Numerical computation of diffusion on a surface

Peter Schwartz*†, David Adalsteinsson‡, Phillip Colella*†, Adam Paul Arkin§¶, and Matthew Onsum‖

*Applied Numerical Algorithms Group, and ¶Physical Biosciences Division, Lawrence Berkeley National Laboratory, Berkeley, CA 94720; ‡Department of Mathematics, University of North Carolina, Chapel Hill, NC 27599; and Departments of §Bioengineering and ‖Mechanical Engineering, University of California, Berkeley, CA 94720

Contributed by Phillip Colella, June 21, 2005

We present a numerical method for computing diffusive transport on a surface derived from image data. Our underlying discretization method uses a Cartesian grid embedded boundary method for computing the volume transport in a region consisting of all points a small distance from the surface. We obtain a representation of this region from image data by using a front propagation computation based on level set methods for solving the Hamilton–Jacobi and eikonal equations. We demonstrate that the method is second-order accurate in space and time and is capable of computing solutions on complex surface geometries obtained from image data of cells.

We consider the problem of computing the solution to the diffusion equation on a surface.

$$\frac{\partial C^{surf}}{\partial t} = \Delta^{surf} C^{surf} \quad [1]$$

$$C^{surf}: \mathcal{S} \rightarrow \mathbb{R}, \quad [2]$$

where $\mathcal{S}$ is a surface in $\mathbb{R}^3$. Our interest is motivated by problems in systems biology. Processes such as cellular metabolism, locomotion, and chemotaxis are mediated in part by diffusive transport on the membrane, represented by Eq. **1**. In particular, we want to be able to compute high-fidelity solutions to these problems, in which the surface is obtained from image data of actual cells.

Traditional approaches to solving partial differential equations on surfaces have been based either on a global representation of the surface, such as a triangularization, or on local representations, such as with local coordinate representations, stitched together by using techniques such as multiblock or overset grids. In either case, both the construction of such representations and the design of discretizations of Eq. **1** based on them have algorithmic difficulties and complications beyond those arising when the domain is a subset of $\mathbb{R}^2$. In this article, we present an approach to this problem that avoids many of these difficulties. It is based on recent developments in both numerical methods for solving partial differential equations in complex geometries and mathematical methods for detecting features in image data. In our approach, we solve the heat equation on an annular domain consisting of all the points within a small distance ε of $\mathcal{S}$.

$$\frac{\partial C}{\partial t} = \Delta C \quad [3]$$

$$C: \Omega(\mathcal{S}, \varepsilon) \rightarrow \mathbb{R} \quad [4]$$

$$\frac{\partial C}{\partial \boldsymbol{n}} = 0 \text{ on } \partial\Omega(\mathcal{S}, \varepsilon) \quad [5]$$

$$\Omega(\mathcal{S}, \varepsilon) = \{\boldsymbol{x} : \min_{\boldsymbol{x}' \in S} \|\boldsymbol{x} - \boldsymbol{x}'\| < \varepsilon\} \quad [6]$$

This problem is solved by using a Cartesian grid embedded boundary method (1, 2), in which Eq. **3** is discretized on any domain in $\mathbb{R}^3$ on a finite volume grid constructed by intersecting the domain with rectangular grid cells. A representation of the annular region for which the requisite intersection information is obtained from image data by using the previously described methods (3) that represent the surface in terms of a solution to a Hamilton–Jacobi equation. Specifically, we obtain from this process a function whose values are the signed distance from the surface, defined in an annular region around the surface. Using such an implicit function representation for $\Omega(\mathcal{S}, \varepsilon)$, it is routine to compute the required intersection information. In ref. 4, a similar approach to the one described here has been used to simulate biomedical fluid flow problems starting from images derived by using the described methods (3) from MRI data. The application to surface transport both imposes a different set of requirements and provides some opportunities for simplification.

The idea of solving this problem on a Cartesian mesh discretization on $\Omega(\mathcal{S}, \varepsilon)$ was used previously (5). In that case, the partial differential equation being solved was the original surface equation (Eq. **1**), extended in a natural way to the annular region. Such an approach leads to complicated difference approximations of the metric terms in the surface derivatives. In addition, Δ^{surf} is highly degenerate when viewed as an operator on functions in $\mathbb{R}^3$. For example, in the case where $\mathcal{S}$ is a plane, any function that depends on the coordinate direction orthogonal to $\mathcal{S}$ is in the null space of Δ^{surf}. This complicates the construction of implicit time discretizations that require the solution of linear systems derived from discretizing Δ^{surf}.

The present approach avoids both of these problems. The finite volume approximations to Δ are relatively simple, and when combined with standard implicit discretizations in time, lead to linear systems that permit the use of efficient iterative methods such as multigrid. The cost is that the solution to Eq. **3** is only an approximation to the solution to Eq. **1**. However, we show that the approximation is $O(\varepsilon^2)$. Since ε can be chosen to be a fixed multiple of the mesh spacing, this leads to error that is comparable to the other discretizations errors in a second-order accurate method.

Thin-Layer Asymptotics

Let $\mathcal{S} \subset \mathbb{R}^3$ denote a compact, smooth, orientable surface, with orientation defined by a unit normal field $\boldsymbol{n}$. Then there exists a finite collection of smooth maps of the form,

$$\boldsymbol{x}_0 : (0, 1)^2 \rightarrow \mathcal{S} \quad [7]$$

such that union of the range of these maps cover $\mathcal{S}$, and, for ε sufficiently small, the extensions

$$\boldsymbol{x} : (-1, 1) \times (0, 1)^2 \rightarrow \Omega(\mathcal{S}, \varepsilon) \quad [8]$$

$$\boldsymbol{x}(\xi_1, \xi_2, \xi_3) = \varepsilon\xi_1 \boldsymbol{n}(\boldsymbol{x}_0(\xi_2, \xi_3)) + \boldsymbol{x}_0(\xi_2, \xi_3) \quad [9]$$

†To whom correspondence may be addressed. E-mail: pcolella@lbl.gov or poschwartz@lbl.gov.

have nonsingular Jacobians and ranges whose union covers $\Omega(\mathcal{S}, \varepsilon)$. In that case, $\boldsymbol{x}(\xi_1, \xi_2, \xi_3)$ is the unique point in $\Omega(\mathcal{S}, \varepsilon)$ that is a signed distance $\varepsilon\xi_1$ from $\mathbf{x}_0(\xi_2, \xi_3)$.

For any smooth coordinate mapping, we have

$$\Delta C = \frac{1}{det(F)} \nabla_{\xi} \cdot (det(F)\, F^{-T} F^{-1} \nabla_{\xi} C), \; F = \nabla_{\xi} \boldsymbol{x}. \qquad [10]$$

For the coordinate mapping (Eq. **9**) this leads to the following form for the diffusion equation (Eq. **3**).

$$\varepsilon^2 J \frac{\partial C}{\partial t} = \frac{\partial}{\partial \xi_1}\left(J \frac{\partial C}{\partial \xi_1}\right) + \varepsilon^2 \sum_{1<i,j\leq 3} \frac{\partial}{\partial \xi_i}\left(\frac{a_{ij}}{J} \frac{\partial C}{\partial \xi_j}\right), \qquad [11]$$

where

$$J = \left(\frac{\partial \boldsymbol{x}}{\partial \xi_2} \times \frac{\partial \boldsymbol{x}}{\partial \xi_3}\right) \cdot \boldsymbol{n} \qquad [12]$$

and

$$a_{ij} = \pm\left(\frac{\partial \boldsymbol{x}}{\partial \xi_i} \times \boldsymbol{n}\right) \cdot \left(\frac{\partial \boldsymbol{x}}{\partial \xi_j} \times \boldsymbol{n}\right) \text{ for } 1 < i, j \leq 3, \qquad [13]$$

where $\pm = +$ if $i = j$, $\pm = -$ if $i \neq j$. We can compute J and a_{ij} as functions of ε.

$$J = J^0 - \varepsilon\xi_1 H + \varepsilon^2 \xi_1^2 K$$

$$a_{ij} = a_{ij}^0 + \varepsilon\xi_1 A_{ij} + O(\varepsilon^2),$$

where the quantities J^0, a_{ij}^0, H, K, and A_{ij} depend only on ξ_2, ξ_3. In particular, the surface Laplacian Δ^{surf} appearing in Eq. **1** can be written in terms of the ξ_2, ξ_3 coordinate system.

$$\frac{\partial C^{surf}}{\partial t} = \Delta^{surf} C^{surf} \qquad [14]$$

$$= \frac{1}{J^0} \sum_{1<i,j\leq 3} \frac{\partial}{\partial \xi_i}\left(\frac{a_{ij}^0}{J^0} \frac{\partial C^{surf}}{\partial \xi_j}\right). \qquad [15]$$

We now show that as $\varepsilon \to 0$, a solution to Eq. **3** differs from a solution to Eq. **1** by $O(\varepsilon^2)$. To do this, we expand C in powers of ε,

$$C = \sum_{p=0}^{3} C^{(p)} \varepsilon^p, \qquad [16]$$

and equate terms in Eq. **11** corresponding to the same power of ε. We will further assume that the derivatives of C with respect to all of the original spatial variables and time are independent of ε, so that no inverse powers of ε appear from differentiating with respect to time or the mapped variables. This will be the case, for example, if the initial data are independent of ξ_1.

First, we note that the homogeneous Neumann boundary condition in Eq. **3** becomes

$$\frac{\partial C}{\partial \xi_1} = 0 \text{ at } \xi_1 = \pm 1. \qquad [17]$$

Furthermore, differentiating Eq. **16** with respect to ε implies that Eq. **17** must hold for each of the $C^{(p)}$ separately. Then we have:

- $p = 0$:

$$\frac{\partial}{\partial \xi_1}\left(J^0 \frac{\partial C^{(0)}}{\partial \xi_1}\right) = 0 \qquad [18]$$

and Eq. **17** implies that $C^{(0)}$ is independent of ξ_1.

- $p = 1$:

$$\frac{\partial}{\partial \xi_1}\left(J^0 \frac{\partial C^{(1)}}{\partial \xi_1}\right) = \frac{\partial}{\partial \xi_1}\left(\xi_1 \mathrm{H} \frac{\partial C^{(0)}}{\partial \xi_1}\right) = 0. \qquad [19]$$

Again, by Eq. **17** $C^{(1)}$ is independent of ξ_1.

- $p = 2$: After rearranging terms, and recalling that $C^{(0)}$ and $C^{(1)}$ are independent of ξ_1, we have

$$-\frac{\partial}{\partial \xi_1}\left(J^0 \frac{\partial C^{(2)}}{\partial \xi_1}\right) = J^0 \frac{\partial C^{(0)}}{\partial t} + \sum_{1<i,j\leq 3} \frac{\partial}{\partial \xi_i}\left(\frac{a_{ij}^0}{J^0} \frac{\partial C^{(0)}}{\partial \xi_j}\right). \qquad [20]$$

The right side is independent of ξ_1, and by Eq. **17** it must be identically zero. i.e., $C^{(0)}$ satisfies Eq. **14**. In addition, $C^{(2)}$ is independent of ξ_1.

- $p = 3$: Similarly to the case $p = 2$ we have

$$-\frac{\partial}{\partial \xi_1}\left(J^0 \frac{\partial C^{(3)}}{\partial \xi_1}\right) = J^0 \frac{\partial C^{(1)}}{\partial t} + \sum_{1<i,j\leq 3} \frac{\partial}{\partial \xi_i}\left(\frac{a_{ij}^0}{J^0} \frac{\partial C^{(1)}}{\partial \xi_j}\right) + \xi_1 G(\xi_2, \xi_3). \qquad [21]$$

It follows from $C^{(1)}$ being independent of ξ_1 and Eq. **17** that $C^{(1)}$ satisfies Eq. **14**.

From this argument we see that $C^{(0)} + \varepsilon C^{(1)}$ satisfies Eq. **14**, which implies that C itself differs from a solution to Eq. **14** by $O(\varepsilon^2)$.

Even though the analysis was carried out given specific assumptions about the initial data, the conclusion appears to be robust relative relaxing the assumptions. In particular, we continue to observe in our numerical calculations nearly invariant behavior of the solution in the direction normal to the surface for problems with forcing of Eq. **1** with a source term; for long-time integration of the equations; and for initial data that varies in the normal direction. Qualitatively, this is not surprising: diffusion in the normal direction relaxes to a local steady state very rapidly relative to the time scale for diffusion in the tangential direction.

Embedded Boundary Discretization

The underlying discretization of space is given by rectangular control volumes on a Cartesian grid: $\mathrm{Y}_{\boldsymbol{i}} = [\boldsymbol{i}h, (\boldsymbol{i} + \boldsymbol{u})h]$, $\boldsymbol{i} \in \mathbb{Z}^3$, h is the mesh spacing, and $\boldsymbol{u}$ is the vector whose entries are all ones. The geometry is represented by the intersection of the irregular domain Ω with the Cartesian grid. We obtain control volumes $V_{\boldsymbol{i}} = \mathrm{Y}_{\boldsymbol{i}} \cap \Omega$ and faces $A_{\boldsymbol{i}\pm\frac{1}{2}\boldsymbol{e}_d}$, which are the intersection of $\partial V_{\boldsymbol{i}}$ with the coordinate planes $\{\mathbf{x} : x_d = (i_d + 1/2 \pm 1/2)h\}$. Here $\boldsymbol{e}_d$ is the unit vector in the d direction. We also define $A_{\boldsymbol{i}}^B$ to be the intersection of the boundary of the irregular domain with the Cartesian control volume: $A_{\boldsymbol{i}}^B = \partial\Omega \cap \mathrm{Y}_{\boldsymbol{i}}$.

To construct finite difference methods, we will need only a small number of real-valued quantities that are derived from these geometric objects.

- Areas and volumes are expressed in dimensionless terms: volume fractions $\kappa_{\boldsymbol{i}} = |V_{\boldsymbol{i}}|h^{-3}$, face apertures $\alpha_{\boldsymbol{i}\pm\frac{1}{2}\boldsymbol{e}_d} = |A_{\boldsymbol{i}\pm\frac{1}{2}\boldsymbol{e}_d}|h^{-2}$ and boundary apertures $\alpha_{\boldsymbol{i}}^B = |A_{\boldsymbol{i}}^B|h^{-2}$.
- The locations of centroids, and the average outward normal to the boundary are given exactly by:

$$\text{face centroid: } \boldsymbol{x}_{\boldsymbol{i}+\frac{1}{2}\boldsymbol{e}_d} = \frac{1}{|A_{\boldsymbol{i}+\frac{1}{2}\boldsymbol{e}_d}|} \int_{A_{\boldsymbol{i}+\frac{1}{2}\boldsymbol{e}_d}} \boldsymbol{x} dA$$

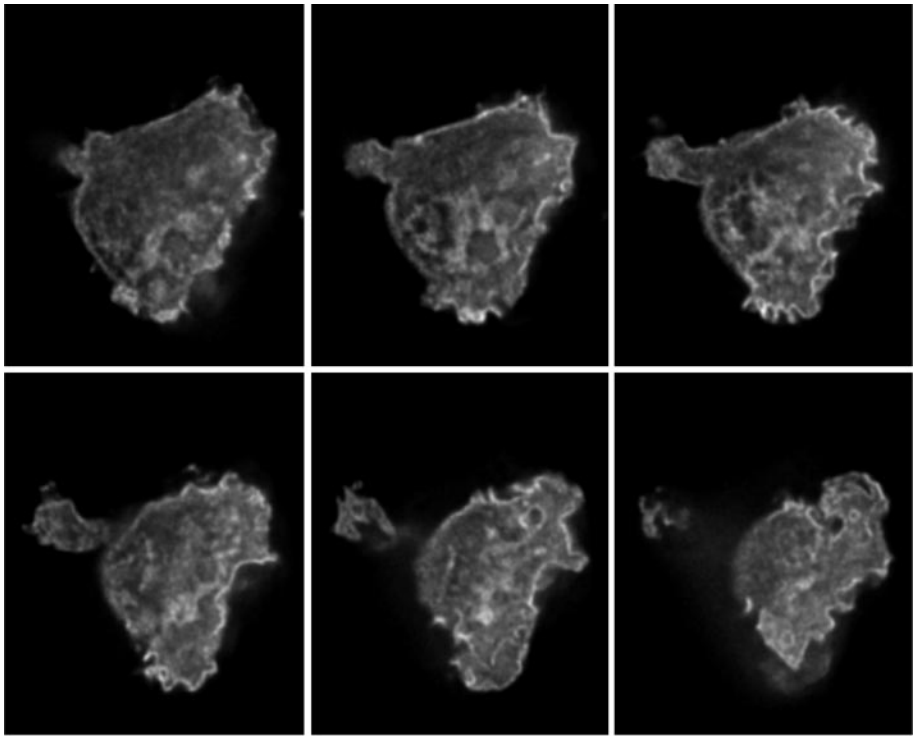

Fig. 1. 2D slices through a gray-scale image of an hl-60 cell (15) obtained by using deconvolution microscopy. Image is ≈2,000 times actual size.

$$\text{boundary face centroid: } \boldsymbol{x}_i^B = \frac{1}{|A_i^B|}\int_{A_i^B} \boldsymbol{x}\,dA$$

$$\text{outward normal: } \boldsymbol{n}_i^B = \frac{1}{|A_i^B|}\int_{A_i^B} \boldsymbol{n}^B\,dA,$$

where $\boldsymbol{n}^B$ is the outward normal to $\partial\Omega$, defined for each point on $\partial\Omega$. We assume that we can compute estimates of these quantities that are accurate to $O(h^2)$.

Using just these quantities, we can define conservative discretizations for the divergence operator. Let $\vec{F} = (F^1, F^2, F^3)$ be a function of $\boldsymbol{x}$. Then

$$\nabla\cdot\vec{F} \approx \frac{1}{|V_i|}\int_{V_i} \nabla\cdot\vec{F}\,dV = \frac{1}{|V_i|}\int_{\partial V_i} \vec{F}\cdot\boldsymbol{n}\,dA \approx \frac{1}{\kappa_i h}\sum_{\pm=+,-}\sum_{d=1}^{3} \pm\alpha_{\boldsymbol{i}\pm\frac{1}{2}\boldsymbol{e}_d} F^d(\boldsymbol{x}_{\boldsymbol{i}\pm\frac{1}{2}\boldsymbol{e}_d}), \quad [22]$$

where Eq. **22** is obtained by replacing the integrals of the normal components of the vector field $\vec{F}$ with the values at the face centroids. We obtain the spatial discretization from replacing $F^d(\boldsymbol{x}_{\boldsymbol{i}\pm\frac{1}{2}\boldsymbol{e}_d})$ with difference approximations. Following refs. 1 and 2, we define the discrete Laplacian,

$$\Delta^h C = \frac{1}{\kappa_i h}\sum_{\pm=+,-}\sum_{d=1}^{3} \pm\alpha_{\boldsymbol{i}\pm\frac{1}{2}\boldsymbol{e}_d} F^{\boldsymbol{d}}_{\boldsymbol{i}\pm\frac{1}{2}\boldsymbol{e}_d}, \quad [23]$$

where the fluxes satisfy

$$F^{\boldsymbol{d}}_{\boldsymbol{i}+\frac{1}{2}\boldsymbol{e}_d} = \sum a_s \frac{(C_{\boldsymbol{i}+\boldsymbol{s}+\boldsymbol{e}_d} - C_{\boldsymbol{i}+\boldsymbol{s}})}{h} = F^d(\boldsymbol{x}_{\boldsymbol{i}+\frac{1}{2}\boldsymbol{e}_d}) + O(h^2). \quad [24]$$

Here the sum over faces and the weights correspond to bilinear interpolation of the centered difference approximations to the centroid location. Then we solve

$$\frac{dC_i}{dt} = (\Delta^h C)_i \quad [25]$$

by discretizing in time with a second-order accurate, L_0-stable implicit Runge–Kutta method (6). The resulting method provides uniformly second-order accurate solutions and is amenable to the use of geometric multigrid solvers for solving the resulting

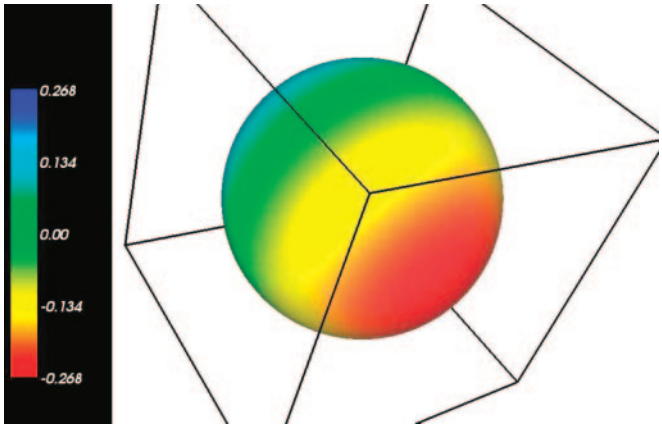

Fig. 2. Computed solution on the spherical shell using 128^3 grid points.

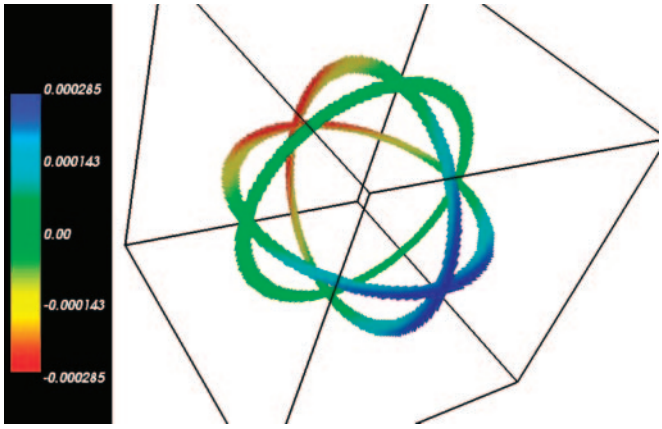

Fig. 3. Slices through the spherical shell showing the difference between the computed solution and the exact solution using 128^3 grid points.

linear systems. This approach can be generalized to include the effect of source terms; for details, see refs. 1 and 2.

Grid Generation

To carry out the numerical procedure outlined in the previous section, it is necessary to generate the geometric data obtained from intersecting $\Omega(\mathcal{S}, \varepsilon)$ with rectangular grid cells, i.e., the areas, volumes, and centroids defined above. To do this, we compute on a Cartesian grid the representation of the domain as an implicit function:

$$\Omega(\mathcal{S}, \varepsilon) = \{\boldsymbol{x} : |\psi(\boldsymbol{x})| < \varepsilon\}. \qquad [26]$$

Given the values of ψ on the grid, it is a routine exercise in quadrature to compute the intersection information we require to $O(h^2)$ accuracy. In this section, we will describe how we obtain such an implicit function starting from image data.

Typically, we are given image data in the form of intensities $G = G(\boldsymbol{x})$ evaluated on a rectangular grid in three dimensions. In this work, the images are given as a collection of deconvolution microscopy images where each $x - y$ slice contains gray scale values in the range [0, 1] (Fig. 1). The goal of this method is to find a surface that lies along high values of the gradient ∇G, as that indicates a sharp change in image intensity. Additional requirements need to be imposed, since in a typicalimage, the gradient is noisy, and there can be both

Table 1. Solution error for $h = 1/32, 1/64, 1/128$

L^1 norm	Rate	L^2 norm	Rate	L^∞ norm	Rate
1.989444e-03		2.424191e-03		5.636316e-03	
4.859636e-04	2.03	5.719458e-04	2.08	1.255820e-03	2.17
1.201429e-04	2.02	1.390271e-04	2.04	2.828283e-04	2.15

missing edges caused by imaging effects and multiple possible edges caused by internal structures. Following ref. 3, we can formulate this problem as a front propagation problem, to be solved by using level set methods to solve the associated Hamilton–Jacobi equation (7):

$$\psi_t + F|\nabla\psi| + \vec{U}\cdot\nabla\psi + g\boldsymbol{\kappa}|\nabla\psi| = 0. \qquad [27]$$

Here the set $\{\boldsymbol{x} : \psi(\boldsymbol{x}, t) = 0\}$ corresponds to the location of the front at time t, with the front located initially outside the surface to be detected, and

$$\boldsymbol{\kappa} = \nabla\cdot\left(\frac{\nabla\psi}{|\nabla\psi|}\right)$$

is the curvature. The functions F and $\vec{U}$ are chosen so that the front is attracted to the maximum value of $|\nabla G|$, while g is chosen to constrain the curvature of the front, thus preventing the front from propagating through small gaps in the image data representation of the surface.

$$k_I = \frac{1}{1 + |\nabla\mathcal{S}(G)|}$$

$$F = \alpha k_I, \quad g = \gamma k_I, \quad \vec{U} = \beta\nabla(|\nabla(\mathcal{S}(G))|)$$

The operator $\mathcal{S}$ is a Gaussian smoothing operator, chosen to reduce the noise in the image data. The parameters α, β, and γ are currently chosen by trial and error, usually by running the detection code on 2D slices of the data, which takes a few seconds per run on a workstation (computing a 3D solution typically takes a few minutes). When the solution to Eq. **27** reaches a steady state, we expect the zero set of ψ to correspond to the outermost surface in the image. In practice, one solves the equations for a fixed time (e.g., $t = 1$) and adjusts the parameters α, β, and γ so that a solution sufficiently close to a steady state is obtained. Since we are interested in the

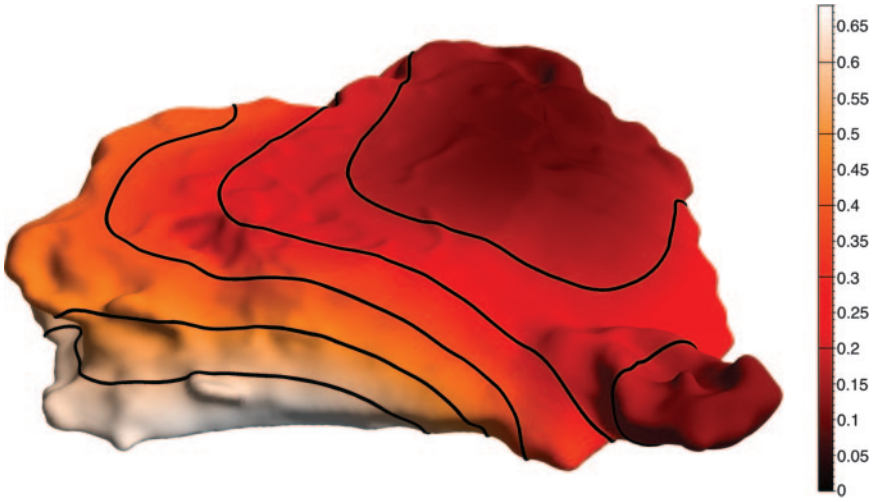

Fig. 4. Solution for diffusion equation for a $256 \times 256 \times 128$ grid at time = 50 s. The time step is 5.0 s.

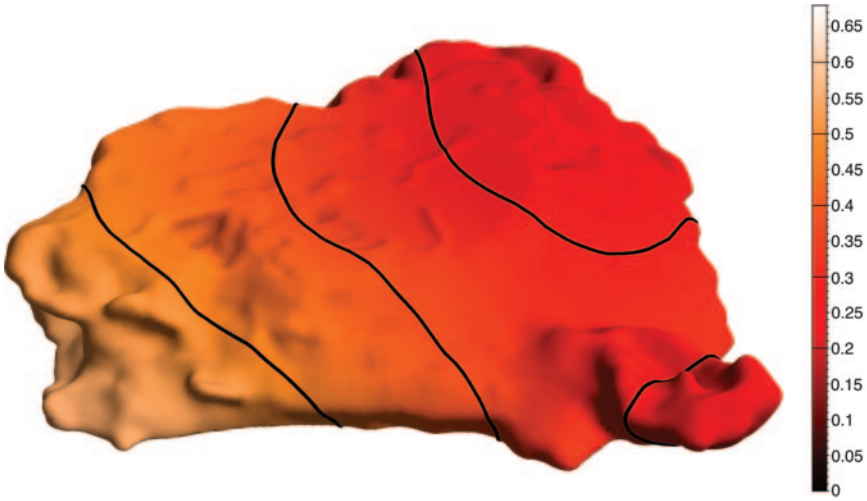

Fig. 5. Solution for diffusion equation for the 256 × 256 × 128 grid at time = 100 s. The time step is 5.0 s.

solution only within a small distance of the propagating front, we use the previously described technique (8) to perform the calculation only in a narrow band near the front. Finally, we require not only the location of the front, but that ψ be a distance function defined within an ε distance on either side of the front. We use the previously described method (9) to compute extensions of F, $g\boldsymbol{\kappa}$, and $\tilde{U}$ away from the zero set at each time step so that the time evolution tends to preserve the property that the solution satisfies the eikonal equation $|\nabla\psi| = 1$ and is therefore a signed distance function. In particular at the end of the calculation we expect the condition (Eq. **26**) to be satisfied. We also postprocess the solution by solving the eikonal equation by using the previously described method (10, 11) to eliminate numerical error in the signed distance property that may have accumulated in the course of the time evolution.

Results

We show results for two examples. In the first, the surface $\mathcal{S}$ in a sphere of radius $r_0 = 0.4$. In spherical coordinates, the initial data is $C^{surf}(\theta, \phi, t = 0) = cos(\phi)$, for which the exact solution is

$$C^{surf}(\theta, \phi, t) = \cos(\phi)e^{-2t/r_0^2}.$$

We compute the solution on a spherical shell as in Eq. **3** for $h = 1/32, 1/64, 1/128$, with $\varepsilon = 3h$. Since the sphere can be specified analytically as a signed distance function, this calculation will test only the accuracy of the method for discretizing the diffusion equation. We advance the solution in time until the accumulated time is 0.1 by using a time step $\Delta t = h/2$. At the final time the magnitude of the solution has decreased by approximately a factor of 4. Fig. 2 shows the computed solution on the outer surface of the sphere. Fig. 3 shows the corresponding error, given as the difference between the computed solution and the exact solution extended to the entire spherical shell to be constant in the radial direction. Table 1 contains various norms of the error, where the integral norms (L^1 and L^2) are computed by computing a consistent approximation to the integrals over $\Omega(\mathcal{S}, \varepsilon)$ divided by 2ε. In the limit of vanishing ε, these estimates converge to an estimate of the appropriate integrals over $\mathcal{S}$. The L^∞ norm is computed as the maximum over all cells of the absolute value. For all three norms, the method is seen to be second order accurate. This is consistent with the modified equation analysis (1, 2, 12).

The second example demonstrates the end-to-end capability. We generate a signed distance representation of the image in Fig. 1, then use it to compute the grid intersection information required to discretize the solution to the diffusion equation in the annular region. In Figs. 4 and 5, the initial condition for this problem was a two-valued function: on a circular patch on the flat underside of the surface we set $C = 10$; everywhere else $C = 0$. The time step was 5.0 s, which is ≈30 times the maximum time step for an explicit method on this problem.

There are a number of directions in which we intend to take this work. The diffusion solver described here is the core of a multicompartment model currently under development for reaction-diffusion processes in cells. Transport in both the membrane and the cytosol is represented by using the embedded boundary approach, with coupling to chemical reaction terms in both regions, and spatially and state-dependent fluxes representing transport coupling the membrane and the interior of the cell. We would also like to extend this approach to other partial differential equations representing mechanical processes on the surface of the cell, including the representation of the membrane as an elastic or viscoelastic medium, coupling the ideas discussed here to the versions of embedded boundary method for hyperbolic problems described (13), extended to moving boundaries following ref. 14.

We thank Prof. Henry Bourne of the University of California, San Francisco, for the use of his laboratory facilities in obtaining the image data shown in Fig. 1. This work was supported at the Lawrence Berkeley National Laboratory by the U.S. Department of Energy: Director, Office of Science, Office of Advanced Scientific Computing, Mathematical, Information, and Computing Sciences Division under Contract DE-AC03-76SF00098; the Defense Advanced Research Planning Agency BioComp program, BAA-01-26-0126517; and the Howard Hughes Medical Institute. Work at the University of North Carolina was supported by the Defense Advanced Research Planning Agency BioComp program under Contract FA8750-05-1-0118.

1. McCorquodale, P., Colella, P. & Johansen, H. (2001) *J. Comput. Phys.* **173,** 620–635.
2. Schwartz, P., Barad, M., Colella, P. & Ligocki, T. J. (2005) *J. Comput. Phys.*, in press.
3. Malladi, R., Sethian, J. A. & Vemuri, B. C. (1995) *IEEE Trans. Pattern Anal. Machine Intell.* **17,** 158–175.
4. Deschamps, T., Schwartz, P., Trebotich, D., Colella, P., Malladi, R. & Saloner, D. (2004) *Technical Report UCRL-CONF-208523* (Lawrence Livermore National Laboratory, Berkeley, CA).
5. Adalsteinsson, D. & Sethian, J. A. (2003) *J. Comput. Phys.* **185,** 271–288.

6. Twizell, E. H., Gumel, A. B. & Arigu, M. A. (1996) *Adv. Comput. Math.* **6,** 333–352.
7. Osher, S. & Sethian, J. A. (1988) *J. Comput. Phys.* **79,** 12–49.
8. Adalsteinsson, D. & Sethian, J. A. (1995) *J. Comput. Phys.* **118,** 269–277.
9. Adalsteinsson, D. & Sethian, J. A. (1999) *J. Comput. Phys.* **138,** 2–33.
10. Helmsen, J. J., Puckett, E. G., Colella, P. & Dorr, M. (1996) *Proc. SPIE* **2726,** 253–261.
11. Sethian, J. A. (1996) *Proc. Natl. Acad. Sci. USA* **93,** 1591–1595.
12. Johansen, H. & Colella, P. (1998) *J. Comput. Phys.* **147,** 60–85.
13. Colella, P., Graves, D. T., Keen, B. J. & Modiano, D. (2005) *J. Comput. Phys.*, in press.
14. Bell, J. B., Colella, P. & Welcome, M. L. (1991) in *AIAA 10th Computational Fluid Dynamics Conference* (Am. Inst. of Aeronautics and Astronautics, Washington, DC), pp. 814–822.
15. Gallagher, R. E., Collins, S. J., Trujilo, J., McCredie, K., Ahearn, M., Tsai, S., Metzgar, R., Aulakh, G., Ting, R., Ruscettti, F. W. & Gallo, R. C. (1979) *Blood* **54,** 713–733.