Transport and Diffusion of Material Quantities on Propagating Interfaces via Level Set Methods [1]

David Adalsteinsson
david@amath.unc.edu

Dept. of Mathematics
University of North Carolina, Chapel Hill

J.A. Sethian

Dept. of Mathematics
University of California, Berkeley

Submitted: Journal of Computational Physics
April, 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.

Traditionally, interface motion has been characterized by a discrete version of the differential geometry representation; in this view, a discretization of a parameterized representation is used to characterize the interface. Known in different fields with different names, these marker methods (string methods, point methods) allow accurate characterizations of evolving interfaces. They

[1]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, U.S. Dept. of Energy, #DE-AC03-76SF00098, and the Division of Mathematical Sciences, National Science Foundation.

form the backbone of a large collection of everyday simulations, include those in fluid mechanics, combustion and flame propagation, and materials sciences. Advantages to this approach include the ability to simultaneously represent the interface by a collection of marker points that move and allow each marker to carry variable quantities to be used to solve associated partial differential equations. 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 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.

Some 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.

In response, two alternative Eulerian numerical techniques have been developed to track interfaces. These include Volume of Fluid techniques (also known as cell methods) and level set methods. Volume of fluid techniques characterize the interface by discretizing the computational domain into a collection of cells that are each assigned a volume fraction denoting the portion of the interface contained in that cell. Level set methods work by embedding the interface as the zero level set of a higher dimensional function. Both techniques extend to three–dimensions and easily handle 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, these two techniques are not easily applied to problems in which a material quantity is required to be evaluated along the interface; fundamentally, this is because both methods work 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.

2 Equations of Motion

Consider a problem in which an interface moves under a velocity which depends on the value of a quantity carried on the surface, as well as location and local geometric properties such as normal direction and curvature. As an example, the motion of a fluid interface can depend on the vorticity distribution along the front. Let $G(x)$ be the value of the surface quantity at a point 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 curve $\Gamma(t, s)$ in which s parameterizes the curve at any fixed time t. Associated with this interface is a function $G(\Gamma(s,t))$, which gives the value of some scalar quantity along the interface.

We consider the following motions:

- **Advection:** The interface is passively advected under a velocity field $(u(x, y), v(x, y))$. This motion also transports the material quantity G along with the interface.
- **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 [15, 16, 19]). This motion also propagates the material quantity in conjunction with the propagating front.
- **Diffusion:** The material quantity G is allowed to diffuse along the interface according to the standard diffusion model.

The above description is not necessarily complete. We consider two examples. First, consider a circle with the function G constant along the circle, and suppose we wish to examine propagation normal to the curve with unit speed. It seems reasonable to assume that the total amount of G along the curve should be conserved under this motion, thus G must decrease as the circle expands. Thus, 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, we make the following additional assumption: we assume that under propagation, G is conserved. The implication of this is that 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.

2.2 Level Set Methods

Level set methods, introduced by Osher and Sethian [12], rely in part on the theory of curve and surface evolution given in [15] and on the link between front propagation and hyperbolic conservation laws discussed in [16]. 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 level set method describes the front by defining a function, $\phi(x)$, on all of space, which embeds the interface as its zero level set. The function is then evolved according to the equation

$$\phi_t(x,t) - F(x,t)|\nabla\phi(x,t)| = 0 \tag{1}$$

Here $F(x,t)$ is the normal speed of the front if x lies on the front. Topological changes are handled naturally in this setting, and the front can be tracked with a high degree of accuracy. Terms such as the normal and curvature of the front can be computed with high accuracy; this will be important for our problem.

We point out that there has been previous work on the description of material quantities in the context of level set methods. In [3], work on solving diffusion-type equations on surfaces has been considered in an implicit representation, and in [6], 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.

2.3 Statement of Problem: Implicit Formulation, Two Dimensions

We now derive a set of equations for an implicit view of the above transport, propagation and diffusion of material quantities in association with an evolving front which corresponds to a curve propagating in two space dimensions. We combine each term separately using operator splitting.

We begin by supposing 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 now derive evolution equations for ϕ and G_{ext} which produce implicit descriptions of the desired action on the front itself. We do so term by term.

2.3.1 Diffusion

We begin with the diffusion equation in the case of a stationary front. Let (x, y) be a point on the front, Let $G((x, y), t)$ be the scalar defined in all of space which agrees with G_{front} on the stationary interface, and let $\phi(x, y)$ be the signed distance function from this 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

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

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

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

The equations for x', x'', y', y'' are given by

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

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

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

$$\begin{aligned} G_t = & \left[G_{xx}\frac{\phi_y}{L} - G_{xy}\frac{\phi_x}{L}\right]\frac{\phi_y}{L} - \frac{G_x\phi_x}{L}\kappa \\ - & \left[G_{xy}\frac{\phi_y}{L} - G_{yy}\frac{\phi_x}{L}\right]\frac{\phi_x}{L} - \frac{G_y\phi_y}{L}\kappa \\ = & \frac{G_{xx}\phi_y^2 - 2G_{xy}\phi_x\phi_y + G_{yy}\phi_x^2}{L^2} - \frac{G_x\phi_x + G_y\phi_y}{L}\kappa \end{aligned}$$

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.

2.3.2 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). To find how the values change, we split the advection into two parts, corresponding to passive advection by the velocity field and to local compression/expansion of the front.

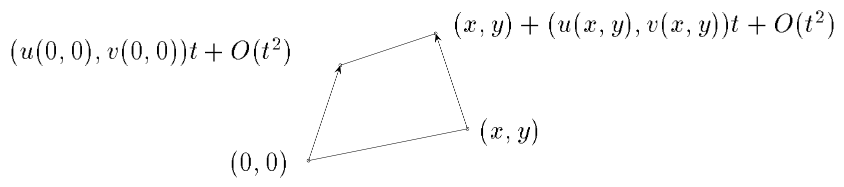

$(u(0,0), v(0,0))t + O(t^2)$
$(x, y) + (u(x, y), v(x, y))t + O(t^2)$
(x, y)
$(0, 0)$

Figure 1: Two-dimensional advection

To get the scaling of the segment length, we expand everything out to get the limit as the segment length and the time step go to zero (see Figure 1.

Expanding everything out, we get that the new segment is

$$(x + u(x, y)t - u(0, 0)t, y + v(x, y)t - v(0, 0)t) + O(t^2)$$

Define the normal of the line segment as (n_x, n_y), and $s = -n_x/n_y$, so $y = sx$ as (x, y) go to $(0, 0)$. If the length at time t is $L(t)$ we have that

$$\begin{aligned} L(t) &= ||(x + (u(x, sx) - u(0, 0))t, sx + (v(x, sx) - v(0, 0))t)|| + O(t^2) \\ &= |x| \cdot ||(1 + (u_x(0, 0) + su_y(0, 0))t, s + (v_x(0, 0) + sv_y(0, 0))t)|| + O(x^2 + t^2) \end{aligned}$$

and

$$L(0) = |x|\sqrt{1 + s^2}$$

Then, omitting the $(0, 0)$, we see that

$$L(t) = |x|\sqrt{(1 + (u_x + su_y)t)^2 + (s + (v_x + sv_y)t)^2} + O(t^2 + x^2)$$

Since the integral of $G(t)$ over the rectangle is preserved, letting $x \to 0$, we get that

$$\begin{aligned} G(t)/G(0) &= \lim_{x\to 0} L(0)/L(t) \\ &= \frac{\sqrt{1 + s^2}}{\sqrt{(1 + (u_x + su_y)t)^2 + (s + (v_x + sv_y)t)^2}} + O(t^2) \\ &= 1 - \sqrt{1 + s^2}(1 + s^2)^{-3/2}(u_x + su_y + s(v_x + sv_y)) + O(t^2) \end{aligned}$$

, and hence the rate of change for $G(t)$ is given by

$$\frac{dG}{dt}(0) = -\frac{u_x + su_y + s(v_x + sv_y)}{1 + s^2} G(0)$$

Using the substitution $s = -n_x/n_y$, we have that the magnification factor is given by

$$\begin{aligned} M &= \frac{u_x + su_y + s(v_x + sv_y)}{1 + s^2} \\ &= \frac{n_y^2 u_x - n_x n_y u_y - n_x(n_y v_x - n_x v_y)}{n_y^2 + n_x^2} \\ &= n_y^2 u_x - n_x n_y (u_y + v_x) + n_x^2 v_y \end{aligned}$$

Thus, if $G(x, y, t)$ is the value as a function of space and time, then the time evolution of G is given by

$$G_t(x, y) = -(u, v) \cdot \nabla G - (n_y^2 u_x - n_x n_y (u_y + v_x) + n_x^2 v_y) G.$$

This term can be defined in all of space. The term (n_x, n_y) is the normal, which can be computed everywhere in space from the gradient of the level set surface. The velocity field (u, v) is either given everywhere, or gotten from the front by extending the u and v components [2].

The first term comes from passive advection, and the second from the compression/expansion. The second part is crucial, even if it is a little counter-intuitive. At first glance, it looks like the advection equation could be easily written down from a conservation argument, but that will give incorrect results. A simple example that can be used to test other suggested equations is the one where a circle is expanding in the velocity field

$$(u(x, y), v(x, y)) = (x, y)$$

away from the origin. Start with the scalar defined as 1 in all of space. From simple conservation arguments, the solution should be given as $G(x, y, t) = e^{-t}$, i.e. $G' = -G$.

2.3.3 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, where κ is the curvature. In this case, the evolution equation becomes

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

2.3.4 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.

$$\begin{aligned} G_t = \quad & \sigma\left[(G_{xx}n_y^2 - 2G_{xy}n_xn_y + G_{yy}n_x^2) - \kappa(\nabla G \cdot N)\right] \\ - \quad & (u,v)\cdot\nabla G - (n_y^2u_x - n_xn_y(u_y+v_x) + n_x^2v_y)G \\ - \quad & F[(N\cdot\nabla G) + \kappa G] \end{aligned}$$

2.4 Statement of Problem: Implicit Formulation, Three Dimensions

The equations for three dimensions are similar, and are derived in the Appendix. 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

$$\begin{aligned} G_t = \quad & \sigma\left[\Xi_\Phi[G_x,G_y,G_z] - \kappa_M(\nabla G\cdot N)\right] \\ - \quad & (u,v,w)\cdot\nabla G - \Xi_\Phi[u,v,w]G \\ - \quad & F[(N\cdot\nabla G) + \kappa_M G] \end{aligned}$$

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

$$\Xi_\Phi[G_x,G_y,G_z] = \begin{bmatrix} (G_{yy}+G_{zz})N_x^2 + (G_{xx}+G_{zz})N_y^2 + (G_{xx}+G_{yy})N_z^2 \\ -2G_{xy}N_xN_y - 2G_{xz}N_xN_z - 2G_{yz}N_yN_z \end{bmatrix}$$

3 Numerical Approximations and Algorithms

3.1 Level Set Methods

3.1.1 Equations of Motion

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. More precisely, given a moving closed hypersurface $\Gamma(t)$, that is, $\Gamma(t=0) : [0,\infty) \rightarrow R^N$, propagating with a speed F in its normal direction, we wish to produce an Eulerian formulation for the motion of the hypersurface propagating along its normal direction with speed F, where F can be a function of various arguments, including the curvature, normal direction, etc. Let $\pm d$ be the signed distance to the interface. If this propagating interface is embedded as the zero level set of a higher dimensional function ϕ, that is, let $\phi(x, t = 0)$, where $x \in R^N$ is defined by

$$\phi(x, t = 0) = \pm d, \tag{2}$$

then an initial value partial differential equation can be obtained for the evolution of ϕ, namely

$$\phi_t + F|\nabla\phi| = 0 \tag{3}$$

$$\phi(x, t = 0) \quad \text{given} \tag{4}$$

This is the implicit formulation of front propagation given in [12]. As discussed in [14, 15, 16], propagating fronts can develop shocks and rarefactions in the slope, corresponding to corners and fans in the evolving interface, and numerical techniques designed for hyperbolic conservation laws can be exploited to construct schemes which produce the correct, physically reasonable entropy solution.

There are certain advantages associated with this perspective. First, it is unchanged in higher dimensions; that is, for surfaces propagating in three dimensions and higher. Second, topological changes in the evolving front Γ are handled naturally; the position of the front at time t is given by the zero level set $\phi(x, y, t) = 0$ of the evolving level set function. This set need not be connected, and can break and merge as t advances. Third, terms in the speed function F involving geometric quantities such as the normal vector n and the curvature κ may be easily approximated through the use of derivative operators applied to the level set function, that is, $n = \frac{\nabla\phi}{|\nabla\phi|}$ and $\kappa = \nabla \cdot \frac{\nabla\phi}{|\nabla\phi|}$ Fourth, the upwind finite difference technology for hyperbolic conservation laws may be used to approximate the gradient operators.

3.1.2 Approximation Schemes

Entropy-satisfying upwind viscosity schemes for this initial value formulation were introduced in [12]. One of the simplest first order scheme is given as

$$\phi_{ijk}^{n+1} = \phi_{ijk}^{n} - \Delta t[\max(F_{ijk}, 0)\nabla^+ + \min(F_{ijk}, 0)\nabla^-], \tag{5}$$

where

$$\nabla^+ = \begin{bmatrix} \max(D_{ijk}^{-x}, 0)^2 + \min(D_{ijk}^{+x}, 0)^2 + \\ \max(D_{ijk}^{-y}, 0)^2 + \min(D_{ijk}^{+y}, 0)^2 + \\ \max(D_{ijk}^{-z}, 0)^2 + \min(D_{ijk}^{+z}, 0)^2 \end{bmatrix}^{1/2}$$

$$\nabla^- = \begin{bmatrix} \max(D_{ijk}^{+x}, 0)^2 + \min(D_{ijk}^{-x}, 0)^2 + \\ \max(D_{ijk}^{+y}, 0)^2 + \min(D_{ijk}^{-y}, 0)^2 + \\ \max(D_{ijk}^{+z}, 0)^2 + \min(D_{ijk}^{-z}, 0)^2 \end{bmatrix}^{1/2}$$

Higher order schemes are available, see [12].

The above formulation reveals two central embeddings. First, in the initialization step (Eqn. 2), the signed distance function is used to build a function ϕ which corresponds to the interface at the level set $\phi = 0$. This step is known as "initialization"; when performed at some later point in the calculation beyond $t = 0$, it is referred to as "re–initialization".

Second, the construction of the initial value PDE given in Eqn. 3 means that the velocity F is now defined for **all** the level sets, not just the zero level set corresponding to the interface itself. We can be more precise by rewriting the level set equation as

$$\phi_t + F|\nabla\phi| = 0, \tag{6}$$

where F is some velocity field which, at the zero level set, equals the given speed F_{front}. In other words,

$$F = F \text{ on } \phi = 0.$$

This new velocity field F is known as the "extension velocity".

Both of these issues need to be confronted in order to efficiently apply level set methods to complex computational problems.

3.1.3 Adaptivity: The Narrow Band Level Set Method

Equation 5 is an explicit scheme, and hence can be solved directly. The time step requirement depends on the nature of the speed function F; for an F that depends only on position, the time step behaves like $\frac{\Delta t}{\Delta x}F \leq 1$. In the case when the speed function F depends on curvature terms (for example, $F = -\kappa$), the equation has a parabolic component, and hence the time step requirement resembles that of a non-linear heat equation; the time step depends roughly on $\frac{\Delta t}{\Delta x^2}$. In the level set formulation, both the level set function and the speed are embedded into a higher dimension. Considerable computational speedup in the level set method comes from the use of the "Narrow Band Level Set Method", introduced by Adalsteinsson and Sethian in [1], which 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 [4], used in recovering shapes from images in Malladi, Sethian and Vemuri [10], 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].

3.1.4 Reinitialization

Reinitialization is the process by which the level set function is periodically recalibrated against the front itself in order to reset the signed distance function. As understood by many practitioners of level set methods (see [19, 22]), reinitialization every time step (or close to every time step) is a poor strategy, since each reinialization causes error in the position of the front. However, occasional reinitialization is required when coupled to the Narrow Band method.

Reinitialization, that is, resetting the level set function ϕ to correspond to the signed distance function, is efficiently and accurate performed using Fast Marching Methods. Fast Marching Methods, [17], are Dijkstra-like upwind finite difference algorithms which solve the Eikonal equation

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

in $ON\log N$ 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 central idea is as follows; begin by use an upwind finite difference approximation to the gradient of the form

$$\left[\begin{array}{c} \max(D^{-x}_{ijk}T, -D^{+x}_{ijk}T, 0)^2 + \\ \max(D^{-y}_{ijk}T, -D^{+y}_{ijk}T, 0)^2 + \\ \max(D^{-z}_{ijk}T, -D^{+z}_{ijk}T, 0)^2 \end{array}\right]^{1/2} = F_{ijk}, \tag{7}$$

(see, for example, Rouy and Tourin [13].) The key observation in Fast Marching Methods [17] is that the above contains 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 [5]. 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 [20], first order unstructured meshes by Kimmel and Sethian [8], and higher order unstructured meshes by Sethian and Vladimirsky [23]. Some early applications include photolithography in [18], a comparison of a similar approach with volume-of-fluid techniques in [7], a fast algorithm for image segmentation in [11] and computation of seismic travel times by Sethian and Popovici [21];

A different Dijsktra-like method for the Eikonal equation was developed by Tsitsiklis [25]; he obtains the viscosity solution through a control-theoretic discretization which hinges on a causality relationship based on the optimality criterion. The Fast Marching Method is an upwind finite difference techniques, while Tsitsiklis' method relies on a minimization scheme based on an optimality criterion. The two techniques use different formulations and different discretizations; in the case of a first order formulation, it was later observed that they ultimately invoke the same quadratic formula in the update For details on the two techniques, see [25, 17]. In addition, we note that Fast Marching Methods lend themselves to higher order schemes in a natural way. For details on using the Fast Marching Method to perform reinitialization in the context of level set methods, see [2]. We also note that in recent work, Sethian and Vladimirsky have been able to produce general schemes for non-Eikonal anisotropic general optimal control problems with the same computational complexity as these Dijkstra-like methods; for details, see [23, 24].

3.1.5 Construction of Extension Velocities

In order to construct extension velocities F, we start with the given velocity F_{front} and choose an appropriate extension velocity. There are several reasons why one needs to build these extension velocities, including the fact that no natural velocity may be available off of the front, the need for sub-grid resolution, the need for accurate representation of front velocities, and the need to maintain a nice level set representation (see [19, 22] for details). In [9], the idea of extrapolating the given front velocity along the gradient of the front to obtain an extension velocity off of the front was introduced, and used in image segmentation. Mathematically, this means that

$$\nabla F \cdot \nabla \phi = 0. \tag{8}$$

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, [26]).

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 Eqn. 8. This velocity is used to update the level set function ϕ^n. For details, see [2].

3.2 Algorithms for Diffusion and Transport of Material Quantities

The general form of the update equation is

$$\begin{aligned} G_t = \quad & \sigma\left[\Xi_\Phi[G_x, G_y, G_z] + \kappa_M(\nabla G \cdot N)\right] \\ + \quad & (u, v, w) \cdot \nabla G - \Xi_\Phi[u, v, w]G \\ + \quad & F[(N \cdot \nabla G) - \kappa_M G] \end{aligned}$$

We update with the discrete value for ϕ_{ij}^n using the upwind schemes in [12]. We update the values of the material quantity G_{ij}^n using central differences for the diffusion terms and upwinding for the advection terms.

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

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

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, $\left(r\cos\left(\frac{s}{r}\right), r\sin\left(\frac{s}{r}\right)\right)$, the initial data is given in terms of arc length by $r\cos\left(\frac{s}{r}\right)$. 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. When you look at the scalar as a 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 that we get by interpolating G onto uniformly spread points on the interface. We then use 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 Figure 2. From this we see that convergence is second order overall.

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. The initial front that is chosen is a perturbed circle, 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].

See Figure 3 for a picture of the front and a plot of G as a function of arc length. This scalar has a full spectrum of frequencies. We can solve this problem exactly with a Fourier series method. To find the exact solution at a later time T, extract the initial scalar as a function of arc length, decompose it into it's Fourier modes, 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

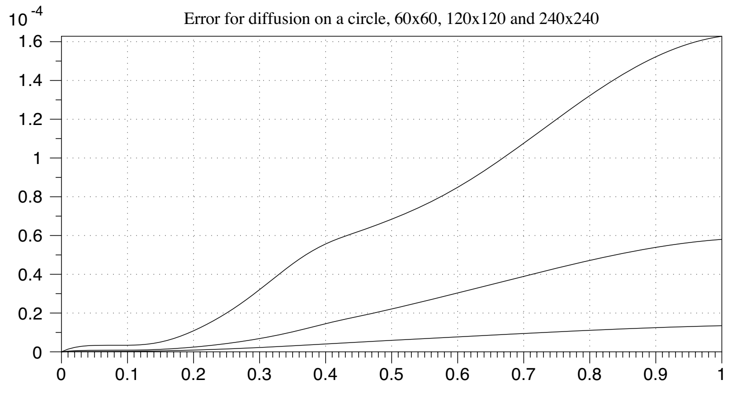


Figure 2: Front fixed, values diffused on a circle.

points and subtracted. This will give the error as a function of arc length. Then compute the L_2 norm of that function and divide by the length to get the average error. This error is shown in Figure 4; 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. Start with an ellipse with major axis 0.4 and 0.3, centered in a unit square box (Figure 5). 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 Figure 6. 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 us a good indication of what the overall error is for the front evolution. The infinity norm gives a very similar result, since the pointwise error is fairly

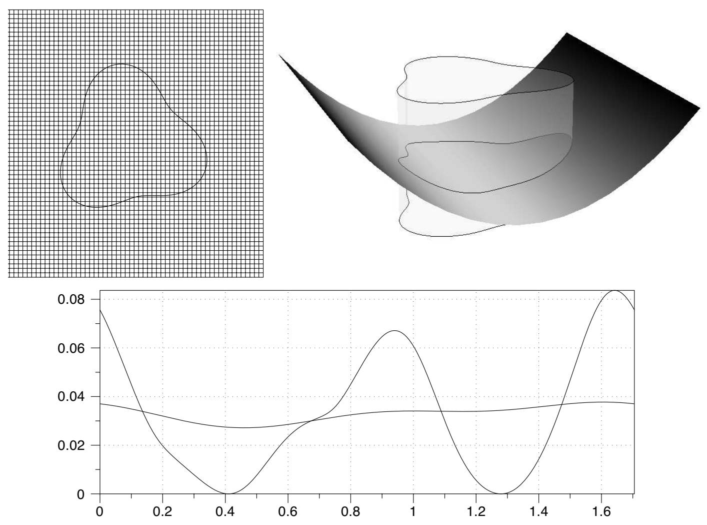


Figure 3: Diffusion on a more general front. Initial value given by $F(x, y) = x^2$. Later value at $T = 1$. Both values are drawn as functions of arc length.

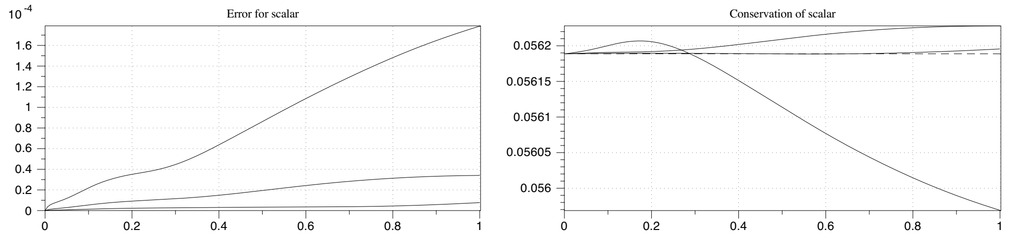


Figure 4: Diffusion on a more general front. Error along the front as a function of time for the grid sizes 60×60,120×120 and 240×240.

continuous. We do this for two hundred time values and plot the result as a function of time for both grid resolutions.

- 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. This gives us 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

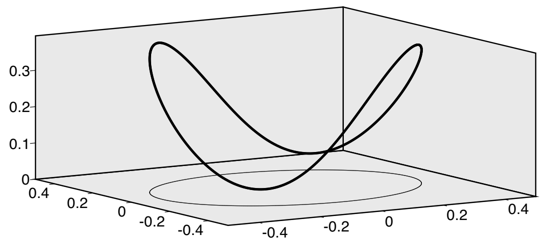
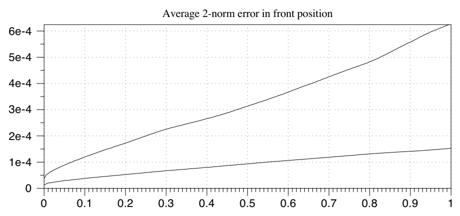


Figure 5: Initial scalar value along the ellipse.

time in Figure 7. 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.



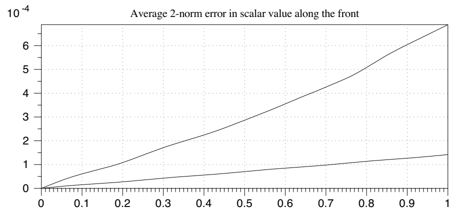


Figure 6: How accurately the level set method captures the rotation of the front, and the error along the front.

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, and use that in that representation, the scalar satisfies the heat equation with diffusion constant 0.1, which is solved using Fourier series.

We run the same numerical experiments as in the section before, but in order to compute the exact solution, we extract the values on the initial front by arc length (using bi–cubic interpolation), define it on an evenly spaced one dimensional grid in arc length. 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.

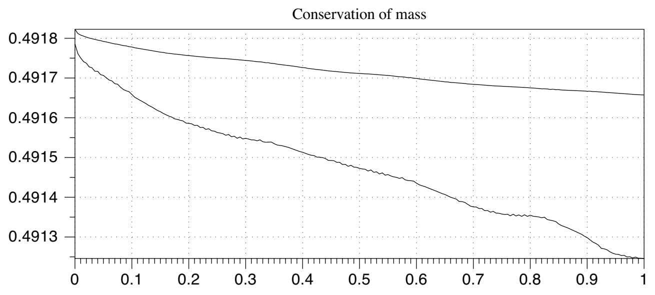
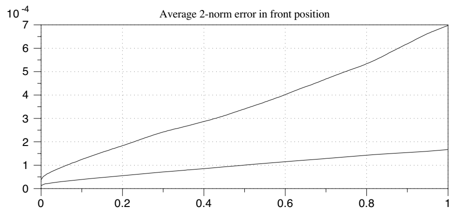


Figure 7: How well mass is conserved along the ellipse. No diffusion, rotated around the origin.



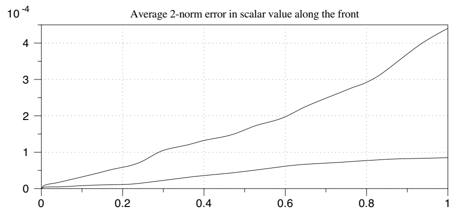


Figure 8: How accurately the level set method captures the rotation of the front, and the error along the front.

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, the error should be very smooth. High noise in the error might indicate a problem with the interpolation routine.

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

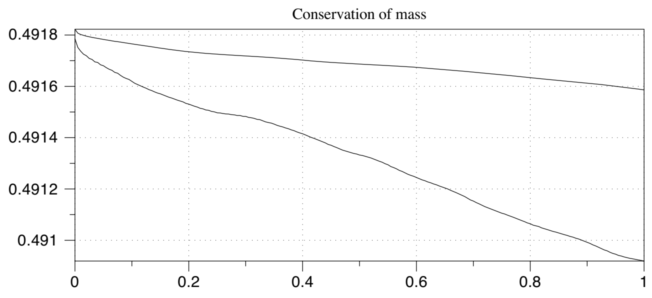


Figure 9: How well mass is conserved along the ellipse. Rotated around the origin, and diffused.

diffusion is proportional to the flux gives

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

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 figures 10 and 11.

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 Figure 12. Initial values along the surface are given by the expression $x+y+z$.
To analyze the accuracy, we track three measurements.

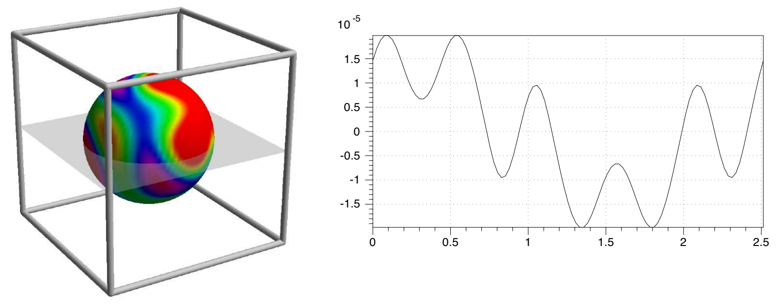


Figure 10: The error along the surface is colored according to value. The plane slices through those values and the resulting error is shown along the front as a function of arc length. This is the error at $t = 0.25$ for the grid spacing $1/100$.

- 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 Figure 13.
- Along the surface, compute the error in the value of the scalar. Then compute the norm of the error over the whole surface. The result is shown in Figure 14.
- The integral of the scalar over the surface. This should be conserved over time. The result is shown in Figure 15.

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 $240 \times 240 \times 120$.

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 rotatational 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 Figure 16.

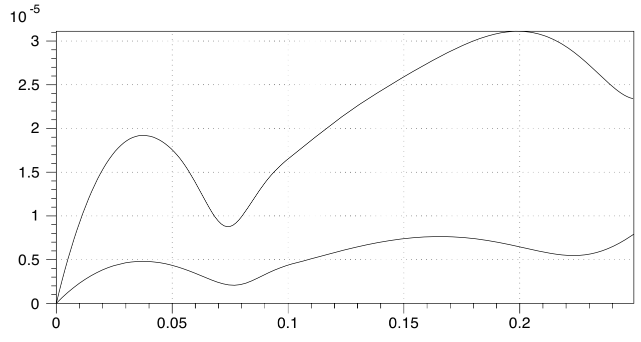


Figure 11: Diffusion on a stationary sphere. How the error changes over time for the exact solution. Plot of the average two-norm (L^2) of the error as a function of time for both $h = 1/50$ and $h = 1/100$. The error is computed by interpolating the values onto the front, and subtracting the known solution 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.

6 Appendix: Derivation of Three-Dimensional Equations

6.1 Diffusion

For a point $\bar{x}$ on a level surface, let Γ be the surface going through that point. We will derive the rate of change for G at that point, using that flux across a boundary is proportional to the gradient, with proportionality constant σ, which we will set to 1 for the derivation.

Assume $\bar{x} = 0$. Let M be an orthagonal matrix which maps the normal n at $\bar{x}$ into $(0,0,1)$. If Φ is the level set function, let Ψ be the rotated level set function and W the rotated G function.

$$G(\bar{x}) = W(M\bar{x}), \qquad \Phi(\bar{x}) = \Psi(M\bar{x})$$

In this new coordinate system, we can write the z coordinate in terms of the first two components of the point, i.e. we can parametrize a rectangular patch with $(x, y, z(x, y))$ that will satisfy

$$\Psi(x, y, z(x, y)) = 0.$$

By differentiating this equation with respect to x and y we get

$$\begin{aligned} z_x(0,0) &= 0 \\ z_y(0,0) &= 0 \end{aligned}$$

and a differentiating it twice with respect to x and y, we get

$$z_{xx}(0,0) = -\frac{\Psi_{xx}(0)}{L}$$

Figure 12: Four time values for rotating a sphere.

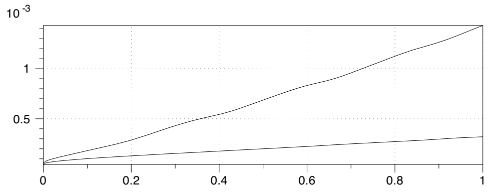
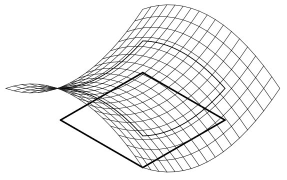


Figure 13: Error for location of front.

$$z_{yy}(0,0) \quad = \quad -\frac{\Psi_{yy}(0)}{L}$$

.

Take a small region of the zero level set of Ψ around 0. Take the region defined by the box $[0,h] \times [0,h]$ in the xy coordinate plane. We compute the flux out of this rectangle, and how that will affect the W value, and then let $h \to 0$.



First, take the edges where $x = 0$ and $x = h$. The gradient of W in that surface along the edges is

$$\nabla W(x, y, z(x,y)) \cdot (1, 0, z_x(x,y)).$$

Figure 16: Advection and Diffusion in three dimensions The motion of the surface. Initial condition is transparent. Surface is rotated around the origin.

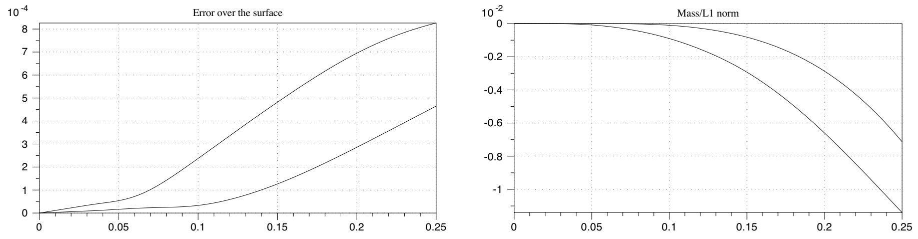


Figure 17: Advection and Diffusion in three dimensions. Error behavior for $70 \times 70 \times 50$ and $140 \times 140 \times 100$. On the left is a measurement of the error compared to the exact solution, on the right how close it is to conserving mass.

equations for z_x and z_y to get

$$\begin{aligned}
\text{Flux}_x &= \int_0^h [W_x(0) + W_{xx}(0)h + W_{xy}(0)y + W_z(0)z_{xx}(0,0)h + W_z(0)z_{xy}(0,0)y \\
&\quad -(W_x(0) + W_{xy}(0)y + W_z(0)z_{xy}(0,0)y) + O(h^2)]dy \\
&= \int_0^h [W_{xx}(0)h + W_z(0)z_{xx}(0,0)h]dy + O(h^3) \\
&= [W_{xx}(0) + W_z(0)z_{xx}(0,0)]h^2 + O(h^3)
\end{aligned}$$

Similar equation holds for the flux through $y = 0$ and $y = h$, so the total flux is

$$(W_{xx}(0) + W_{yy}(0) + W_z(0)z_{xx}(0,0) + W_z(0)z_{yy}(0,0))h^2 + O(h^3)$$

Dividing by the area of the rectangle and taking the limit as $h \to 0$, we get that

$$W_t(0) = W_{xx}(0) + W_{yy}(0) + W_z(0)(z_{xx}(0,0) + z_{yy}(0,0)).$$

We have already derived equations for z_{xx} and z_{yy} in terms of Ψ, so

$$W_t(0) = \sigma \left[W_{xx}(0) + W_{yy}(0) - \frac{W_z(0)}{L}(\Psi_{xx}(0) + \Psi_{yy}(0)) \right]$$

The next step is to move this back to an expression in terms of F and Φ. We know that the Laplace operator is independent of rotation, so that $\Delta V = \Delta W$, and $\Delta\Phi = \Delta\Psi$. Therefore

$$\begin{aligned}
V_t(0) &= W_t(0) = \sigma \left[\Delta W(0) - W_{zz}(0) - \frac{W_z(0)}{L}(\Delta\Psi(0) - \Psi_{zz}(0)) \right] \\
&= \sigma \left[\Delta W(0) - \mathbf{e}_3^T D^2 W(0)\mathbf{e}_3 - \frac{\mathbf{e}_3^T \nabla W}{L}(\Delta\Psi(0) - \mathbf{e}_3^T D^2\Psi(0)\mathbf{e}_3) \right] \\
&= \sigma \left[\Delta V(0) - n^T D^2 V(0) n - \frac{n^T \nabla V}{L}(\Delta\Phi(0) - n^T D^2\Phi(0) n) \right]
\end{aligned}$$

The term

$$\Delta\Phi(0) - n^T D^2 \Phi(0) n$$

is the mean curvature of the surface at the point. $n^T \nabla F = F_n$, so if the F is computed through an extension algorithm, this is zero. Since it wil not remain zero, the term should be included as a correction.

The term

$$\Delta F(0) - n^T D^2 F(0) n$$

where

$$n = \frac{1}{L}(\Phi_x, \Phi_y, \Phi_z)$$

appears for $F = V$ and $F = \Phi$ in the above equation. The next step is to simplify this term, and write it in terms of derivatives that are easy to compute.

$$\begin{aligned} \Delta F(0) - n^T D^2 F(0) n &= \frac{1}{L^2}\left[\Phi_x^2(F_{yy} + F_{zz}) + \Phi_y^2(F_{xx} + F_{zz}) + \Phi_z^2(F_{xx} + F_{yy})\right. \\ &\quad \left. - 2\Phi_x\Phi_y F_{xy} - 2\Phi_x\Phi_z F_{xz} - 2\Phi_y\Phi_z F_{yz}\right] \end{aligned}$$

If the normal $N = (n_x, n_y, n_z)$, this simplifies to

$$\begin{aligned} \Delta F(0) - n^T D^2 F(0) n &= n_x^2(F_{yy} + F_{zz}) + n_y^2(F_{xx} + F_{zz}) + n_z^2(F_{xx} + F_{yy}) \\ &\quad - 2n_x n_y F_{xy} - 2n_x n_z F_{xz} - 2n_y n_z F_{yz} \end{aligned}$$

6.2 Advection

We begin with a small paralleloid at the origin spanned with two vectors $(h, 0, sh)$, $(0, h, rh)$ where $s = -n_x/n_z$, $r = -n_y/n_z$. We compute the rate at which the area changes, which will give the rate of change for the scalar value V at that point (opposite sign). Consider the three points $(0,0,0)$, $(h, 0, sh)$, $(0, h, rh)$. In a short time t, these points map into

$$\begin{gathered} (u(0,0,0)t, v(0,0,0)t, w(0,0,0)t) \\ (h + u(h,0,sh)t, v(h,0,sh)t, sh + w(h,0,sh)t) \\ (u(0,h,rh)t, h + v(0,h,rh)t, rh + w(0,h,rh)t) \end{gathered}$$

The area of these two paralleloids is produced using the cross product. For the first one, we have that

$$(0, h, rh) \times (h, 0, sh) = (sh^2, rh^2, -h^2)$$

that has length

$$h^2\sqrt{1 + r^2 + s^2}$$

The new paralleloid is spanned by the vectors

$$(u(0,h,rh)t - u(0,0,0)t, h + v(0,h,rh)t - v(0,0,0)t, rh + w(0,h,rh)t - w(0,0,0)t)$$

$(h+u(h,0,sh)t-u(0,0,0)t,\, v(h,0,sh)t-v(0,0,0)t,\, sh+w(h,0,sh)t-w(0,0,0)t)$

Expanding them in terms of h around the origin, ignoring $O(h^2)$ terms and dropping $(0,0,0)$, we have that

$$(th(u_y+ru_z), h+th(v_y+rv_z), rh+th(w_y+rw_z))$$
$$(h+th(u_x+su_z), th(v_x+sv_z), sh+th(w_x+sw_z)).$$

The length of the cross product of these two vectors is

$$\sqrt{1+s^2+r^2+2tD+O(t^2)} = \sqrt{1+s^2+r^2}\left(1+\frac{tD}{1+s^2+r^2}+O(t^2),\right)$$

where the value of D is

$$\begin{aligned} D = & \quad s(w_x+sw_z+s(v_y+rv_z)-r(v_x+sv_z)) \\ + & \quad r(r(u_x+su_z)+(w_y+rw_z)-s(u_y+ru_z)) \\ + & \quad (u_x+su_z+v_y+rv_z) \\ = & \quad (u_x+v_y)+s(w_x+u_z)+r(w_y+v_z) \\ + & \quad s^2(w_z+v_y)+r^2(u_x+w_z)-rs(v_x+u_y) \end{aligned}$$

By computing the ratios of these lengths, we get as before, since $s=-n_x/n_z$, $r=-n_y/n_z$, $1+s^2+r^2=1/n_z^2$, the change due to the compression/expansion is given by

$$V_t = -\frac{D}{1+s^2+r^2}V = -n_z^2 DV = -KV$$

where

$$\begin{aligned} K = n_z^2 D \quad = & \quad n_z^2(u_x+v_y)-n_xn_z(w_x+u_z)-n_yn_z(w_y+v_z) \\ + & \quad n_x^2(w_z+v_y)+n_y^2(u_x+w_z)-n_xn_y(v_x+u_y) \end{aligned}$$

In addition, we need to passively advect the scalar field. Combining these two terms gives us the total evolution equation

$$V_t = -(u,v,w)\cdot\nabla V - KV$$

6.3 Normal Advection

When $(u,v,w)=F(n_x,n_y,n_z)$, where F can be a function of space, the above equation simplifies into

$$V_t = -F[(N\cdot\nabla V)+\kappa_M V]$$

where κ_M is the mean curvature

$$\begin{aligned} \kappa_M = (& \quad \Phi_z^2(\Phi_{xx}+\Phi_{yy})-2\Phi_x\Phi_z\Phi_{xz}-2\Phi_y\Phi_z\Phi_{yz} \\ + & \quad \Phi_x^2(\Phi_{yy}+\Phi_{zz})+\Phi_y^2(\Phi_{xx}+\Phi_{zz})-2\Phi_x\Phi_y\Phi_{xy}) \\ & \quad /(\Phi_x^2+\Phi_y^2+\Phi_z^2)^{3/2} \end{aligned}$$

6.4 Final Equations

Combining the previous equations, we develop an expression for the case in which σ is the diffusion coefficient and the front is moved with a combination of an advection field (u, v, w) and a normal speed F. 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

$$\kappa_M = \Xi_\Phi[\Phi_x, \Phi_y, \Phi_z]$$

and the differential equation simplifies to

$$\begin{aligned} V_t = & \quad \sigma\left[\Xi_\Phi[V_x, V_y, V_z] + \kappa_M(\nabla V \cdot N)\right] \\ - & \quad (u,v,w)\cdot\nabla V - \Xi_\Phi[u,v,w]V \\ - & \quad F[(N\cdot\nabla V) + \kappa_M V] \end{aligned}$$

, with first term equal to

$$\Xi_\Phi[V_x, V_y, V_z] = \begin{bmatrix} (V_{yy} + V_{zz})N_x^2 + (V_{xx} + V_{zz})N_y^2 + (V_{xx} + V_{yy})N_z^2 \\ -2V_{xy}N_xN_y - 2V_{xz}N_xN_z - 2V_{yz}N_yN_z \end{bmatrix}$$

References

[1] Adalsteinsson, D., and Sethian, J.A., *A Fast Level Set Method for Propagating Interfaces*, J. Comp. Phys., 118, 2, pp. 269–277, 1995.

[2] Adalsteinsson, D., and Sethian, J.A., *The Fast Construction of Extension Velocities in Level Set Methods*, J. Comp. Phys., 148, 1999, pp. 2-22.

[3] Bertalmio, M., Sapiro, G., Cheng, L.T., Osher, S. *A Framework for Solving Surface Partial Differential Equations for Computer Graphics Applications*, CAM report (43) 2000.

[4] Chopp, D.L., *Computing Minimal Surfaces via Level Set Curvature Flow*, Jour. of Comp. Phys., 106, pp. 77–91, 1993.

[5] Dijkstra, E.W., *A Note on Two Problems in Connection with Graphs*, Numerische Mathematic, 1:269–271, 1959.

[6] Harabetian, E., Osher, S., and Shu, C-W., *An eulerian approach for vortex motion using a level set regularization procedure*, Journal Computational Physics, 127, 1, 15 - 26, 1996.

[7] Helmsen, J.J., *A Comparison of Three-Dimensional Photolithography Development Methods,* Ph.D. Dissertation, EECS, University of California, Berkeley, CA, 1994.

[8] Kimmel, R., and Sethian, J.A., *Fast Marching Methods on Triangulated Domains*, Proc. Nat. Acad. Sci., 95, pp. 8341-8435, 1998.

[9] Malladi, R., Sethian, J.A., and Vemuri, B.C., *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, 800, pp. 3–13, 1994.

[10] Malladi, R., Sethian, J.A., and Vemuri, B.C., *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, 800, pp. 3–13, 1994.

[11] Malladi, R., and Sethian, J.A., *An $O(N \log N)$ Algorithm for Shape Modeling*, Proc. Nat. Acad. Sci., Vol. 93, pp. 9389-9392, 1996.

[12] Osher, S., and Sethian, J.A., *Fronts Propagating with Curvature-Dependent Speed: Algorithms Based on Hamilton–Jacobi Formulations*, Journal of Computational Physics, 79, pp. 12–49, 1988.

[13] Rouy, E. and Tourin, A., *A Viscosity Solutions Approach to Shape-From-Shading*, SIAM J. Num. Anal, 29, 3, pp. 867–884, 1992.

[14] Sethian, J.A., *An Analysis of Flame Propagation*, Ph.D. Dissertation, Dept. of Mathematics, University of California, Berkeley, CA, 1982.

[15] Sethian, J.A., *Curvature and the Evolution of Fronts*, Comm. in Math. Phys., 101, pp. 487–499, 1985.

[16] Sethian, J.A., *Numerical Methods for Propagating Fronts*, in Variational Methods for Free Surface Interfaces, Eds. P. Concus and R. Finn, Springer-Verlag, NY, 1987.

[17] Sethian, J.A., *A Fast Marching Level Set Method for Monotonically Advancing Fronts*, Proc. Nat. Acad. Sci., 93, 4, pp.1591–1595, 1996.

[18] Sethian, J.A., *Fast Marching Level Set Methods for Three-Dimensional Photolithography Development*, Proceedings, SPIE 1996 International Symposium on Microlithography, Santa Clara, California, June, 1996.

[19] Sethian, J.A., Level Set Methods and Fast Marching Methods: Evolving Interfaces in Geometry, Fluid Mechanics, Computer Vision, and Materials Sciences, Cambridge University Press, 1999.

[20] Sethian, J.A., *Fast Marching Methods*, SIAM Review, 41, July, 1999.

[21] Sethian, J.A., and Popovici, M., *Fast Marching Methods Applied to Computation of Seismic Travel Times*, Geophysics, 64, 2, 1999.

[22] Sethian, J.A., *Evolution, Implementation, and Application of Level Set and Fast Marching Methods for Advancing Fronts* J. Comp. Phys., 169,2, pp. 503-555, 2001.

[23] Sethian, J.A. & Vladimirsky, A. (2000) *Fast Methods for the Eikonal and Related Hamilton–Jacobi Equations on Unstructured Meshes*, Proc. Nat. Acad. Sci., 97, 11, 5699-5703.

[24] Sethian, J.A. & Vladimirsky, A. (2000) *Ordered Upwind Methods for Static Hamilton-Jacobi Equations*, Proc. Nat. Acad. Sci., 98, 11069-11074.

[25] Tsitsiklis, J.N., *Efficient Algorithms for Globally Optimal Trajectories*, IEEE Transactions on Automatic Control, Volume 40, pp. 1528-1538, 1995.

[26] Zhao, H-K., Chan, T., Merriman, B., and Osher, S., *A Variational Level Set Approach to Multiphase Motion*, Jour. Comp. Phys., 127, pp. 179–195, (1996).