SIAM J. Sci. Comput.
Vol. 22, No. 6, pp. 2178–2193

AN $\mathcal{O}(N)$ LEVEL SET METHOD FOR EIKONAL EQUATIONS[*]

SEONGJAI KIM[†]

Abstract. A propagating interface can develop corners and discontinuities as it advances. Level set algorithms have been extensively applied for the problems in which the solution has advancing fronts. One of the most popular level set algorithms is the so-called fast marching method (FMM), which requires total $\mathcal{O}(N \log_2 N)$ operations, where N is the number of grid points. The article is concerned with the development of an $\mathcal{O}(N)$ level set algorithm called the *group marching method* (GMM). The new method is based on the narrow band approach as in the FMM. However, it is incorporating a correction-by-iteration strategy to advance a group of grid points at a time, rather than sorting the solution in the narrow band to march forward a single grid point. After selecting a group of grid points appropriately, the GMM advances the group in two iterations for the cost of slightly larger than one iteration. Numerical results are presented to show the efficiency of the method, applied to the eikonal equation in two and three dimensions.

Key words. level set method, narrow band approach, eikonal equation, first-arrival traveltime

AMS subject classifications. 65M06, 86A15

PII. S1064827500367130

1. Introduction.

A propagating interface can develop corners and discontinuities as it advances. It is the case for the solution of, e.g., the eikonal equation, a Hamilton–Jacobi differential equation, in heterogeneous media. Since we do not know which values to assign to the gradient components at the corners and even do not know which values to assign to the function at discontinuities, the solution does not satisfy the equation in the normal (classical) sense. An immediate consideration is to introduce a nonclassical weak solution: a continuous function having possibly discontinuous gradients that solves the equation in average.

Unfortunately, the class of weak solutions is too big; the initial data for these problems does not determine weak solutions uniquely. One way to enforce the uniqueness is to add viscosity described by a dissipative term, e.g., a multiple of the negative Laplace operator, to the equation. The solution for a given viscosity is smooth and therefore uniquely determined. Then one may take the limit of these smooth solutions by letting the coefficient of the Laplace operator approach zero. If the limit exists, its solution is uniquely determined for the appropriate initial data; it is called a *viscosity solution*.

Properties of viscosity solutions were first studied by Lax [14] in the context of hyperbolic conservation laws; good references to work on them are [3, 4, 15].

It has been conjectured that *the first-arrival traveltime (FATT) field is a viscosity solution of the eikonal equation* [30]. The conjecture is supported by some theoretical results and by a great deal of numerical evidence. Techniques for computing viscosity solutions were developed first for hyperbolic conservation laws. Numerical methods for these conservation laws have utilized *upwind* finite differences (FDs) and have adapted to produce viscosity solutions of the eikonal equation [6, 12, 19, 25, 30]. Upwind FD techniques and their applications to Hamilton–Jacobi equations can be found in [16, 17, 18].

[*]Received by the editors February 7, 2000; accepted for publication (in revised form) December 12, 2000; published electronically April 26, 2001.
http://www.siam.org/journals/sisc/22-6/36713.html

[†]Department of Mathematics, University of Kentucky, Lexington, KY 40506-0027 (skim@ms.uky.edu).

The level set method is a numerical technique to compute advancing fronts and has been applied to a wide range of important physical problems; see, e.g., [26, 27] and references therein. Even though its solution shows first-order accuracy, the level set method has been widely used mostly due to its built-in stability. It has been adapted for the computation of the FATT and implemented with the narrow band technique, called the *fast marching method* (FMM), in which the narrow band points form a neighborhood of advancing wavefronts [16, 19, 25, 26, 27, 29]. The FMM was first developed by Tsitsiklis [29] and later rederived by Sethian [25]. The FMM requires sorting the solution at each step of the narrow band. For the binary tree sorting, the total cost of the method becomes $\mathcal{O}(N \log_2 N)$. The main object of the article is to develop an $\mathcal{O}(N)$ level set method for the FATT, called the *group marching method* (GMM).

An outline of the paper is as follows. In section 2, the eikonal equation is introduced as the model equation, along with its numerical techniques to solve. The first-order upwind FD scheme is presented, and the FMM is briefly reviewed. In section 3, we suggest the GMM; a pseudocode is presented. Section 4 is devoted to the introduction of the *average normal slowness* which can improve accuracy of the solution. An approximate average slowness which can be easily implemented is discussed in the same section. In section 5, an alternative update formula is considered to improve flexibility in incorporating the average slowness. In section 6, the GMM is tested for various velocity models, update formulae, and average slowness incorporation, in two and three dimensions, and is compared with the FMM. It is numerically verified that the new method performs with the computation cost $\mathcal{O}(N)$. Section 7 discusses some important aspects on accuracy, efficiency, and applicability for the level set methods. The last section includes conclusions.

2. Preliminaries.

2.1. The eikonal equation.

The eikonal equation in an isotropic medium is given by

$$|\nabla \tau(\mathbf{x}_s, \mathbf{x})|^2 = \frac{1}{v^2(\mathbf{x})}, \tag{2.1}$$

where $\tau(\mathbf{x}_s, \mathbf{x})$ is the traveltime of the acoustic wave from the source $\mathbf{x}_s$ to the location $\mathbf{x}$ and $v(\mathbf{x})$ denotes the velocity of the propagating wavefront at $\mathbf{x}$. For down-going wavefronts (advancing in the $(z+)$-direction), for example, the equation can be rewritten in the evolutionary form

$$\tau_z = \sqrt{s^2 - \tau_x^2 - \tau_y^2}, \tag{2.2}$$

where $s(= 1/v)$ is called the slowness, the reciprocal of the velocity.

As mentioned earlier, the FATT is a (continuous) viscosity solution of the eikonal equation. There have been various numerical techniques for FATTs: ray-tracing methods, FD methods, and algorithms based on Fermat's principle [24].

The use of ray tracing followed by interpolation of traveltimes is a popular and robust method for computing diffraction trajectories for small or moderate velocity contrasts. For regions with high velocity contrasts, ray tracing methods can produce quite large shadow zones where the computed traveltime field should be interpolated. Such ray-tracing/interpolation processes are cumbersome and computationally expensive, especially in three dimensions. For a comprehensive treatment of ray theory, see [1, 2, 7].

An alternative to the method is to compute traveltimes by solving directly the eikonal equation on a regular grid by FD schemes; see [5, 6, 12, 30, 31, 32] for standard expanding-box methods and [19, 21] for the level set method. A drawback of the standard FD eikonal solvers is that it is not easy to control the propagation angle of rays as it is with ray tracing methods. The level set method incorporating the narrow band method overcomes the drawback of the standard FD schemes; however, it has first-order accuracy and is hardly extendable for a higher-order scheme in realistic media. Nonetheless the FD eikonal solvers have their advantages: they are more efficient computationally than ray tracers and rarely produce shadow zones. The author recently suggested a stable FD eikonal solver incorporating a postsweeping iteration [12], which is second-order and readily extendable to higher-order schemes.

The current level set methods require sorting the traveltimes in the narrow band, a neighborhood of the wavefront, at each stage of the computation. The FMM [25, 28], a narrow band level set method, has adopted the binary tree sorting algorithm; the method turns out to require a total of $\mathcal{O}(N \log_2 N)$ operations, where N is the number of grid points. Even though the level set methods have first-order accuracy, they are still attractive due to their built-in stability and a wide range of applications. It is worth developing an optimal cooperating technique with which the level set method costs $\mathcal{O}(N)$ operations in total.

2.2. The FD scheme. For a numerical scheme for (2.1), consider a cubic domain

$$(x_{\min}, x_{\max}) \times (y_{\min}, y_{\max}) \times (z_{\min}, z_{\max});$$

partition it into $N_x \times N_y \times N_z$ cells of the uniform size $\Delta x \times \Delta y \times \Delta z$ with their vertices

$$\mathbf{x}_{i,j}^k = (x_i, y_j, z_k) = (x_{\min} + i\Delta x, y_{\min} + j\Delta y, z_{\min} + k\Delta z).$$

Let $\tau_{i,j}^k = \tau(\mathbf{x}_s, \mathbf{x}_{i,j}^k)$, and define the forward $(+)$ and backward $(-)$ difference operators for τ_x at the point $\mathbf{x}_{i,j}^k$:

$$D_x^{\pm} \tau_{i,j}^k = \pm \frac{\tau_{i\pm1,j}^k - \tau_{i,j}^k}{\Delta x}. \tag{2.3}$$

Define the *upwind* FD scheme for τ_x incorporating the first-arrivals:

$$\widehat{D}_x \tau_{i,j}^k = \text{mod\_max}\Big(\max\left(D_x^- \tau_{i,j}^k, 0\right), \, \min\left(D_x^+ \tau_{i,j}^k, 0\right) \Big), \tag{2.4}$$

where mod_max returns the larger value in modulus. Note that

$$|\widehat{D}_x \tau_{i,j}^k| = \max\left(D_x^- \tau_{i,j}^k, \, -D_x^+ \tau_{i,j}^k, \, 0\right),$$

which can be utilized in implementation.

After introducing the analogues for τ_y and τ_z, i.e., $\widehat{D}_y \tau_{i,j}^k$ and $\widehat{D}_z \tau_{i,j}^k$, we can formulate the upwind FD scheme for (2.1):

$$\left(\widehat{D}_x \tau_{i,j}^k\right)^2 + \left(\widehat{D}_y \tau_{i,j}^k\right)^2 + \left(\widehat{D}_z \tau_{i,j}^k\right)^2 = \left(s_{i,j}^k\right)^2. \tag{2.5}$$

For down-going wavefronts as in (2.2), the scheme can be explicitly rewritten as

$$\tau_{i,j}^{k+1} = \tau_{i,j}^k + \Delta z \cdot H[\tau]_{i,j}^k, \tag{2.6}$$

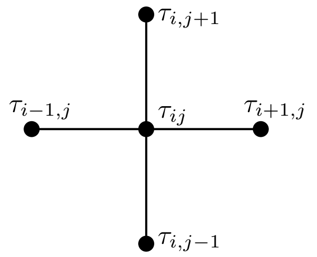

FIG. 1. *The update procedure of the FMM for τ_{ij}.*

where

$$H[\tau]^k_{i,j} = \sqrt{\left(s^k_{i,j}\right)^2 - \left(\widehat{D}_x\tau^k_{i,j}\right)^2 - \left(\widehat{D}_y\tau^k_{i,j}\right)^2}.$$

The quantity inside the radical can be negative in practical simulation. Then it can be either ignored or adjusted by a nonnegative value, depending on the algorithm adopted. In either case, it should be carefully designed not to violate causality or stability and not to deteriorate accuracy.

2.3. Review of the FMM. In this section we review the FMM, a narrow band level set algorithm, presented in [25, 27]. The FMM begins from the source and updates the traveltimes at neighboring grid points. The six neighboring points (four in two dimensions) form the first stage of the narrow band. Then, the FMM expands the narrow band by updating traveltimes at neighboring downwind points of a *Trial* point in the narrow band, and accepting the trial point as an *Alive* (completed) point. The trial point is selected such that its traveltime is smallest among all values available from the narrow band, for which the FD scheme easily satisfies causality.

We summarize the FMM as follows [27]. First, tag the source point as *Alive*, where the traveltime is zero. Then compute traveltimes at all points one grid point away from the source, and tag them as *Close*. Finally, tag *Far* all other grid points, and set the traveltimes for the *Far* points large. Then the FMM loop is carried out as follows.

(a) Let *Trial* be the point in *Close* having the smallest traveltime.
(b) Tag as *Close* all neighbors of *Trial* that are not *Alive*. (If the neighbor is in *Far*, remove it from the list and add it to the set *Close*.)
(c) Recompute the traveltimes at all *Close* neighbors of *Trial* using (2.5).
(d) Remove the point *Trial* from *Close* and add it to *Alive*.
(e) If *Close* is not empty, go to top of loop.

Note that in step (a) of the FMM loop, the point *Trial* is chosen to have the smallest traveltime among all values in *Close*. This implies that the traveltimes in *Close* should be sorted from the smallest to the largest, at each stage of the narrow band. When a binary tree sorting algorithm is applied, the cost is $\mathcal{O}(\log_2 N_B)$, where N_B is the number of grid points in *Close*. Due to the sorting algorithm, the total cost for the FMM becomes $\mathcal{O}(N \log_2 N)$, where N is the total number of grid points.

Now, we present the update procedure (c) of the FMM loop [27]. For a simple presentation, we illustrate it in two dimensions. Let $\mathbf{x}_{ij}$ be a *Close* neighbor of *Trial* to be updated from nearby values: $\tau_{i-1,j}$, $\tau_{i+1,j}$, $\tau_{i,j-1}$, and $\tau_{i,j+1}$. (See Figure 1.) The four nearby points connected by grid line segments form four quadrants; the FMM attempts to solve the quadratic equation given by each quadrant according to

$$\left(\widehat{D}_x\tau_{ij}\right)^2 + \left(\widehat{D}_y\tau_{ij}\right)^2 = \left(s_{ij}\right)^2. \tag{2.7}$$

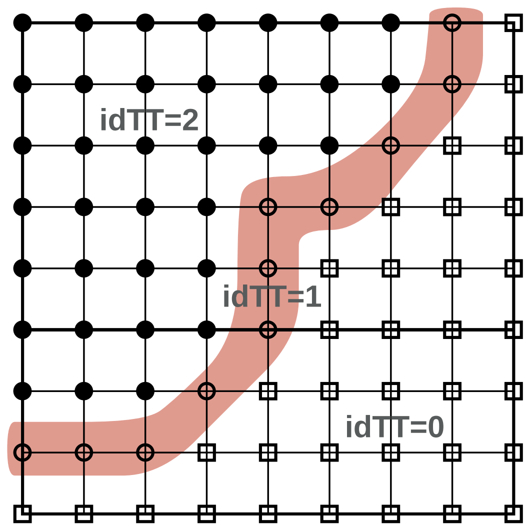

FIG. 2. *The GMM illustrated in two dimensions. The closed circles indicate the points already computed, the open circles form a neighborhood of a wavefront, and the open boxes correspond to the downwind points to be computed at the current stage or later.*

For example, we refer to possible contributors $\tau_{i+1,j}$ and $\tau_{i,j+1}$ (the first quadrant). Without loss of generality, there are two cases.

1. If only $\tau_{i+1,j}$ is known, then we find the larger solution τ of

$$\left(\frac{\tau - \tau_{i+1,j}}{\Delta x}\right)^2 = (s_{ij})^2. \tag{2.8}$$

2. If both $\tau_{i+1,j}$ and $\tau_{i,j+1}$ are known, then we find the larger real solution of

$$\left(\frac{\tau - \tau_{i+1,j}}{\Delta x}\right)^2 + \left(\frac{\tau - \tau_{i,j+1}}{\Delta y}\right)^2 = (s_{ij})^2. \tag{2.9}$$

For each of four quadrants, we construct all possible real solutions τ; we choose the smallest value for the traveltime τ_{ij}.

3. The GMM.

A costly component of the narrow band methods is the sorting of the solution in each step of the narrow band of wavefronts, which costs $\mathcal{O}(\log_2 N)$ per grid point. Here we introduce a new narrow band level set algorithm, called the GMM, whose total computation cost is $\mathcal{O}(N)$.

Consider a neighborhood Γ of a wavefront; see the shaded area in Figure 2. (Γ corresponds to *Close* in the FMM.) In the current stage of the GMM, we will select a group of points G out of Γ, recompute the traveltimes at neighboring points of G that are not completed, register the neighboring points as members of Γ if they are not already registered, and finally tag "completed" for the points in G. The group of points should be carefully chosen in such a way that the computed solution does not violate causality, since in our algorithm the traveltimes are not sorted from the smallest to the largest. The main objective in this section is to develop a way to select such a group of points from the narrow band.

In the FMM, the traveltimes on the narrow band are sorted at each stage of the narrow band to find the smallest value, and the traveltime at a point is updated from the real solutions of eight "quadratic equations" (in three dimensions), each of which associates with a west-east, south-north, and up-down combination of neighboring grid points. The main reason for the sorting and abundant computation is that we do not know where the wavefront is coming from and advancing to. If one wishes to advance the narrow band by a group of points, it is necessary to explicitly incorporate some aspects of wavefront directions into the computation algorithm. Also, it may require updating traveltimes more than once at neighboring downwind points of the group.

We begin with the two-dimensional (2D) case with $h = \Delta x = \Delta y$ for a simple presentation. Recall that in the FMM, the traveltime at a point $\mathbf{x}_{ij}$ is updated by choosing the minimum of the real solutions of four quadratic equations, each of which corresponds to one of four quadrants. So, given neighboring traveltimes, the minimum of the solutions of the quadratic equations must be associated with the quadrant where the raypath is passing before reaching at $\mathbf{x}_{ij}$. Let the first quadrant be in the upwind direction for $\mathbf{x}_{ij}$. (See Figure 1.) Then the updated traveltime at $\mathbf{x}_{ij}$, τ_{ij}, should satisfy

$$\left(\frac{\tau_{ij} - \tau_{i+1,j}}{h}\right)^2 + \left(\frac{\tau_{ij} - \tau_{i,j+1}}{h}\right)^2 = (s_{ij})^2. \tag{3.1}$$

Since τ_{ij} is the larger solution of the above equation,

$$\tau_{ij} \geq \frac{1}{2}(\tau_{i+1,j} + \tau_{i,j+1}). \tag{3.2}$$

Thus it follows from (3.1) and (3.2) that

$$\tau_{ij} \geq \min(\tau_{i+1,j}, \tau_{i,j+1}) + \delta\tau_{ij}, \quad \delta\tau_{ij} = \frac{1}{\sqrt{2}} \cdot h \cdot s_{ij}. \tag{3.3}$$

Defining $s_{\Gamma,\min} = \min\{s_{ij} : \mathbf{x}_{ij} \in \Gamma\}$, we can interpret (3.3) as follows. *Given two points $\mathbf{x}_{i_1,j_1}$ and $\mathbf{x}_{i_2,j_2}$, if their traveltime difference is less than*

$$\delta\tau \equiv \frac{1}{\sqrt{2}} \cdot h \cdot s_{\Gamma,\min},$$

the angle between the line segment $\overline{\mathbf{x}_{i_1,j_1}\mathbf{x}_{i_2,j_2}}$ and the wavefront normal is larger than 45 degrees, i.e., rather perpendicular than parallel.

Remark. In three dimensions, $\sqrt{2}$ in (3.3) and $\delta\tau$ should be replaced by $\sqrt{3}$. It can be shown with a physical insight as follows. We begin with the 2D problem. The equality in (3.3) holds only if $\tau_{i+1,j} = \tau_{i,j+1}$, i.e., $\mathbf{x}_{i+1,j}$ and $\mathbf{x}_{i,j+1}$ are on the same wavefront; the distance between the wavefront and $\mathbf{x}_{ij}$ is $h/\sqrt{2}$. When $\tau_{i+1,j} \neq \tau_{i,j+1}$, the traveltime increment at $\mathbf{x}_{ij}$, from $\min(\tau_{i+1,j}, \tau_{i,j+1})$, is clearly larger than $\delta\tau_{ij}$. Therefore, we can claim that $|\tau_{ij} - \min(\tau_{i+1,j}, \tau_{i,j+1})|$ is minimized when the wavefront including $\mathbf{x}_{i+1,j}$ and $\mathbf{x}_{i,j+1}$ (at the same time) reaches the target point $\mathbf{x}_{ij}$, which is the slowness multiplied by the distance between the wavefront and $\mathbf{x}_{ij}$. For the three-dimensional (3D) problem, there is no physical restriction to claim the same; here the distance between the target point and the wavefront including the adjacent three points becomes $h/\sqrt{3}$.

Now we are ready to choose the group G to be *completed* at a time. Define $\tau_{\Gamma,\min} = \min\{\tau_{ij} : \mathbf{x}_{ij} \in \Gamma\}$ and select G as follows:

$$G = \Big\{\mathbf{x}_{ij} \in \Gamma \,:\, \tau(\mathbf{x}_{ij}) \leq \tau_{\Gamma,\min} + \delta\tau\Big\}. \tag{3.4}$$

Let $\mathbf{x}_{i_1,j_1}$ and $\mathbf{x}_{i_2,j_2}$ be two points in G. When they are not adjacent, it is clear that their traveltimes do not affect each other in the update procedure. If they are adjacent, one can barely affect the other, since the wavefront normal is nearer to perpendicular, rather than parallel, to the line segment formed by the points. However, whether the two points are adjacent or not, their neighboring downwind points can be affected by both points. It can be the case, in particular, for intersecting wavefronts, i.e., at or near shocks. The neighboring points may have different traveltimes for a different order of updates, which implies that the group update may not be stable.

To fix instability, we update all the neighboring points of the group G twice—one in an order and the other in the opposite order. The double computation fixes instability. To see it, imagine that we try to update the neighboring points of G one more time. We can readily see that none of the neighboring points changes its traveltime during the extra updates. Of course, it also holds for the 3D problem with $\delta\tau = \frac{1}{\sqrt{3}} \cdot h \cdot s_{\Gamma,\min}$.

Now we summarize the above arguments as in the following algorithm (in three dimensions), called the GMM.

```
• Initialization.
  (I1) Assign a large number to the traveltime array TT, e.g., $\texttt{TT}(\cdot,\cdot,\cdot) \equiv 1.0e5$;
  (I2) Set zero for the traveltime index idTT, i.e., $\texttt{idTT}(\cdot,\cdot,\cdot) \equiv 0$;
  (I3) Set $\texttt{delTAU} = \frac{1}{\sqrt{3}} \cdot \min(\Delta x, \Delta y, \Delta z) \cdot \min_{i,j,k} s^k_{i,j}$;
  (I4) On the box of $(2 \times 2 \times 2)$ cells having the source at its center,
         – assign the analytic value of TT on the box;
         – set $\texttt{idTT}(\cdot,\cdot,\cdot) = 2$, at the source;
         – set $\texttt{idTT}(\cdot,\cdot,\cdot) \equiv 1$, on the surface of the box;
           save those point indices to the interface indicator array $\texttt{GAMMA}(\cdot,\cdot,\cdot)$;
         – set TM to be the minimum of TT on the surface of the box;
• Marching Forward.
  (M1) Set $\texttt{TM} = \texttt{TM} + \texttt{delTAU}$;
  (M2) For each $(i,j,k)$ in GAMMA, in the reverse order, if $(\texttt{TT}(i,j,k) \le \texttt{TM})$,
       recompute traveltimes of neighboring points $(\ell,m,n)$ where $\texttt{idTT} \le 1$;
  (M3) For each $(i,j,k)$ in GAMMA, in the forward order, if $(\texttt{TT}(i,j,k) \le \texttt{TM})$,
       (a) recompute traveltimes of neighboring points $(\ell,m,n)$ where $\texttt{idTT} \le 1$;
       (b) if $\texttt{idTT} = 0$ at a neighboring point $(\ell,m,n)$,
           set $\texttt{idTT}(\ell,m,n) = 1$ and save $(\ell,m,n)$ into GAMMA;
       (c) remove the index $(i,j,k)$ out of GAMMA; set $\texttt{idTT}(i,j,k) = 2$;
  (M4) If $\texttt{GAMMA} \neq \emptyset$, go to (M1);
```

Remark. Apparently, the reverse computation (M2) is cheaper to carry out than (M3); the double-computation (M2)–(M3) does not increase the computation cost twice. It can be made cheaper as follows. A step of the GMM advances a group of points including not only the global minimum (of the narrow band) but also all local minima that are less than or equal to TM. It is not difficult to see if a point in the group is a local minimum of the narrow band. Since it is not necessary to compute twice at local minima, one can modify the algorithm to skip the double-computation there. Let the point (i,j,k) in (M2) be recognized as a local minimum during the recomputation of neighboring downwind points. Then one can update GAMMA and idTT as in (b) and (c) of (M3); the point (i,j,k) would be skipped by (M3), since it is already out of GAMMA. More than half the points in a group seems a local minimum, in practice. The double-computation increases the computation cost, not twice, but *slightly.*

Remark. The GMM is in fact an iterative update procedure, converging in two iterations. One may want to select G with a larger $\delta\tau$. In this case, the number of iterations must become larger. Rouy and Tourin [23] has chosen all the grid points as one group and carried out iterations up to convergence. The GMM can be viewed as an intermediate algorithm between the FMM [25, 27] ($\delta\tau = 0$) and the purely iterative algorithm of Rouy and Tourin [23] ($\delta\tau = \infty$).

Remark. When the slowness is constant, two group marchings advance the wavefront to a completely different outer surface. Such a feature can make the GMM more efficient than pointwise methods such as the FMM [27].

4. Average normal slowness.

It is well known that the *exact* traveltime from the source $\mathbf{x}_s$ to a location $\mathbf{x}$ can be computed along the raypath

$$\tau(\mathbf{x}_s, \mathbf{x}) = \int_{\mathbf{x}_s}^{\mathbf{x}} \frac{1}{v(\mathbf{x}')} d\sigma = \int_{\mathbf{x}_s}^{\mathbf{x}} s(\mathbf{x}') d\sigma, \tag{4.1}$$

where σ is the length element along the raypath. This implies that accuracy of algorithm (2.5) can be improved by replacing $s_{i,j}^k$ by the *average normal slowness* $\widehat{s_{i,j}^k}$, which is obtained by integrating the slowness along the wavefront normal direction. The wavefront normal, by definition, is $v(\mathbf{x})\nabla\tau$ and is tangent to the raypath for isotropic media.

Velocities are often provided at grid points in practice; the velocity needs to be interpolated over the whole computation domain. For simplicity, we adopt the trilinear spline interpolation in three dimensions; i.e., the velocity is treated as a trilinear function in each cell.

As an example for the computation of the average normal slowness, consider a point $\mathbf{x}_{i,j}^k$ at which the wavefront normal is pointing the $(z+)$-direction. Let us compute $\tau_{i,j}^{k+1}$ out of $\tau_{i,j}^k$ and its adjacent traveltimes $\tau_{i\pm1,j}^k$ and $\tau_{i,j\pm1}^k$. Since $\widehat{D}_x \tau_{i,j}^k$ and $\widehat{D}_y \tau_{i,j}^k$ are likely zero at the point $\mathbf{x}_{i,j}^k$, we can see from (2.6), with $s_{i,j}^k$ replaced by $\widehat{s_{i,j}^k}$, that

$$\tau_{i,j}^{k+1} = \tau_{i,j}^k + \Delta z \cdot \widehat{s_{i,j}^k}, \tag{4.2}$$

where

$$\widehat{s_{i,j}^k} = \frac{1}{\Delta z} \cdot \int_{\mathbf{x}_{i,j}^k}^{\mathbf{x}_{i,j}^{k+1}} \frac{1}{v(\mathbf{x}')} d\sigma = \begin{cases} \dfrac{1}{v_{i,j}^k} = s_{i,j}^k & \text{if } v_{i,j}^{k+1} = v_{i,j}^k, \\[2ex] \dfrac{\log(v_{i,j}^{k+1}) - \log(v_{i,j}^k)}{v_{i,j}^{k+1} - v_{i,j}^k} & \text{else.} \end{cases}$$

In realistic media, it is difficult to compute the average normal slowness without raypath information. A reasonable approximation can be simply obtained as follows. For example, for the update from the first quadrant as in (2.8)–(2.9), we first check which one is smaller between $\tau_{i+1,j}$ and $\tau_{i,j+1}$. If $\tau_{i+1,j}$ is smaller, the wavefront normal would have a smaller angle to the line segment $\overline{\mathbf{x}_{i+1,j}\mathbf{x}_{i,j}}$ than to $\overline{\mathbf{x}_{i,j+1}\mathbf{x}_{i,j}}$. So it is reasonable to utilize

$$\widehat{s}_{i,j} \approx \begin{cases} s_{i,j} & \text{if } s_{i+1,j} = s_{i,j}, \\[1ex] \dfrac{\log(v_{i+1,j}) - \log(v_{i,j})}{v_{i+1,j} - v_{i,j}} & \text{else.} \end{cases} \tag{4.3}$$

When $\tau_{i,j+1}$ is smaller, the index $(i+1,j)$ in (4.3) should be replaced by $(i,j+1)$.

The practical accuracy of numerical algorithms for traveltimes seems strongly dependent on the ability of the numerical code to compute an *average slowness* close to the average normal slowness. No matter what the order of the numerical scheme is, the solution would turn out to have a first-order accuracy in general media if the point slowness is incorporated. (In principle, the energy does not propagate jumping over discrete points but advances their fronts through all the points in the medium.) Another degraded accuracy problem can appear at singularities such as the source points and caustics; it can be effectively treated by employing numerical techniques such as the locally uniform mesh refinement [12] and adaptive gridding approach [20].

5. An alternative update procedure in two dimensions.

In this section, we consider an alternative update formula for the 2D problem which is slightly different from (2.8)–(2.9). We first define it as follows. To update $\tau_{i,j}$, choose the minimum from the four values

$$\begin{aligned}&\tau_{i\pm1,j} + \Delta x\cdot \max\left(\alpha\widehat{s}_{ij},\ \sqrt{(\widehat{s}_{ij})^2 - (\widehat{D}_z\tau_{i\pm1,j})^2}\right),\\ &\tau_{i,j\pm1} + \Delta z\cdot \max\left(\beta\widehat{s}_{ij},\ \sqrt{(\widehat{s}_{ij})^2 - (\widehat{D}_x\tau_{i,j\pm1})^2}\right),\end{aligned} \tag{5.1}$$

where

$$\alpha = \frac{\Delta x}{\sqrt{(\Delta x)^2+(\Delta z)^2}}, \quad \beta = \frac{\Delta z}{\sqrt{(\Delta x)^2+(\Delta z)^2}}.$$

Here the maximization with $\alpha\widehat{s}_{ij}$ or $\beta\widehat{s}_{ij}$ is incorporated so as not to violate stability.

The above update procedure utilizes delayed traveltime derivatives (purely forward), and therefore it is more efficient in implementation and performance than (2.8)–(2.9). However, it may be *unstable* without the maximization. For example, ignoring the factor $\alpha\widehat{s}_{ij}$, the updating value obtainable from the west (the direction of $\tau_{i-1,j}$) becomes

$$\tau_{i-1,j} + \Delta x\cdot\sqrt{(\widehat{s}_{ij})^2 - (\widehat{D}_z\tau_{i-1,j})^2}.$$

Consider the point $(i,j) = (1,2)$ in Figure 3. Since, apparently, $\widehat{D}_x\tau_{i-1,j-1} = 0$ for a constant slowness, we see $(\widehat{s}_{ij})^2 = (\widehat{D}_z\tau_{i-1,j})^2$. So the updated final value of τ_{ij} would become at most $\tau_{i-1,j}$, which is wrong (an underestimation)!

The maximization is equivalent to imposing the *maximum angle condition* which takes care of only the rays coming to the point $\mathbf{x}_{ij}$ through the *numerical domain of dependence*. The maximum angle condition was first introduced in [8] and was adopted by Symes and his colleagues [13, 20]; see also [12]. When τ_{ij} is to be updated from the west, the numerical domain of dependence is the line segment $\overline{\mathbf{x}_{i-1,j-1}\mathbf{x}_{i-1,j+1}}$, and every ray approaching from the line segment to $\mathbf{x}_{ij}$ increases the traveltime τ_{ij} (from $\tau_{i-1,j}$) by $\alpha\widehat{s}_{ij}$ at minimum.

The purely forward update procedure (5.1) is as accurate as (2.8)–(2.9) in practical computation. We will see it here for a constant slowness and in section 6 for general media by numerical simulation. Let $h = \Delta x = \Delta z$ and $s(\mathbf{x}) \equiv s_0$ in Figure 3. Let the traveltimes on the narrow band have been computed accurately. Let us try to compute $\tau_{1,2}$, whose analytic value is $\sqrt{5}hs_0 \approx 2.236\cdot hs_0$. First, we utilize the formula (2.8)–(2.9):

$$(\tau_{1,2}-\tau_{0,2})^2 + (\tau_{1,2}-\tau_{1,1})^2 = h^2 s_0^2. \tag{5.2}$$

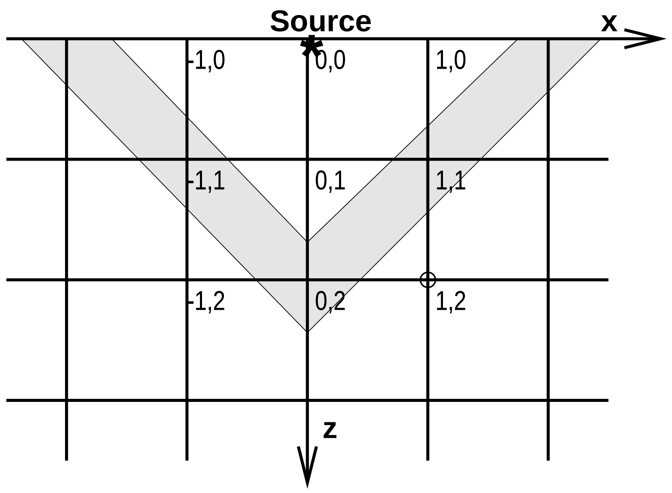

FIG. 3. *The narrow band method in two dimensions. The slowness is assumed to be constant. The shaded area denotes the narrow band at a certain moment. A point source is located at the origin marked by an asterisk.*

Replacing $\tau_{0,2}$ and $\tau_{1,1}$ by $2hs_0$ and $\sqrt{2}hs_0$, respectively, and solving the quadratic equation, we have

$$\tau_{1,2} \approx 2.351 \cdot hs_0,$$

which is the best value for $\tau_{1,2}$ from all former and other updates of (2.8)–(2.9). Now we apply (5.1):

$$\tau_{1,2} = \tau_{1,1} + h \cdot \max\left(\tfrac{1}{\sqrt{2}}s_0,\ \sqrt{s_0^2 - (\widehat{D}_x\tau_{1,1})^2}\right),$$
$$\tau_{1,2} = \tau_{0,2} + h \cdot \max\left(\tfrac{1}{\sqrt{2}}s_0,\ \sqrt{s_0^2 - (\widehat{D}_z\tau_{0,2})^2}\right). \tag{5.3}$$

Again, substituting the analytic values, we have the minimum

$$\tau_{1,2} \approx 2.324 \cdot hs_0,$$

which slightly improves accuracy over (2.8)–(2.9). The above argument can be applied to every point.

The main reason for the introduction of (5.1) is not for such a slight improvement in accuracy, but for more efficient incorporation of the average normal slowness. Note that the formula (5.1) incorporates ray directions more explicitly than (2.8)–(2.9). Since chasing raypaths is expensive, an approximation can be utilized as follows. For example, for the update from the west, an approximate average slowness can be found as

$$\widehat{s}_{i,j} \approx \begin{cases} s_{ij} & \text{if } s_{i-1,j} = s_{i,j}, \\ \dfrac{\log(v_{i-1,j}) - \log(v_{i,j})}{v_{i-1,j} - v_{i,j}} & \text{else.} \end{cases} \tag{5.4}$$

The above is compared with (4.3) in the same line. In heterogeneous media, the above average slowness produces more accurate traveltimes with (5.1) than with (2.8)–(2.9); see numerical results in section 6.

Table 1

Accuracy and efficiency of the GMM in two dimensions. A point source is located at $(x,z)=(3000,0)$.

		Point slowness				Average slowness			
		(2.8)–(2.9)		(5.1)		(2.8)–(2.9)		(5.1)	
v	M^2	CPU	$E(\tau^h)$	CPU	$E(\tau^h)$	CPU	$E(\tau^h)$	CPU	$E(\tau^h)$
v_1	100^2	0.13	4.6e-2	0.09	3.5e-2	0.15	3.4e-2	0.11	2.1e-2
	200^2	0.52	2.5e-2	0.37	1.8e-2	0.62	1.9e-2	0.44	8.2e-3
	400^2	2.14	1.3e-2	1.44	9.3e-3	2.43	1.0e-2	1.78	3.6e-3
v_2	100^2	0.14	3.6e-2	0.09	2.3e-2	0.17	3.5e-2	0.12	1.5e-2
	200^2	0.52	2.0e-2	0.35	1.1e-2	0.69	1.9e-2	0.51	7.2e-3
	400^2	2.10	1.1e-2	1.43	5.9e-3	2.71	1.0e-2	2.04	3.5e-3

Remark. In three dimensions, the numerical domain of dependence forms a solid quadrilateral whose edges have the same length. Since six of these surfaces cannot surround the point $\mathbf{x}_{ij}^k$ to be updated (so instability can happen), some auxiliary difference formulas should be introduced including the points diagonal from $\mathbf{x}_{ij}^k$ such that the union of numerical domains of dependence is all around in every direction to the target point. For the case $\Delta x=\Delta y=\Delta z$, it is not difficult to introduce such FDs. However, it would be very complicated for general cases. Alternatively, one may combine (2.8)–(2.9), the cell-oriented version, and (5.1), the line-oriented version.

6. Numerical experiments.

The GMM in section 3 is implemented in two and three dimensions for the FATTs of (2.1). Set the domain $\Omega=(0,6000\,\text{m})^d$, $d=2,3$. We consider four different velocity models: for $\mathbf{x}\in\Omega$,

$$\begin{aligned} v_1(\mathbf{x}) &= 1000+z \text{ m/s},\\ v_2(\mathbf{x}) &= 1000+0.2x+0.5z \text{ m/s},\\ v_3(\mathbf{x}) &= 1000+0.3x+0.2y+0.4z \text{ m/s},\\ v_4(\mathbf{x}) &= \begin{cases} 4500 \text{ m/s} & \text{if } \mathbf{x}\in[1500,4500]^d,\\ 2000 \text{ m/s} & \text{else.}\end{cases} \end{aligned} \tag{6.1}$$

(For 2D cases, the y-components are dropped.) The domain is partitioned with $h=\Delta x=\Delta y=\Delta z=6000/M$ for $M>0$. Point sources are imposed inside or on the surface of the domain. The traveltimes in linear velocities can be computed analytically; the numerical error is measured as

$$E(\tau^h)=\|\tau^h-\tau_{\text{analytic}}\|_\infty,$$

where τ^h denotes the computed traveltime with a grid size h. The average normal slowness is approximated by (4.3) and (5.4).

The main/driver routines are written in C++, and the core computation routines are in F77. The computation is carried out on a Gateway Solo, a 266 MHz laptop having 128M memory and a Linux operating system. The elapsed time CPU is the user time measured in seconds.

Table 1 compares the accuracy and efficiency of the GMM in two dimensions between the point slowness and the average slowness, and between the quadratic formula (2.8)–(2.9) and the purely forward formula (5.1). Two different velocities (v_1 and v_2) are selected here, and a point source is located at $(x,z)=(3000,0)$. As one can see from the table, incorporating the average slowness increases about 20–25% computation cost, while the formula (5.1) decreases about 30% the cost of (2.8)–(2.9). The velocity v_1 has a larger variance (in particular, in the vertical direction)

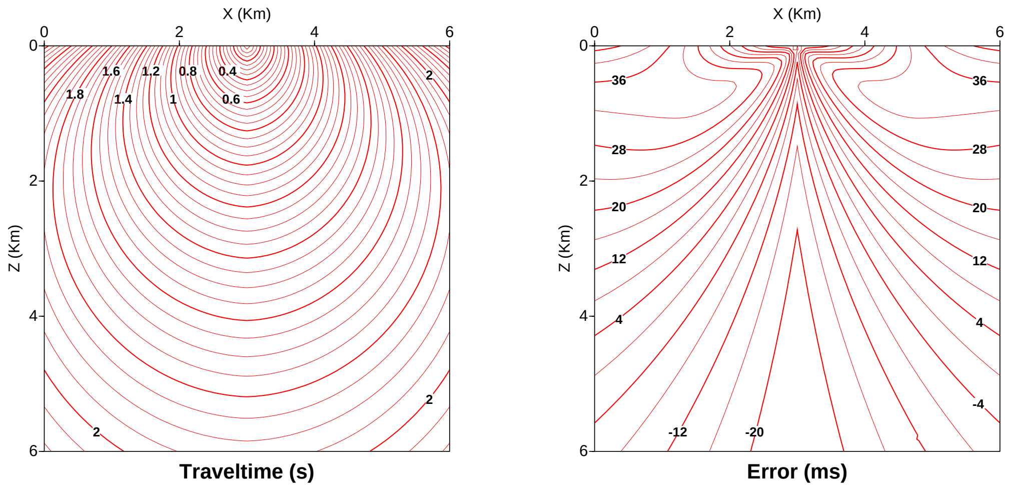

FIG. 4. *The computed traveltime (in seconds) and the error* $\tau^h - \tau_{\text{analytic}}$ *(in milliseconds). Set* $v = v_1$*, and the point source is located at* $(x, z) = (3000, 0)$*. The point slowness is incorporated with the updated formula* (2.8)–(2.9).

than v_2; the formula (5.1) incorporating the average slowness has improved accuracy more dramatically for v_1 than for v_2. We can see from the last panel of the table that its accuracy is slightly better than linear, even measured in the maximum norm. Such *superlinear convergence* has been observed for most linear velocity models. The following should also be noticed.

- The computation cost of the GMM is $\mathcal{O}(N)$ for any choice of slowness and update formula.
- The formula (5.1) is more efficient and accurate than (2.8)–(2.9), whether the slowness is averaged or not.
- The average slowness better improves accuracy when incorporated with the purely forward formula (5.1).

Here we have employed a roughly averaged slowness; however, the numerical error has decreased by a factor of two to three. When the velocity is highly oscillatory or when a higher-order FD scheme is employed, the average slowness is essential to incorporate. The point slowness will impose a first-order accuracy for heterogeneous media, no matter how accurate the FD scheme is.

Figure 4 depicts the computed traveltime (in seconds) and the error $\tau^h - \tau_{\text{analytic}}$ (in milliseconds) for the example discussed in Table 1 ($v = v_1$). The point slowness is incorporated with the update formula (2.8)–(2.9). The error (right side) indicates that the method is both over- and underestimating. Overestimation is not a danger but just an accuracy problem; it can be improved through the minimizing process using more available updating values. On the other hand, underestimation has no way to be fixed.

In Figure 5, we present the error $\tau^h - \tau_{\text{analytic}}$ (in milliseconds) for the point slowness (left) and the average slowness (right), both incorporated with the update formula (5.1). As in Figure 4, $v = v_1$, and the point source is located at $(x, z) = (3000, 0)$. No underestimation has been observed for the formula (5.1), as one can see from these pictures. The error for the average slowness is smaller and smoother than that of the

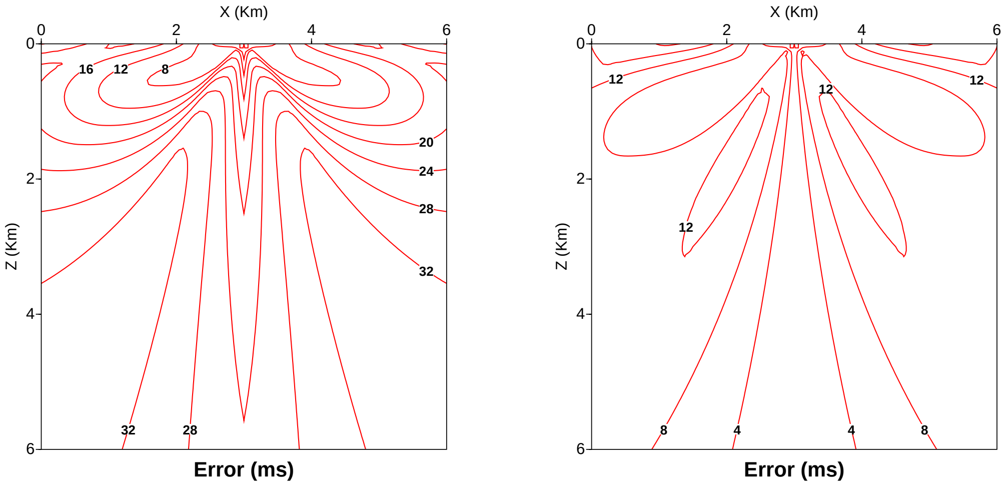

Fig. 5. *The error* $\tau^h - \tau_{\text{analytic}}$ *(in milliseconds) for the point slowness (left) and the average slowness (right). For the update formula,* (5.1) *is utilized. Set* $v = v_1$*, and the point source is located at* $(x, z) = (3000, 0)$.

Table 2
Accuracy and efficiency of the FMM and GMM in three dimensions. Set $v = v_3$*, and locate a point source at* $(x, y, z) = (3000, 3000, 1000)$*. The updated formula* (2.8)–(2.9) *is applied.*

	FMM				GMM			
	Point slowness		Average slowness		Point slowness		Average slowness	
M^3	CPU	$E(\tau^h)$	CPU	$E(\tau^h)$	CPU	$E(\tau^h)$	CPU	$E(\tau^h)$
30^3	1.37	1.2e-1	2.04	1.0e-1	0.98	1.2e-1	1.14	8.2e-2
60^3	19.81	6.2e-2	27.62	5.5e-2	7.88	6.2e-2	9.13	4.5e-2
120^3	395.8	3.2e-2	460.5	2.8e-2	64.06	3.2e-2	73.73	2.3e-2

point slowness. Note that the ray direction is vertical on the line segment $\{x = 3000\}$, where the solution incorporating the average slowness is computed without error. (It also can be figured out mathematically.)

In Table 2, we show numerical results for the FMM and GMM in three dimensions. The velocity is chosen as $v = v_3$, and a point source is located at $(x, y, z) = (3000, 3000, 1000)$. The update formula (2.8)–(2.9) is applied. We observe in 3D simulation that the computation cost of the FMM increases more rapidly than the problem size, while GMM costs $\mathcal{O}(N)$. One can easily expect that the GMM is three to five times faster than the FMM for problems of reasonable sizes. It should be noticed that the average slowness improves accuracy better for the GMM.

In Figure 6, we depict the computed traveltime of the GMM in three dimensions superimposed on the cross section $\{y = 3000\}$ of the velocity model $v = v_4$. The number of grid points is 100^3 and the source is located at $(x, y, z) = (1000, 3000, 1000)$. The update formula (2.8)–(2.9) is utilized incorporating the average slowness; the GMM takes 39.89 seconds. There one can see the shocks developed during the advancement of the wavefronts, due to the headwave out from the high velocity region.

7. Discussion.

The GMM is yet to be tested for realistic models, in particular, in three dimensions. Since the performance of the GMM is weakly dependent on the

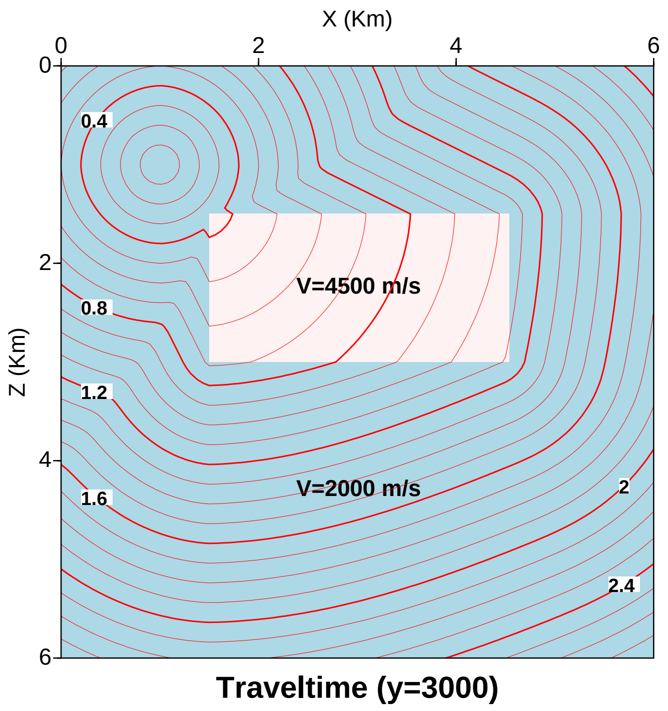

FIG. 6. *The computed traveltime of the GMM in three dimensions superimposed on the cross section* $\{y = 3000\}$ *of the velocity* $v = v_4$. *The number of grid points is* 100^3*, and the source is located at* $(x, y, z) = (1000, 3000, 1000)$.

velocity model, as one can see from Table 1, we can conclude that a 3D problem of a million unknowns can be solved in 40 seconds on a 266 MHz laptop. Nonetheless, I would like to discuss some important aspects on the accuracy, efficiency, and applicability of the level set methods, both the FMM [25, 27] and the GMM.

Higher-order extension. The level set methods are hard to incorporate high-order FD schemes. Sethian [27] suggested one-sided high-order schemes to be considered whenever the traveltimes at the corresponding points are available. Here one should notice that one-sided high-order FD schemes can produce a less accurate solution than the first-order scheme, in particular, in heterogeneous media. Even if the medium is constant, the one-sided second-order scheme easily produces underestimated solutions, as one can see from [27]. An interested reader can also check it for the computation of $\tau_{1,2}$ in Figure 3 using the second-order version of (5.2) presented in [27]. (Your result would be approximately $2.205 \cdot hs_0$; the exact value is $2.236 \cdot hs_0$.)

Computation cost. Now the level set method can be carried out for the computation cost of $\mathcal{O}(N)$ instead of $\mathcal{O}(N \log N)$. However, both the FMM and the GMM access the data (such as the solution, the narrow band points, and the binary tree (FMM)) in an almost random manner, rather than a systematic way along the loop indices. As a consequence, the algorithms can be more expensive than expected, depending on the computers adopted.

Applicability to the traveltime computation in seismology. Seismic image processing methods require an accurate computation of traveltimes of acoustic or elastic waves. When a set of seismic survey data is acquired utilizing the main frequency of 40Hz, for example, the period for the propagation of one wavelength is 25 milliseconds. For a reasonable image processing, the error should be much less than 25 milliseconds over the whole domain whose edge lengths are often 4,000 m to 9,000 m. For the example in Table 2, one has to choose approximately 300 points ($h = 20\,\text{m}$) in each direction to keep the error below 10 milliseconds; the expected computation time is 16 to 19 minutes in the same machine (ignoring the memory problem), which is extremely expensive. Note that for a section of seismic image, the traveltimes are often computed for more than 10,000 shot-gathers. (If a machine of the same speed tested here is used, it will take at least four months for the traveltime computation.) See [8, 9, 10, 12, 13, 21, 22, 31] for numerical methods and related issues in the traveltime computation and seismic image processing.

Overall, the level set methods should be far more improved (particularly, in accuracy) for a wider range of applications. The update formula discussed at the end of section 5 for the 3D problem, a second-order FD scheme applicable to heterogeneous media, and applications to anisotropic media will be the subjects of a forthcoming paper [11].

8. Conclusions.

An optimal level set method called the GMM has been suggested, based on a physical investigation for the solution on the narrow band of wavefronts. It has been applied to the computation of the FATTs of the eikonal equation in two and three dimensions. The GMM incorporates both the narrow band approach [25, 27] and the iterative scheme [23]; it converges in two iterations for a group of points in the narrow band. Strategies have been discussed to reduce the cost of the double-computation. We have introduced the average normal slowness to improve accuracy; its approximation is discussed for an efficient implementation. A new alternative update formula is suggested for the 2D problem, and it is numerically verified to be superior to that of the conventional level set methods in both efficiency and accuracy. For various velocities, grid sizes, and update formulae, the GMM has carried out its job for the cost $\mathcal{O}(N)$, where N is the total number of grid points. The GMM performs three to five times faster than the FMM for reasonable-size problems in three dimensions.

Acknowledgments. The author expresses his sincere thanks to Professor S. Osher, UCLA, for his interests and valuable comments. Constructive comments by two anonymous reviewers improved the paper and are much appreciated.

REFERENCES

[1] V. Cerveny, *Ray synthetic seismograms for complex two- and three-dimensional structures*, J. Geophys., 58 (1985), pp. 2–26.
[2] V. Cerveny, I. A. Molotkov, and I. Psencik, *Ray Methods in Seismology*, University of Karlova Press, Prague, 1977.
[3] M. Crandall and P. Lions, *Viscosity of solutions of Hamilton-Jacobi equations*, Trans. Amer. Math. Soc., 277 (1983), pp. 1–42.
[4] M. Crandall and P. Souganidis, *Developments in the theory of nonlinear first-order partial differential equations*, in Differential Equations, W. Knowles and R. Lewis, eds., North-Holland Math. Stud. 92, North-Holland, Amsterdam, 1984, pp. 131–143.
[5] J. Dellinger, *Anisotropic finite-difference traveltimes*, in Expanded Abstracts, Society of Exploration Geophysicists, Tulsa, OK, 1991, pp. 1530–1533.

[6] J. Dellinger and W. W. Symes, *Anisotropic finite-difference traveltimes using a Hamilton-Jacobi solver*, in Expanded Abstracts, Society of Exploration Geophysicists, Tulsa, OK, 1997, pp. 1786–1789.
[7] F. Friedlander, *Sound Pulses*, Cambridge University Press, Cambridge, UK, 1958.
[8] S. Gray and W. May, *Kirchhoff migration using eikonal equation traveltimes*, Geophysics, 59 (1994), pp. 810–817.
[9] S. Kim, *ENO-DNO-PS: A stable, second-order accuracy eikonal solver*, in Expanded Abstracts, Society of Exploration Geophysicists, Tulsa, OK, 1999, pp. 1747–1750.
[10] S. Kim, *On eikonal solvers for anisotropic traveltimes*, in Expanded Abstracts, Society of Exploration Geophysicists, Tulsa, OK, 1999, pp. 1875–1878.
[11] S. Kim, *An $\mathcal{O}(N)$ Second-Order Level Set Method*, in preparation, 2000.
[12] S. Kim and R. Cook, *3D traveltime computation using second-order ENO scheme*, Geophysics, 64 (1999), pp. 1867–1876.
[13] S. Kim, W. Symes, and M. A. El-Mageed, *Superconvergent difference formulas for traveltimes and amplitudes*, in Mathematical and Numerical Aspects of Wave Propagation, J. A. DeSanto, ed., SIAM, Philadelphia, 1998, pp. 591–593.
[14] P. Lax, *Hyperbolic Systems of Conservation Laws and the Mathematical Theory of Shock Waves*, SIAM, Philadelphia, 1973.
[15] P. L. Lions, *Generalized Solutions of Hamilton-Jacobi Equations*, Res. Notes Math. 69, Pitman, New York, 1982.
[16] R. Malladi and J. A. Sethian, *An $\mathcal{O}(N \log N)$ algorithm for shape modeling*, Proc. Natl. Acad. Sci. USA, 93 (1996), pp. 9389–9392.
[17] 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.
[18] S. Osher and C.-W. Shu, *High-order essentially nonoscillatory schemes for Hamilton–Jacobi equations*, SIAM J. Numer. Anal., 28 (1991), pp. 907–922.
[19] A. Popovici and J. A. Sethian, *Three dimensional traveltime computation using the fast marching method*, in Expanded Abstracts from the 67th Annual International Meeting of Society of Exploration Geophysicists, Dallas, TX, 1997, Society of Exploration Geophysicists, Tulsa, OK, 1997, pp. 1778–1781.
[20] J. Qian, C. Belfi, and W. Symes, *Adaptive finite-difference method for traveltime and amplitude*, in Expanded Abstracts, Society of Exploration Geophysicists, Tulsa, OK, 1999, pp. 1763–1766.
[21] F. Qin, Y. Luo, K. B. Olsen, W. Cai, and G. T. Schuster, *Finite-difference solution of the eikonal equation along expanding wavefronts*, Geophysics, 57 (1992), pp. 478–487.
[22] M. Reshef and D. Kosloff, *Migration of common shot gathers*, Geophysics, 51 (1986), pp. 324–331.
[23] E. Rouy and A. Tourin, *A viscosity solutions approach to shape-from-shading*, SIAM J. Numer. Anal., 29 (1992), pp. 867–884.
[24] W. A. J. Schneider, K. A. Ranzinger, A. H. Balch, and C. Kruse, *A dynamic programming approach to first arrival traveltime computation in media with arbitrarily distributed velocities*, Geophysics, 57 (1992), pp. 39–50.
[25] J. A. Sethian, *A fast marching level set method for monotonically advancing fronts*, Proc. Natl. Acad. Sci. USA, 93 (1996), pp. 1591–1595.
[26] J. A. Sethian, *Theory, algorithms, and applications of level set methods for propagating interfaces*, Acta Numerica, 1996 Acta Numer. 5, Cambridge University Press, Cambridge, UK, 1996, pp. 309–395.
[27] J. A. Sethian, *Fast marching methods*, SIAM Rev., 41 (1999), pp. 199–235.
[28] J. A. Sethian and A. Popovici, *3D traveltime computation using the fast marching method*, Geophysics, 64 (1999), pp. 516–523.
[29] J. Tsitsiklis, *Efficient algorithms for globally optimal trajectories*, IEEE Trans. Automat. Control, 40 (1995), pp. 1528–1538.
[30] J. van Trier and W. W. Symes, *Upwind finite-difference calculation of travel-times*, Geophysics, 56 (1991), pp. 812–821.
[31] J. E. Vidale, *Finite-difference calculation of travel times*, Bull. Seismol. Soc. Amer., 78 (1988), pp. 2062–2076.
[32] J. E. Vidale, *Finite difference calculation of traveltimes in three dimensions*, Geophysics, 55 (1990), pp. 521–526.