Journal of Computational Physics 228 (2009) 6440–6455

Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

Fast sweeping method for the factored eikonal equation ☆

Sergey Fomel [a], Songting Luo [b,*], Hongkai Zhao [b]

[a] Jackson School of Geosciences, The University of Texas at Austin, Austin, TX 78713, United States
[b] Department of Mathematics, University of California, Irvine, CA 92697, United States

ARTICLE INFO

Article history:
Received 30 October 2008
Received in revised form 6 April 2009
Accepted 16 May 2009
Available online 27 May 2009

Keywords:
Fast sweeping method
Eikonal equation
Factored eikonal equation
Source singularity

ABSTRACT

We develop a fast sweeping method for the factored eikonal equation. By decomposing the solution of a general eikonal equation as the product of two factors: the first factor is the solution to a simple eikonal equation (such as distance) or a previously computed solution to an approximate eikonal equation. The second factor is a necessary modification/correction. Appropriate discretization and a fast sweeping strategy are designed for the equation of the correction part. The key idea is to enforce the causality of the original eikonal equation during the Gauss–Seidel iterations. Using extensive numerical examples we demonstrate that (1) the convergence behavior of the fast sweeping method for the factored eikonal equation is the same as for the original eikonal equation, i.e., the number of iterations for the Gauss–Seidel iterations is independent of the mesh size, (2) the numerical solution from the factored eikonal equation is more accurate than the numerical solution directly computed from the original eikonal equation, especially for point sources.

1. Introduction

The eikonal equation

$$|\nabla T|^2 = S^2(\mathbf{x}) \tag{1}$$

describes the traveltime $T(\mathbf{x})$ of a wave propagating with slowness (refraction index) $S(\mathbf{x})$ in space $\mathbf{x} \in \mathbb{R}^n$. In the case of anisotropic wave propagation, S depends additionally on $\nabla T/|\nabla T|$. When $S(\mathbf{x})$ is equal to one, the traveltime $T(\mathbf{x})$ corresponds to the distance function.

The eikonal equation plays an important role in many practical applications: computer vision, material science, computational geometry, etc. [16]. In seismic imaging, in particular, finite-difference solutions of the eikonal equation are used routinely for computing traveltime tables for numerical modeling and migration of seismic waves [22,20,17,10]. Although limited for computing only first-arrival traveltimes [7], eikonal solvers can be extended in several different ways to image multiple arrivals [3].

In this paper, we derive the factored eikonal equation by assuming that either an analytical or a numerical solution is available for Eq. (1) in the same domain but with different right-hand side. The solution $T(\mathbf{x})$ is then represented as a product of the known solution and an unknown factor, which satisfies the factored eikonal equation. The hope is that if $T(\mathbf{x})$ is some perturbation of the available solution, solving the factored eikonal equation either is easier or produces more accurate solution numerically.

☆ Research partially supported by NSF Grant DMS-0513073, ONR Grant N00014-02-1-0090 and DARPA Grant N00014-02-1-0603.
* Corresponding author. Tel.: +1 949 824 2456.
E-mail addresses: sergey.fomel@beg.utexas.edu (S. Fomel), luos@math.uci.edu (S. Luo), zhao@math.uci.edu (H. Zhao).

0021-9991/$ - see front matter © 2009 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2009.05.029

We develop a numerical algorithm based on the fast sweeping method to solve the factored eikonal equation and to evaluate the resultant gain in accuracy. The fast sweeping method (FSM) is an efficient iterative method that uses Gauss–Seidel iterations with alternating orderings to solve a wide range of Hamilton–Jacobi equations and other type of hyperbolic problems [4,27,19,26,8,9,25,15,14,24]. With an appropriate upwind scheme that captures the causality of the underlying partial differential equation, the iteration can converge in a finite number of iterations independent of the mesh size, which was proved for special cases in [26]. The intuition is the following: Information propagates along characteristics. Using a systematic alternating ordering strategy, all directions of characteristics can be divided into a finite number of groups and each group is covered simultaneously by one of the orderings. Moreover, any characteristics can be covered by a finite number of orderings [26]. With an appropriate upwind scheme that enforces the causality of the underlying partial differential equation, a Gauss–Seidel iteration propagates correct information in each updating along characteristics whose directions agree with the orderings.

After outlining the theory and the numerical algorithm, we conduct a series of numerical experiments, where numerical solutions are compared with analytical solutions for model problems. A significant improvement in accuracy is observed in comparison with FSM applied directly to the original eikonal equation. Finally, we apply our method to compute traveltime tables for the benchmark Marmousi model.

2. Factored eikonal equation

A fundamental property of Eq. (1) is that scaling slowness S by a constant corresponds to scaling traveltime T by the same constant. This property was used in seismic reflection imaging in the method of common-reflection-point scans [2,1].

Let us consider a factored decomposition

$$S(\mathbf{x}) = S_0(\mathbf{x})\,\alpha(\mathbf{x}), \tag{2}$$

$$T(\mathbf{x}) = T_0(\mathbf{x})\,\tau(\mathbf{x}) \tag{3}$$

and assume that

$$|\nabla T_0|^2 = S_0^2(\mathbf{x}). \tag{4}$$

If both T_0 and S_0 are known (either from an analytical solution or from a previous numerical computation), one can pose the problem of a numerical evaluation of the correction $\tau(\mathbf{x})$ on a computational grid with appropriate boundary conditions (see Remark 1 in the next section for assigning boundary conditions for τ). The function substitutions transform Eq. (1) to the *factored eikonal equation*

$$T_0^2(\mathbf{x})\,|\nabla\tau|^2 + 2\,T_0(\mathbf{x})\,\tau(\mathbf{x})\,\nabla T_0\cdot\nabla\tau + [\tau^2(\mathbf{x}) - \alpha^2(\mathbf{x})]\,S_0^2(\mathbf{x}) = 0. \tag{5}$$

When $\alpha(\mathbf{x})$ is a constant, the solution of Eq. (5) is trivial. When $\alpha(\mathbf{x})$ is not a constant but slowly varying, the hope is that accuracy of evaluating $T(\mathbf{x})$ from solving the factored eikonal equation can be greatly improved compared to a direct numerical solution of the original eikonal Eq. (1). One scenario is that point source singularities in the original solution $T(\mathbf{x})$ are well captured by $T_0(\mathbf{x})$. So the correction $\tau(\mathbf{x})$ is a smooth function in a neighborhood of the point sources. For example, when computing the traveltime for a point source the solution is singular at the source. Special treatment, such as using local grid refinement near the source, has to be implemented in order to achieve high order accuracy for the numerical solution to the eikonal equation [13]. However, locally the singularity of the solution to a regular eikonal equation (assuming $S(\mathbf{x})$ is smooth and strictly positive) at a point source is the same as the singularity of the distance function to that point source up to a smooth modification. Numerical tests in Section 4 show that the numerical solution based on the factored eikonal equation can be significantly more accurate than the numerical solution computed directly from the original eikonal equation. Also note that although ∇T_0 does not exist at the source point, it is well-defined away from the source point. It provides good approximation of all ray directions near the point source. This is crucial for computing accurate solutions away from the point source, which cannot be approximated easily on a discrete mesh.

When taking $S_0(\mathbf{x}) = 1, T_0(\mathbf{x})$ is the distance function. In the case of simple domains and boundary conditions, the distance can be evaluated analytically. For example, the distance from a point source at $\mathbf{x_0}$ is $T_0(\mathbf{x}) = |\mathbf{x} - \mathbf{x_0}|$, which transforms Eq. (5) to

$$|\mathbf{x} - \mathbf{x_0}|^2\,|\nabla\tau|^2 + 2\,\tau(\mathbf{x})\,(\mathbf{x} - \mathbf{x_0})\cdot\nabla\tau + \tau^2(\mathbf{x}) - S^2(\mathbf{x}) = 0. \tag{6}$$

A numerical solution of Eq. (6) was investigated previously in geophysical applications [12,23]. Simple analytical solutions exist for several other particular cases of slowness distributions such as a constant gradient of the slowness squared, a constant gradient of the velocity (inverse slowness), etc. [5].

Remark. In general T may have other singularities, e.g., due to the intersections of different characteristics in addition to point singularities at source points. Hence τ may also have singularities away from sources. In practice it is impossible to know the exact locations of these singularities without knowing the exact solution. However, for numerical computation (especially if the scheme is upwind), singularities caused by the intersections of different characteristics are less damaging or polluting than point singularities at source points from where characteristics emanate. In principle we only need to choose a

T_0 that captures point singularities of T while being smooth elsewhere. In Section 4, numerical Example 2 and 4 show that as long as T_0 captures point singularities of T and is smooth elsewhere, solving the factored eikonal equation improves the accuracy and achieves uniform first order convergence even if T has other singularities (due to multiple sources) away from source points.

3. Numerical algorithm

The solution to the eikonal Eq. (1) is the first-arrival time. So the causality, required by FSM, is very simple, the value of T at a grid point should be determined by its neighbors whose values are smaller. The key point in discretizing Eq. (5) is to design an upwind scheme following the causality of the original eikonal equation.

We present the algorithm on a rectangular mesh in 2D (the algorithm can be easily extended to higher dimensions). Fig. 1 shows an interior grid point C with neighbors W, E, N, S. We discretize Eq. (5) on four triangles: $\Delta CEN, \Delta CNW, \Delta CWS$ and ΔCSE. For example, on triangle ΔCWS, we get a discretized equation:

$$T_0^2(C)\left|\left(\frac{\tau_C-\tau_W}{h},\frac{\tau_C-\tau_S}{h}\right)\right|^2+2T_0(C)\tau_C\nabla T_0(C)\cdot\left(\frac{\tau_C-\tau_W}{h},\frac{\tau_C-\tau_S}{h}\right)+\{\tau_C^2-\alpha^2(C)\}S_0^2(C)=0. \tag{7}$$

By solving this equation, we enforce the causality condition.

Causality condition: assume τ_C^h is an appropriate root of Eq. (7), then we require $\tau_C^h T_0(C)\geqslant\tau_W T_0(W),\ \tau_C^h T_0(C)\geqslant\tau_S T_0(S)$, e.g., Fig. 1.

Eq. (7) may not have real roots or may have real roots that don't satisfy the causality condition. Under such circumstances, it means that the values of τ_W and τ_S don't support a valid solution τ_C within the triangle ΔCWS using linear approximations. This may happen especially when the initial guess is arbitrary. However, note that Eq. (5) is a hyperbolic equation, we can use the classical method of characteristics to pass the information of τ from W to C and from S to C along edges $\vec{WC}, \vec{SC}$ respectively.

The characteristic equation for Eq. (5), with $(\tau_x,\tau_y)=(p,q)$, is:

$$\begin{cases}\left(\frac{dx}{dt},\frac{dy}{dt}\right)=(2T_0^2p+2\tau T_0T_{0x},2T_0^2q+2\tau T_0T_{0y})=2T_0\nabla T,\\ \frac{d\tau}{dt}=2T_0^2(p^2+q^2)+2\tau T_0(T_{0x}p+T_{0y}q)=2T_0\nabla\tau\cdot\nabla T.\end{cases} \tag{8}$$

The first equation shows that the characteristics for the factored eikonal equation in the physical space are the same as those for the original eikonal equation up to a scaling/reparametrization by T_0. By using the method of characteristics on edges $\vec{WC}, \vec{SC}$ respectively, one can calculate two values of $\tau:\tau_{WC}$ and τ_{SC}. For example, on edge $\vec{WC}=(\delta x,\delta y), |\vec{WC}|=L$ (in this case, $\delta x=h,\delta y=0,L=h$), using first equation in Eq. (8), we have

$$\begin{cases}\delta x=(2T_0^2p+2\tau T_0T_{0x})\delta t\\ \delta y=(2T_0^2q+2\tau T_0T_{0y})\delta t\end{cases}\Rightarrow\begin{cases}p=\left(\frac{\delta x}{\delta t}-2\tau T_0T_{0x}\right)\frac{1}{2T_0^2},\\ q=\left(\frac{\delta y}{\delta t}-2\tau T_0T_{0y}\right)\frac{1}{2T_0^2}.\end{cases}$$

Note that either using $\left(\frac{dx}{dt}\right)^2+\left(\frac{dy}{dt}\right)^2=4T_0^2S^2$ or plugging (p,q) into Eq. (5), we get an equation of δt, solve for δt to get $\delta t=\frac{L}{2T_0S(C)}$. Now using second equation in Eq. (8), we have $\tau=\tau_W+[2T_0^2(p^2+q^2)+2\tau T_0(T_{0x}p+T_{0y}q)]\delta t$, solve for τ to get $\tau_{WC}=\frac{LS(C)+\tau_WT_0}{T_0+(\delta xT_{0x}+\delta yT_{0y})}$.

We present the local solver.

Local Solver:

1. Initialization: assign initial boundary values on boundary grid points.
2. Gauss–Seidel iteration: sweeping the domain with four alternating orderings repeatedly:

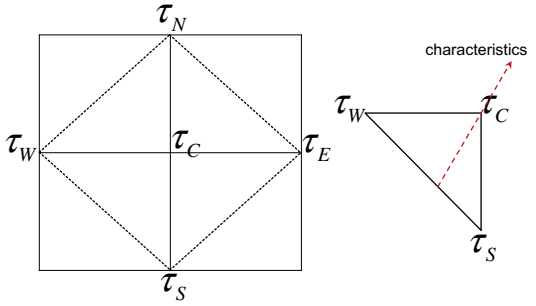

Fig. 1. Rectangular mesh.

(1) $i = 1 : I, j = 1 : J$, (2) $i = 1 : I, j = J : 1$,
(3) $i = I : 1, j = 1 : J$, (4) $i = I : 1, j = J : 1$.

- At each gird point, discretize the factored eikonal equation on 4 triangles ΔCEN, ΔCNW, ΔCWS and ΔCSE, and solve the discretized equations on each triangle. For example, on triangle ΔCWS, solve Eq. (7) for two possible roots, $\tau_{C,1}$ and $\tau_{C,2}$.
- If there are two real roots, $\tau_{C,1}$ and $\tau_{C,2}$, then
  - if both $\tau_{C,1}, \tau_{C,2}$ satisfy the causality condition, then

    $$T_{WS} = \min\{\tau_{C,1} T_0(C), \tau_{C,2} T_0(C)\}.$$

  - else if $\tau_{C,1}$ satisfies the causality condition, then $T_{WS} = \tau_{C,1} T_0(C)$.
  - else if $\tau_{C,2}$ satisfies the causality condition, then $T_{WS} = \tau_{C,2} T_0(C)$.
  - else if none of the two roots satisfies the causality condition, then use the method of characteristics on edges $\overrightarrow{WC}, \overrightarrow{SC}$ to get τ_{WC} and τ_{SC}. Enforce causality $\tau_{WC} T_0(C) \geqslant T_W, \tau_{SC} T_0(C) \geqslant T_S$. And choose $T_{WS} = \min\{\tau_{WC} T_0(C), \tau_{SC} T_0(C)\}$.
- else, use the method of characteristics on edges $\overrightarrow{WC}, \overrightarrow{SC}$ to get τ_{WC} and τ_{SC}. Enforce causality $\tau_{WC} T_0(C) \geqslant T_W$, $\tau_{SC} T_0(C) \geqslant T_S$. And choose $T_{WS} = \min\{\tau_{WC} T_0(C), \tau_{SC} T_0(C)\}$.
- Choose the minimum from 4 triangles, $T_C = \min\{T_{EN}, T_{NW}, T_{WS}, T_{SE}\}$.
- $\tau_C = \frac{T_C}{T_0(C)}$.

Remark 1. The boundary condition can be assigned by two ways. At a boundary point $\mathbf{x_0}$, (1), if $T_0(\mathbf{x_0}) \neq 0$, then $\tau(\mathbf{x_0}) = \frac{T(\mathbf{x_0})}{T_0(\mathbf{x_0})}$. (2), if $T_0(\mathbf{x_0}) = 0$, e.g., at a source point, $T(\mathbf{x_0})$ is also 0. From l'Hospital rule or Eq. (5), we have $\tau(\mathbf{x_0}) = \alpha(\mathbf{x_0})$.

Remark 2. If $T_0 = 0$ at some point away from the source point(s), then the division by T_0 doesn't hold anymore. A shift of T_0 by adding a large enough positive constant is needed in this case. The shifted T_0 is positive and satisfies the same eikonal Eq. (4).

Remark 3. Analytically, the original eikonal equation for T and the factored eikonal equation for τ, including the method of characteristics are all equivalent. However, the latter approach will give better approximations for ∇T near a source point. With the factorization $T = \tau T_0$ we avoid taking ∇T directly. Instead we have $\nabla T = \nabla \tau T_0 + \tau \nabla T_0$ in which T_0 captures the point singularity and ∇T_0 is precisely known away from the source point. By solving a differential equation for τ we achieve better approximations. Therefore, τT_0 is also more accurate. Another important point is that the causality condition for τ is equivalent to the one for T and we know the enforcement of the causality condition during the fast sweeping iteration guarantees the convergence of T for the original eikonal equation [26].

4. Numerical tests

In this section we test our numerical algorithm on a variety of examples to demonstrate both accuracy and efficiency.

Eq. (1) has analytical solutions for the case of point source and some special forms of the slowness function $S(\mathbf{x})$. In Examples 1–7, we use analytical examples to test the proposed numerical scheme. In Examples 2 and 4, we test cases with two source points. In these tests errors are measured in maximum norm. Convergence criterion for all fast sweeping iterations is when maximum difference between two consecutive iterations is less than 10^{-9}.

For completion analytical derivations of these examples are included in the Appendix.

Example 1 (*Constant gradient of slowness squared*). As shown in Appendix A, when a point source is located at point $\mathbf{x}_0$ and the slowness distribution has the form

$$S^2(\mathbf{x}) = S_0^2 + 2\,\mathbf{g}_0 \cdot (\mathbf{x} - \mathbf{x}_0) \tag{9}$$

with constant S_0 and $\mathbf{g}_0$, then the analytical solution is

$$T(\mathbf{x}) = \overline{S}^2\,\sigma - |\mathbf{g}_0|^2\,\frac{\sigma^3}{6}, \tag{10}$$

where

$$\sigma = \frac{\sqrt{2\left(\overline{S}^2 \pm \sqrt{\overline{S}^4 - |\mathbf{g}_0|^2\,|\mathbf{x} - \mathbf{x}_0|^2}\right)}}{|\mathbf{g}_0|} \tag{11}$$

and

$$\overline{S}(\mathbf{x}) = \sqrt{\frac{S^2(\mathbf{x}) + S_0^2}{2}} = \sqrt{S_0^2 + \mathbf{g}_0 \cdot (\mathbf{x} - \mathbf{x}_0)}. \tag{12}$$

In general, two different solutions exist for two different signs in Eq. (11). The branch of the solution that turns into the constant-slowness solution for $\mathbf{g}_0$ approaching $\mathbf{0}$ corresponds to the negative sign and can be written in the alternative form

$$\sigma^2 = \frac{2\left(\bar{S}^2 - \sqrt{\bar{S}^4 - |\mathbf{g}_0|^2 |\mathbf{x}-\mathbf{x}_0|^2}\right)}{|\mathbf{g}_0|^2} = \frac{2|\mathbf{x}-\mathbf{x}_0|^2}{\bar{S}^2 + \sqrt{\bar{S}^4 - |\mathbf{g}_0|^2 |\mathbf{x}-\mathbf{x}_0|^2}}, \tag{13}$$

which avoids division by zero.

We test one setup with parameters:

- $\mathbf{x}_0 = (0,0)$
- computational domain (km): $[0,1.5]\times[0,0.5]$
- $S_0 = 2$ s/km
- $\mathbf{g}_0 = \{0,-3\}$ s^2/km^3

Table 1 shows the results by both the factored and the original eikonal equation. Fig. 2 shows the plots of analyticalical solutions. The analytical solution is plotted on a 150×50 mesh. The comparison of solutions is plotted in the region $[0,0.5]\times[0,0.5]$ on a 50×50 mesh. As shown in the figure, there is a "shadow zone" for the analytical solution where no characteristics of T can enter, resulting in the non-existence of an analytical solution in this region. In other words, the square root in Eq. (11) does not hold in this region since it is a square root of some negative value. For comparison purpose, we measure errors only in the sub-domain $[0,0.5]\times[0,0.5]$, although the numerical solution can still be computed in the whole domain. The results show that the factored eikonal equation captures the right causality of the solution. The computation is as efficient as solving the original eikonal equation using fast sweeping method. At the same time the numerical solution is much more accurate in terms of both the error magnitude and order of convergence thanks to the fact that the singularity at the point source is well captured by the distance function. For our first order upwind scheme a perfect first order of convergence is achieved. On the other hand a first order discretization of the original eikonal equation cannot achieve first order accuracy due to the singularity at the point source.

Example 2. Constant gradient of slowness squared continued).
In this example, we test a case with two source points located at (0,0), (1.5,0). The equation and computational domain are the same as in Example 1. Three choices of T_0 are tested:

- (1) $T_0 = \min\{dist(x,y,0,0), dist(x,y,1.5,0)\}$, which is the distance function to the two source points and satisfies $|\nabla T_0| = 1$ in the viscosity sense. Both T_0 and T have singularities at the same equal distance locations.
- (2) $T_0 = dist(x,y,0,0)\times dist(x,y,1.5,0)$, in which case $|\nabla T_0|$ can be computed exactly and T_0 is smooth away from the two point sources.
- (3) $T_0 = \min\{dist(x,y,0,0), dist(x,y,1.5,0), dist(x,y,0.75,-0.1)\}$, in which case $|\nabla T_0| = 1$ in the viscosity sense. Both T and T_0 have singularities but at different locations.

To make sense of the factorized equation mathematically, ∇T_0 has to exist. For choice 1 and choice 3, T_0 has singularities in addition to point singularities. We use different approximations of ∇T_0 to cope with the singularities. For choice 2, T_0 is smooth away from source points, so exact ∇T_0 is used. Since the exact solution T has a "shadow zone" for the same reason as in Example 1, we again measure error only in the region $[0,1.5]\times[0,0.25]$.

Table 1
Example 1: numerical error.

Mesh	Error of T on $[0,0.5]\times[0,0.5]$	# Iterations
Factored eikonal equation		
150×50	0.0010702	3
300×100	0.0005348	3
600×200	0.0002673	3
1200×400	0.0001336	3
Original eikonal equation		
150×50	0.0214127	3
300×100	0.0129566	3
600×200	0.0076381	3
1200×400	0.0044107	3

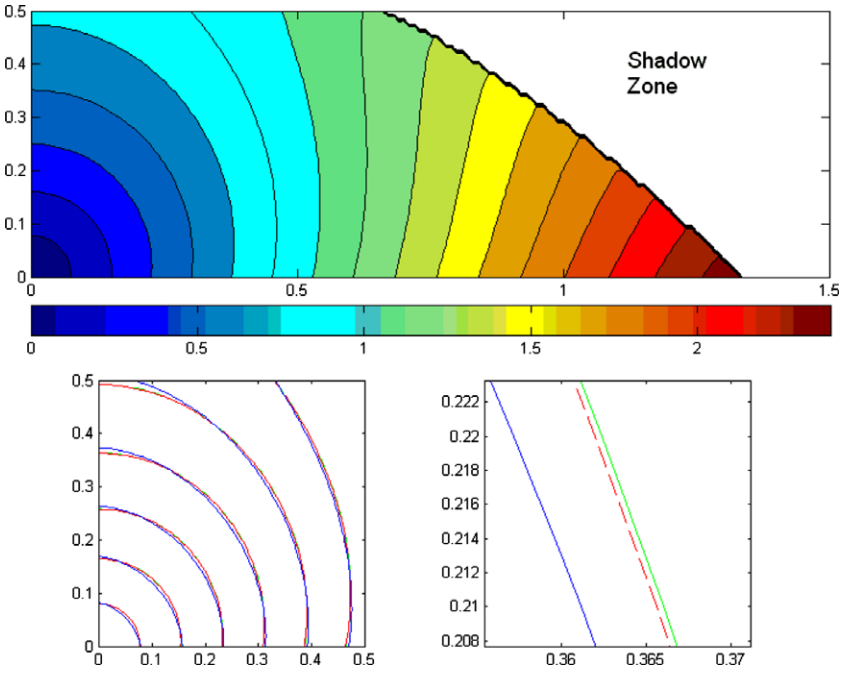

Fig. 2. Example 1. Top: analytical solution with a shadow zone. Bottom (left: contour plots, right: zoom in): green line – analytical solution, red dashed line – factored eikonal equation, blue dotted line – original eikonal equation. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 2 shows the results with ∇T_0 chosen in the following way. For choice 1, T_0 has singularities at equal distance line, we choose $\nabla T_0 = \nabla dist(x, y, 0, 0)$ at equal distance line and exact ∇T_0 elsewhere in our test. Similarly, for choice 3, at equal distance line, if $dist(x, y, 0, 0) < dist(x, y, 1.5, 0)$ we choose $\nabla T_0 = \nabla dist(x, y, 0, 0)$, if $dist(x, y, 1.5, 0) < dist(x, y, 0, 0)$ we choose $\nabla T_0 = \nabla dist(x, y, 1.5, 0)$, and exact ∇T_0 is used elsewhere.

First we can see that with choice 1 and 2, the factored eikonal equation has much better results than the original eikonal equation. Although choice 1 has better results than choice 2, since the singularities of T and T_0 coincide at the equal distance line and characteristics flow into singularities similarly in both cases, choice 1 is impractical in general. For choice 3, T_0 has singularities at different locations from those of T, it needs more iterations and produces much worse results since the

Table 2
Example 2: numerical error.

Mesh	Error of T on $[0, 1.5] \times [0, 0.25]$	# Iterations
Factored eikonal equation: $T_0 = \min\{dist(x, y, 0, 0), dist(x, y, 1.5, 0)\}$		
150×50	0.0004933	5
300×100	0.0002456	5
600×200	0.0001225	5
1200×400	0.0000611	5
Factored eikonal equation: $T_0 = dist(x, y, 0, 0) \times dist(x, y, 1.5, 0)$		
150×50	0.0050798	5
300×100	0.0025370	5
600×200	0.0012679	5
1200×400	0.0006338	5
Factored eikonal equation: $T_0 = \min\{dist(x, y, 0, 0), dist(x, y, 1.5, 0), dist(x, y, 0.75, -0.1)\}$		
150×50	0.0607602	9
300×100	0.0285019	9
600×200	0.0139638	9
1200×400	0.0066409	9
Original eikonal equation		
150×50	0.0205986	5
300×100	0.0124967	5
600×200	0.0073907	5
1200×400	0.0042815	5

characteristics (or causality) of the original eikonal equation are messed up at singularities of T_0. However, it is interesting to note that results are still first order. In conclusion, one should choose T_0 that captures source singularities while being smooth elsewhere like the product of distance functions to point sources.

If for choice 1 and choice 3, ∇T_0 is approximated with upwind finite differences instead of being chosen as above, as shown in Table 3, choice 1 still has good results, but not as good as in Table 2 since ∇T_0 is not exact, however, choice 3 does not work anymore, the iterations don't converge due to the poor approximation of ∇T_0 at kinks where ∇T_0 is not well-defined.

Example 3 (*Constant gradient of velocity*). When a point source is located at point $\mathbf{x}_0$ and the slowness distribution has the form

$$\frac{1}{S(\mathbf{x})} = \frac{1}{S_0} + \mathbf{G}_0 \cdot (\mathbf{x} - \mathbf{x}_0) \tag{14}$$

with constant S_0 and $\mathbf{G}_0$ (constant gradient of velocity), the analytical solution is

$$T(\mathbf{x}) = \frac{1}{|\mathbf{G}_0|} \operatorname{arccosh}\left(1 + \frac{1}{2} S(\mathbf{x}) S_0 |\mathbf{G}_0|^2 |\mathbf{x} - \mathbf{x}_0|^2\right), \tag{15}$$

where arccosh is the inverse hyperbolic cosine function

$$\operatorname{arccosh}(z) = \ln\left(z + \sqrt{z^2 - 1}\right).$$

We test one setup with parameters:

- $\mathbf{x}_0 = (0, 0)$
- computational domain (km): $[0, 1] \times [0, 0.5]$
- $S_0 = 2$ s/km
- $\mathbf{G}_0 = \{0, -1\}\, 1/s$

Table 4 shows the results by both the factored and the original eikonal equation. Fig. 3 shows the plots of analytical and numerical solutions. The analytical solution is plotted on a 160×80 mesh. The comparison of solutions is plotted on an 80×80 mesh. As shown in bottom figure (circled region), some of the characteristics of T don't stay in the computational domain all the time. They leave the computational domain and come back. In our numerical computation we don't assume knowledge of this behavior (as in practice) and just use outflow boundary condition at the computational domain. Hence we don't measure error due to the wrong boundary condition. We only record error in the region $[0, 0.5] \times [0, 0.5]$, although the numerical computation is done in the whole domain with outflow boundary condition. The results show the same observations as in Example 1.

Table 3
Example 2: upwind approximation of ∇T_0.

Mesh	Error of T on $[0, 1.5] \times [0, 0.25]$	# Iterations
Factored eikonal equation: $T_0 = \min\{dist(x, y, 0, 0), dist(x, y, 1.5, 0)\}$		
150×50	0.0033254	5
300×100	0.0016804	5
600×200	0.0008447	5
1200×400	0.0004235	5

Table 4
Example 3: numerical error.

Mesh	Error of T on $[0, 0.5] \times [0, 0.5]$	# Iterations
Factored eikonal equation		
160×80	0.0007115	3
320×160	0.0003555	3
640×320	0.0001777	3
1280×640	0.0000888	3
Original eikonal equation		
160×80	0.0140801	3
320×160	0.0084309	3
640×320	0.0049316	3
1280×640	0.0028312	3

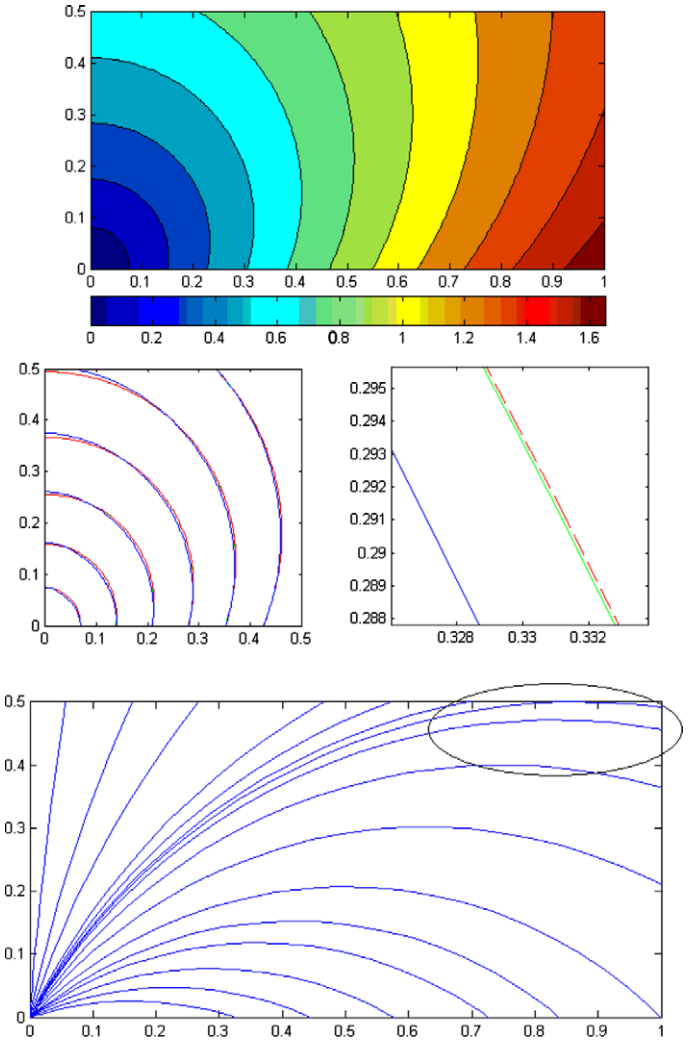

Fig. 3. Example 3. Top: analytical solution. Middle (left: contour plots, right: zoom in): green line – analytical solution, red dashed line – factored eikonal equation and blue dotted line – original eikonal equation. Bottom: rays. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Example 4. [Constant gradient of velocity continued] In this example, we test one case with two source points (0,0), (1,0). Similar choices of T_0 are tested as in Example 2:

- (1), $T_0 = \min\{dist(x,y,0,0), dist(x,y,1,0)\}$,
- (2), $T_0 = dist(x,y,0,0) \times dist(x,y,1,0)$,
- (3), $T_0 = \min\{dist(x,y,0,0), dist(x,y,1,0), dist(x,y,0.5,-0.1)\}$.

Tables 5 and 6 show the results with ∇T_0 chosen as in Example 2. For upwind approximations of ∇T_0, choice 1 has similar results as in Example 2, however, choice 3 works in this case with first order accuracy. But error magnitude is worse than the original eikonal equation and more iterations are needed to converge.

Example 5. In this example, we take the analytical solution from Example 3 as T_0 in Example 1. Table 7 shows the results by the factored eikonal equation. Fig. 4 shows the analytical solution on a 150×50 mesh and the comparison of solutions on a 50×50 mesh.

Example 6. In this example, we take the analytical solution from Example 1 as T_0 in Example 3. Since T_0 has a "shadow zone", we restrict our computation to the domain $[0, 0.5] \times [0, 0.5]$ (km).

Table 8 shows the results by the factored eikonal equation. Fig. 5 shows the plots of analytical and numerical solutions on an 80×80 mesh.

Table 5
Example 4: numerical error.

Mesh	Error of T	# Iterations
Factored eikonal equation: $T_0 = \min\{dist(x,y,0,0), dist(x,y,1,0)\}$		
160×80	0.0007115	5
320×160	0.0003555	5
640×320	0.0001777	5
1280×640	0.0000888	5
Factored eikonal equation: $T_0 = dist(x,y,0,0) \times dist(x,y,1,0)$		
160×80	0.0031731	5
320×160	0.0015822	5
640×320	0.0007902	5
1280×640	0.0003949	5
Factored eikonal equation: $T_0 = \min\{dist(x,y,0,0), dist(x,y,1,0), dist(x,y,0.5,-0.1)\}$		
160×80	0.0198242	9
320×160	0.0097473	9
640×320	0.0046399	9
1280×640	0.0022761	9
Original eikonal equation		
160×80	0.0140801	5
320×160	0.0084309	5
640×320	0.0049316	5
1280×640	0.0028312	5

Table 6
Example 4: upwind approximation of ∇T_0.

Mesh	Error of T	# Iterations
Factored eikonal equation: $T_0 = \min\{dist(x,y,0,0), dist(x,y,1,0)\}$		
160×80	0.0022748	5
320×160	0.0011893	5
640×320	0.0006069	5
1280×640	0.0003067	5
Factored eikonal equation: $T_0 = \min\{dist(x,y,0,0), dist(x,y,1,0), dist(x,y,0.5,-0.1)\}$		
160×80	0.0396539	9
320×160	0.0185494	9
640×320	0.0087090	9
1280×640	0.0042085	9

Remark. These two examples demonstrate that as long as T_0 captures the singularity of T at source points, using factored eikonal equation can improve the numerical results.

Example 7. In this example, we test a model with a plane-wave source, the analytical solution is:

$$T(\mathbf{x}) = \bar{S}^2\sigma - |g_0|^2\frac{\sigma^3}{6}, \tag{16}$$

where

$$\sigma = \frac{\sqrt{S^2 - S_0^2 - 2g_z(z - z_0)}}{|g_x|} = \frac{\sqrt{S^2 - S_0^2 - 2g_z z}}{|g_x|},$$

$\bar{S}$ is as in Eq. (12), $\mathbf{x} = (x,z)$ and $g_0 = (g_x, g_z)$.

Table 7
Example 5: numerical error.

Mesh	Error of T on $[0,0.5] \times [0,0.5]$	# Iterations
Factored eikonal equation		
150×50	0.0021033	3
300×100	0.0010510	3
600×200	0.0005253	3
1200×400	0.0002626	3

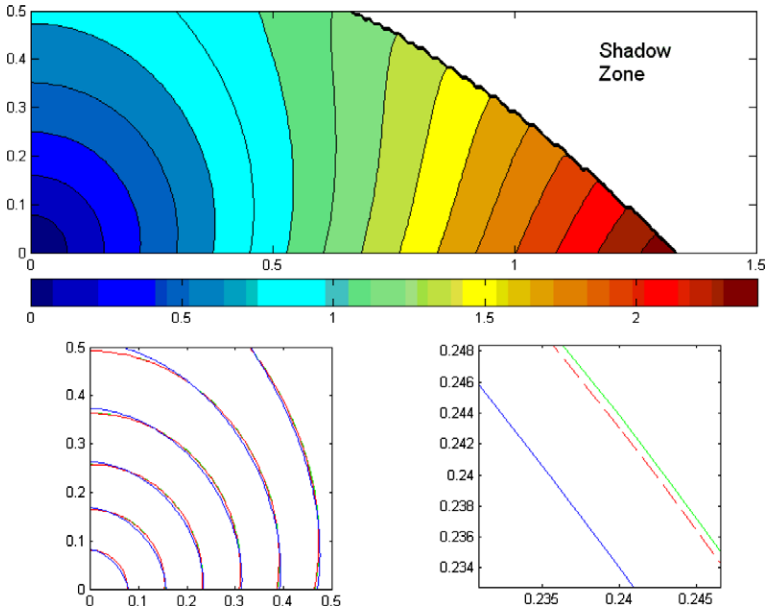

Fig. 4. Example 5. Top: analytical solution with a shadow zone. Bottom (left: contour plots, right: zoom in): green line – analytical solution, red dashed line – factored eikonal equation, blue dotted line – original eikonal equation. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 8
Example 6: numerical error.

Mesh	Error of T on $[0,0.5]\times[0,0.5]$	# Iterations
Factored eikonal equation		
80×80	0.0012770	3
160×160	0.0006377	3
320×320	0.0003186	3
640×640	0.0001592	3

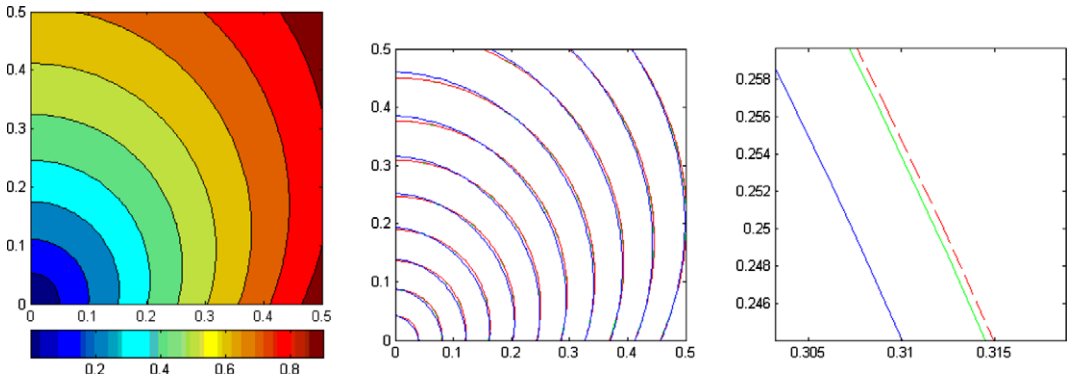

Fig. 5. Example 6. Left: analytical solution. Middle and right (contour plots and zoom in): green line – analytical solution, red dashed line – factored eikonal equation, blue dotted line – original eikonal equation. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

We test the model with parameters:

- plane-wave source on $z = 0$;
- computational domain (km): $[0, 1.5]\times[0, 0.5]$;

Table 9
Example 7: $T_0(x,z) = S_0 z$.

Mesh	Error of T on $[0.75, 1.5] \times [0, 0.5]$	# Iterations
Factored eikonal equation		
150×50	0.0003085	2
300×100	0.0001540	2
600×200	0.0000770	2
1200×400	0.0000385	2
Original eikonal equation		
150×50	0.0039623	2
300×100	0.0019791	2
600×200	0.0009891	2
1200×400	0.0004944	2

- $S_0 = 2$ s/km;
- $g_0 = (1, -3)$ s^2/km^3.

First we choose $T_0(x,z) = S_0 z$. Table 9 shows the results by both the factored and the original eikonal equation. Fig. 6 shows the plots of analytical and numerical solutions on a 150×50 mesh. It is easy to see that characteristics of T starting from the prescribed boundary cannot cover the whole computational domain, especially the up-left region (circled region) of the computational domain. Again we don't assume any knowledge of this scenario and use outflow boundary condition for our numerical computation. For this reason we record error only in the region $[0.75, 1.5] \times [0, 0.5]$ although the numerical computation is done in the whole domain. Since there is no point source singularity in this problem, we see both methods achieve first order accuracy. However, the factored eikonal equation still produces much better accuracy.

Next we choose $T_0(x,z) = \int_0^z S(z)dz$ with $S(z) = \int_{x_{\min}}^{x_{\max}} S(x,z)dx$, using the factored eikonal equation.

Table 10 shows the results by the factored eikonal equation. Fig. 7 shows the plots of numerical solutions on a 150×50 mesh.

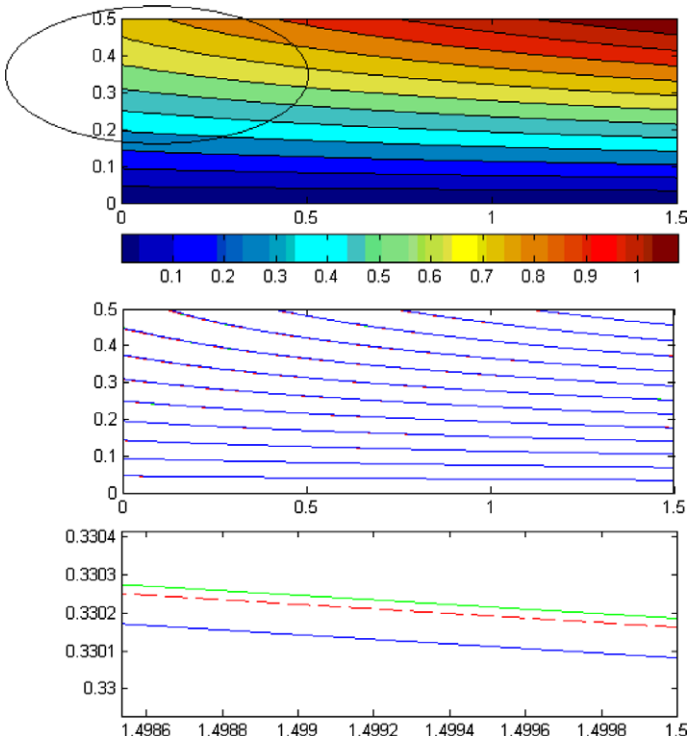

Fig. 6. Example 7 with $T_0(x,z) = S_0 z$. Top: analytical solution. Middle and bottom (contour plots and zoom in): green line – analytical solution, red dashed line – factored eikonal equation, blue dotted line – original eikonal equation. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 10
Example 7: $T_0(x,z) = \int_0^z S(z)dz$.

Mesh	Error of T on $[0.75, 1.5] \times [0, 0.5]$	# Iterations
Factored eikonal equation		
150×50	0.0002538	2
300×100	0.0001267	2
600×200	0.0000633	2
1200×400	0.0000316	2
Original eikonal equation		
150×50	0.0039623	2
300×100	0.0019791	2
600×200	0.0009891	2
1200×400	0.0004944	2

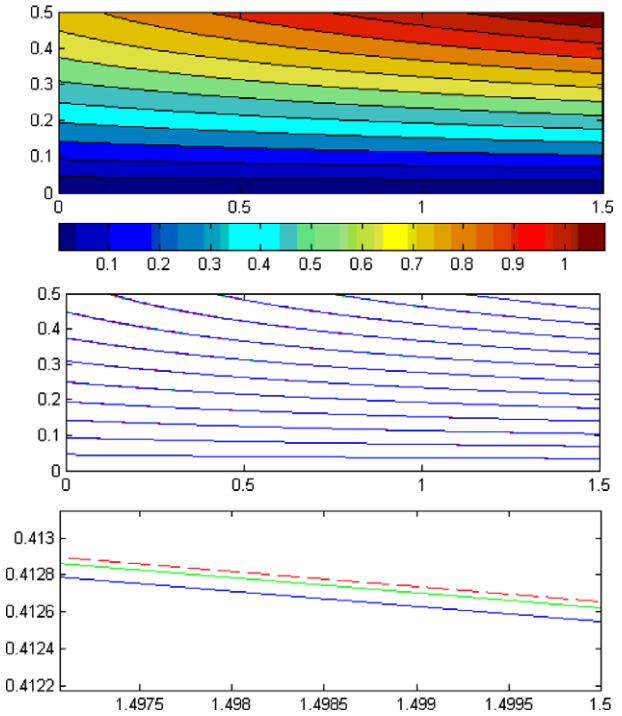

Fig. 7. Example 7 with $T_0(x,z) = \int_0^z S(z)dz$. Top: analytical solution. Middle and bottom (contour plots and zoom in): green line – analytical solution, red dashed line – factored eikonal equation, blue dotted line – original eikonal equation. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

4.1. Marmousi model

In this example, we apply our method to the famous Marmousi model [21] using both a point source and a plane-wave source. We choose T_0 as the distance function to the source point and $T_0(x,z) = 2z$ respectively for the factored eikonal equation.

Fig. 8 shows the results by both the factored and the original eikonal equation on a 2301×751 mesh (grid size 4 m).

The results are computed with C code programmed by Microsoft Visual C++ 6.0 on PC(Intel(R)Core(TM)2 CPU T5500@1.66GHZ). With *tolerance* $= 10^{-9}$, for the point source case, the original eikonal equation takes 28 iterations, CPU time: 54 s, the factored eikonal equation takes 18 iterations, CPU time: 74 s. For the plane-wave source case, the original eikonal equation takes 26 iterations, CPU time: 48 s, the factored eikonal equation takes 18 iterations, CPU time: 70 s. We see that the factored eikonal equation takes a little more time due to more complicated algebraic operations for the local solver at each grid point.

5. Conclusions

We have demonstrated that the fast sweeping method based on the factored eikonal equation improves the accuracy of the numerical solution to the original eikonal equation while maintaining the efficiency of the classical fast sweeping meth-

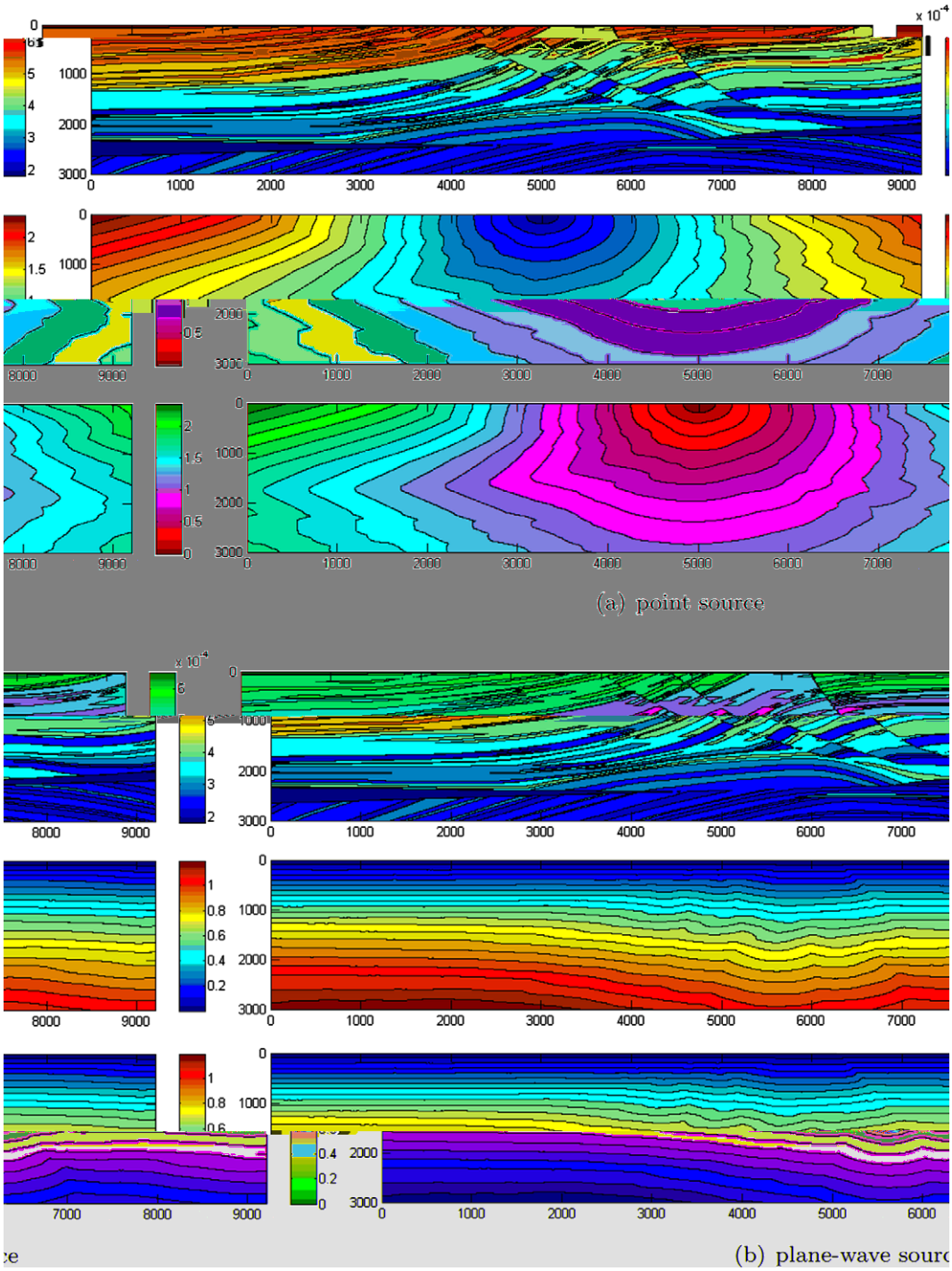

Fig. 8. Marmousi model with (a) point source and (b) plane-wave source. From top to bottom: slowness field, factored eikonal equation and original eikonal equation.

od. In the case of a point source problem, the singularity at the source can be removed effectively by the factored eikonal equation. For multi-point sources, the smoothness of T_0 away from the source points is essential to obtain better accuracy without losing efficiency. Both for individual point sources and simultaneous point sources, as well as plane-wave sources, we observe a significant improvement in accuracy.

Appendix A. Analytical traveltimes in the linear sloth[1] model

In this appendix, we provide analytical solutions of the eikonal Eq. (1) for the special case of a constant gradient of slowness squared

$$S^2(\mathbf{x}) = S_0^2 + 2\,\mathbf{g}_0 \cdot (\mathbf{x} - \mathbf{x}_0) \tag{17}$$

and a point-source or plane-wave boundary conditions. More general solutions of this kind are provided in [11].

We use the classic Hamilton–Jacobi theory and the method of characteristics [6]. The characteristics of Eq. (1) are described by the following system of ODEs

$$\frac{d\mathbf{x}}{d\sigma} = \mathbf{p}, \tag{18}$$

$$\frac{d\mathbf{p}}{d\sigma} = S(\mathbf{x})\,\nabla S, \tag{19}$$

$$\frac{dT}{d\sigma} = S^2(\mathbf{x}). \tag{20}$$

Solving the initial-value problem $\mathbf{x}(0) = \mathbf{x}_0, \mathbf{p}(0) = \mathbf{p}_0$ for Eq. (19), we obtain

$$\mathbf{p}(\sigma) = \mathbf{p}_0 + \mathbf{g}_0\,\sigma. \tag{21}$$

Solving Eq. (18) leads to

$$\mathbf{x}(\sigma) = \mathbf{x}_0 + \mathbf{p}_0\,\sigma + \mathbf{g}_0\,\frac{\sigma^2}{2}. \tag{22}$$

Rays, described by Eq. (22) are parabolic trajectories, analogous to trajectories of mechanical particles traveling with a constant acceleration. In the special case of a zero gradient, the trajectories are straight lines. Eq. (20) transforms to

$$\frac{dT}{d\sigma} = \mathbf{p}(\sigma) \cdot \mathbf{p}(\sigma) = S_0^2 + 2\,\mathbf{p}_0 \cdot \mathbf{g}_0\,\sigma + \mathbf{g}_0 \cdot \mathbf{g}_0\,\sigma^2. \tag{23}$$

with $\mathbf{p}_0 \cdot \mathbf{p}_0 = S_0^2$. Its solution takes the form

$$T(\sigma) = S_0^2\,\sigma + \mathbf{p}_0 \cdot \mathbf{g}_0\,\sigma^2 + \mathbf{g}_0 \cdot \mathbf{g}_0\,\frac{\sigma^3}{3}. \tag{24}$$

A.1. Point source

From Eq. (22), we derive

$$S^2(\mathbf{x}_1) - S^2(\mathbf{x}_0) = 2\,\mathbf{g}_0 \cdot (\mathbf{x}_1 - \mathbf{x}_0) = 2\,\mathbf{g}_0 \cdot \mathbf{p}_0\,\sigma + \mathbf{g}_0 \cdot \mathbf{g}_0\,\sigma^2 \tag{25}$$

or

$$\mathbf{g}_0 \cdot \mathbf{p}_0 = \frac{S^2(\mathbf{x}_1) - S^2(\mathbf{x}_0) - |\mathbf{g}_0|^2\,\sigma^2}{2\,\sigma}. \tag{26}$$

Additionally, Eq. (22) leads to

$$|\mathbf{x}_1 - \mathbf{x}_0|^2 = S^2(\mathbf{x}_0)\,\sigma^2 + \mathbf{g}_0 \cdot \mathbf{p}_0\,\sigma^3 + |\mathbf{g}_0|^2\,\frac{\sigma^4}{4} \tag{27}$$

or, substituting Eq. (26),

$$|\mathbf{x}_1 - \mathbf{x}_0|^2 = S^2(\mathbf{x}_0)\,\sigma^2 + \frac{S^2(\mathbf{x}_1) - S^2(\mathbf{x}_0) - |\mathbf{g}_0|^2\,\sigma^2}{2}\,\sigma^2 + |\mathbf{g}_0|^2\,\frac{\sigma^4}{4} = \frac{S^2(\mathbf{x}_1) + S^2(\mathbf{x}_0)}{2}\,\sigma^2 - |\mathbf{g}_0|^2\,\frac{\sigma^4}{4}. \tag{28}$$

The last equation is a quadratic equation for σ^2 and can be solved analytically, as follows:

$$\sigma = \frac{\sqrt{2\,(\bar{S}^2 \pm \sqrt{\bar{S}^4 - |\mathbf{g}_0|^2\,|\mathbf{x}_1 - \mathbf{x}_0|^2})}}{|\mathbf{g}_0|}, \tag{29}$$

where

$$\bar{S}(\mathbf{x}_0, \mathbf{x}_1) = \sqrt{\frac{S^2(\mathbf{x}_1) + S^2(\mathbf{x}_0)}{2}} = \sqrt{S^2(\mathbf{x}_0) + \mathbf{g}_0 \cdot (\mathbf{x}_1 - \mathbf{x}_0)}. \tag{30}$$

[1] Square of the slowness.

Finally, we can substitute Eqs. (26) and (30) into (24) to find the analytical two-point traveltime for the constant-slowness-gradient case

$$T(\mathbf{x}_0,\mathbf{x}_1) = \overline{S}^2\,\sigma - 3|\mathbf{g}_0|^2\,\frac{\sigma^3}{6}, \tag{31}$$

where σ can be one of the two solutions in Eq. (29). Eq. (31) coincides with (10) in the main text.

A.2. Plane-wave source

In the case of a plane-wave source $\mathbf{p}_0 = \{0, p_z\}$ at the surface $\mathbf{x}_0 = \{x_0, 0\}$, Eqs. (25), (22), and (24) transform to

$$S^2 = S_0^2 + 2\,g_z\,p_z\,\sigma + (g_x^2 + g_z^2)\,\sigma^2, \tag{32}$$

$$z = z_0 + p_z\,\sigma + g_z\,\frac{\sigma^2}{2}, \tag{33}$$

$$T = S_0^2\,\sigma + p_z\,g_z\,\sigma^2 + (g_x^2 + g_z^2)\,\frac{\sigma^3}{3}, \tag{34}$$

where g_x and g_z are components of $\mathbf{g}_0$. Simple calculation (34) $-0.5\sigma\cdot$ (32) gives

$$T(\mathbf{x}) = \overline{S}^2\sigma - (g_x^2 + g_z^2)\,\frac{\sigma^3}{6},$$

where $\overline{S} = \sqrt{\frac{S^2(x)+S_0^2}{2}}$. Notice that the formula is the same as point source. The only difference is that

$$\sigma = \frac{\sqrt{S^2 - S_0^2 - 2g_z(z-z_0)}}{|g_x|},$$

which is obtained by solving the quadratic equation of (32) $-2g_z\cdot$ (33).

Appendix B. Analytical traveltimes in the linear velocity model

To derive analytical traveltimes for the case of the slowness distribution of the form

$$\frac{1}{S(\mathbf{x})} = \frac{1}{S_0} + \mathbf{G}_0\cdot(\mathbf{x}-\mathbf{x}_0), \tag{35}$$

it is convenient to change the variable in the ODE system (18)–(20) to put it in the form

$$\frac{d\mathbf{x}}{d\xi} = \frac{\mathbf{p}}{S^3(\mathbf{x})}, \tag{36}$$

$$\frac{d\mathbf{p}}{d\xi} = \frac{\nabla S}{S^2(\mathbf{x})} = -\nabla\left(\frac{1}{S(\mathbf{x})}\right), \tag{37}$$

$$\frac{dT}{d\xi} = \frac{1}{S(\mathbf{x})}, \tag{38}$$

where $d\xi = S^3(\mathbf{x})\,d\sigma$. The solution of Eq. (37) for the initial condition $\mathbf{p}(0) = \mathbf{p}_0$ is

$$\mathbf{p}(\xi) = \mathbf{p}_0 - \mathbf{G}_0\,\xi. \tag{39}$$

Therefore,

$$S^2(\mathbf{x}) = \mathbf{p}\cdot\mathbf{p} = \mathbf{p}_0\cdot\mathbf{p}_0 - 2\,\mathbf{p}_0\cdot\mathbf{G}_0\,\xi + \mathbf{G}_0\cdot\mathbf{G}_0\,\xi^2. \tag{40}$$

Substituting (40) in Eq. (38), we can solve this ODE analytically to obtain

$$T(\xi) = \frac{1}{|\mathbf{G}_0|}\,\operatorname{arccosh}\left(1 + \frac{|\mathbf{G}_0|^2\,\xi^2}{S_0^2 + S_0\,S(\mathbf{x}) - \mathbf{p}_0\cdot\mathbf{G}_0\,\xi}\right). \tag{41}$$

Let $a = \mathbf{p}\cdot(\mathbf{x}-\mathbf{x}_0)$. The chain rule shows that

$$\frac{da}{d\xi} = \frac{d\mathbf{p}}{d\xi}\cdot(\mathbf{x}-\mathbf{x}_0) + \mathbf{p}\cdot\frac{d\mathbf{x}}{d\xi} = -\mathbf{G}_0\cdot(\mathbf{x}-\mathbf{x}_0) + \frac{\mathbf{p}\cdot\mathbf{p}}{S^3(\mathbf{x})} = -\mathbf{G}_0\cdot(\mathbf{x}-\mathbf{x}_0) + \frac{1}{S(\mathbf{x})} = \frac{1}{S_0}. \tag{42}$$

Therefore $a = \xi/S_0$.

Let $r = (\mathbf{x}-\mathbf{x}_0)\cdot(\mathbf{x}-\mathbf{x}_0)$. The chain rule shows that

$$\frac{dr}{d\xi} = 2\,\frac{d\mathbf{x}}{d\xi}\cdot(\mathbf{x}-\mathbf{x}_0) = 2\,\frac{\mathbf{p}\cdot(\mathbf{x}-\mathbf{x}_0)}{S^3(\mathbf{x})} = \frac{2\,\xi}{S^3(\mathbf{x})\,S_0}. \tag{43}$$

Substituting Eq. (40) into (43) and integrating, we obtain

$$r(\xi) = \frac{4\,\xi^2}{S(\mathbf{x})\,S_0\left(\left[S(\mathbf{x})+S_0\right]^2 - \mathbf{G}_0\cdot\mathbf{G}_0\,\xi^2\right)}. \tag{44}$$

Finally, expressing ξ from equation Eq. (44) and substituting into Eq. (41), we obtain

$$T(\mathbf{x}) = \frac{1}{|\mathbf{G}_0|}\,\operatorname{arccosh}\left(1+\frac{1}{2}\,S(\mathbf{x})\,S_0\,|\mathbf{G}_0|^2\,|\mathbf{x}-\mathbf{x}_0|^2\right), \tag{45}$$

which is equivalent to Eq. (15) in the main text. Eq. (45) is well-known [18], its derivation is included here for completeness.

References

[1] F. Audebert, J.P. Diet, P. Guillaume, I.F. Jones, X. Zhang, CRP-scans: 3-D preSDM velocity analysis via zero-offset tomographic inversion, in: 67th Ann. Internat. Mtg., Soc. of Expl. Geophys., 1997, pp. 1805–1808.
[2] F. Audebert, J.P. Diet, X. Zhang, CRP-scans from 3-D pre-stack depth migration: a powerful combination of CRP-gathers and velocity scans, in: 66th Ann. Internat. Mtg., Soc. of Expl. Geophys., 1996, pp. 515–518.
[3] D. Bevc, Imaging complex structures with semirecursive Kirchhoff migration, Geophysics 62 (02) (1997) 577–588.
[4] M. Boué, P. Dupuis, Markov chain approximations for deterministic control problems with affine dynamics and quadratic cost in the control, SIAM J. Numer. Anal. 36 (3) (1999) 667–695.
[5] V. Červený, Seismic Ray Theory, Cambridge University Press, 2001.
[6] R. Courant, D. Hilbert, Methods of Mathematical Physics, John Wiley & Sons, 1989.
[7] S. Geoltrain, J. Brac, Can we image complex structures with first-arrival traveltime?, Geophysics 58 (04) (1993) 564–575.
[8] C.Y. Kao, S. Osher, J. Qian, Lax–Friedrichs sweeping schemes for static Hamilton–Jacobi equations, J. Comput. Phys. 196 (2004) 367–391.
[9] C.Y. Kao, S. Osher, Y.H. Tsai, Fast sweeping method for static Hamilton–Jacobi equations, SIAM J. Numer. Anal. 42 (2005) 2612–2632.
[10] S. Kim, 3-D eikonal solvers: first arrival traveltimes, Geophysics 67 (4) (2002) 1225–1231.
[11] M. Körnig, Cell ray tracing for smooth, isotropic media: a new concept based on a generalized analytical solution, Geophys. J. Int. 123 (1995) 391–408.
[12] A. Pica, Fast and accurate finite-difference solutions of the 3d eikonal equation parametrized in celerity, in: 67th Ann. Internat. Mtg., Soc. of Expl. Geophys., 1997, pp. 1774–1777.
[13] J. Qian, W.W. Symes, An adaptive finite-difference method for traveltimes and amplitudes, Geophysics 67 (2002) 167–176.
[14] J. Qian, Y.-T. Zhang, H.-K. Zhao, A fast sweeping methods for static convex Hamitlon–Jacobi equations, J. Sci. Comput. 31 (1/2) (2007) 237–271.
[15] J. Qian, Y.-T. Zhang, H.-K. Zhao, Fast sweeping methods for eiknonal equations on triangulated meshes, SIAM J. Numer. Anal. 45 (2007) 83–107.
[16] J.A. Sethian, Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science, Cambridge University Press, 1999.
[17] J.A. Sethian, A.M. Popovici, 3-D traveltime computation using the fast marching method, Geophysics 64 (2) (1999) 516–523.
[18] M.M. Slotnick, Lessons in seismic computing, in: R.A. Geyer (Ed.), Soc. of Expl. Geophys., 1959.
[19] Y.-H.R. Tsai, L.-T. Cheng, S. Osher, H.-K. Zhao, Fast sweeping algorithms for a class of Hamilton–Jacobi equations, SIAM J. Numer. Anal. 41 (2003) 673–694.
[20] J. van Trier, W.W. Symes, Upwind finite-difference calculation of traveltimes, Geophysics 56 (06) (1991) 812–821.
[21] R. Versteeg, The Marmousi experience: velocity model determination on a synthetic complex data set, The Leading Edge 13 (09) (1994) 927–936.
[22] J.E. Vidale, Finite-difference calculation of traveltimes in three dimensions, Geophysics 55 (05) (1990) 521–526.
[23] L. Zhang, J.W. Rector, G.M. Hoversten, Eikonal solver in the celerity domain, Geophys. J. Int. 162 (2005) 1–8.
[24] Y.-T. Zhang, H.-K. Zhao, S. Chen, Fixed-point iterative sweeping methods for static Hamilton–Jacobi equations, Methods Appl. Anal. 13 (2006) 299–320.
[25] Y.-T. Zhang, H.-K. Zhao, J. Qian, High order fast sweeping methods for static Hamilton–Jacobi equations, J. Sci. Comput. 29 (2006) 25–56.
[26] H.-K. Zhao, A fast sweeping method for eikonal equations, Math. Comput. 74 (2005) 603–627.
[27] H.-K. Zhao, S. Osher, B. Merriman, M. Kang, Implicit and non-parametric shape reconstruction from unorganized points using variational level set method, Comput. Vis. Image Understand. 80 (2000) 295–319.