SIAM J. NUMER. ANAL.
Vol. 42, No. 6, pp. 2612–2632

FAST SWEEPING METHODS FOR STATIC HAMILTON–JACOBI EQUATIONS*

CHIU-YEN KAO†, STANLEY OSHER†, AND YEN-HSI TSAI‡

Abstract. We propose a new sweeping algorithm which discretizes the Legendre transform of the numerical Hamiltonian using an explicit formula. This formula yields the numerical solution at a grid point using only its immediate neighboring grid values and is easy to implement numerically. The minimization that is related to the Legendre transform in our sweeping scheme can either be solved analytically or numerically. We illustrate the efficiency and accuracy approach with several numerical examples in two and three dimensions.

Key words. fast sweeping method, Godunov Hamiltonian, static Hamilton–Jacobi equation

AMS subject classifications. 35F30, 65N06

DOI. 10.1137/S0036142902419600

1. Introduction.

The Hamilton–Jacobi equation

$$\psi_t(x,t) + H(x, \nabla\psi(x,t)) = 0 \tag{1.1}$$

arises in many applications ranging from classical mechanics to contemporary problems of optimal control. These include geometrical optics, crystal growth, etching, computer vision, obstacle navigation, path planning, photolithography, and seismology. In general, these nonlinear PDEs cannot be solved analytically. The solutions usually develop singularities in their derivatives even with smooth initial conditions. In these cases, the solutions do not satisfy the equation in the classical sense. The weak solution that is usually sought is called the viscosity solution [10]. Numerically, in general, one looks for a consistent and monotone scheme to construct approximate viscosity solutions [27].

In this paper, we focus on static Hamilton–Jacobi equations of the following form:

$$\begin{cases} H(x, \nabla\phi(x)) = R(x) & \text{for } x \in \Omega, \\ \phi(x) = q(x) & \text{for } x \in \Gamma \subset \partial\Omega, \end{cases} \tag{1.2}$$

where H, q, and $R > 0$ are Lipschitz continuous and H is also convex and homogeneous of degree one in $\nabla\phi(x)$. A special case of this type of equation is the eikonal equation,

$$|\nabla\phi| = r(x) \tag{1.3}$$

with the same type of Dirichlet boundary condition as in (1.2). Many numerical methods have been developed for this problem. Rouy and Tourin [24] used an iterative method to solve the discretized eikonal equation and proved that it converges to the viscosity solution. The key is to use an upwind, monotone, and consistent discretization for $|\nabla\phi|$. Instead of using iterative methods, Tsitsiklis [29], later Sethian [25],

*Received by the editors December 11, 2002; accepted for publication (in revised form) April 3, 2004; published electronically April 19, 2005. This research was supported by ONR grant N00014-03-1-0071 and ONR MURI grant N00014-02-1-0720.

http://www.siam.org/journals/sinum/42-6/41960.html

†Department of Mathematics, University of California Los Angeles, Los Angeles, CA 90095 (ckao@math.ucla.edu, sjo@math.ucla.edu).

‡Department of Mathematics and PACM, Princeton University, Princeton, NJ 08544 (ytsai@math.princeton.edu).

and Helmsen et. al. [14] proposed single-pass methods. Based on the monotonicity of the solution along the characteristics, they combined the heap-sort data structure with a variation of the classical Dijkstra algorithm to solve the steady state equation $|\nabla\phi| = r(\mathbf{x})$. This became known as the fast marching method whose complexity is $O(N \log N)$, where N is the total number of grid points in the domain. Later Sethian and Vladimirsky [26] generalized the method of [29] to solve (1.2).

Osher [18] provided a link between time-independent and time-dependent Hamilton–Jacobi equations. The zero level set of the viscosity solution ψ of (1.1) with suitable initial conditions at various time t is the solution $\phi(x, y) = t$ of (1.2). This gives an approach that one can try to solve the time-dependent equation by the level set formulation [19] with high order approximations on the partial derivatives [20], [15]. Falcone and Ferretti studied a class of semi-Lagrangian schemes which can be interpreted as a discrete version of the Hopf–Lax–Oleinik representation formula for first order time-dependent Hamilton–Jacobi equations. In semi-Lagrangian schemes, ψ needs to be interpolated using its grid values, the Legendre transformation of H needs to be obtained, and the minimum must be computed on an unbounded set. See [11] and the references therein for more details.

Another approach to obtaining a “time-dependent” Hamilton–Jacobi equation from a time-independent Hamilton–Jacobi equation comes by using the so-called paraxial formulation, i.e., by assuming that there is a preferred direction in the wave propagation. In [13], the paraxial formulation was first proposed for the eikonal equation (1.3). Later in [22], [23], a paraxial formulation was proposed for the static general eikonal equation (1.2) in geophysical applications.

An important application for (1.2) is obtaining geodesic distance on a manifold. Suppose that $P = (x, y)$ is a point on a manifold M defined as the graph of a smooth function $f(x, y)$ and that γ are the curves connecting P and $\Gamma \subset M$ on the manifold. The minimizing curve of γ is called the geodesic. Let ϕ be the distance function such that

$$\phi(x, y) = \min_{\gamma \subset M} \int_\gamma ds.$$

Then ϕ is the solution of

$$(1.4)\quad \sqrt{\left(\frac{1+f_y^2}{f_x^2+f_y^2+1}\right)\phi_x^2 + \left(\frac{1+f_x^2}{f_x^2+f_y^2+1}\right)\phi_y^2 - 2\frac{f_x f_y}{f_x^2+f_y^2+1}\phi_x\phi_y} = 1, \quad \phi|_\Gamma = 0.$$

This equation can be easily generalized to higher dimensions. For example, in three dimensions we again write down the formula for M as the graph of a smooth function $f(x, y, z)$. The distance function ϕ then satisfies

$$(1.5)\qquad \sqrt{a\phi_x^2 + b\phi_y^2 + c\phi_z^2 - 2d\phi_x\phi_y - 2e\phi_y\phi_z - 2f\phi_z\phi_x} = 1, \quad \phi|_\Gamma = 0,$$

where

$$a = \frac{1+f_y^2+f_z^2}{1+f_x^2+f_y^2+f_z^2}, \quad b = \frac{1+f_x^2+f_z^2}{1+f_x^2+f_y^2+f_z^2}, \quad c = \frac{1+f_x^2+f_y^2}{1+f_x^2+f_y^2+f_z^2},$$
$$d = \frac{f_x f_y}{1+f_x^2+f_y^2+f_z^2}, \quad e = \frac{f_y f_z}{1+f_x^2+f_y^2+f_z^2}, \quad f = \frac{f_z f_x}{1+f_x^2+f_y^2+f_z^2}.$$

We will apply our new algorithm to compute the geodesic distance later. There are other approaches that are designed to compute geodesic distances on manifolds. Kimmel and Sethian [16] extended the fast marching method to triangulated manifolds and provided an algorithm for computing the geodesic distances, thereby extracting shortest paths on triangulated manifolds. Barth [2] used the discontinuous Galerkin method to find the distance on graphs of functions that are represented by spline functions. In [7], the authors embeded the manifold as the zero level set of a Lipschitz continuous function and solved the corresponding eikonal equation (1.4) in the embedding space. In [17], the authors based their work on the theory of geodesics on Riemannian manifolds with boundaries and adapted the standard fast marching method to compute weighted distance functions and geodesics on implicit surfaces efficiently. Tsai et. al. [28] used a fast Gauss–Seidel-type iteration method and a monotone upwind Godunov flux for the numerical Hamiltonian.

We propose a new interpretation of the monotone upwind Godunov flux for the numerical Hamiltonian to solve (1.2). The complexity of our method appears to be $O(N)$. We illustrate the approach with several numerical examples in two and three dimensions.

2. A new numerical scheme for convex Hamiltonians. Our new numerical algorithm for static Hamilton–Jacobi equations is composed of a sweeping process and an update formula. The sweeping process we use here is a version of Gauss–Seidel iteration. It is motivated originally by Boué and Dupuis [3], who first suggested that the complexity of this approach for the eikonal case is $O(N)$. In [31], the fast sweeping algorithm was first formulated in PDE framework for the eikonal equation and was used to compute the distance function. In the sweeping process, we sweep through the grids with alternating directions in order to follow the characteristics and use the most recent values as we update the solution. This means that we overwrite an old value with its new value as soon as we obtain the latter. In one dimension, we sweep through the grids from left to right followed by right to left because the characteristics have only two possible directions. In two dimensions, the characteristics may have an infinite number of possible directions. We use four sweeping directions so that a specific sweeping direction covers a group of characteristics at the same time. We denote these four sweeping directions as one iteration. In n dimensions, we will use 2^n alternating directions per iteration. We stop our iterations when the L_1 norm of the difference of two successive iteration results is less than the given tolerance, which is $O(h)$, where h is the grid size.

The new update formula we derive here comes from using the Legendre transformation. The Legendre transformation can be applied to the Wulff problem [21], which is usedto determine the equilibrium shape of crystalline materials. We give the definitions in the following.

DEFINITION 2.1. *Let $\gamma : S^{d-1} \rightarrow R^+$ be a continuous function defined on a curved space S^{d-1}.*

1. The first Legendre transformation of γ is

$$\gamma_*(\nu) = \min_{\theta\cdot\nu>0,\, |\theta|=1} \left[\frac{\gamma(\theta)}{(\theta\cdot\nu)}\right].$$

2. The second Legendre transformation of γ is

$$\gamma^*(\nu) = \max_{\theta\cdot\nu>0,\, |\theta|=1} [\gamma(\theta)(\theta\cdot\nu)].$$

The first and second Legendre transformations are dual to each other in a certain sense, i.e., $(\gamma_*)^* = \gamma$ if γ is convex and $(\gamma^*)_* = \gamma$ if γ is polar-convex. See, e.g., [21]. We can extend γ to the whole space R^d by defining

$$\widetilde{\gamma}(x) = |x|\gamma\left(\frac{x}{|x|}\right),$$

where the extension $\widetilde{\gamma}$ is homogeneous to degree 1 and $x \in R^d$.

The convex Hamiltonian using the Bellman formula or the Legendre transformation is

$$H(\nabla\phi(x)) = \max_\theta[(\nabla\phi\cdot\theta)w(\theta)], \quad \theta \in S^{d-1},$$

where

$$w(\theta) = \min_{\nu\cdot\theta>0,\,|\nu|=1}\left[\frac{H(\nu)}{(\nu\cdot\theta)}\right] \quad \text{and} \quad \nu = \frac{\nabla\phi(x)}{|\nabla\phi(x)|}. \tag{2.1}$$

We define the numerical Hamiltonian as follows:

$$\hat{H}(D_-^i\phi; D_+^j\phi) = \max_\theta\left\{\left(\sum_k D_\mp^k\phi\cdot\theta_k^\pm\right)w(\theta)\right\},$$

where $D_-^i\phi$ $(D_+^j\phi)$ are the backward (forward) difference in i (j) direction, $\theta^+ = \max(\theta, 0)$, and $\theta^- = \min(\theta, 0)$. This numerical Hamiltonian is monotone and consistent. It also turns out to be Godunov's numerical Hamiltonian. In order to describe this clearly without loss of generality, we discuss the two-dimensional case here,

$$H(\phi_x, \phi_y) = \max_\theta(\phi_x\cos\theta + \phi_y\sin\theta)w(\theta),$$

where

$$w(\theta) = \min_{-\frac{\pi}{2}\le\nu-\theta\le\frac{\pi}{2}} \frac{H(\cos\nu, \sin\nu)}{\cos(\nu-\theta)}.$$

The new numerical Hamiltonian is

$$\hat{H}(D_-^x\phi, D_+^x\phi; D_-^y\phi, D_+^y\phi) = \max_\theta\{((\cos\theta)^\pm D_\mp^x\phi + (\sin\theta)^\pm D_\mp^y\phi)w(\theta)\}.$$

We say a function $H(x_1, x_2, \dots, x_n)$ is nondecreasing in x_j by writing $H(x_1, x_2, \dots, x_{j-1}, \uparrow, x_{j+1}, \dots, x_n)$ and nonincreasing by writing $H(x_1, x_2, \dots, x_{j-1}, \downarrow, x_{j+1}, \dots, x_n)$.

LEMMA 2.2. *$\hat{H}$ is monotone; i.e., $\hat{H}(\uparrow, \downarrow, \uparrow, \downarrow)$.*

Proof. Since $w > 0$, this conclusion is straightforward. □

LEMMA 2.3. *$\hat{H}$ is consistent; i.e., $\hat{H}(p, p; q, q) = H(p, q)$.*

Proof. This is a simple manipulation of the following definitions:

$$\begin{aligned}
\hat{H}(p_-, p_+; q_-, q_+) &:= \max_\theta\{((\cos\theta)^\pm p_\mp + (\sin\theta)^\pm q_\mp)w(\theta)\},\\
\hat{H}(p, p; q, q) &= \max_\theta\{((\cos\theta)^\pm p + (\sin\theta)^\pm q)w(\theta)\}\\
&= \max_\theta\{(p\cos\theta + q\sin\theta)w(\theta)\}\\
&=: H(p, q). \qquad \square
\end{aligned}$$

By solving the Riemann problem for Hamilton–Jacobi equations (a generalization of Godunov's procedure), Bardi and Osher [1] proved the following result for Godunov's scheme:

$$H^G(p_-,p_+;q_-,q_+) = \mathrm{ext}_{p\in I[p_-,p_+]}\,\mathrm{ext}_{q\in I[q_-,q_+]}\,H(p,q), \tag{2.2}$$

where

$$\begin{aligned}\mathrm{ext}_{p\in I[a,b]} &= \min_{p\in[a,b]} \quad \text{if } a\le b,\\ \mathrm{ext}_{p\in I[a,b]} &= \max_{p\in[b,a]} \quad \text{if } a> b,\\ H^G(D_-^x\phi_{ij},D_+^x\phi_{ij};D_-^y\phi_{ij},D_+^y\phi_{ij}) &= H^G(p_-,p_+;q_-,q_+),\end{aligned}$$

and $I[a,b]$ denotes the closed interval bounded by a and b.

PROPOSITION 2.4. *$\hat{H}$ is Godunov's numerical Hamiltonian; i.e., $\hat{H}=H^G$.*

Proof. We first assume $p_- < p_+$ and $q_- < q_+$,

$$\begin{aligned}H^G(p_-,p_+;q_-,q_+) &:= \min_{p_-\le p\le p_+}\ \min_{q_-\le q\le q_+} H(p,q)\\ &= \min_{p_-\le p\le p_+}\ \min_{q_-\le q\le q_+}\left\{\max_\theta\{(p\cos\theta+q\sin\theta)w(\theta)\}\right\}\\ &= \max_\theta\left\{\min_{p_-\le p\le p_+}\ \min_{q_-\le q\le q_+}(p\cos\theta+q\sin\theta)w(\theta)\right\}\\ &= \max_\theta\{((\cos\theta)^\pm p_\mp+(\sin\theta)^\pm q_\mp)w(\theta)\}\\ &=: \hat{H}(p_-,p_+;q_-,q_+).\end{aligned}$$

The proof for the other 3 cases is equally straightforward. ◻

Now we use our new numerical Hamiltonian to solve (1.2). In order to write our scheme in an explicit form, we prove the following property first.

LEMMA 2.5. $\max_\theta(af(\theta)-g(\theta))=0$ *with* $f(\theta)>0 \iff a=\min_\theta\frac{g(\theta)}{f(\theta)}$.

Proof.

$$\max_\theta(af(\theta)-g(\theta)) = \max_\theta f(\theta)\left(a-\frac{g(\theta)}{f(\theta)}\right)=0.$$

Since $f(\theta)>0$, we have $\max_\theta(a-\frac{g(\theta)}{f(\theta)})=0$, which implies $a=\min_\theta\frac{g(\theta)}{f(\theta)}$. ◻

Apply this property to

$$\hat{H}(D_-^x\phi,D_+^x\phi;D_+^y\phi,D_-^y\phi)=R(x,y).$$

Let $\phi_0=\phi_{i,j}$, $\phi_W=\phi_{i-1,j}$, $\phi_E=\phi_{i+1,j}$, $\phi_S=\phi_{i,j-1}$, and $\phi_N=\phi_{i,j+1}$. Breaking down the expressions, we have

$$\max_{\theta,\phi_{W,E,S,N}}\left\{\left\{\begin{matrix}(\cos\theta)^+(\phi_O-\phi_W)\\ -(\cos\theta)^-(\phi_O-\phi_E)\end{matrix}\right\}+\left\{\begin{matrix}(\sin\theta)^+(\phi_O-\phi_S)\\ -(\sin\theta)^-(\phi_O-\phi_N)\end{matrix}\right\}\right\}w(\theta)-hR(x_i,y_j)=0,$$

$$\begin{aligned}\max_{\theta,\phi_{W,E,S,N}}\ &\phi_O((\cos\theta)^+-(\cos\theta)^-+(\sin\theta)^+-(\sin\theta)^-)w(\theta)\\ &+\left\{\begin{matrix}-(\cos\theta)^+\phi_W\\ (\cos\theta)^-\phi_E\end{matrix}\right\}+\left\{\begin{matrix}-(\sin\theta)^+\phi_S\\ (\sin\theta)^-\phi_N\end{matrix}\right\}w(\theta)=hR(x_i,y_j).\end{aligned}$$

Thus

$$\begin{aligned}\phi_O &= \min_\theta \left\{ \frac{\begin{Bmatrix} (\cos\theta)^+\phi_W \\ -(\cos\theta)^-\phi_E \end{Bmatrix} + \begin{Bmatrix} (\sin\theta)^+\phi_S \\ -(\sin\theta)^-\phi_N \end{Bmatrix} w(\theta) + hR(x_i, y_j)}{(|\cos\theta| + |\sin\theta|)w(\theta)} \right\} \\ &= \min_\theta K(\theta). \end{aligned} \tag{2.3}$$

We can also derive the three-dimensional numerical Hamiltonian and the update formula in the same way. Let $\phi_0 = \phi_{i,j.k}$, $\phi_W = \phi_{i-1,j,k}$, $\phi_E = \phi_{i+1,j,k}$, $\phi_S = \phi_{i,j-1,k}$, $\phi_N = \phi_{i,j+1,k}$, $\phi_D = \phi_{i,j,k-1}$, and $\phi_U = \phi_{i,j,k+1}$. We have

$$\begin{aligned}&\hat{H}(D_-^x\phi, D_+^x\phi; D_-^y\phi, D_+^y\phi; D_-^z\phi, D_+^z\phi) \\ &\quad = \max_{\theta_1,\theta_2}\{((\sin\theta_1\cos\theta_2)^\pm D_\mp^x\phi + (\sin\theta_1\sin\theta_2)^\pm D_\mp^y\phi + (\cos\theta_1)^\pm D_\mp^z\phi)w(\theta_1,\theta_2)\}, \\ &\phi_O = \min_{\theta_1,\theta_2}\left\{ \frac{\begin{Bmatrix} (\sin\theta_1\cos\theta_2)^+\phi_W \\ -(\sin\theta_1\cos\theta_2)^-\phi_E \end{Bmatrix} + \begin{Bmatrix} (\sin\theta_1\sin\theta_2)^+\phi_S \\ -(\sin\theta_1\sin\theta_2)^-\phi_N \end{Bmatrix} + \begin{Bmatrix} (\cos\theta_1)^+\phi_D \\ -(\cos\theta_1)^-\phi_U \end{Bmatrix} w + hR}{(|\sin\theta_1\cos\theta_2| + |\sin\theta_1\cos\theta_2| + |\cos\theta_1|)w} \right\}. \end{aligned} \tag{2.4}$$

Sometimes it is possible to obtain explicit expression for w from (2.1), but in general, one has to use numerical approximations by the fast Legendre transform developed by Brenier [4] and Corrias [9]. The minimization in the update formulas (2.3) and (2.4) can be achieved either analytically or numerically. For a Hamiltonian of quadratic form in the gradient, we solve the minimization analytically in the next section. For other cases, we find the minimizer by using some well-developed numerical optimization techniques, e.g., L-BFGS-B [5], [32] and trust region methods that employ quadratic interpolation [12], [8].

3. Analytically solving a class of Hamilton–Jacobi equations.

The quadratic form Hamiltonian

$$\sqrt{a(x,y)\phi_x^2 + b(x,y)\phi_y^2 - 2c(x,y)\phi_x\phi_y} = R(x,y) \tag{3.1}$$

is of special interest because computing geodesic distances on a manifold leads to this type of equation. Here we show that the minimization of (2.3) can be solved explicitly. Using the Legendre transformation, we have (after some simple calculations)

$$H(\cos\nu, \sin\nu) = \sqrt{a\cos^2\nu + b\sin^2\nu - 2c\sin\nu\cos\nu}$$

and

$$w(\theta) = \sqrt{\frac{ab - c^2}{a\sin^2\theta + b\cos^2\theta + 2c\cos\theta\sin\theta}}.$$

Finding the minimum of (2.3) when $0 < \theta < \pi/2$ first, $\frac{dK}{d\theta} = 0$ leads to

$$(-\phi_W + \phi_S)w^2 - hR[(\cos\theta + \sin\theta)w' + (-\sin\theta + \cos\theta)w] = 0. \tag{3.2}$$

Thus

$$\frac{-\phi_W + \phi_S}{hR} = \frac{-a\sin\theta + b\cos\theta + c(\sin\theta - \cos\theta)}{\sqrt{(ab - c^2)(a\sin^2\theta + b\cos^2\theta + 2c\sin\theta\cos\theta)}} = T(\theta) \tag{3.3}$$

and

$$T'(\theta) = -\frac{\sqrt{ab - c^2}(\cos\theta + \sin\theta)}{(a\sin^2\theta + b\cos^2\theta + 2c\sin\theta\cos\theta)^{3/2}} < 0.$$

The solvability condition for θ_1 is

$$\frac{c - a}{\sqrt{a(ab - c^2)}} < \frac{-\phi_W + \phi_S}{hR} < \frac{b - c}{\sqrt{b(ab - c^2)}}. \tag{3.4}$$

If (3.4) is satisfied, we will have a unique solution for $0 < \theta < \pi/2$ because of the monotonicity of T. Let $m = (-\phi_W + \phi_S)/hR$. We have

$$\theta = \tan^{-1}\left(\frac{-cm^2(ab - c^2) - (a - c)(b - c) \pm m(ab - c^2)\sqrt{(a + b - 2c) - m^2(ab - c^2)}}{am^2(ab - c^2) - (a - c)^2}\right)$$

if both m and the denominator are not zero. Here we have two choices for θ because we square both sides while we do the calculation. We need to plug in (3.3) and pick up the right one. Also

$$\theta = \tan^{-1}\left(\frac{b - a}{c - a}\right)$$

if the denominator is zero, and

$$\theta = \tan^{-1}\left(\frac{c - b}{c - a}\right)$$

if $m = 0$. Using similar arguments, we can write down solvability conditions and explicit formulas for θ in other ranges. This can be summarized in the following algorithm.

Algorithm (Quadratic Hamilton–Jacobi solver using the Bellman formula). We assume that $\phi(i, j)$ is given in a small neighborhood of Γ. We initialize the unknown ϕ by setting $\phi(i, j)$ to ∞[1] and mask (i, j) = unknown.

We begin by setting $\phi^{(0)} = \phi$.

Do the following steps while $|\phi^{(n+1)} - \phi^{(n)}| > \delta$: ($\delta > 0$ is the given tolerance which is $O(h)$).

Sweeping Process: A compact way of writing these sweeping iterations in C/C++ is

```
for(s1=-1;s1<=1;s1+=2)
for(s2=-1;s2<=1;s2+=2)
for(i=(s1<0?nx:0);(s1<0?i>=0:i<=nx);i+=s1)
for(j=(s2<0?ny:0);(s2<0?j>=0:j<=ny);j+=s2)
  update $\phi_{i,j}$
```

Update Formula: For each grid point (i, j) visited in the sweeping iteration, if mask (i, j) = unknown, do the following:

For $(s_x, s_y) = (\pm 1, \pm 1)$

[1] Notice that we only need to use a large value in actual implementation.

1. Check the solvability condition

$$
\begin{aligned}
&m = \frac{s_x s_y(\phi^{(n)}(i, j - s_y) - \phi^{(n)}(i - s_x, j))}{hR},\\
&\text{check } \frac{c-a}{\sqrt{a(ab-c^2)}} < m < \frac{b-c}{\sqrt{b(ab-c^2)}} \quad \text{when } s_x s_y > 0,\\
&\text{check } \frac{-(b+c)}{\sqrt{b(ab-c^2)}} < m < \frac{a+c}{\sqrt{a(ab-c^2)}} \quad \text{when } s_x s_y < 0.
\end{aligned}
$$

2. If the condition is satisfied,

$$
\begin{aligned}
\theta = \tan^{-1} \Bigg(&\frac{-cm^2(ab-c^2) - (as_x - cs_y)(bs_y - cs_x)}{am^2(ab-c^2) - (as_x - cs_y)^2}\\
&\pm \frac{m(ab-c^2)\sqrt{(a+b-2cs_x s_y) - m^2(ab-c^2)}}{am^2(ab-c^2) - (as_x - cs_y)^2} \Bigg) + (1 - s_x)\frac{\pi}{2}
\end{aligned}
$$

if both m and the denominator are not zero. Plug in the test function

$$
T(\theta) = \frac{(-as_x + cs_y)\sin\theta + (bs_y - cs_x)\cos\theta}{\sqrt{(ab-c^2)(a\sin^2\theta + b\cos^2\theta + 2c\sin\theta\cos\theta)}}
$$

and pick up the right one which equals m, not $-m$. Also

$$
\theta = \tan^{-1}\left(\frac{b-a}{c - as_x s_y}\right) + (1 - s_x)\frac{\pi}{2}
$$

if the denominator is zero, and

$$
\theta = \tan^{-1}\left(\frac{cs_x - bs_y}{cs_y - as_x}\right) + (1 - s_x)\frac{\pi}{2}
$$

if $m = 0$.

3. Add

$$
\phi_{tmp} = \frac{(s_x\phi(i - s_x, j)\cos\theta + s_y\phi(i, j - s_y)\sin\theta)w(\theta) + hR}{(|\cos\theta| + |\sin\theta|)w(\theta)}
$$

to the list `phi_candidate.`

4. Add $K(0)$, $K(\frac{\pi}{2})$, $K(\pi)$, $K(\frac{3\pi}{2})$ to the list `phi_candidate.`
5. Let $\phi_{\min}$ be the minimum element of `phi_candidate`.
6. Update

$$
\phi^{(n+1)}(i, j) = \min(\phi^{(n)}(i, j), \phi_{\min}).
$$

4. Numerical minimization. For a more general sweeping algorithm, we use numerical optimization to calculate ϕ_0. There are many minimization methods that are readily available to us. Some methods need only evaluations of the function while others also require evaluations of the derivative of the function. For our multidimensional cases, we use the L-BFGS-B method [5], [32], [6] because the cost of the iteration is low and the storage requirements of the algorithm are modest. L-BFGS-B is a limited memory quasi-Newton method for a large-scale bound-constrained problem. The minimizer $\widetilde{\theta}$ of (2.3) and the minimizer $(\widetilde{\theta_1}, \widetilde{\theta_2})$ of (2.4) at a grid point is constructed to be within a given tolerance through iterations, and the number of iterations depends on the initial condition and the tolerance. In our algorithm, we use the minimizer obtained in the previous sweep as our initial guess. In the first sweep, we use the minimizers of the upwind neighboring grid pronts as initial conditions for the quasi-Newton method. This implies that the initial conditions that we end up using are, in most cases, close enough to the minimizers. In practice, with the tolerance of 10^{-6}, we observed that, in average, only four to five iterations are needed. There is an alternative approach of discretizing θ and then searching for the minimum in the corresponding discretized space. Take the two-dimensional case, for example,

$$\phi_O = \min_{\theta} K(\theta) = K(\widetilde{\theta}) \sim \min_{\theta_j} k(\theta_j), \tag{4.1}$$

where $\theta_j = j \triangle \theta / 2\pi$. Used in a straightforward manner, this kind of approach would require that the grid size $\triangle\theta$ is comparable to the given tolerance. In the following, we briefly describe how the L-BFGS-B method works.

Consider finding a minimum by Newton's method to search for a zero of the gradient of the function $f(\theta) : R^n \to R$. The iteration formula is given by

$$\theta^{K+1} = \theta^k - A^{-1} \cdot \nabla f(\theta),$$

where A is the Hessian matrix of f. The BFGS method is a quasi-Newton method because it doesn't use the actual Hessian matrix of f, but it constructs a sequence of H^k to approximate A^{-1}. The iteration formula for unconstrained optimization is given by

$$\theta^{k+1} = \theta^k - \lambda^k H^k g^k, \quad k = 0, 1, 2, \dots ,$$

where λ^k is a step size, g^k is the gradient of f at θ^k, and H^k is updated at every iteration by the following formula:

$$H^{k+1} = (V^k)^T H^k V^k + \rho^k s^k (s^k)^T, \tag{4.2}$$

where

$$\rho^k = 1/(y^k)^T s^k, \qquad V^k = I - \rho^k y^k (s^k)^T,$$

and

$$s^k = \theta^{k+1} - \theta^k, \qquad y^k = g^{k+1} - g^k.$$

The limited memory BFGS method only stores the m most recent pairs $\{s^i, y^i\}_{i=k-m}^{k-1}$ to update H^k. Suppose that the current iteration is θ^k and the initial limited memory matrix $H^k_{(0)}$ (usually a diagonal matrix) is updated by $\{s^i, y^i\}_{i=k-m}^{k-1}$. From (4.2) we have

$$\begin{aligned} H^k = {} & ((V^{k-1})^T \cdots (V^{k-m})^T) H^k_{(0)} (V^{k-m} \cdots V^{k-1}) \\ & + \rho^{k-m} ((V^{k-1})^T \cdots (V^{k-m+1})^T) s^{k-m} (s^{k-m})^T (V^{k-m+1} \cdots V^{k-1}) \\ & + \rho^{k-m+1} ((V^{k-1})^T \cdots (V^{k-m+2})^T) s^{k-m+1} (s^{k-m+1})^T (V^{k-m+2} \cdots V^{k-1}) \\ & + \cdots + \rho^{k-1} s^{k-1} (s^{k-1})^T. \end{aligned} \tag{4.3}$$

For bound constrained problems, the direct Hessian approximation $B^k = (H^k)^{-1}$ is used. The detail derivation and efficient algorithm for computing H^k and B^k are found in [6]. This B^k is used to define a quadratic model of f at θ^k,

$$Q^k(\theta) = f(\theta^k) + (g^k)^T(\theta - \theta^k) + \frac{1}{2}(\theta - \theta^k)^T B^k(\theta - \theta^k).$$

In order to find the minimizer of Q^k subject to the bound constrained, the gradient projection method is first used to determine a set of active bounds. Suppose we have $\Theta = \{\theta \mid l_i \le \theta_i \le u_i,\ i = 1, \dots, n\}$; the ith coordinate of the projection of vector θ is given by

$$P(\theta, l, u)_i = \begin{cases} l_i & \text{if } \theta_i \le l_i, \\ u_i & \text{if } \theta_i \ge u_i, \\ \theta_i & \text{otherwise.} \end{cases}$$

We can then find the generalized Cauchy point that is the first local minimizer θ^c of

$$Q_L^k(t) = Q^k(P(\theta^k - tg^k, l, u)).$$

Use θ^c to identify a set of active variables and then find the minimizer $\overline{\theta}^{k+1}$of the quadratic model with respect to the free variables. Perform a line search

$$\theta^{k+1} = \theta^k + \alpha^k \left(\overline{\theta}^{k+1} - \theta^k\right), \tag{4.4}$$

where α^k is the step size, to find θ^{k+1} that satisfies the sufficient decrease condition

$$f(\theta^{k+1}) \le f(\theta^k) + 10^{-4}(g^k)^T \left(\overline{\theta}^{k+1} - \theta^k\right).$$

For more details, please refer to [5]. In our calculation, we choose $m = 5$ and the stopping criterion is then

$$\|P(\theta^k - g^k, l, u) - \theta^k\|_\infty < 10^{-6}.$$

5. Examples. We implement our new numerical scheme in the following examples. We choose $\delta = 10^{-15}$ for two dimensional cases and $\delta = 10^{-12}$ for three dimensional cases for simplicity. Ideally the δ should be chosen as a small constant times the grid size. We test an anisotropic case with constant coefficients a, b, and c in Figures 1 and 2 to show a very degenerate case with varied coefficients and a box-shape boundary condition. The equation is

$$\sqrt{0.375\phi_x^2 + 0.25\phi_y^2 - 0.58\phi_x\phi_y} = (2.1 - \cos(4\pi^2 xy))/4.$$

Thus $a = 0.375$, $b = 0.25$, $c = 0.29$, and $R(x, y) = (2.1 - \cos(4\pi^2 xy))/4$. Notice that in this case, $ab = 0.0938$ is barely greater than $c^2 = 0.0841$ and R is highly oscillatory. That is why it needs more iterations. In general, we usually need more iterations when the characteristics are very curvy. Figures 3 and 4 show the geodesic distances on manifolds. In Figure 3, there are two boundary points. The contour plot has kinks on the equal distance places. In Figure 4, the boundary point is in the center and on the

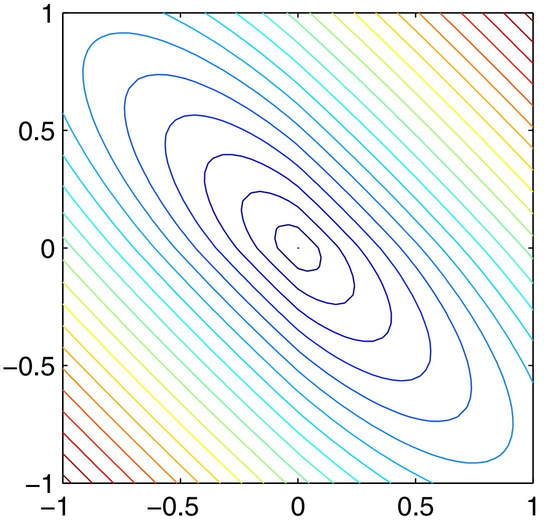


FIG. 1. *A sweeping result after* 2 *sweeping iterations on a* 50×50 *grid. The boundary is a single point in the center.* $a = 1.0$, $b = 1.0$, $c = 0.9$, *and* $R = 1$.

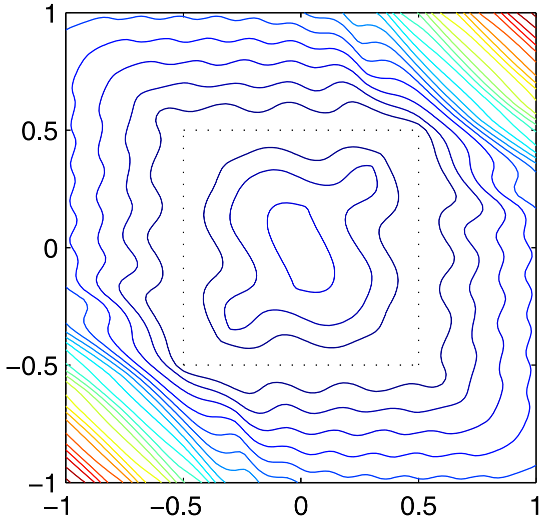


FIG. 2. $a = 0.375$, $b = 0.25$, $c = 0.29$, *and* $R(x, y) = (2.1 - \cos(4\pi^2 xy))/4.0$ *on a* 100×100 *grid. Convergence is reached after* 45 *sweeping iterations.*

top of the mountain-shaped manifold. The contour plot shows the geodesic distance to the boundary point. Figure 5 is an example of the first arrival travel times to seismic imaging. The computational domain suggests material layering under a sinusoidal profile with layer shapes $C(x) = 0.1225 \sin(4\pi x)$. Suppose the domain is split into four parts by $y_i(x) = 0.1225 \sin(4\pi x) + p_i$ where $i = 1, 2, 3,$ and $p_i = (-0.25, 0, 0.25)$. In each layer, the anisotropic speed at (x, y) is given by an ellipse with the long axis (of length $2F_2$) tangential to the curve $C(x)$ and the short axis (of length $2F_1$) normal

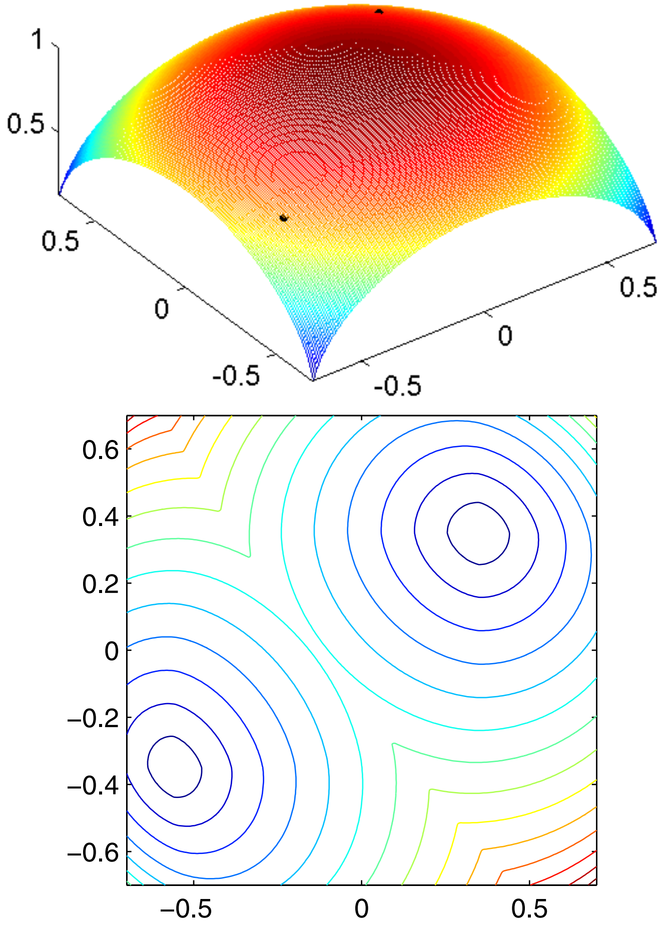


Fig. 3. *This is an example of the distance on a half sphere. The sweeping algorithm was applied to the graph of* $f(x,y)=\sqrt{1.0-(x^2+y^2)}$ *with* $\phi(-0.56,-0.35)=\phi(0.35,0.35)=0$ *as a boundary condition on a* 200×200 *grid. The convergence was reached after* 3 *sweeping iterations.*

to the curve. F_1 and F_2 are constants in each layer. This leads to

$$F_2\sqrt{((1+n^2)\phi_x^2+(1+m^2)\phi_y^2-2mn\phi_x\phi_y)/(1+m^2+n^2)}=1,$$

where

$$(m,n)=\frac{\sqrt{(F_2/F_1)^2-1}}{\sqrt{1+\left(\frac{dC(x)}{dx}\right)^2}}\left(\frac{dC(x)}{dx},-1\right).$$

From the results, we know that the algorithm is stable even with discontinuous coefficients. Figures 6 and 7 are the solutions for three-dimensional eikonal equation with one and two point boundary conditions. Figures 8 and 9 are the more general cases for

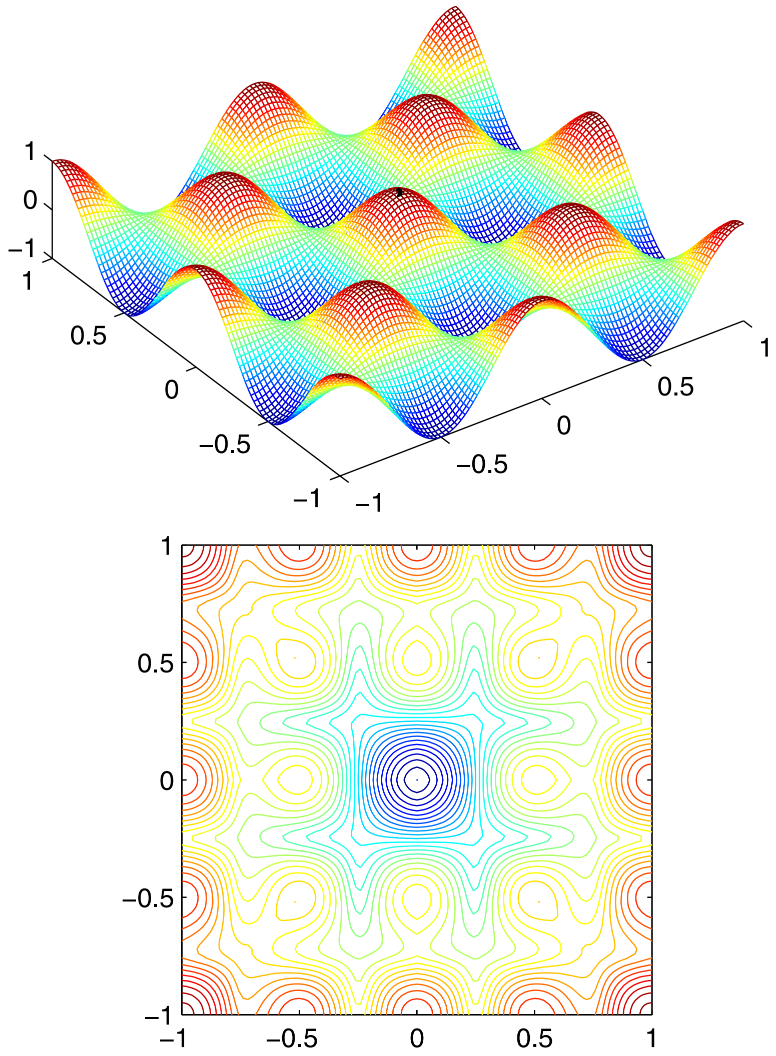


Fig. 4. *The distance contour from* $(0, 0)$ *on the graph of* $f(x, y) = \cos(2\pi x)\cos(2\pi y)$. *The convergence was obtained after* 12 *iterations on a* 100×100 *grid.*

three dimensions. Figure 8 has a boundary point $\phi(0, 0, 0) = 0$ and Figure 9 has a cubic boundary condition with sides of length one. The governing equation we solved is

$$\sqrt{a\phi_x^2 + b\phi_y^2 + c\phi_z^2 - 2d\phi_x\phi_y - 2e\phi_y\phi_z - 2f\phi_z\phi_x} = 1,$$

where

$$a = \frac{1 + f_y^2 + f_z^2}{1 + f_x^2 + f_y^2 + f_z^2}, \quad b = \frac{1 + f_x^2 + f_z^2}{1 + f_x^2 + f_y^2 + f_z^2}, \quad c = \frac{1 + f_x^2 + f_y^2}{1 + f_x^2 + f_y^2 + f_z^2},$$

$$d = \frac{f_x f_y}{1 + f_x^2 + f_y^2 + f_z^2}, \quad e = \frac{f_y f_z}{1 + f_x^2 + f_y^2 + f_z^2}, \quad f = \frac{f_z f_x}{1 + f_x^2 + f_y^2 + f_z^2},$$

and $f(x, y, z) = \cos(2\pi x)\cos(2\pi y)\cos(2\pi z)$, and the corresponding

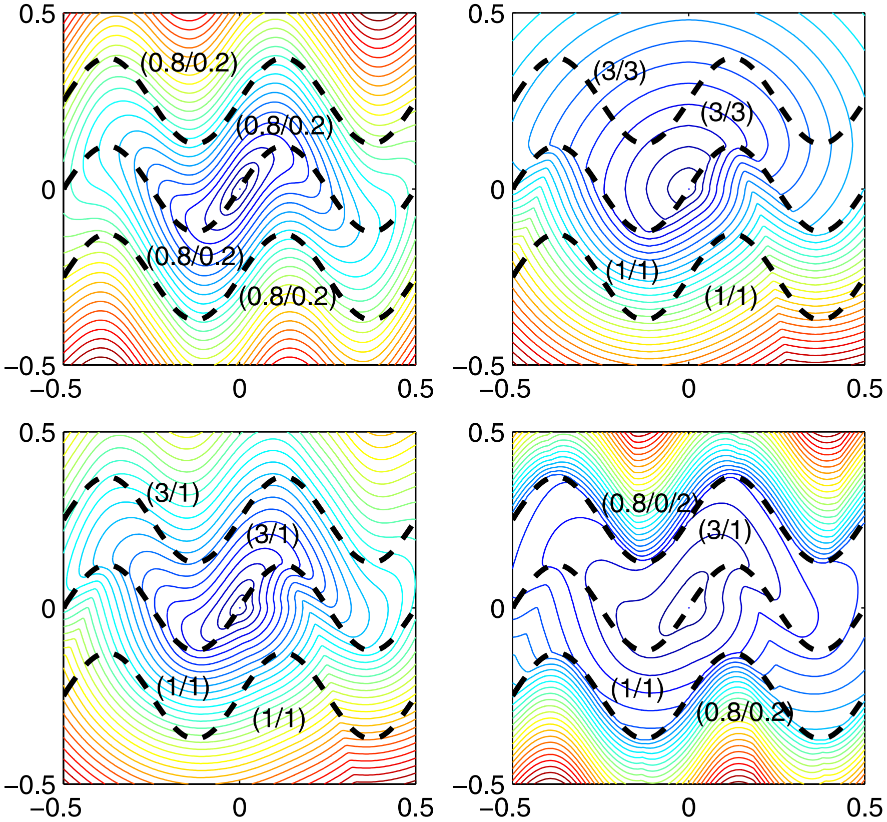


FIG. 5. *This is an example of first arrival travel times in seismic imaging* [26]*. The* (F_2, F_1) *pair for each layer is given in the above figures. The convergence was obtained after* $5, 4, 16,$ *and* 16 *iterations on a* 200×200 *grid.*

$$w(\theta_1, \theta_2) = \sqrt{\frac{1}{1 + (f_x \sin\theta_1 \cos\theta_2 + f_y \sin\theta_1 \sin\theta_2 + f_z \cos\theta_1)^2}}.$$

This seems to be the first successful rapid computation in three dimensions for such problems. In [30], it was proved that the results from the fast sweeping method for the eikonal equation with $R(x) = 1$ need only one iteration, which is exactly 2^n Gauss–Seidel alternating sweepings for the problem in R^n, to reach a solution with global error $O(h \log(1/h))$. We provide the numerical evidence by testing our methods on an eikonal equation with $R(x) = 1$ on two and three dimensions. The results are given in Tables 1 and 2. For anisotropic cases, we found out that the number of iterations depend on the anisotropy of the Hamiltonian, but it is always reasonable and appears to be independent of the grid size.

6. Conclusion. In this paper, we have presented a new numerical method for Hamilton–Jacobi equations written in the form of Bellman's formula. We proved that the numerical Hamiltonian we proposed is monotone and consistent and is in fact also the Godunov Hamiltonian. We implemented this new scheme and showed some results in two- and three-dimensional cases.

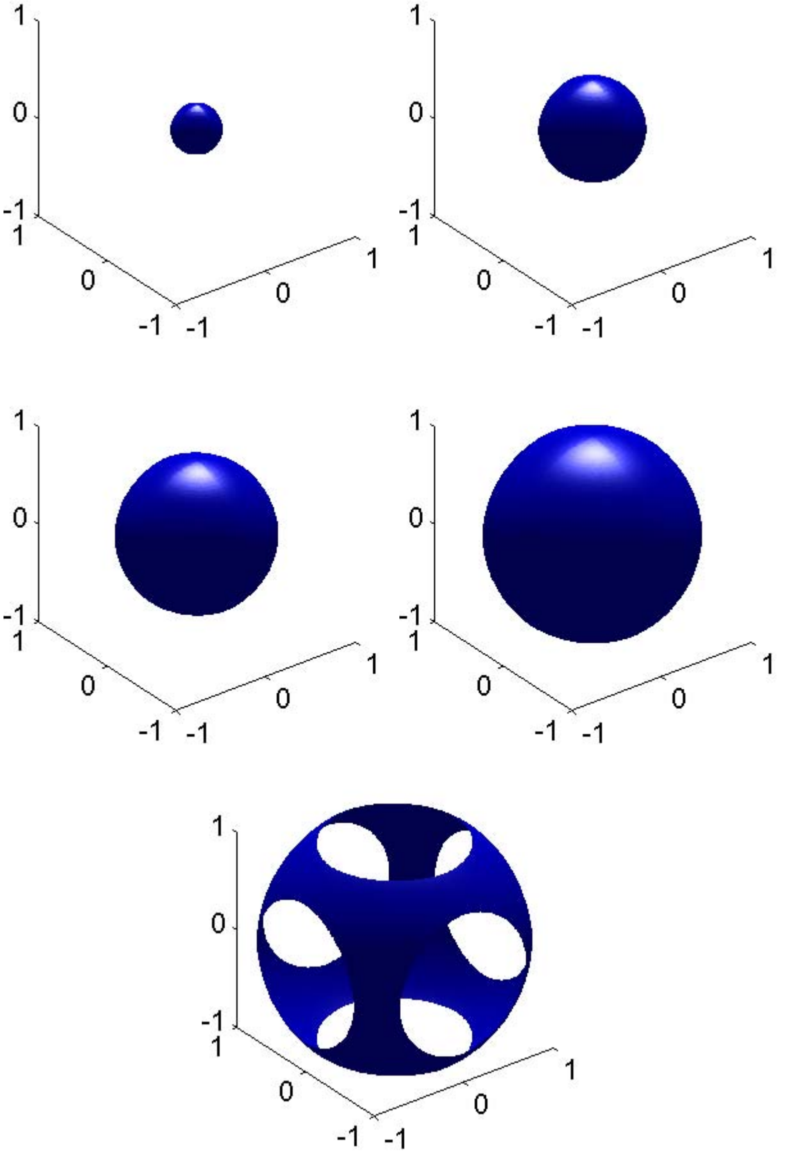


FIG. 6. *This is the one-iteration result of the* 3*D eikonal equation with the boundary* $(0, 0)$ *in the center of the graph. The corresponding contours are* 0.25*,* 0.5*,* 0.75*,* 1.0*, and* 1.25.

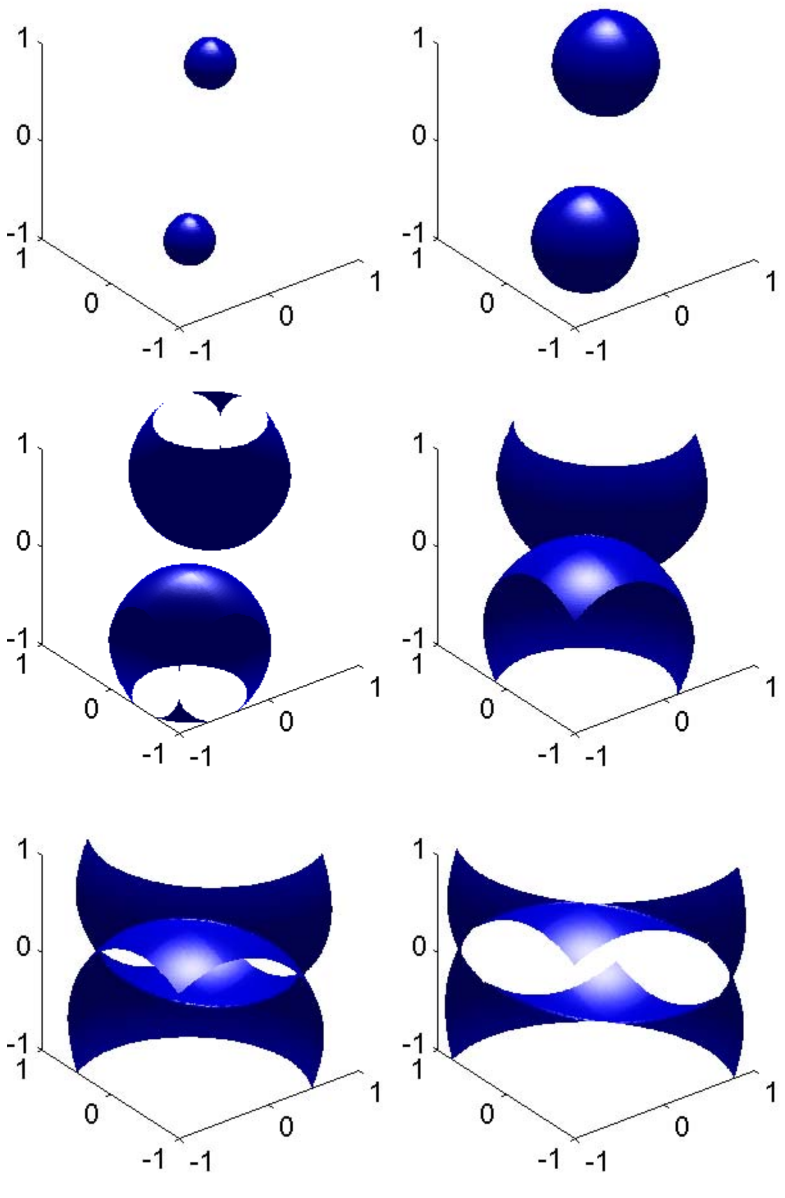


Fig. 7. *This is the one-iteration result of the* 3*D eikonal equation with two boundary points* $(-0.5, -0.5)$ *and* $(0.5, 0.5)$*. The corresponding contours are* 0.25*,* 0.5*,* 0.75*,* 1*,* 1.25*, and* 1.5.

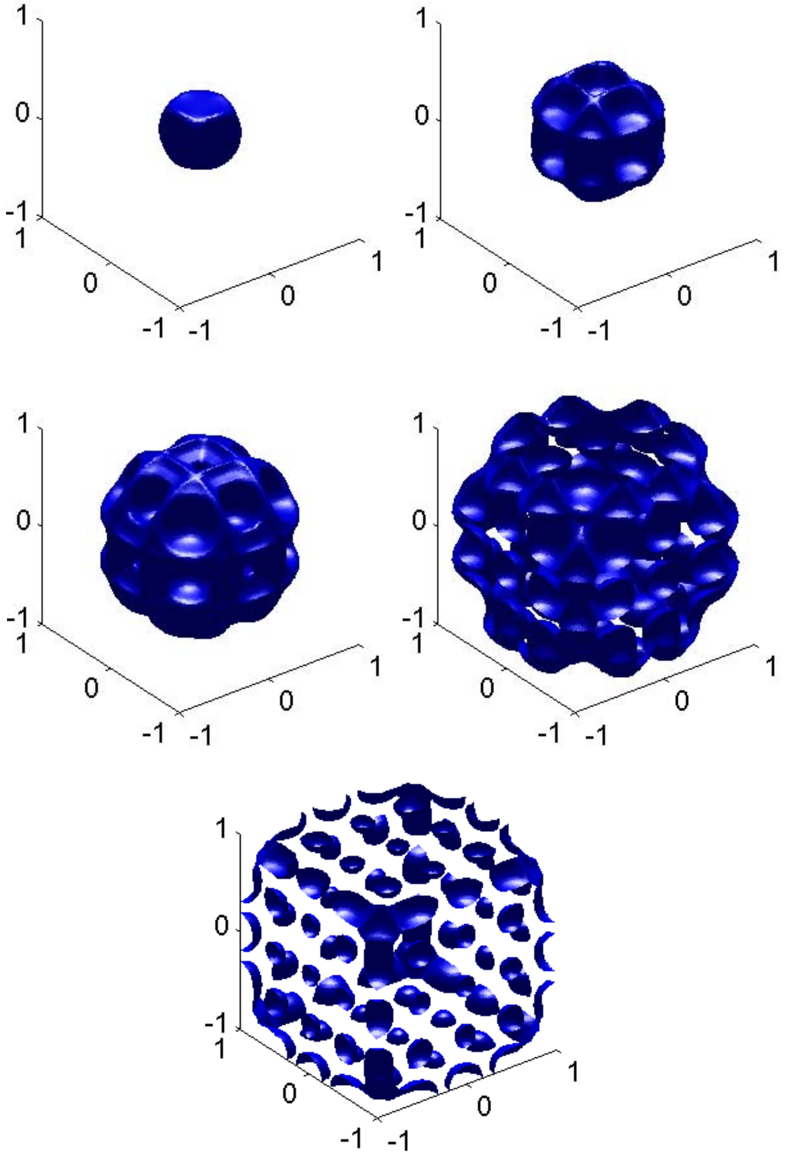


FIG. 8. *This is a 3D example with* $f(x,y,z)=\cos(2\pi x)\cos(2\pi y)\cos(2\pi z)$*, the corresponding* $w=(1+(\bigtriangledown f\cdot\theta)^2)^{-1/2}$*, and a boundary point at the center. The convergence was obtained after* 10 *iterations on a* $100\times100\times100$ *grid. The contours shown here are* 1.2*,* 1.5*,* 1.8*,* 2.2*, and* 2.5*.*

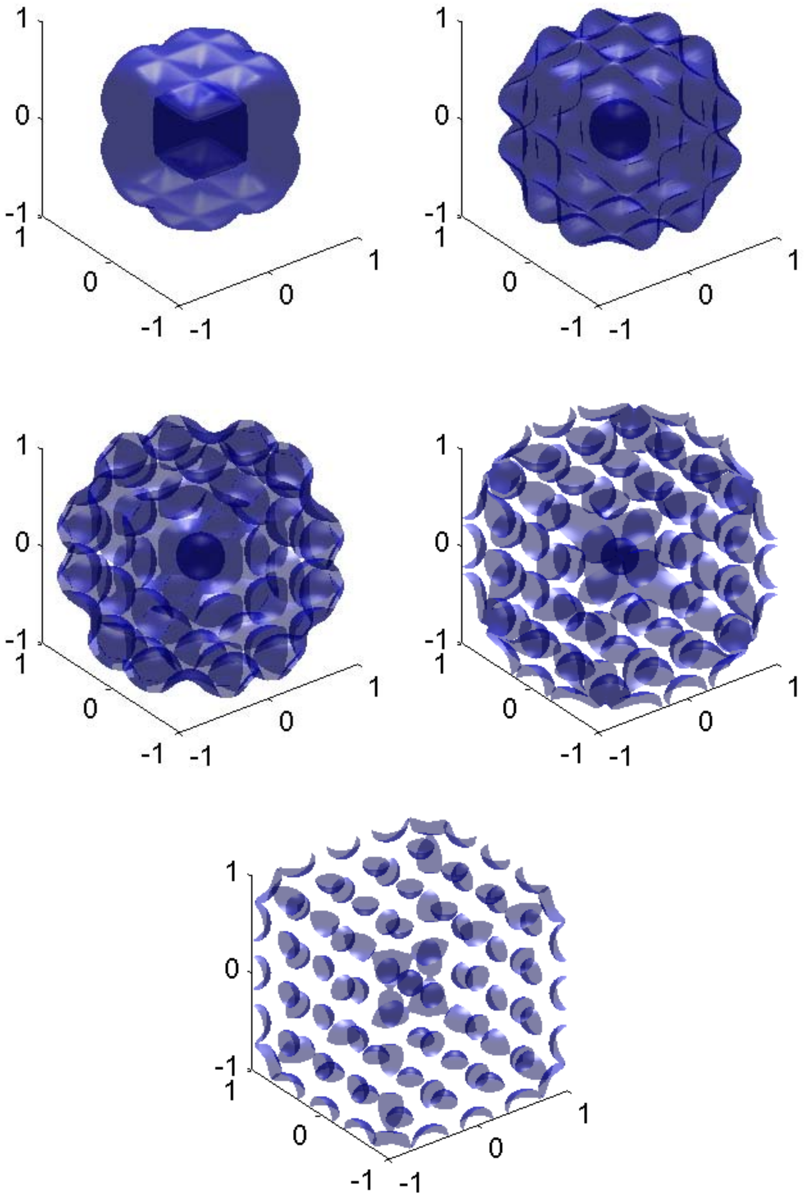


FIG. 9. *This is a 3D example with* $f(x,y,z) = \cos(2\pi x)\cos(2\pi y)\cos(2\pi z)$*, the corresponding* $w = (1 + (\bigtriangledown f \cdot \theta)^2)^{-1/2}$*, and the cubic boundary condition. The convergence was obtained after* 9 *iterations on a* $100 \times 100 \times 100$ *grid. The contours shown here are* 0.2*,* 0.4*,* 0.6*,* 0.8*, and* 1.0.

Table 1
The errors of a 2D eikonal case.

2D eikonal equation $dx =$	2/50	2/100	2/200	2/400	2/800	2/1600	1/3200
L_1 error	0.102158	0.060888	0.0358203	0.0207759	0.0118848	0.0067128	0.00374894
L_∞ error	0.0437414	0.0262969	0.0154506	0.00890583	0.00505242	0.00282877	0.00156648

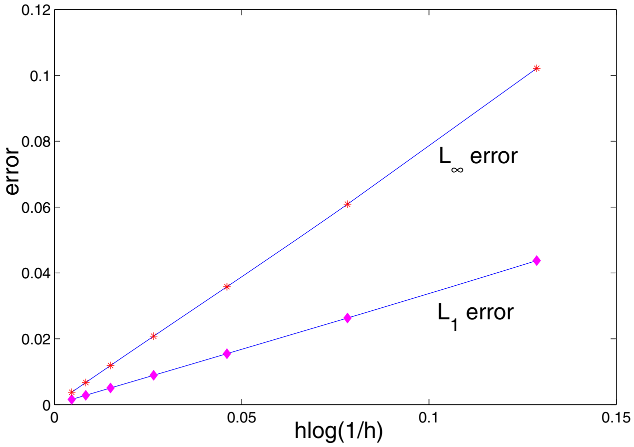
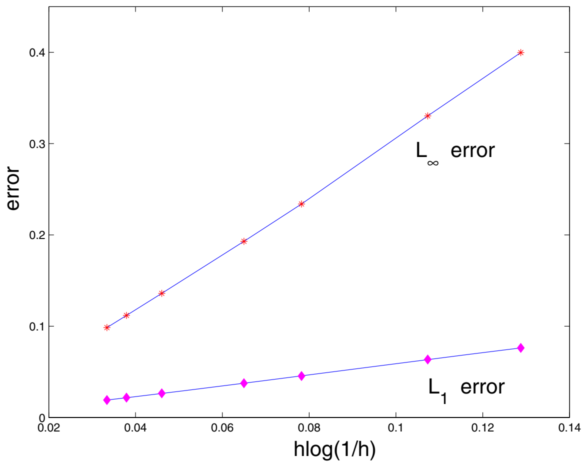


Table 2
The errors of a 3D eikonal case.

3D eikonal equation $dx =$	2/50	2/64	2/100	2/128	2/200	2/256	2/300
L_1 error	0.399696	0.330305	0.233834	0.192961	0.135946	0.111793	0.0985156
L_∞ error	0.0761747	0.0635267	0.0454065	0.0375639	0.0264938	0.0217706	0.0191687



REFERENCES

[1] M. Bardi and S.Osher, *The nonconvex multidimentional Riemann problem for Hamilton-Jacobi equations*, SIAM J. Math. Anal., 22 (1991), pp. 344–351.

[2] T. J. Barth, *On the Marchability of Interior Stabilized Discontinuous Galerkin Approximations of the Eikonal and Related Pdes with Non-Divergence Structure*, NASA Technical Report, NAS-01-010, NASA Ames Research Center, Moffett Field, CA, 2001.

[3] M. Boué and P. Dupuis, *Markov chain approximations for deterministic control problems with affine dynamics and quadratic cost in the control*, SIAM J. Numer. Anal., 36 (1999), pp. 667–695.

[4] Y. Brenier, *Un algorithme rapide pour le calcul de transformées de Legendre-Fenchel discretes*, C. R. Acad. Sci. Paris Sér. I Math., 308 (1989), pp. 587–589.

[5] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu, *A limited memory algorithm for bound constrained optimization*, SIAM J. Sci. Comput., 16 (1995), pp. 1190–1208.

[6] R. H. Byrd, J. Nocedal, and R. B. Schnabel, *Representation of quasi-Newton matrices and their use in limited memory methods*, Math. Programming, 63 (1994), pp. 129–156.

[7] L.-T. Cheng, P. Burchard, B. Merriman, and S. Osher, *Motion of curves constrained on surfaces using a level-set approach*, J. Comput. Phys., 175 (2002), pp. 604–644.

[8] T. F. Coleman and Y. Li, *An interior trust region approach for nonlinear minimization subject to bounds*, SIAM J. Optim., 6 (1996), pp. 418–445.

[9] L. Corrias, *Fast Legendre-Fenchel transform and applications to Hamilton-Jacobi equations and conservation laws*, SIAM J. Numer. Anal., 33 (1996), pp. 1534–1558.

[10] M. G. Crandall and P. L. Lions, *Two approximations of solutions of Hamilton-Jacobi equations*, Math. Comp., 43 (1984), pp. 1–19.

[11] M. Falcone and R. Ferretti, *Semi-Lagrangian schemes for Hamilton-Jacobi equations, discrete representation formulae and Godunov methods*, J. Comput. Phys., 175 (2002), pp. 559–575.

[12] A. Friedlander, J. M. Martinez, and S. A. Santos, *A new trust region algorithm for bound constrained minimization*, Appl. Math. Optim, 30 (1994), pp. 235–266.

[13] S. Gray and W. May, *Kirchhoff migration using eikonal equation traveltimes*, Geophysics, 59 (1994), pp. 810–817.

[14] J. Helmsen, E. Puckett, P. Colella, and M. Dorr, *Two new methods for simulating photolithography development in* 3*d*, in SPIE 2726, 1996, pp. 253–261.

[15] G.-S. Jiang and D. Peng, *Weighted ENO schemes for Hamilton-Jacobi equations*, SIAM J. Sci. Comput., 21 (2000), pp. 2126–2143.

[16] R. Kimmel and J. A. Sethian, *Computing geodesic paths on manifolds*, Proc. Natl. Acad. Sci. USA, 95 (1998), pp. 8431–8435.

[17] F. Memoli and G. Sapiro, *Fast computation of weighted distance functions and geodesics on implicit hyper-surfaces*, J. Comput. Phys., 173 (2001), pp. 730–764.

[18] S. Osher, *A level set formulation for the solution of the Dirichlet problem for Hamilton-Jacobi equations*, SIAM J. Math. Anal., 24 (1993), pp. 1145–1152.

[19] S. Osher and J. A. Sethian, *Fronts propagating with curvature-dependent speed: algorithms based on Hamilton-Jacobi formulations*, J. Comput. Phys., 79 (1988), pp. 12–49.

[20] S. Osher and C.-W. Shu, *High-order essentially nonoscillatory schemes for Hamilton-Jacobi equations*, SIAM J. Numer. Anal., 28 (1991), pp. 907–922.

[21] D. Peng, S. Osher, B. Merriman, and H.-K. Zhao, *The geometry of Wulff crystal shapes and its relations with Riemann problems*, in Nonlinear Partial Differential Equations (Evanston, IL, 1998), AMS., Providence, RI, 1999, pp. 251–303.

[22] J. Qian and W. W. Symes, *Paraxial eikonal solvers for anisotropic quasi-p travel times*, J. Comput. Phys., 174 (2001), pp. 256–278.

[23] J. Qian and W. W. Symes, *Finite-difference quasi-p traveltimes for anisotropic media*, Geophysics, 67 (2002), pp. 147–155.

[24] E. Rouy and A. Tourin, *A viscosity solutions approach to shape-from-shading*, SIAM J. Numer. Anal., 29 (1992), pp. 867–884.

[25] J. A. Sethian, *Fast marching level set methods for three dimensional photolithography development*, in SPIE 2726, 1996, pp. 261–272.

[26] J. A. Sethian and A. Vladimirsky, *Ordered upwind methods for static Hamilton-Jacobi equations*, Proc. Natl. Acad. Sci. USA, 98 (2001) pp. 11069–11074.

[27] P. E. Souganidis, *Approximation schemes for viscosity solutions of Hamilton-Jacobi equations*, J. Differential Equations, 59 (1985), pp. 1–43.

[28] Y.-H. R. Tsai, L.-T. Cheng, S. Osher, and H.-K. Zhao, *Fast sweeping methods for a class of Hamilton-Jacobi equations*, SIAM J. Numer. Anal., 41 (2003), pp. 673–694.

[29] J. Tsitsiklis, *Efficient algorithms for globally optimal trajectories*, IEEE Trans. Automat. Control, 40 (1995), pp. 1528–1538.

[30] H.-K. Zhao, *Fast sweeping method for eikonal equations* I, *Distance function*, www.math.uci.edu/~zhao, 2002.

[31] H.-K. Zhao, S. Osher, B. Merriman, and M. Kang, *Implicit and non-parametric shape reconstruction from unorganized points using variational level set method*, Computer Vision and Image Understanding, 80 (2000), pp. 295–314.

[32] C. Zhu, R. H. Byrd, P. Lu, and J. Nocedal, *L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization*, ACM Trans. Math. Software, 23 (1997), pp. 550–560.