Available online at www.sciencedirect.com
SCIENCE DIRECT®
Journal of Computational Physics 185 (2003) 271–288
JOURNAL OF COMPUTATIONAL PHYSICS
www.elsevier.com/locate/jcp

Transport and diffusion of material quantities on propagating interfaces via level set methods ☆

David Adalsteinsson [a,*], J.A. Sethian [b]

[a] *Department of Mathematics, University of North Carolina, Chapel Hill, NC, USA*
[b] *Department of Mathematics, University of California, Berkeley, USA*

Received 24 April 2002; received in revised form 18 September 2002; accepted 24 November 2002

Abstract

We develop theory and numerical algorithms to apply level set methods to problems involving the transport and diffusion of material quantities in a level set framework. Level set methods are computational techniques for tracking moving interfaces; they work by embedding the propagating interface as the zero level set of a higher dimensional function, and then approximate the solution of the resulting initial value partial differential equation using upwind finite difference schemes. The traditional level set method works in the trace space of the evolving interface, and hence disregards any parameterization in the interface description. Consequently, material quantities on the interface which themselves are transported under the interface motion are not easily handled in this framework. We develop model equations and algorithmic techniques to extend the level set method to include these problems. We demonstrate the accuracy of our approach through a series of test examples and convergence studies.

1. Introduction and overview

In this paper, we develop numerical algorithms for tracking interfaces in which material quantities are transported along an evolving front. The algorithms are designed to be robust, accurate, efficient, handle topological change, and work in both two and three space dimensions.

One way to characterize interface motion is by a discrete parameterized view. These methods allow accurate calculations, and include the ability to simultaneously move an interface while transporting variable quantities. This last capability is of considerable importance in a variety of physical problems, including boundary integral formulations of interface problems, in which single and double layer

☆ D. Adalsteinsson is supported by the Alfred P. Sloan Foundation. J.A. Sethian is supported in part by the Applied Mathematical Sciences, Office of Energy Research, US Department of Energy, #DE–AC03-76SF00098, and the Division of Mathematical Sciences, National Science Foundation.
* Corresponding author.
E-mail addresses: david@amath.unc.edu (D. Adalsteinsson), sethian@math.berkeley.edu (J.A. Sethian).

0021-9991/02/$ - see front matter
doi:10.1016/S0021-9991(02)00057-8

distributions must be updated and advanced in tandem with the moving interface, and problems in semiconductor manufacturing under etching and deposition processes, in which associated quantities such as step coverage are critical in accurately capturing the dynamics of the evolving profile. Possible drawbacks of these marker Lagrangian approaches include difficulties with evaluating topological change, the need to remove parts of the evolving front (known as delooping) to correctly characterize the viscosity solution, the need to adaptively add and remove points to provide numerical resolution and ameliorate strict time step requirements, and complications in three dimensions.

One alternative Eulerian numerical formulation is provided by the level set method, which embeds the interface as the zero level set of a higher dimensional function. This approach extend to three dimensions and easily handles topological change. Level set techniques have the additional advantage that they are naturally linked to numerical schemes for hyperbolic conservation laws, and can easily provide accurate values for geometric terms such as the normal direction and curvature. Nonetheless, the technique is not easily applied to problems in which a material quantity is required to be evaluated along the interface; fundamentally, this is because the method works in the range space of the domain, and ignore past information such as tangential stretch. While some extensions of these techniques have been developed, a general and numerically sound algorithm is not yet available.

This paper presents techniques to design level set methods to accurately compute the evolution of interfaces which carry material properties. The key idea involves the use of a single level set function to characterize the interface position, and an associated field suitably extended off the interface to keep control of the material quantities. We derive the relevant equations of motion, and then present our numerical techniques, followed by a series of numerical convergence tests and model problems to verify the algorithm.

One straightforward example is the motion of a fluid interface which depends, in a boundary integral formulation, on the vorticity distribution along the front. A more complex example involves copper electroplating, which concerns the deposition of a substance which both diffuses on and is transported by the evolving interface. In order to improve the filling of deep holes, an inhibitor is sometimes added to the copper salt solution; the additive slows the copper deposition rate. The substance is thus consumed at the surface and depletes towards the bottom of the trench (see [11,10] for similar examples.) An accurate numerical simulation would allow for both diffusion and transport of the coverage material.

2. Equations of motion

We consider a problem in which an interface moves under a velocity which depends on location and local geometric properties such as normal direction and curvature. Let $G(\mathbf{x})$ be the value of a surface quantity at a point $\mathbf{x}$ on the interface. Our goal is to characterize the transport of G under the interface motion.

2.1. Statement of problem: Lagrangian formulation

For ease of explanation, we discuss the problem in terms of curves propagating in two space dimensions; ultimately, the equations we derive will apply to surface propagation in three space dimensions as well. Consider a moving interface given by a curve; associated with this interface is a function G, which gives the value of some scalar quantity along the interface.

We consider the following motions.

2.1.1. Interface motion

- *Advection:* The interface is passively advected under a velocity field $(u(x,y), v(x,y))$. Here, we assume that this advection velocity is defined on the front itself; it may also have meaning off of the front.

- *Propagation:* The interface propagates normal to itself with speed F, which may depend on position, normal direction, local curvature, as well as the solution of some associated partial differential equation. Here, the solution is meant to be the viscosity solution, selected through the appropriate entropy condition (see [5,18,19,22]).

2.1.2. Evolution of material quantity G

The material quantity G which lies on the front itself also evolves. We allow for the following types of evolution.

- *Diffusion:* The material quantity G is allowed to diffuse along the interface according to a standard diffusion model; this can be thought of as occurring for a fixed interface.
- *Conservative transport under front motion:* At a particular point of the interface (x, y), G is carried with the propagating interface. As a model, we take conservation of surface coverage. That is, as an element of the interface stretches or shrinks under the front evolution, this corresponds to a commensurate decrease or increase (respectively) of $G(x, y)$ so that the G along the entire interface is conserved. Thus, if L is the length of a parameterized tangent vector at (x, y), then LG is conserved under front motion. This is described more precisely below.

2.2. Advection, propagation, smoothness, conservation

Our particular model for propagation of the surface scalar G has some subtlety. To begin, we require that the total amount of G be conserved along the interface as it moves. This is not the same as assuming that the underlying advection velocity is conservative. As an example, consider a unit circle with total amount G_0 of substance. Then at any point (x, y) of the circle, we have that $G(x, y) = G_0/2\pi$. Imagine that the circle propagates in its normal direction with unit speed. Then since the total amount G along the front is constant, we have that

$$G(x, y, t) = \frac{G_0}{2\pi(1 + t)}.$$

It is easy to see that this is not the solution of the conservative advection equation for G. Instead, our model corresponds to a conservative advection equation for LG, where L is a length of a surface element. Geometrically, this means that if we imagine a painted balloon, as the balloon is blown up, the amount of paint per unit surface area must decrease.

Second, non-smooth fronts pose an additional dilemma. Consider an initial square with constant G. Suppose the square moves outwards with unit speed. The sides should move outwards carrying the material quantity G, and the expanding corners should be arcs in which G is zero. This conserves the total amount of G as the front moves, yet it is clear that the distribution of G is no longer smooth. Similarly, if the front moves inwards, we view G as a delta function at the corners. Thus, in the presence of corners in the evolving front, we must examine the regularized problem of the limit of smooth fronts which produce the non-smooth front.

2.3. Level set methods

Level set methods, introduced by Osher and Sethian [16], rely in part on the theory of curve and surface evolution given by Sethian in [17,18] and on the link between front propagation and hyperbolic conservation laws discussed in [19]. These techniques recast interface motion as a time-dependent Eulerian initial value partial differential equation, and rely on viscosity solutions to the appropriate differential equations to update the position of the front, using an interface velocity that is derived from the relevant physics both

on and off the interface. These viscosity solutions are obtained by exploiting schemes from the numerical solution of hyperbolic conservation laws. Level set methods are specifically designed for problems involving topological change, dependence on curvature, formation of singularities, and host of other issues that often appear in interface propagation techniques. The equation of motion for the evolving level set function ϕ is given by

$$\phi_t + F|\nabla\phi| + u \cdot \nabla\phi = 0,$$

where F is the speed in the normal direction to the interface and u is the (vector) advection velocity. For details, see [16,22].

Our goal now is to use this embedded interface approach to develop linked equations of motion for the propagating interface and the material quantity G. We note that equations for a non-embedded model, where properties are defined only the interface, have been previously developed in parameterized frameworks, see, for example [29,32]. We also point out that there has been previous work on the description of material quantities in the context of level set methods. In [4], work on solving diffusion-type equations on static (non-moving) surfaces has been considered in an implicit representation, and in [8], a level set formulation was used to track the evolution of vortex sheets, keeping track of the circulation on the evolving interface. We refer the reader to these papers for background and earlier work.

2.4. Statement of problem: implicit formulation, two dimensions

Suppose that the material function G_{front}, defined on the front, is in fact defined throughout the computational domain, and use the notation G to represent this function. Thus, G is a function from R^2 to R which agrees with G_{front} on the given initial curve Γ. We shall later make suggestions as to appropriate choices for G, for now, we shall simply assume that it can be constructed and is sufficiently smooth. Following the level set approach, let the initial curve Γ be embedded as the zero level set of some function $\phi(x, y, t = 0)$. We derive evolution equations for ϕ and G which produce implicit descriptions of the desired action on the front itself. We do so term by term.

2.4.1. Diffusion

Begin with the diffusion equation in the case of a stationary front. Let $G((x, y), t)$ be defined in all space and agrees with G_{front} on the stationary interface, and let $\phi(x, y)$ be an implicit representation of the interface. Finally, let $\alpha(s) = (x(s), y(s))$ parametrize the front by arc length, such that $\alpha(0) = (x, y)$. Diffusion along the interface then implies that

$$G_t((x, y), t) = \sigma G_{ss}((x, y), t),$$

where σ is the diffusion constant. Setting $\sigma = 1$, we have that

$$\begin{aligned} G_t((x,y)) &= \frac{\partial}{\partial s}\left[\frac{\partial}{\partial s}[G(\alpha(s))]\right] = \frac{\partial}{\partial s}\left[G_x(\alpha(s))x'(s) + G_y(\alpha(s))y'(s)\right] \\ &= [G_{xx}(\alpha(s))x'(s) + G_{xy}(\alpha(s))y'(s)]x'(s) + G_x(\alpha(s))x'(s) + [G_{xy}(\alpha(s))x'(s) + G_{yy}(\alpha(s))y'(s)]y'(s) \\ &\quad + G_y(\alpha(s))y''(s), \end{aligned}$$

where $(x', y') = (-\phi_y, \phi_x)/L$, $L = \sqrt{\phi_x^2 + \phi_y^2}$, and

$$x'' = -\frac{\phi_x}{L}\kappa, \quad y'' = -\frac{\phi_y}{L}\kappa, \quad \kappa = \frac{\phi_y^2\phi_{xx} - 2\phi_x\phi_y\phi_{xy} + \phi_x^2\phi_{yy}}{L^3},$$

κ is the curvature of the front. Assembling the terms, we have

$$G_t = \left[G_{xx}\frac{\phi_y}{L} - G_{xy}\frac{\phi_x}{L}\right]\frac{\phi_y}{L} - \frac{G_x\phi_x}{L}\kappa - \left[G_{xy}\frac{\phi_y}{L} - G_{yy}\frac{\phi_x}{L}\right]\frac{\phi_x}{L} - \frac{G_y\phi_y}{L}\kappa$$

$$= \frac{G_{xx}\phi_y^2 - 2G_{xy}\phi_x\phi_y + G_{yy}\phi_x^2}{L^2} - \frac{G_x\phi_x + G_y\phi_y}{L}\kappa.$$

We now note that this equation may be defined on all of space, and defines the evolution of the material function G both on and off the front. Additionally, we note that if (n_x, n_y) is the value of the normal, the above equation simplifies to

$$G_t = (G_{xx}n_y^2 - 2G_{xy}n_xn_y + G_{yy}n_x^2) - (G_xn_x + G_yn_y)\kappa.$$

Here, $G_xn_x + G_yn_y$ is the dot product of ∇G with the normal. Finally, we note that the above equation has been derived in [4] using more general approach; we present this derivation to emphasize the link with the parametrized view.

2.4.2. Advection

Consider a front with scalar field G which is also undergoing advection by means of a vector field (u, v) that depends on (x, y).

We define the quantity $L = \sqrt{\phi_x^2 + \phi_y^2}$, which represents the length of the tangent vector along the interface. Then conservation of the material quantity LG under advection by (u, v) means that we have

$$(LG)_t + (uLG)_x + (vLG)_y = 0.$$

The first term is thus

$$(LG)_t = GL_t + LG_t = G\frac{\phi_x\phi_{xt} + \phi_y\phi_{yt}}{L} + LG_t,$$

$$\phi_{xt} = -(u\phi_x + v\phi_y)_x = -u\phi_{xx} - v\phi_{xy} - u_x\phi_x - v_x\phi_y,$$

$$\phi_{yt} = -(u\phi_x + v\phi_y)_y = -u\phi_{xy} - v\phi_{yy} - u_y\phi_x - v_y\phi_y,$$

$$(LG)_t = -G\frac{u\phi_x\phi_{xx} + (u\phi_y + v\phi_x)\phi_{xy} + v\phi_y\phi_{yy}}{L} - G\frac{u_x\phi_x^2 + (v_x + u_y)\phi_y\phi_x + v_y\phi_y^2}{L} + LG_t.$$

The second term is

$$(uLG)_x = u_xLG + uL_xG + uLG_x = u_xLG + u\frac{\phi_x\phi_{xx} + \phi_y\phi_{xy}}{L}G + uLG_x.$$

The third term is similar, and leads to

$$-G\frac{u_x\phi_x^2 + (v_x + u_y)\phi_y\phi_x + v_y\phi_y^2}{L^2} + G_t + u_xG + uG_x + v_yG + vG_y = 0.$$

Rearranging, and using $L^2 = \phi_x^2 + \phi_y^2$, we have

$$G_t + G\frac{u_x\phi_y^2 - (v_x + u_y)\phi_y\phi_x + v_y\phi_x^2}{L^2} + uG_x + vG_y = 0$$

yields the final advection equation

$$G_t + uG_x + vG_y + (n_y^2 u_x - n_x n_y (u_y + v_x) + n_x^2 v_y)G = 0.$$

2.4.3. Normal advection

If $u = Fn_x$, $v = Fn_y$, where F can be a function of (x, y), we observe that the term

$$n_y^2 u_x - n_x n_y (u_y + v_x) + n_x^2 v_y$$

simplifies to κF, and the evolution equation becomes

$$G_t(x, y) = -F[(N \cdot \nabla G) + \kappa G].$$

2.4.4. Final equations

Combining the previous equations, we get that if σ is the diffusion coefficient, and the front is moved with a combination of an advection field (u, v) and a normal speed F:

$$G_t = \sigma\Big[(G_{xx} n_y^2 - 2G_{xy} n_x n_y + G_{yy} n_x^2) - \kappa(\nabla G \cdot N)\Big] - (u, v) \cdot \nabla G - (n_y^2 u_x - n_x n_y (u_y + v_x) + n_x^2 v_y)G$$
$$- F[(N \cdot \nabla G) + \kappa G].$$

2.5. Statement of problem: implicit formulation, three dimensions

The equations for three dimensions are derived in a similar manner; for a complete derivation, see [3]. We assume a diffusion coefficient σ and an advection velocity field (u, v, w).

To simplify the notation, introduce the differential operator

$$\Xi_\Phi[u, v, w] = \frac{1}{\|\nabla \Phi\|^2} \begin{bmatrix} (v_y + w_z)\Phi_x^2 + (u_x + w_z)\Phi_y^2 + (u_x + v_y)\Phi_z^2 \\ -(v_x + u_y)\Phi_x \Phi_y - (w_x + u_z)\Phi_x \Phi_z - (w_y + v_z)\Phi_y \Phi_z \end{bmatrix}.$$

Using this operator, the mean curvature is given by

$$\kappa_M = \frac{\Xi_\Phi[\Phi_x, \Phi_y, \Phi_z]}{\|\nabla \Phi\|}$$

and the differential equation simplifies to

$$G_t = \sigma\big[\Xi_\Phi[G_x, G_y, G_z] - \kappa_M(\nabla G \cdot N)\big] - (u, v, w) \cdot \nabla G - \Xi_\Phi[u, v, w]G - F[(N \cdot \nabla G) + \kappa_M G].$$

Expanding out, we note that the first term is equal to

$$\Xi_\Phi[G_x, G_y, G_z] = \Big[(G_{yy} + G_{zz})N_x^2 + (G_{xx} + G_{zz})N_y^2 + (G_{xx} + G_{yy})N_z^2 - 2G_{xy}N_x N_y - 2G_{xz}N_x N_z - 2G_{yz}N_y N_z\Big].$$

For a non-embedded model in which quantities are defined only on the interface, a parameterized set of equations which model this phenomena were derived using a different approach in [29], see also [32].

3. Numerical approximations and algorithms

3.1. Level set methods

Level set methods rely on two central embeddings; first the embedding of the interface as the zero level set of a higher dimensional function, and second, the embedding (or extension) of the interface's velocity to this higher dimensional level set function. As mentioned earlier, this leads to an initial value partial differential equation for the implicitly defined function ϕ of the form

$$\phi_t + F|\nabla\phi| = 0 \quad \phi(x, t = 0) \text{ given.} \tag{1}$$

This is the implicit formulation of front propagation given in [16]. Schemes for solving this equation were provided in [16]. In almost all computational situations, those schemes are inefficient and wasteful. Practical implementation of these schemes requires addressing the computational cost of both the embedding of the level set function ϕ and the embedding of the underlying velocity field F.

3.1.1. Efficiency

The Narrow Band Level Set Method, introduced by Adalsteinsson and Sethian in [1], limits work to a neighborhood (or "narrow band") of the zero level set. This is a significant cost reduction; it also means that extension velocities need only be constructed at points lying in the narrow band, as opposed to all points in the computational domain. This idea of limiting computation to a narrow band around the zero level set was introduced in Chopp [6], used in recovering shapes from images in Malladi et al. [14], and explored in depth in [1]. Details on the accuracy, typical tube sizes, and number of times a tube must be rebuilt may be found in Adalsteinsson and Sethian [1].

This Narrow Band Method requires some sort of reinitialization, that is, reconstruction of the signed distance function. As understood by many practitioners of level set methods (see [22,25]), reinitialization every time step (or close to every time step) is a poor strategy, since each reinitialization causes error in the position of the front. However, occasional reinitialization is required when coupled to the Narrow Band method.

One technique for performing this reinitialization comes from using Fast Marching Methods [20], which are Dijkstra-like upwind finite difference algorithms which solve the Eikonal equation

$$|\nabla T|F(x, y, z) = 1 \quad T = 0 \text{ on } \Gamma,$$

in $O(N \log N)$ time, where N is the total number of points in the computational domain. Reinitialization comes as a special case of the Eikonal equation when $F = 1$. The key observation in Fast Marching Methods is that an upwind approximation to the gradient implies an ordering on mesh points, obtained by a sort algorithm which updates the points in ascending value of T, similar to Dijkstra's network path algorithm [7]. This ordering is computed as the calculation unfolds, and yields an algorithm which avoids all iteration. The Fast Marching Method has been extended to higher order finite difference approximations by Sethian in [23], first-order unstructured meshes by Kimmel and Sethian [12], and higher order unstructured meshes by Sethian and Vladimirsky [26]. Some early applications include photolithography in [21], a comparison of a similar approach with volume-of-fluid techniques in [9], a fast algorithm for image segmentation in [15] and computation of seismic travel times by Sethian and Popovici [24]. A different Dijsktra-like method for the Eikonal equation, obtained through a control-theoretic discretization which hinges on the optimality criterion, was developed by Tsitsiklis [31]. For discussion of various Dijkstra-like methods and extensions beyond the Eikonal equation to general static Hamilton–Jacobi equations, see [27,28].

3.1.2. Extension F and G off the interface

We need to build values for both the velocity F and the material quantity G off the interface in order to achieve the embedding. We shall exploit the extension velocity methodology developed in earlier works.

To begin, in order to construct extension velocities F, one starts with the given velocity F_{front} and chooses an appropriate extension velocity (see [22,25] for details). In [13], the idea of extrapolating the given front velocity along the gradient of the front to obtain an extension velocity off of the front was first introduced and used for image segmentation; an earlier extension velocity formulation was given in [30]. Mathematically, this means that

$$\nabla F \cdot \nabla \phi = 0. \tag{2}$$

It is straightforward to show that this choice of extension velocity maintains the signed distance function for the level sets of ϕ for all time (see, for example [33]).

In [2], a strategy for constructing these extension velocities was introduced, using a two-tiered system. Given a level set function at time n, namely ϕ_{ij}^n, one first constructs a signed distance function $\bar{\phi}_{ij}^n$ around the zero level set. Simultaneous with this construction, one then constructs the extension velocity F_{ext} satisfying Eq. (2). This velocity is used to update the level set function ϕ^n. For details, see [2].

We shall use these ideas to extend G off of the interface.

3.2. Algorithms for diffusion and transport of material quantities

We use standard upwind difference operators to advance both the level set function ϕ and the scalar G.

3.2.1. Level set advection

We use a second-order scheme to advect and propagate the level set function. We consider the general form

$$\phi_t + u \cdot \nabla \phi + F|\nabla \phi| = \epsilon\kappa,$$

where u is the multi-dimensional advection field, F is the propagation speed normal to the front, and ϵ is a constant. We note that F and u can depend on G. Following [16], a spatially second-order upwind scheme is used for the two transport terms on the left, and a central difference scheme is used for the diffusive term on the right.

For the time advancement, we use a second-order Runge–Kutta approximation. Abstractly, we write

$$\phi_t = L(u, F, \phi, \epsilon).$$

Then the time advancement is

$$\begin{aligned} U_0 &= L(u(t_n), F(t_n), \phi, \epsilon)\mathrm{d}t, \\ \phi &= \phi + U_0, \\ U_1 &= L(u(t_{n+1}), F(t_{n+1}), \phi, \epsilon)\mathrm{d}t, \\ \phi &= \phi + (U_1 - U_0)/2. \end{aligned}$$

The advantage to this formulation is that one does not construct a separate copy of ϕ.

3.2.2. Scalar advection

The general form of the update equation is

$$G_t = \sigma\left[\Xi_\Phi[G_x, G_y, G_z] + \kappa_M(\nabla G \cdot N)\right] + (u, v, w) \cdot \nabla G - \Xi_\Phi[u, v, w]G + F[(N \cdot \nabla G) - \kappa_M G].$$

The update for the G scalar is similar to that for the level set function. The diffusion term is computed using central differences. The advection term $u \cdot \nabla G$ is approximated using the above upwind differences. The term

$$-(n_y^2 u_x - n_x n_y (u_y + v_x) + n_x^2 v_y) G$$

is an exponential term, decaying or expanding depending on the sign. This is computed using central derivatives. Again, a second order in time approximation is used.

3.2.3. Alternative formulation

We note here an alternative formulation, which is to work directly with the conservation equation given in Section 2.4.2 and approximate the evolution of LG using conservative schemes. We are pursuing this approach, and will report on it elsewhere.

4. Numerical tests of algorithm: two dimensions

In this section, we consider some two-dimensional test cases to analyze the accuracy of the algorithms.

4.1. Diffusion, fixed front

We begin by studying the diffusion of a scalar quantity on a fixed front. As a simple test, let the front be given by a circle with radius 0.3 in a unit box. Define the ϕ surface by computing the signed distance rather than using the exact distance. The initial value for G is defined everywhere as

$$G(x, y) = x.$$

We then evolve the G array according to

$$G_t = \sigma \Big[(G_{xx} n_y^2 - 2 G_{xy} n_x n_y + G_{yy} n_x^2) + \kappa (\nabla G \cdot N) \Big].$$

The normal N is taken from the level surface, and is constant in time since the surface never moves.

An exact solution may be produced as follows. Parametrizing the front by arc length, $(r\cos(s/r), r\sin(s/r))$, the initial data is given in terms of arc length by $r\cos(s/r)$. With unit diffusion coefficient, the exact solution is given by

$$r \cos\left(\frac{s}{r}\right) \exp\left(-\frac{\sigma t}{r^2}\right) = x \exp\left(-\frac{\sigma t}{x^2 + y^2}\right).$$

4.1.1. Diffusion, fixed front, simple

Calculations are performed up to time $T = 1$ for three grid sizes, 60×60, 120×120, and 240×240. Initial value for G is $G(x, y) = x$ in all of space. As a scalar function of arc length, this is a wave on the circle, i.e. the lowest wave number. Stability is ensured by a time step which is scaled as the square of the step size. The error is evaluated as follows. For a large number of time values, we compute the exact solution by the above expression and subtract it from the computed solution obtained by interpolating G onto uniformly spread points on the interface. We then use the trapezoid rule to compute the integral of the square of this error, and divide through by the total length of the circle. This gives the average two norm of the error along the front. The results are shown in Fig. 1. The observed convergence is second order overall.

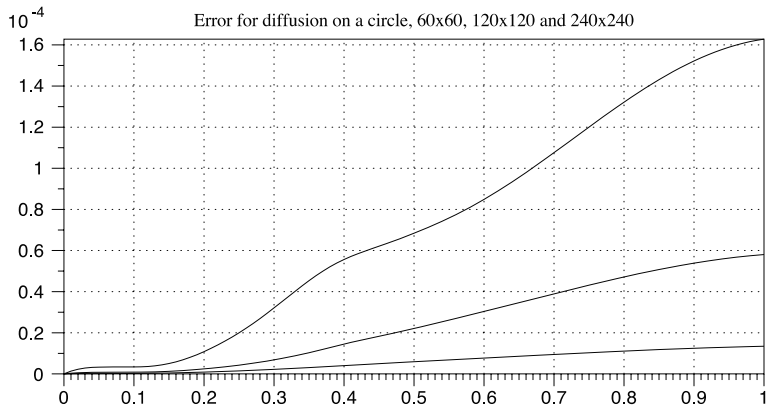

Fig. 1. Front fixed, values diffused on a circle.

4.1.2. Diffusion, front fixed, general case

A more challenging problem occurs when the front is no longer a circle and the initial value of the scalar is more complicated than a single eigenvector of the diffusion operator. We chose as a perturbed circle as the initial front, and the scalar G is defined for points (x, y) on the front with $G(x, y) = x^2$. Note that if the scalar is defined only on the front, for example as a function of arc length, the initialization of $G(x, y)$ will be done by the fast extension method [2].

Fig. 2 gives a view of the front and a plot of G as a function of arc length. This scalar has a full spectrum of frequencies, which can be solved exactly with a Fourier series method. To find the exact solution at a later time T, we extract the initial scalar as a function of arc length, decompose it into its Fourier modes, and then scale each mode using the coefficient of diffusion $\sigma = 0.05$. At that time T, the numerical solution is interpolated (bi-cubic) onto the same points and subtracted. This will give the error as a function of arc

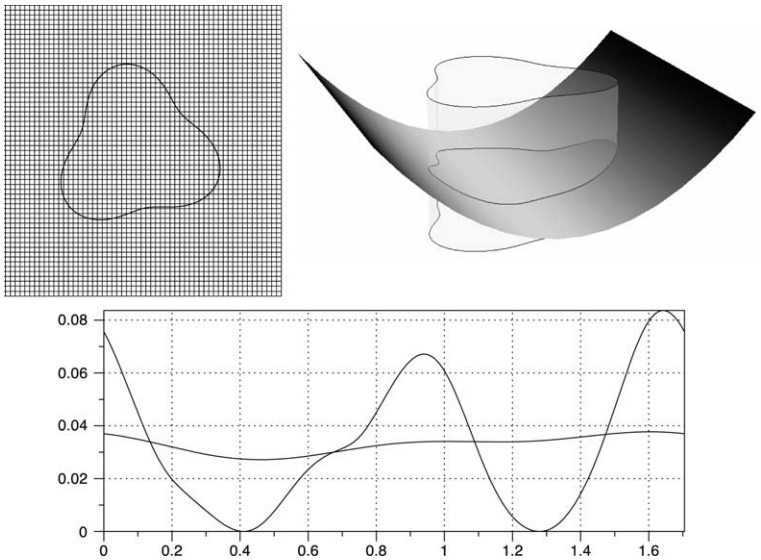

Fig. 2. Diffusion on a more general front. Initial value given by $F(x, y) = x^2$. Later value at $T = 1$. Both values are drawn as functions of arc length.

length. We then compute the L_2 norm of that function and divide by the length to get the average error. This error is shown in Fig. 3, and the run is simulated until time $T = 1.0$.

4.2. Advection

4.2.1. Advection – two dimensions

Next, we consider the advection of the material quantity as the front moves. We start with an ellipse with major axis 0.4 and 0.3, centered in a unit square box (Fig. 4). The initial value of the scalar is defined as

$$G(x,y) = x^2 + y^2 - \frac{xy}{\sqrt{x^2+y^2}} + \frac{1}{10}$$

everywhere in space. We consider a flow which rotates this ellipse around the center a full circle. For this problem, we know the exact solution, since the values will just rotate with the front. We run this problem for two grid sizes, $h = 1/121$ and $h = 1/241$.

We measure the error both for the front evolution as well as the scalar evolution, presented in the two plots shown in Fig. 5. The error is measured as follows.

- For the front, we compute the contour of the level set. Using the exact solution, we compute the pointwise difference between the computed and exact front. This will give us the error along the front. Then we take that error and compute the average two norm of this error. This gives a good indication of the overall error for front evolution. The infinity norm gives a very similar result, since the pointwise error is fairly continuous. This was done for two hundred time values and the results are plotted as a function of time for both grid resolutions.

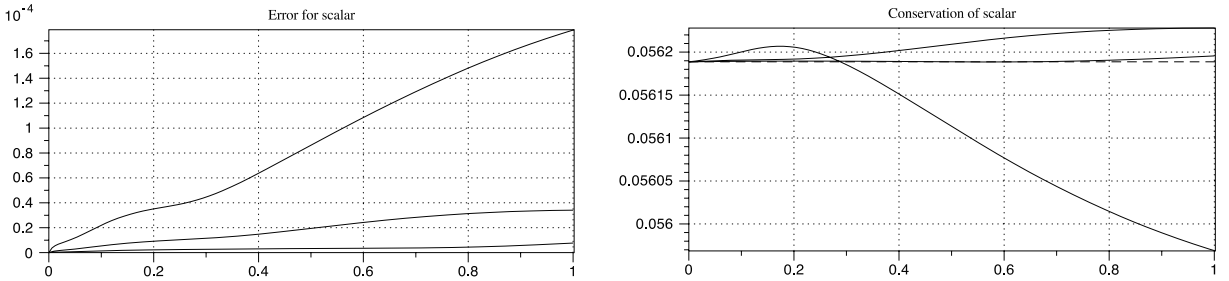

Fig. 3. Diffusion on a more general front. Error along the front as a function of time for the grid sizes 60×60, 120×120, and 240×240.

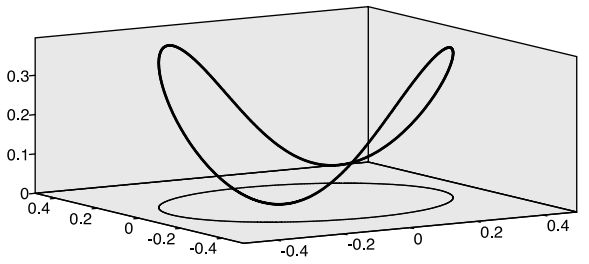

Fig. 4. Initial scalar value along the ellipse.

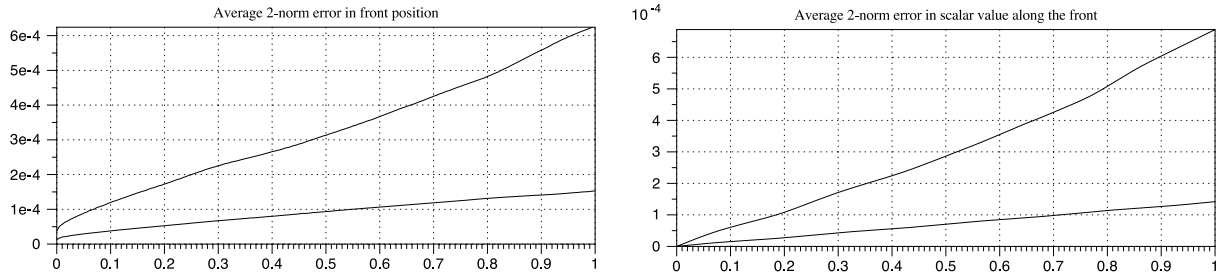

Fig. 5. Error for rotating front.

- For the scalar values along the front, we interpolate the values from the scalar field onto the points on the front by using a bi-cubic interpolation. We then subtract off the exact solution at those points gotten by rotating the exact values, giving the error along the front. Take the average two norm of this error for the same two hundred time values as before. To get the relative error size, we note that the maximum value of the scalar is about 0.4.

A different and equally useful measurement of scheme accuracy is to track mass conservation. To do so, we compute the path integral of the values along the front for each of these two hundred time values and plot it as a function of time in Fig. 6. Note that for all of these runs, we get a second-order reduction of the error; even on the coarse grid, the error is very small.

4.2.2. Advection and diffusion

We now rerun the above problem, but in addition to the rotation we include diffusion of the values along the surface. We use a diffusion coefficient 0.1 in order for the scalar to reduce in amplitude by about a factor of 10.

The exact solution can be obtained, since it is a combination of a rotation and a diffusion on a fixed elliptical front. To compute the exact solution of the diffusion process on a fixed front, we write the solution in terms of arc length. In this representation, the scalar satisfies the heat equation with diffusion constant 0.1 and can be solved using Fourier series.

We then ran the same numerical experiments as in the previous section. In order to compute the exact solution, we extract the values on the initial front by arc length (using bi-cubic interpolation), defined on an evenly spaced one dimensional grid. We then compute the numerical Fourier series to provide the exact solution at later times. This is then rotated and subtracted from the computed solution (see Fig. 7 and Fig. 8).

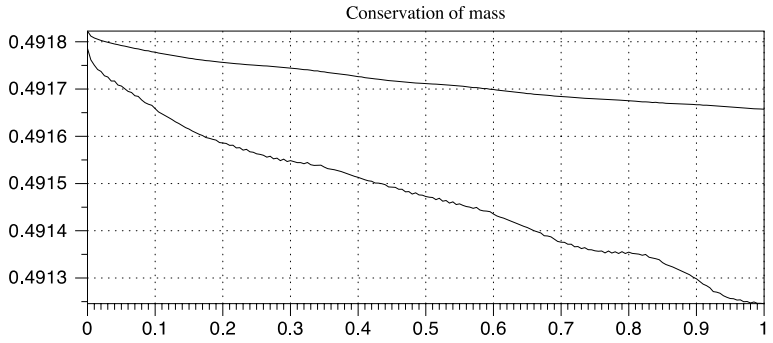

Fig. 6. Mass conservation along the ellipse; no diffusion, with rotation.

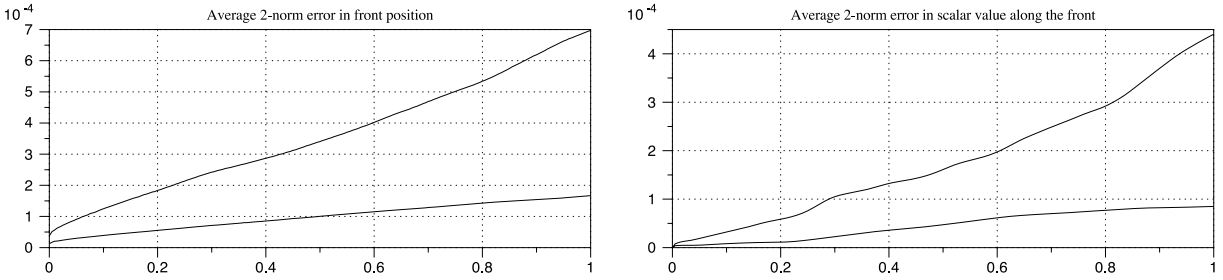

Fig. 7. Error of front rotation and scalar.

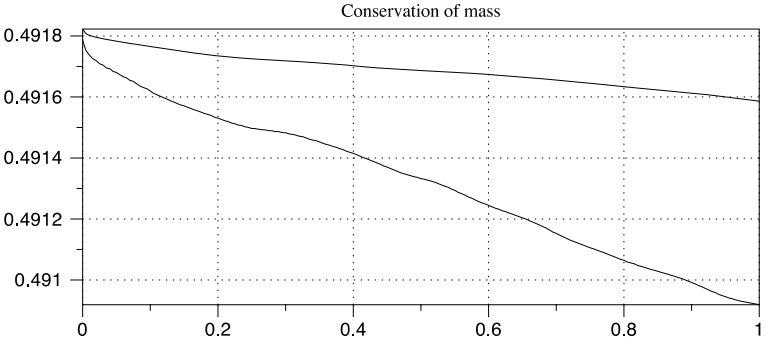

Fig. 8. Mass conservation of rotated ellipse with scalar diffusion.

5. Numerical tests of algorithm: three dimensions

We now test the ability of the algorithm to track the evolution to material quantities in conjunction with three-dimensional interface evolution. In all of the tests, it is necessary to use cubic interpolation at all time. Lower order interpolation will cause convergence to stall. In all of the results, we expect that the error should be very smooth.

5.1. Diffusion on sphere

When the front is fixed, and $\sigma = 1$, the evolution equation becomes

$$G_t = \Xi_\Phi[G_x, G_y, G_z] + \kappa_M(\nabla G \cdot N).$$

In this test we compare the simulation with an exact solution. To find an exact solution, we look for a solution on a sphere which only depends on the z value. It can therefore be described as $G(x) = G(\varphi)$ in terms of spherical coordinates. The diffusion equation for the sphere, using this symmetry and the fact that diffusion is proportional to the flux gives

$$G_t(\varphi) = \lim_{\Delta\varphi\to 0} \frac{(2\pi r\sin(\varphi+\Delta\varphi))G_\varphi(\varphi+\Delta\varphi)/r - (2\pi r\sin(\varphi))G_\varphi(\varphi)/r}{2\pi r^2\sin(\varphi)\Delta\varphi + \mathrm{O}((\Delta\varphi)^2)} = \frac{\cos(\varphi)G_\varphi(\varphi) + \sin(\varphi)G_{\varphi\varphi}(\varphi)}{r^2\sin(\varphi)}$$

The function $G(\varphi) = \cos(\varphi)$ is an eigenfunction of this operator with eigenvalue $-2/r^2$, so with that as the initial condition, we get an exact solution

$$G(t,\varphi) = \cos(\varphi)\exp\left(-\frac{2t}{r^2}\right).$$

We can do this on concentric spheres by defining the initial function to be

$$G(x,y,z) = z.$$

The calculations are run on successive grids of $50 \times 50 \times 50$ and $100 \times 100 \times 100$ on the box $[-0.6, 0.6] \times [-0.6, 0.6] \times [-0.6, 0.6]$. Here, the front is initialized as a sphere with radius 0.4, and the simulation are run up to $T = 0.25$ with diffusion constant $\sigma = 0.5$. The results are shown in Figs. 9 and 10.

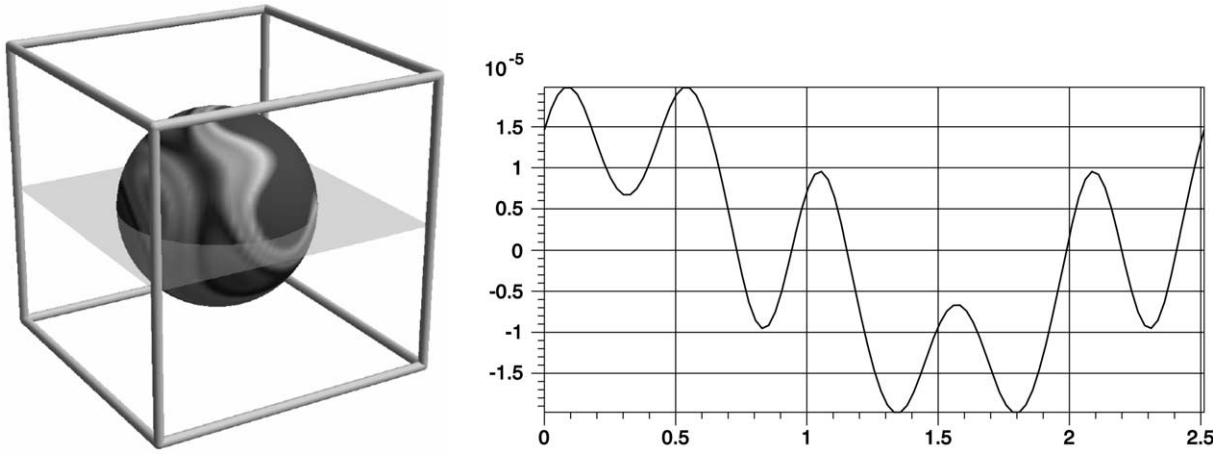

Fig. 9. The error along the surface is shaded according to value. The plane slices through those values and the resulting error is shown along the front as a function of arc length. This is the error at $t = 0.25$ for the grid spacing 1/100.

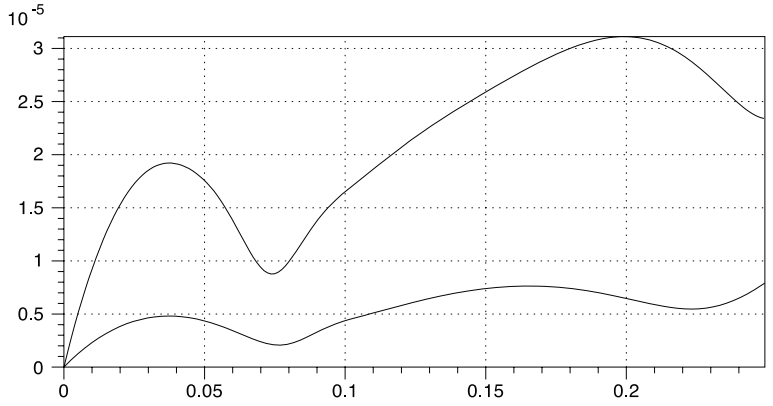

Fig. 10. Diffusion on a stationary sphere. How the error changes over time for the exact solution. Plot of the average two norm (L^2) of the error as a function of time for both $h = 1/50$ and $h = 1/100$. The error is computed by interpolating the values onto the front, and subtracting the known solution there. This defines a function on the surface, and we compute a surface integral to find the two norm. Divide by the area to find the average two norm.

5.2. Advection

We now study how well the algorithm tracks the advection of scalar values. We rotate a sphere with values defined on it. We know the exact solution, and can compare both the position of the front and the values along the front with the exact solution. We use a sphere with radius 0.4, centered around $(-0.5, 0, 0)$ and rotate the sphere in the xy plane around the origin. The sphere completes a full circle at $T = 1$. Four time values for this rotation are shown in Fig. 11. Initial values along the surface are given by the expression $x + y + z$.

To analyze the accuracy, we track three measurements.

- The pointwise error in the position of the front. The error is computed at each point on the surface, and then the average norm is computed along the surface. The result is shown in Fig. 12.
- Along the surface, we compute the error in the value of the scalar and then compute the norm of the error over the whole surface. The result is shown in Fig. 13.
- The integral of the scalar over the surface. This should be conserved over time. The result is shown in Fig. 14.

We use a surface integral to compute the average two norm of the error along the surface. This computation is done for the grid sizes $120 \times 120 \times 60$ and $240 \times 240 \times 120$.

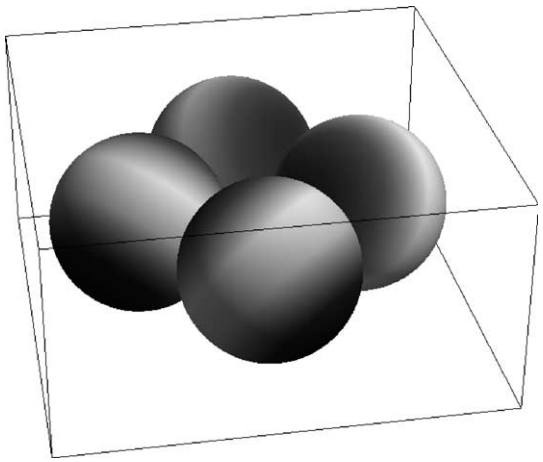

Fig. 11. Four time values for rotating a sphere.

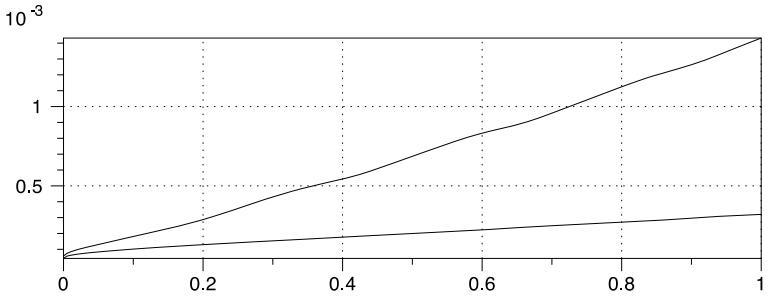

Fig. 12. Error for location of front.

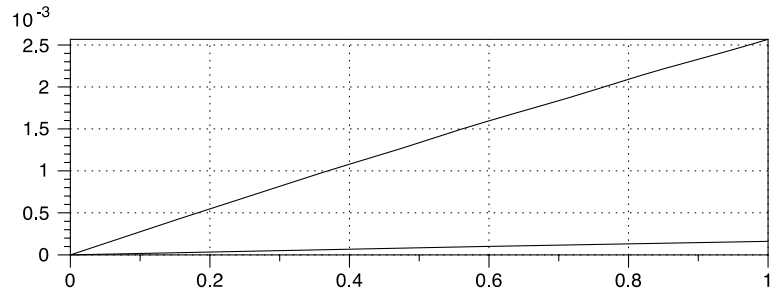

Fig. 13. Difference between computed and exact.

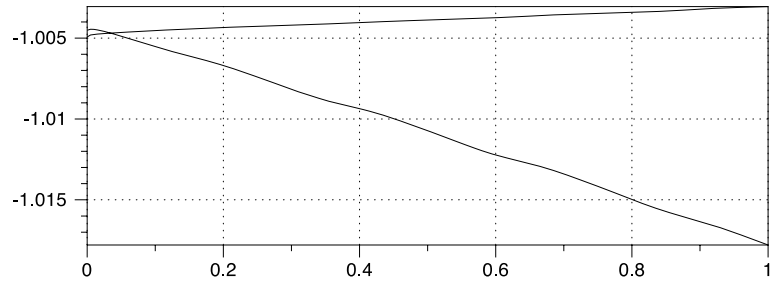

Fig. 14. The integral over the scalar changes.

5.3. Advection and diffusion

Consider the problem where we have simultaneous advection and a diffusion in three dimensions. We begin with a sphere with center at $(-1/5, 0, 0)$ and radius 2/5 in the box

$$[-4/5, 3/5] \times [-3/5, 4/5] \times [-1/2, 1/2]$$

and now rotate it a quarter turn in the xy plane to $(0, -1/5, 0)$ by the rotational field $2\pi(y, -x, 0)$ and final time $T = 1/4$. We set the diffusion coefficient to $1/2$ and run two different grid sizes 1/50 and 1/100. The rotation is shown in Fig. 15 with the error shown in Fig. 16.

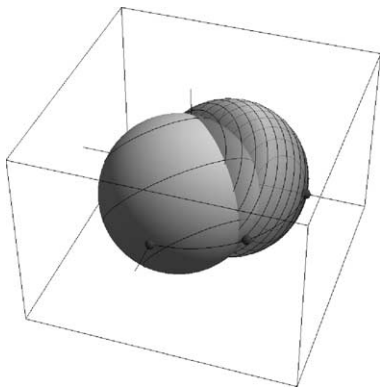

Fig. 15. Advection and diffusion in three dimensions. The motion of the surface. Initial condition is transparent. Surface is rotated around the origin.

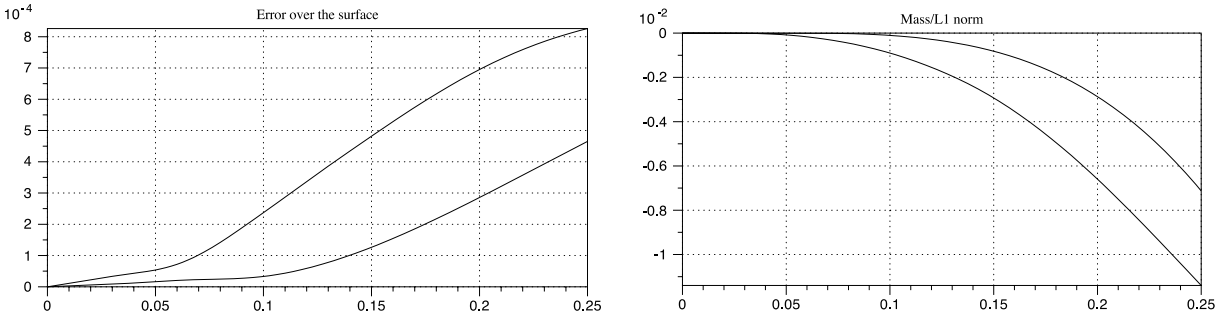

Fig. 16. Advection and diffusion in three dimensions. Error behavior for $70 \times 70 \times 50$ and $140 \times 140 \times 100$. On the left is a measurement of the error compared to the exact solution, on the right how close it is to conserving mass.

In future work, we plan to apply these algorithms to semiconductor problems and surface/fluid transport.

Acknowledgements

We would like to thank M. Garzon for many valuable comments and the reviewers for their many constructive suggestions.

References

[1] D. Adalsteinsson, J.A. Sethian, A fast level set method for propagating interfaces, J. Comput. Phys. 118 (2) (1995) 269–277.
[2] D. Adalsteinsson, J.A. Sethian, The fast construction of extension velocities in level set methods, J. Comput. Phys. 148 (1999) 2–22.
[3] D. Adalsteinsson, J.A. Sethian, Transport and diffusion of material quantities on propagating interfaces via level set methods. CPAM Report, University of California, Berkeley, 2002.
[4] M. Bertalmio, L.T. Cheng, S. Osher, G. Sapiro, Variational problems and partial differential equations on implicit surfaces; the framework and examples in image processing and pattern formation, J. Comput. Phys. 174 (2001) 759–780.
[5] M.G. Crandall, P.-L. Lions, Viscosity solutions of Hamilton–Jacobi equations, Trans. AMS 277 (1983) 1–43.
[6] D.L. Chopp, Computing minimal surfaces via level set curvature flow, J. Comput. Phys. 106 (1993) 77–91.
[7] E.W. Dijkstra, A note on two problems in connection with graphs, Numer. Math. 1 (1959) 269–271.
[8] E. Harabetian, S. Osher, C.-W. Shu, An Eulerian approach for vortex motion using a level set regularization procedure, J. Comput. Phys. 127 (1) (1996) 15–26.
[9] J. Helmsen, E.G. Puckett, P. Colella, M. Dorr, Two new methods for simulating photolithography development. SPIE 1996 International Symposium on Microlithography, SPIE 2726, June, 1996.
[10] G.C.A.M. Jansen, J.F. Jongste, A.H. Verbruggen, S. Radelaar, H.J. Barth, W. Robl, The Role of TiN Liner in forcefill, Electrochem. Solid State Lett. 1 (2) (1998) 97–99.
[11] D. Josell, D. Wheeler, W.H. Huber, T.P. Moffat, Superconformal Electrodeposition in Submicron Features, Physical Review Letters 87(1) (2001) 16012–1–4.
[12] R. Kimmel, J.A. Sethian, Fast Marching Methods on Triangulated Domains, Proc. Natl. Acad. Sci. USA 95 (1998) 8341–8435.
[13] R. Malladi, J.A. Sethian, B.C. Vemuri, Evolutionary fronts for topology-independent shape modeling and recovery, in: Proceedings of Third European Conference on Computer Vision, Stockholm, Sweden, Lecture Notes in Computer Science, vol. 800, 1994, pp. 3–13.
[14] R. Malladi, J.A. Sethian, B.C. Vemuri, Evolutionary fronts for topology-independent shape modeling and recovery, in: Proceedings of Third European Conference on Computer Vision, Stockholm, Sweden, Lecture Notes in Computer Science, vol. 800, 1994, pp. 3–13.
[15] R. Malladi, J.A. Sethian, An $O(N \log N)$ algorithm for shape modeling, Proc. Natl. Acad. Sci. USA 93 (1996) 9389–9392.
[16] S. Osher, J.A. Sethian, Fronts propagating with curvature-dependent speed: algorithms based on Hamilton–Jacobi formulations, J. Comput. Phys. 79 (1988) 12–49.

[17] J.A. Sethian, An analysis of flame propagation, Ph.D. Dissertation, Department of Mathematics, University of California, Berkeley, CA, 1982.
[18] J.A. Sethian, Curvature and the evolution of fronts, Comm. Math. Phys. 101 (1985) 487–499.
[19] J.A. Sethian, Numerical methods for propagating fronts, in: P. Concus, R. Finn (Eds.), Variational Methods for Free Surface Interfaces, Springer, New York, 1987.
[20] J.A. Sethian, A fast marching level set method for monotonically advancing fronts, Proc. Natl. Acad. Sci. USA 93 (4) (1996) 1591–1595.
[21] J.A. Sethian, Fast marching level set methods for three-dimensional photolithography development, in: Proceedings, SPIE 1996 International Symposium on Microlithography, Santa Clara, California, June, 1996.
[22] J.A. Sethian, Level Set Methods and Fast Marching Methods: Evolving Interfaces in Geometry, Fluid Mechanics, Computer Vision, and Materials Sciences, Cambridge University Press, Cambridge, 1999.
[23] J.A. Sethian, Fast marching methods, SIAM Review, 41, July, 1999.
[24] J.A. Sethian, M. Popovici, Fast marching methods applied to computation of seismic travel times, Geophysics 64 (2) (1999).
[25] J.A. Sethian, Evolution, implementation, and application of level set and fast marching methods for advancing fronts, J. Comput. Phys. 169 (2) (2001) 503–555.
[26] J.A. Sethian, A. Vladimirsky, Fast methods for the eikonal and related Hamilton–Jacobi equations on unstructured meshes, Proc. Natl. Acad. Sci. USA 97 (11) (2000) 5699–5703.
[27] J.A. Sethian, A. Vladimirsky, Ordered upwind methods for static Hamilton–Jacobi equations, Proc. Natl. Acad. Sci. USA 98 (2000) 11069–11074.
[28] Sethian JA, Vladimirsky A. Ordered Upwind Methods for Static Hamilton–Jacobi Equations: Theory and Algorithms, CPAM Report 2001, Berkeley, SIAM Numerical Analysis, 2002, in press.
[29] H.A. Stone, A simple derivation of the time-dependent convection–diffusion equation for surfactant transport along a deforming interface, Phys. Fluids A 2 (1) (1989) 111.
[30] J.A. Sethian, J.D. Strain, Crystal growth and dendrite formation, J. Comput. Phys. 98 (2) (1992) 231–253.
[31] J.N. Tsitsiklis, Efficient algorithms for globally optimal trajectories, IEEE Trans. Autom. Control 40 (1995) 1528–1538.
[32] H. Wong, D. Rumschitzki, Maldarelli, On the surfactant mass balance at a deforming fluid interface, Phys. Fluids 8 (11) (1996) 3202.
[33] H.-K. Zhao, T. Chan, B. Merriman, S. Osher, A variational level set approach to multiphase motion, J. Comput. Phys. 127 (1996) 179–195.