdot \mathbf{n}\, ds. \tag{2.13}$$

Applying the divergence theorem, (2.13) evaluates to 0 if (x, y) is outside Γ and 1 if (x, y) is inside. Equation (2.12) says that each particle represents an oriented arc with a given length and implies that the larger the mean radius of Γ, the more particles of a given mass are required to represent a jump of given height across Γ. In $\mathbb{R}^3$, particles would represent oriented patches of surface area, and so on for arbitrary dimension.

Figure 2.1 shows the u-surface recovered from placing 1, 2, 4, and 10,000 particles equiangularly on the circle of radius 10. These figures illustrate that the particles can be thought of as oriented steps only in a collective sense: Each one individually is a singular dipole, and only by cancellation can a simple step be constructed.

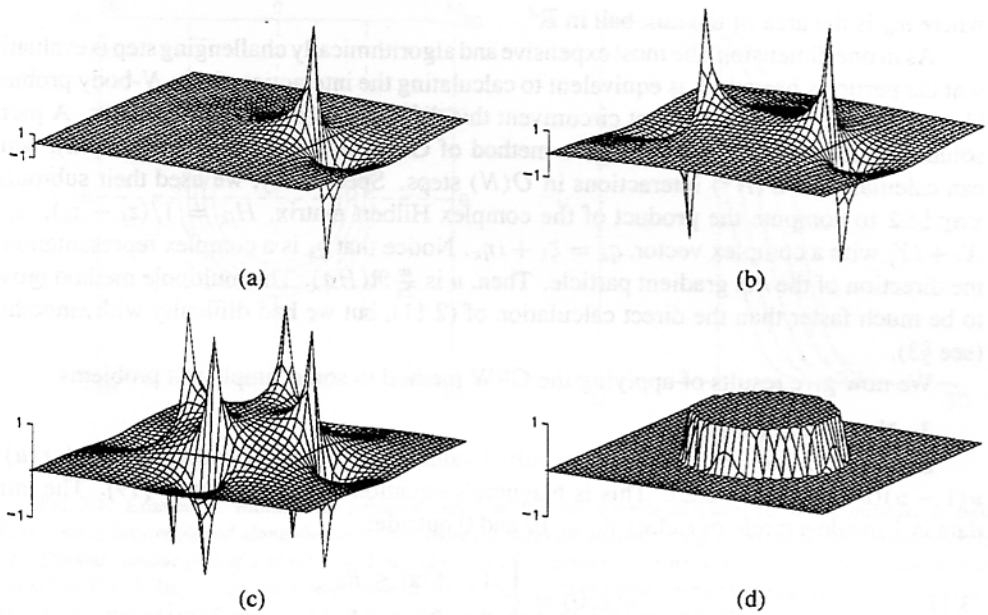

FIG. 2.1. *Representing a step with gradient particles: u-surface recovered using* (2.11) *without smoothing from* (a) 1, (b) 2, (c) 4, *and* (d) 10,000 *particles equally spaced around a circle of radius* 10. *Note changes of scale: u-axis scales for* (a), (b), (c), *and* (d), *are in the ratio* 1:2:4:4. *The grid covers the square from* $(-20, -20)$ *to* $(20, 20)$.

Moreover, the interpretation of the gradient particles as steps is only valid at $t = 0$; after the first random walk step the particles will not lie on any meaningful curve. The Poisson formula representation, (2.11), is more general than (2.13), however, and continues to apply to any collection of particles in the plane.

After discretizing the initial data, one uses the same fractional step iteration as in $\mathbb{R}^1$, replacing (2.5) with independent Gaussian steps for X_j and Y_j:

- Gaussian random walk step:

$$X_j(t + \Delta t) = X_j(t) + \sigma_j^x,$$

$$Y_j(t + \Delta t) = Y_j(t) + \sigma_j^y,$$

  where σ_j^x and σ_j^y are independent $N(0, 2\Delta t)$ random variables.
- Evaluate $u_j = u(X_j(t + \Delta t), Y_j(t + \Delta t))$, $j = 1, \ldots, N$ using (2.11).
- Kill or replicate particles with probability $|f'(u_j)|\Delta t$:
  1. Kill particle if $f' < 0$;

2. Create a new particle at (X_j, Y_j) if $f' > 0$.

The differences between this 2-D algorithm and the 1-D algorithm given above are that: (1) a 2-D random walk is performed and (2) the Green function of the Laplacian in two dimensions is used to compute u on the particles using (2.11). To extend this to d-dimensions we merely use a d-dimensional random walk and substitute the Green function of the d-dimensional Laplacian in (2.11) to perform the recovery:

$$u(\mathbf{x}, t) = \frac{1}{\omega_d} \sum_{j=1}^{N} \frac{\mathbf{r}_j \cdot m_j \mathbf{n}_j}{|\mathbf{r}_j|^d}, \tag{2.14}$$

where ω_d is the area of the unit ball in $\mathbb{R}^d$.

As in one dimension, the most expensive and algorithmically challenging step is evaluating u at the particles because it is equivalent to calculating the interactions in an N-body problem. Unlike the 1-D case, one cannot circumvent this difficulty by sorting the particles. A partial solution is to apply the fast multipole method of Greengard and Rokhlin [13], [12], which can calculate the $O(N^2)$ interactions in $O(N)$ steps. Specifically, we used their subroutine `rapif2` to compute the product of the complex Hilbert matrix, $H_{jk} = 1/(z_j - z_k)$, $z_j = X_j + iY_j$ with a complex vector, $q_k = \xi_k + i\eta_k$. Notice that q_k is a complex representation of the direction of the kth gradient particle. Then, u is $\frac{m}{2\pi}\Re(Hq)$. The multipole method proved to be much faster than the direct calculation of (2.11), but we had difficulty with smoothing (see §3).

We now give results of applying the GRW method to some simple test problems.

3. Numerical examples.

Example 1. We begin with a pure initial value problem: (1.1) with $d = 2$ and $f(u) = u(1-u)(u-a)$, $a = 0.25$. This is Nagumo's equation without recovery [19]. The initial data is 1 inside a circle of radius $R_0 = 10$ and 0 outside:

$$u(\mathbf{x}, 0) = \begin{cases} 1 & \text{if } |\mathbf{x}| \le R_0, \\ 0 & \text{if } |\mathbf{x}| > R_0. \end{cases} \tag{3.1}$$

With this initial data the solution takes the form of an "excited region" (where $u \approx 1$), which expands radially and asymptotically approaches a traveling wave as the radius $\to \infty$.

To solve this problem, we place $N = 10,000$ particles, each with mass $2\pi R_0/N$, on the circle of radius $R_0 = 10$ at $T = 0$ and advance up to $T = 25$ with a timestep of $\Delta t = 0.1$. The initial data is shown in Fig. 2.1(d), and Figs. 3.1(a), (b) show u as a stacked contour plot at $T = 5$ and 25. Figure 3.1(c) shows the expansion of the $u = 0.5$ contour in time. It retains approximate cylindrical symmetry although this is not imposed in the algorithm. Although we have evaluated the solution on a rectangular grid for display purposes, the particles cluster in a narrow band around the $u = 0.5$ contour (Fig. 3.4(c), (d), Example 3).

As indicated in §2, it was necessary to smooth the singularity in (2.11) when particles approached each other closely, or when evaluating the solution at a grid point near a particle. We thus replaced (2.11) with

$$u(x, y, t) = \frac{m}{2\pi} \sum_{j=1}^{N} \frac{\mathbf{r}_j \cdot \mathbf{n}_j}{\max(|\mathbf{r}_j|^2, \varepsilon^2)}. \tag{3.2}$$

The choice $\varepsilon = 0.3$ gave good results.

For comparison with the exact solution we solved the radial problem,

$$u_t = u_{rr} + \frac{1}{r} u_r + f(u), \tag{3.3}$$

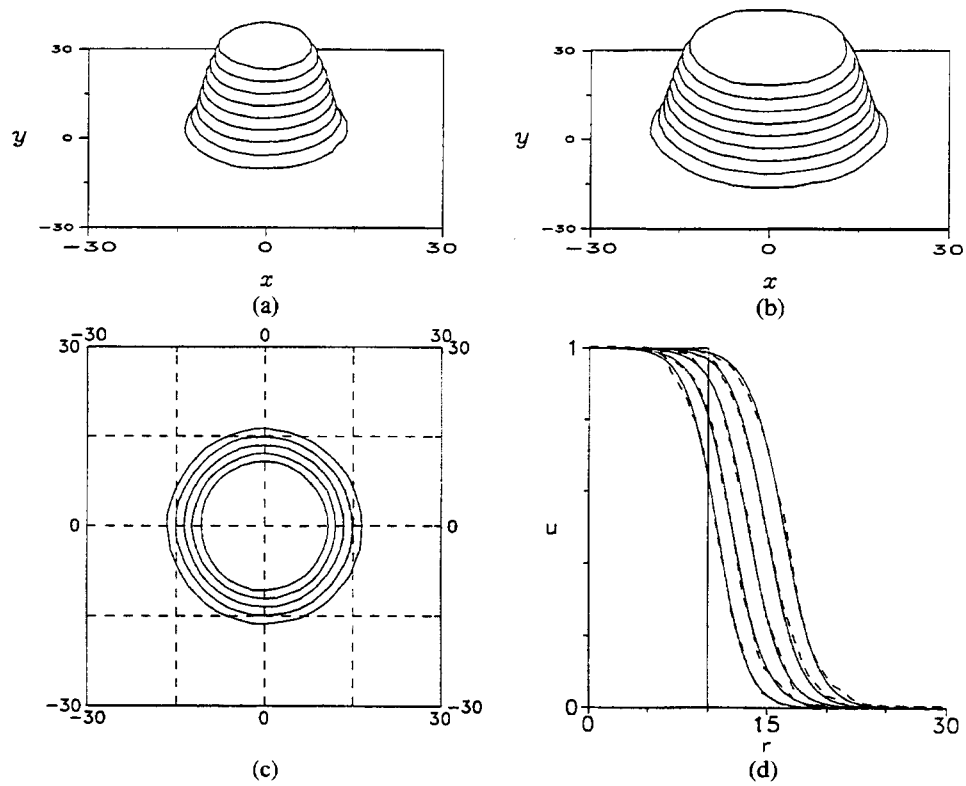

FIG. 3.1. *Example* 1. *Initial value problem with* $u = 1$ *inside a circle of radius* 10, *and* 0 *outside. Initially* 10,000 *particles are placed along the circle, and more particles are automatically added as the front expands (Fig.* 3.2). *Stacked contour plot of* u *at* (a) $T = 5$ *and* (b) $T = 25$. *Contour levels are* 0.1, 0.2, ..., 0.9. (c) *Contour for* $u = 0.5$ *at* $T = 5, 10, \ldots, 25$. (d) *Comparison at* $T = 0, 5, \ldots, 25$ *of* GRW *profiles (dashed) along ray* $\theta = 0$ *with the "exact" solution (solid) computed by a finite difference method for the radial* (3.3).

with a finite-difference method using a small uniform mesh. Figure 3.1(d) shows the wave profile of the GRW solution as compared to the finite-difference solution along a ray. Although the shape of the profile fluctuates, the wave speed is computed accurately.

As in the 1-D algorithm, the propagation of the wave is the result of transfer of mass. In one dimension, total particle mass and particle number, N, are conserved, so that, in effect, particles jump across the front. In two dimensions, total mass is still conserved (u remains near 1 at the origin), but particle number grows with time. Since in a radially symmetric traveling wave, the circumference of the front grows linearly, (2.12) suggests that N must also increase linearly. After a transient, during which the shape of the profile relaxes, linear growth is observed (Fig. 3.2). Note that N increases faster than the average radius, so that the effective arc-length per particle decreases with time. We believe that this is because the particles lose their radial orientation as they diffuse. If at each timestep the particles are reoriented to point radially outward, then N grows at the same rate as the radius (not shown).

The calculation for Fig. 3.1 was programmed in C on a Cray YMP and required about 1.5 CPU hours, virtually all of which was spent computing the readily vectorized N-body sum 3.2. The same calculation took only one-fifth as long using the fast multipole FORTRAN subroutine, `rapif2`, even though this mostly did not vectorize. However, this level of performance was attainable only if no smoothing was done, and the resulting solutions (not shown) suffered from large spike-like errors. Smoothing is implemented by calling a close approach subroutine,

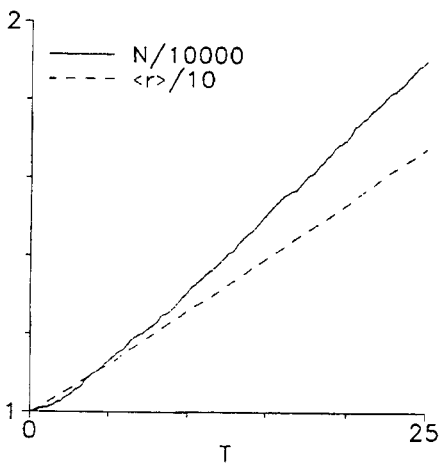

FIG. 3.2. *Growth of particle number with time. Solid curve: The number of particles, normalized by the initial number. Dashed curve: The average radius of the wave (computed as the simple average of the radii of the particles), normalized by the initial radius. Both grow approximately linearly, but N grows faster. Thus, the effective arc-length per particle declines slowly.*

which computes the interactions directly when $|\mathbf{r}| < \varepsilon$. This degrades performance: with $\varepsilon = 0.3$ instability was avoided, but `rapif2` actually took longer than the direct N-body calculation, presumably because the subroutine call inhibited vectorization.

Example 2. The next example addresses the question of what happens when the shape of the excited region changes in time. Fogelson [10] used the same discretization as in Example 1 to represent a circular step of concentration around a localized source, which stretches and elongates into an ellipse due to convection and diffusion. In the case of pure convection he showed that the particles would remain on and normal to the boundary of the region, Γ, because it is the material curve separating regions of high and low concentration. The particles rotate because of the derivatives of the convective term. There are no terms in reaction-diffusion that can turn the normals, which raises the question of how new emerging directions can be represented.

Reconsider Example 1, but with the excited region initially a square. Asymptotically, the square again evolves into a circular traveling wave. Figure 3.3(a) shows the contours for $u = 0.5$ at $T = 5, 10, \ldots, 25$. The shape of the front transforms from a square to a circle. The calculation is in reasonable agreement with results of a 2-D finite-difference method (not shown). The curved front is represented by linear combinations of the four existing directions in the particle population. Diffusion combines with growth and decay of particles to produce, on average, the appropriate mix of particles in the required locations. Note that this depends on the linearity of (2.11), which is inherited from (2.9). Figure 3.3(b) shows the distribution of particle directions (actually the difference between the argument of the particle and the direction of its normal) at $T = 25$, when the front is approximately circular. The distribution closely approximates a cosine, which is sufficient to represent a circular front.

Example 3. In order to illustrate a simple example with boundary conditions we solved (1.1) on the half-space $x > 0$, appending the boundary condition $u_x(0, y) = 0$. This is done by the method of images: a particle at (X_j, Y_j) with orientation (ξ_j, η_j) has an image particle at $(-X_j, Y_j)$ with orientation $(-\xi_j, \eta_j)$. The sum (2.11) is modified to account for the influence of the image particles on the real particles; it is of course not necessary to compute u at the image particles. This calculation is equivalent to two excited regions coalescing, and we have

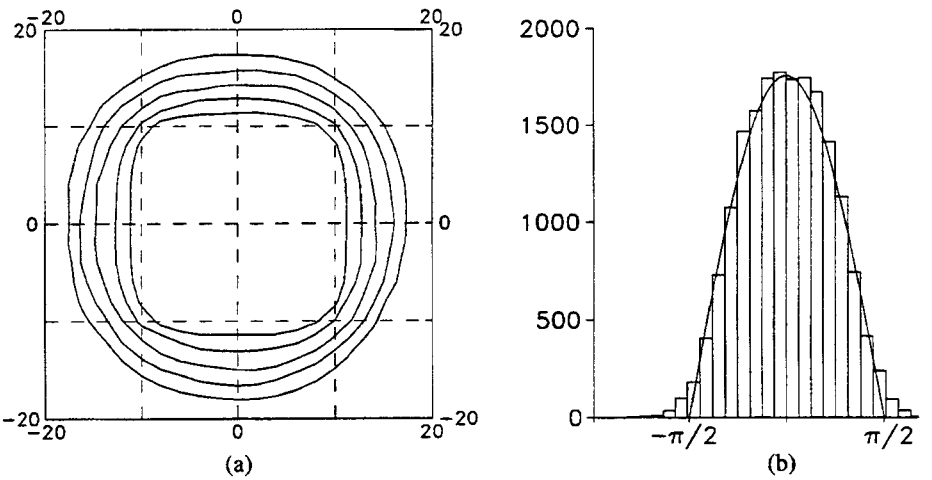

FIG. 3.3. *Example* 2. *Initial value problem with the initial excited region a square circumscribing a circle of radius* 10. (a) *Contours for* $u = 0.5$ *at* $T = 5, 10, \ldots, 25$ *for the* GRW *method.* (b) *Histogram (numbers of particles) versus* $\tan^{-1}\frac{y}{x} - \tan^{-1}\frac{\eta}{\xi}$; *that is, the difference between the argument of the particle and the direction of its normal. The distribution approximates a scaled cosine, shown superimposed.*

displayed it as such in Fig. 3.4. Figure 3.4(a) shows the $u = 0.5$ contours as a function of time, indicating the two regions expanding and merging into one large region. Figure 3.4(b) is a surface plot at $T = 15$. Figures 3.4(c), (d) show the particle positions in the plane to highlight how the method adapts to the solution: The cloud of particles is confined to a narrow band around the front, and, as the regions merge, the particles along the y-axis, which are no longer needed to resolve the solution, are killed off.

Example 4. As a final example, we indicate how to represent more general initial conditions. Consider the heat equation,

$$u_t = \Delta u,$$

with Gaussian initial data and solution

$$u(x, t) = \frac{1}{4\pi(\sigma + t)} e^{-(x^2+y^2)/4(\sigma+t)}.$$

In this example we choose $\sigma = 4$.

We discretize with uniform step size δu along the u direction and construct the corresponding level sets Γ_i (here, circles). By (2.12), each curve Γ_i should contain particles with combined mass

$$M_i = \text{length}(\Gamma_i)\delta u = 2\pi R_i \delta u.$$

The total mass of all the level sets $M = \sum M_i$ approximates the volume under $u(x, 0)$. The particle representation can be uniquely specified by distributing N particles with equal mass, $m = M/N$. Thus, Γ_i gets $N_i = \lfloor M_i/m \rfloor = \lfloor (NM_i/M) \rfloor$ particles. To avoid fractional particles, the mass per particle on Γ_i can be adjusted slightly: $m_i = M_i/N_i$. An alternative is to use random roundoff. The particles are spaced uniformly in arc-length (here, equiangularly) and have the direction of the gradient, which is the outward normal to Γ_i (here, radial).

An alternative method is to distribute particles on a rectangular grid. With uniform spacing h, total mass equal to $|\nabla u(x_i, y_j)|h^2$ is placed at the grid point (x_i, y_j) and subdivided into

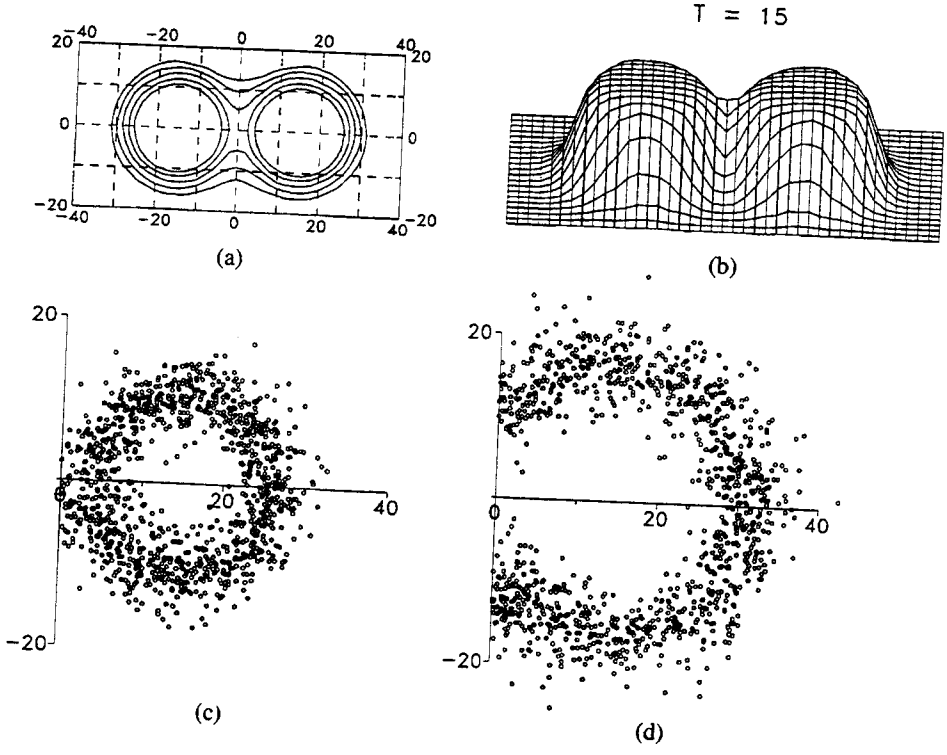

FIG. 3.4. *Example* 3. *Expanding front in the half space* $x > 0$ *with* $u_x(0, y) = 0$, *solved by using image particles.* (a) *Contours for* $u = 0.5$ *at* $T = 5, 10, \ldots, 25$. (b) *Surface plot of* u *at* $T = 15$. (*Plots reflect the influence of both the real particles and their images.*) *Positions of the actual particles* (*only* $\frac{1}{10}$ *of the particles are shown for clarity*) *at* (c) $T = 5$ *and* (d) $T = 25$, *showing that the particles follow the wave front, and that the superfluous particles along* $x = 0$ *are eliminated automatically.*

particles, each with direction $-\nabla u(x_i, y_j)$. This is analogous to initialization in the RVM when an initial vorticity distribution is given. This alternative is easy to implement for our example because ∇u is known analytically, but the level set method is more natural and works better given the symmetry of the problem. One problem with this approach arises when the initial function is given in tabular form. The numerical derivative required to compute the initial gradient reduces the numerical accuracy of the initial conditions passed to the GRW by $O(h^{-1})$.

In the level set approach there are two free numerical parameters, the number of level curves, and the total number of particles; these determine the number of particles per level curve. One must then balance coverage in the radial and azimuthal directions. We have not studied the optimal tradeoff. Figure 3.5(a) shows the representation of the initial data with 100 level curves and initial $N = 10{,}000$ (9948 after adjustments), and Fig. 3.5(b) shows the exact and computed profiles along a ray at $t = 0$ and $t = 4$.

4. Issues for future consideration. We have taken a step toward our goal of a grid-free, adaptive method for reaction-diffusion equations in two or more space dimensions by generalizing a family of 1-D random gradient methods [27], [28], [7], [21], [11]. Example 3 shows spatial adaptivity: The particles are concentrated around the steep gradient of the wavefront (Fig. 3.4(c)) and, when the gradient vanishes due to interaction with a boundary (or another expanding excited region), the particles defining that region die off (Fig. 3.4(d)). Example 2

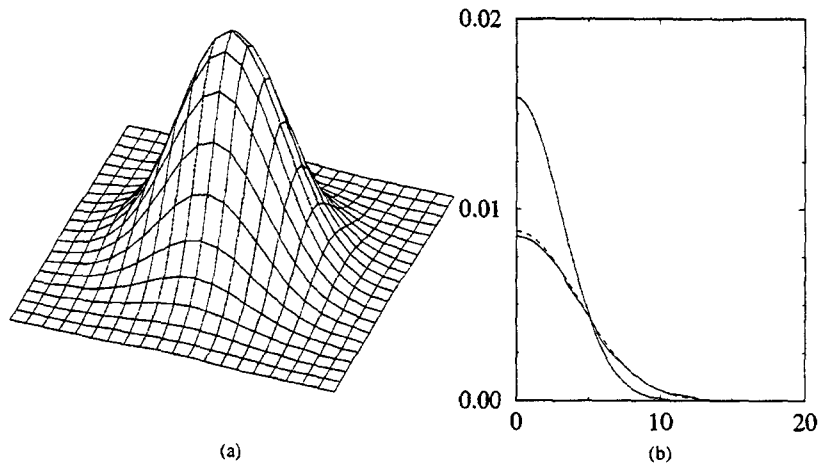

FIG. 3.5. *Example* 4. *Solution of the heat equation with Gaussian initial data.* (a) *Initial surface represented by radially oriented particles along* 100 *level curves uniformly spaced in the u direction. A total of* 9948 *particles were used.* (b) *Comparison of exact (solid) and computed (dashed) profiles along the positive x-axis at time* $t = 0$ *(upper curves) and* $t = 4$.

demonstrates adaptivity with respect to particle directions: Directions not represented in the initial data are represented by linear combinations of existing particles.

We conclude with a discussion of the limitations of the method and suggestions for improvements.

One good feature of the 1-D discretization, which does not generalize to two dimensions, is monotonicity. For equations that have monotonic solutions, such as Nagumo's equation without recovery, the numerical method is guaranteed to have monotonic solutions. This greatly constrains the set of solutions, preventing oscillatory instabilities, for example. In two dimensions there is no such guarantee, and the solutions may go negative as well as exceed 1. Fortunately, in our case $f' < 0$ at $u = 0, 1$, and these errors damp out, but it is evident from Fig. 2.1 that a great deal of cancellation is involved in representing smooth solutions by singular gradient patches.

The main drawback of the GRW method is that large numbers of particles are needed. Although we have not done systematic timing studies, we can estimate how the efficiency of the GRW method with direct calculation of the particle interactions compares to finite-difference methods. Using a 2-D (nonradial) Euler method, an answer more accurate than the GRW solution of Fig. 3.1 can be obtained in 3.6 Cray YMP CPU seconds. The GRW is at least 3 orders of magnitude slower. Accuracy was assessed by comparing along a ray both solutions with that of a second order 1-D (radial) solver using a fine grid. Improvement requires reducing N and speeding up the calculation of the particle interactions.

The GRW may not choose the most economical representation of the solution. In Example 4 the accuracy decreased with time, primarily because the particles lose their radial orientation as they diffuse. In Example 1 we found that efficiency was enhanced by rotating the particles so that they would always point radially. Even then, more particles are needed to resolve the wave as the radius increases.

In order to address this problem, we have experimented with the Chorin–Ghoniem method, which permits masses of individual particles to change with time while GRW conserves particle mass [7], [21]. The result is that in Chorin–Ghoniem while particle number is conserved, individual particles can grow or shrink exponentially in time. This leads to the need to remove

particles that have practically vanished and to split up particles that have grown too massive. In that case, however, the number of particles grows with time just as in the stochastic growth and decay algorithm. We note that Fogelson [10] found that no instability was caused in the convection-diffusion problem by allowing the masses to grow unchecked. This may reflect inherent differences between the convection-dominated and diffusion-dominated cases. We conclude that Chorin–Ghoniem has no practical advantage over GRW in reaction-diffusion problems.

Another modification that could both limit the growth of particles and mitigate loss of optimal particle direction is to recontour the computed solution numerically and reassign particles every so often. That is, N could be reduced by redistributing the mass among a smaller number of particles. This is related to the "rezoning" technique that has been effectively applied to the RVM to reduce the variance of long time solutions [20]. One could apply rezoning by redistributing mass onto regular grids as in the RVM, but this is more complicated in the case of the 2-D GRW because one must compute both the x and y components of the gradient at each point rather than the scalar vorticity.

Another more speculative possibility is to view particle "evolution" in biological terms: particles that help define the solution proliferate while unneeded ones die. Thus, it may be beneficial to introduce "mutations" by randomly or heuristically perturbing the directions. This might also help in problems like Example 2 with direction-deficient initial conditions.

Because of the strong formal similarity between the GRW and RVM [5] methods, much of the machinery that has been developed for the RVM can be applied to the GRW method. For example, a direct implementation of the algorithm requires the computation of $O(N^2)$ interactions between particles at each timestep. We have applied the Greengard–Rokhlin [12], [13] algorithm to accomplish this in $O(N)$ arithmetic operations, increasing the efficiency by almost an order of magnitude. As noted above, however, we had difficulties implementing adequate smoothing to avoid numerical instability without at the same time destroying the computational efficiency. An interesting alternative is to use the Barnes–Hut algorithm to do the N-body calculation [2]. While only $O(N \log N)$ complex, Barnes–Hut permits the use of high-order cut-off functions [3] in a more integral way than Greengard–Rokhlin. Thus Barnes–Hut offers the possibility of incorporating smoothing in a subquadratic N-body algorithm. Further research is required in this area.

An alternative to the multipole method to overcome the quadratic complexity of the GRW would be to do, say, 10 independent runs with 1,000 particles instead of 1 run with 10,000 particles. The solution at a fixed set of grid points would be obtained by averaging. If the variance is $O(N^{-1})$ as in the 1-D GRW methods, then averaging would yield a solution as accurate as that of the single large simulation at $1/10$ the cost. Such an approach also would parallelize in a natural way.

A final, and promising, possibility is to model the diffusion deterministically and thereby replace the random walk with a deterministic motion. This approach has already been applied to simple diffusion and convection-dominated diffusion problems in one and two dimensions, [25], [24], [9], and to collisional dynamics [26], [16]. A distinct advantage of such methods is that deterministic movement of particles to simulate diffusion eliminates the statistical fluctuations of a Monte Carlo method. Instead, these methods have a (deterministic) discretization error associated with their approximate diffusion process. In one dimension, the deterministic algorithms share much in common with the GRW methods, as they are grid free and automatically adaptive. In two dimensions, the existing deterministic methods compute the approximate derivatives required by finite differences on the moving grid defined by the particle coordinates. It is hoped that a variant of a deterministic method can be found that does not require the construction of a moving grid to compute the particle trajectories.

Acknowledgments. The bulk of the calculations were performed on a Cray YMP-8/256 computer at the Advanced Scientific Computing Laboratory of the National Cancer Institute's Frederick Cancer Research and Development Facility. Code development and other computing was done using an IBM 3090-VF600 at the National Institutes of Health's Division of Computer Resources and Technology and a TMC CM-2 at the Advanced Computing Laboratory at the Los Alamos National Laboratory.

We gratefully thank Leslie Greengard for making available the program `rapif2` which implements the fast multipole method [13], [12]. We also thank Matt Wilson for the visualization tools `xplot` and `xview`. Finally, we thank Robert Krasny for pointing out long ago that Anderson's method could be adapted to this problem.

REFERENCES

[1] C. R. ANDERSON, *A vortex method for flows with slight density variation*, J. Comput. Phys., 61 (1985), pp. 417–444.

[2] J. E. BARNES AND P. HUT, *A hierarchical $O(N \log N)$ force-calculation algorithm*, Nature, 324 (1986), pp. 446–449.

[3] J. T. BEALE AND A. MAJDA, *High order accurate vortex methods with explicit kernels*, J. Comput. Phys., 58 (1985), pp. 188–208.

[4] B. CHAUVIN AND A. ROUAULT, *A stochastic simulation for solving scalar reaction-diffusion equations*, Adv. in Appl. Probab., 22 (1990), pp. 88–100.

[5] A. J. CHORIN, *Numerical study of slightly viscous flow*, J. Fluid Mech., 57 (1973), pp. 785–796.

[6] ———, *Vortex sheet approximation of boundary layers*, J. Comput. Phys., 27 (1978), pp. 428–442.

[7] ———, *Numerical methods for use in combustion modeling*, in Computing Methods in Applied Sciences and Engineering, R. Glowinski and J. L. Lions, eds., North Holland, Amsterdam, 1980, pp. 229–235.

[8] R. COURANT, K. FRIEDRICHS, AND H. LEWY, *Über die partiellen differenzengleichungen der mathematischen physik*, Math. Ann., 100 (1928), pp. 32–74.

[9] P. DEGOND AND F.-J. MUSTIELES, *A deterministic approximation of diffusion equations using particles*, SIAM J. Sci. Stat. Comput., 11 (1990), pp. 293–310.

[10] A. L. FOGELSON, *Particle-method solution of two-dimensional convection-diffusion equations*, J. Comput. Phys., 100 (1992), pp. 1–16.

[11] A. F. GHONIEM AND F. SHERMAN, *Grid-free simulation of diffusion using random walk methods*, J. Comput. Phys., 61 (1985), pp. 1–37.

[12] L. GREENGARD, *The Rapid Evaluation of Potential Fields in Particle Systems*, M.I.T. University Press, Cambridge, MA, 1988.

[13] L. GREENGARD AND V. ROKHLIN, *A fast algorithm for particle simulations*, J. Comput. Phys., 73 (1987), pp. 325–348.

[14] O. E. HALD, *Convergence of random methods for a reaction-diffusion equation*, SIAM J. Sci. Stat. Comput., 2 (1981), pp. 85–94.

[15] ———, *Convergence of a random method with creation of vorticity*, SIAM J. Sci. Stat. Comput., 7 (1986), pp. 1373–1386.

[16] F. HERMELINE, *A deterministic particle method for transport diffusion equations: Application to the Fokker-Planck equation*, J. Comput. Phys., 82 (1989), pp. 122–146.

[17] F. JOHN, *Partial Differential Equations*, 4th Edition, Springer-Verlag, New York, 1982, p. 99.

[18] H. P. MCKEAN, *Application of Brownian motion to the equation of Kolmogorov-Petrovskii-Piskunov*, Comm. Pure Appl. Math., 28 (1975), pp. 323–331.

[19] J. NAGUMO, S. ARIMOTO, AND S. YOSHIZAWA, *An active pulse transmission line simulating nerve axon*, Proc. IRE, 50 (1962), pp. 2061–2070.

[20] H. O. NORDMARK, *Rezoning for higer order vortex methods*, J. Comput. Phys., 97 (1991), pp. 366–397.

[21] A. K. OPPENHEIM AND A. GHONIEM, *Application of the random element method to one-dimensional flame propagation problems*, AIAA-83-0600, AIAA 21st Aerospace Sciences Meeting, Reno, NV, 1983.

[22] E. G. PUCKETT, *Convergence of a random particle method to solutions of the Kolmogorov equation $u_t = \nu u_{xx} + u(1-u)$*, Math. Comp., 52 (1989), pp. 615–645.

[23] P. A. RAVIART, *An analysis of particle methods*, in Numerical Methods in Fluid Dynamics, Lecture Notes in Mathematics 1127, Springer-Verlag, New York, 1985, pp. 243–324.

[24] G. RUSSO, *Deterministic diffusion of particles*, Comm. Pure Appl. Math., 43 (1990), pp. 697–733.

[25] G. Russo, *A Lagrangian method for collisional kinetic equations*, in Proc. 1988 AMS SIAM Summer Seminar, Colorado State University, Fort Collins, CO, July 18–29, 1988, E. L. Allgower and K. Georg, eds., Lectures in Applied Mathematics 26, American Mathematical Society, Providence, RI, 1990, pp. 519–539.

[26] ———, *A particle method for collisional kinetic equations*. I. *Basic theory and one-dimensional results*, J. Comput. Phys., 87 (1990), pp. 270–300.

[27] A. S. Sherman and C. S. Peskin, *A Monte-Carlo method for scalar reaction diffusion equations*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 1360–1372.

[28] ———, *Solving the Hodgkin-Huxley equations by a random walk method*, SIAM J. Sci. Statist. Comput., 9 (1988), pp. 170–190.