Journal of Scientific Computing (2019) 79:1456–1476
https://doi.org/10.1007/s10915-019-00905-6

Corner Cases, Singularities, and Dynamic Factoring

Dongping Qi[1] · **Alexander Vladimirsky**[2]

Received: 12 January 2018 / Revised: 19 December 2018 / Accepted: 3 January 2019 /
Published online: 16 January 2019
© Springer Science+Business Media, LLC, part of Springer Nature 2019

Abstract
In Eikonal equations, rarefaction is a common phenomenon known to degrade the rate of convergence of numerical methods. The "factoring" approach alleviates this difficulty by deriving a PDE for a new (locally smooth) variable while capturing the rarefaction-related singularity in a known (non-smooth) "factor". Previously this technique was successfully used to address rarefaction fans arising at point sources. In this paper we show how similar ideas can be used to factor the 2D rarefactions arising due to nonsmoothness of domain boundaries or discontinuities in PDE coefficients. Locations and orientations of such rarefaction fans are not known in advance and we construct a "just-in-time factoring" method that identifies them dynamically. The resulting algorithm is a generalization of the Fast Marching Method originally introduced for the regular (unfactored) Eikonal equations. We show that our approach restores the first-order convergence and illustrate it using a range of maze navigation examples with non-permeable and "slowly permeable" obstacles.

Keywords Eikonal · Rarefaction fans · Factoring · Fast Marching · Robotic navigation

Mathematics Subject Classification 49L20 · 49L25 · 49N90 · 65N12 · 65N22

1 Introduction

The Eikonal equation arises in many application domains including the geometric optics, optimal control theory, and robotic navigation. It is a first-order non-linear partial differential equation of the form

The second author's work is supported in part by the National Science Foundation Grant DMS-1738010 and the Simons Foundation Fellowship.

✉ Alexander Vladimirsky
vladimirsky@cornell.edu

Dongping Qi
dq48@cornell.edu

[1] Center for Applied Mathematics, Cornell University, Ithaca, NY 14853, USA

[2] Department of Mathematics and Center for Applied Mathematics, Cornell University, Ithaca, NY 14853, USA

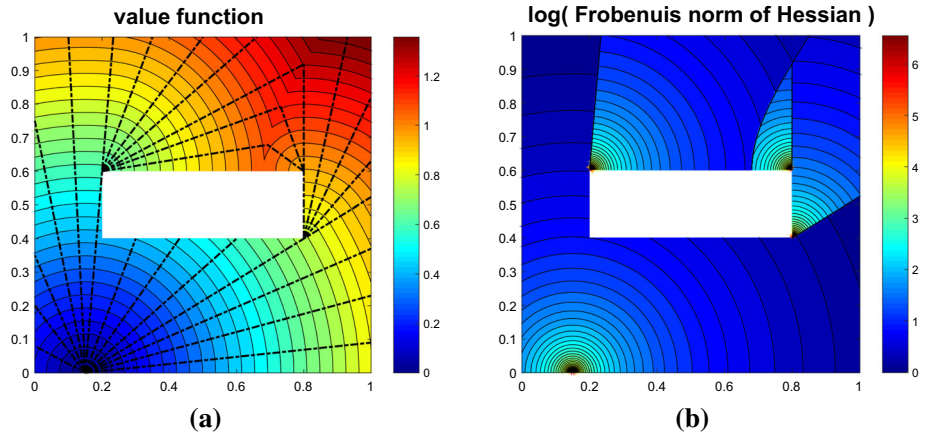


Fig. 1 A simple example with $F = 1$ and the exit set Q consisting of a single "point source" $\mathbf{x}_0 = (0.15, 0)$.(Left) level sets of $u(\mathbf{x})$, the distance to $\mathbf{x}_0$ on the domain with an obstacle. Dashed lines show the optimal/shortest paths, some of which follow a part of the obstacle boundary. The "shockline" (where ∇u is undefined) consists of all points starting from which the shortest path is not unique. The point source and 3 out of 4 obstacle corners generate "rarefaction fans" of characteristics. (Right) The level curves of the log-Frobenius-norm of Hessian, illustrating the blow up of the second derivatives of u due to rarefaction fans (Color figure online)

$$\begin{cases} |\nabla u(\mathbf{x})|\, F(\mathbf{x}) = 1, \\ u(\mathbf{x}) = 0, \qquad\qquad \mathbf{x} \in Q. \end{cases} \tag{1}$$

In control-theoretic context, the function $u(\mathbf{x})$ can be interpreted as the minimum time needed to reach the exit set Q starting from $\mathbf{x}$, with F specifying the speed of motion. The characteristics of this PDE coincide with the gradient lines of u and yield min-time-optimal trajectories to Q. This equation typically does not have a classical solution, making it necessary to select a *weak* Lipschitz-continuous solution that is physically meaningful. The theory of *viscosity solutions* [7] accomplishes this and guarantees the existence and uniqueness for a very broad set of problems. Viscosity solutions exhibit both shocklines (where characteristics run into each other) and rarefaction fans (where many characteristics emanate from the same point). Typically, the former receive most of the attention in numerical treatment since they lead to a discontinuity in ∇u. But in this paper we focus on the latter, which result in a blow up in second derivatives of u on parts of the domain where ∇u remains continuous and bounded. (See Fig. 1.)

Since the local truncation error of numerical discretizations is proportional to higher derivatives, it is not surprising that rarefaction fans degrade the rate of convergence of standard numerical methods. Special "factoring techniques" have been introduced to alleviate this problem for rarefaction fans arising from point source boundary conditions [10,15–17,20,27]. The idea is to represent u as a product [10] or a sum [16] of two functions: one capturing the right type of singularity at the point source and another with bounded derivatives except at shocklines. The former is known based on F restricted to Q; the latter is a priori unknown and recovered by solving a modified PDE by a version of one of the efficient methods (e.g., [21,29] or [32]) originally developed for an "un-factored" Eikonal equation (1). We review this prior work in Sect. 2 and show that, despite its better rate of convergence, factoring can also have detrimental effects on parts of the domain far from the point source. We also consider a *localized* version of factoring, which often improves the accuracy and is more suitable for problems with multiple point sources.

However, point sources are not the only origin of rarefaction fans. They can also arise due to non-smoothness of the boundary; e.g., at some of the "obstacle corners" in 2D maze navigation problems (see Fig. 1). The degradation of accuracy leads to numerical artifacts in computed trajectories passing near such corners. Unlike in the point source case, these rarefaction fans are not radially symmetric; moreover, their locations and geometry have to be determined dynamically. We handle this by developing a "just-in-time localized factoring" method and verifying its rate of convergence numerically (Sect. 3). Even more complicated fans can arise at corners of "slowly permeable obstacles" (i.e., problems with piecewise-continuous speed function F). The extension of our dynamic factoring to this case is covered in Sect. 4.

Throughout the paper, we present our approach using a Cartesian grid discretization of the Eikonal equation with grid aligned obstacles (and discontinuities of F). However, the ideas presented here have a broader applicability. Arbitrary polygonal obstacles can be treated in exactly the same way if the discretization is posed on an obstacle-fitted triangulated mesh. Dynamic factoring for more general Hamilton–Jacobi–Bellman PDEs would work very similarly, though the factoring will need to account for the anisotropy in rarefaction fans. We conclude by discussing these and other future extensions as well as the limitations of our approach in Sect. 5.

2 Point-Sources and Factoring

We begin by examining the classical Eikonal Eq. (1) in 2D with a "single point-source" boundary condition: $Q = \{\mathbf{x}_0\}$ and $u(\mathbf{x}_0) = 0$. Throughout this paper, we will assume that the controlled process is restricted to a closed set $\Omega \subset R^2$. I.e., $u(\mathbf{x})$ is the minimal-time from $\mathbf{x} \in \Omega$ to $Q \subset \Omega$ without leaving Ω though possibly traveling along parts of $\partial\Omega$; see the trajectories traveling along the obstacle boundary in Fig. 1a. This makes u an Ω-*constrained viscosity solution* of (1) (see Chapter 4.5 in [2]), but for the purposes of numerical implementation we can simply impose the boundary condition $u = +\infty$ on $R^2 \backslash \Omega$.

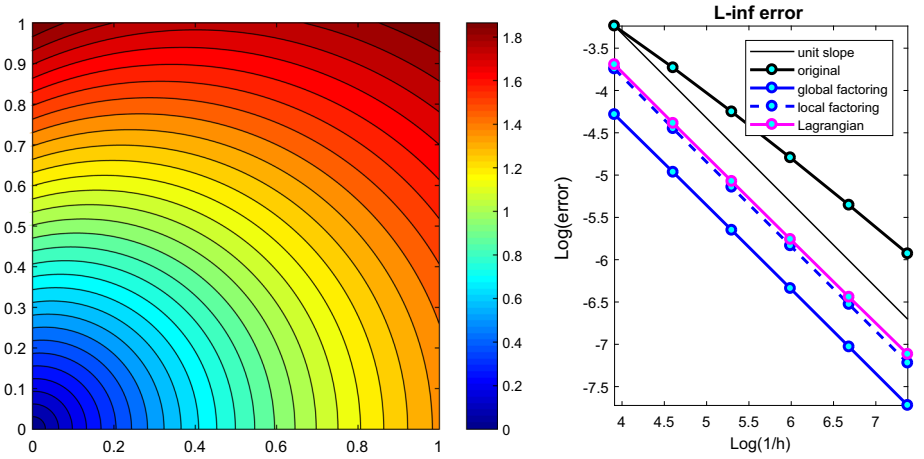
A common approach for discretizing (1) on a uniform Cartesian grid is to use upwind finite differences:

$$\max\left\{D_{i,j}^{-x}u, -D_{i,j}^{+x}u, 0\right\}^2 + \max\left\{D_{i,j}^{-y}u, -D_{i,j}^{+y}u, 0\right\}^2 = \frac{1}{F_{i,j}^2}, \tag{2}$$

using the standard finite difference notation

$$\begin{aligned} D_{i,j}^{-x}u &= \frac{u_{i,j} - u_{i-1,j}}{h}, \qquad D_{i,j}^{+x}u = \frac{u_{i+1,j} - u_{i,j}}{h}, \\ D_{i,j}^{-y}u &= \frac{u_{i,j} - u_{i,j-1}}{h}, \qquad D_{i,j}^{+y}u = \frac{u_{i,j+1} - u_{i,j}}{h}. \end{aligned} \tag{3}$$

The discretized system of Eq. (2) is coupled and non-linear. We postpone the discussion of fast algorithms used to solve it until Sect. 2.1 and for now focus on the rate of convergence of the numerical solution to the viscosity solution of PDE (1). Since this discretization is globally first-order accurate, the local truncation error is proportional to the second derivatives of u, which blow up as we approach $\mathbf{x}_0$. Because of this blow up, even if we assume that local truncation errors accumulate linearly, the global error would decreases as $O(h \log \frac{1}{h})$ instead of the expected $O(h)$.



(a) level sets of $u(\mathbf{x})$ **(b)** L^∞ error

Fig. 2 (Left) level curves of the value function computed from the formula (4) with $\mathbf{x}_0 = (0, 0)$, $s_0 = 2$, $\mathbf{v} = (0.5, 0)$.(Right) the corresponding convergence plot (based on the L^∞ error) for several discretization approaches, with the thin black line of slope (-1) included to aid the visual comparison. For this example, the convergence plot based on the L^1 error is very similar and thus omitted (Color figure online)

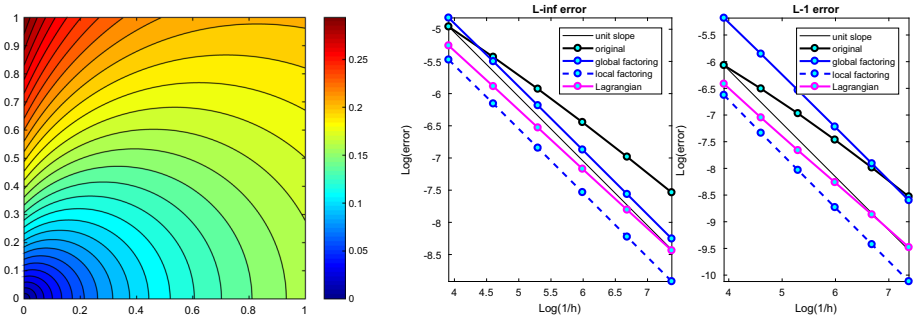
To illustrate the convergence rate, we will consider a simple example with a known analytic solution [16] on a square domain[1]:

$$F(\mathbf{x}) = \frac{1}{s_0} + \mathbf{v} \cdot (\mathbf{x} - \mathbf{x}_0) \quad \Longrightarrow \quad u(\mathbf{x}) = \frac{1}{|\mathbf{v}|} \operatorname{acosh}\left(1 + \frac{1}{2} s_0 |\mathbf{v}|^2 \frac{|\mathbf{x} - \mathbf{x}_0|^2}{F(\mathbf{x})}\right). \tag{4}$$

Figure 2 shows the solution and the convergence plots for the parameter values $\mathbf{x}_0 = (0, 0)$, $s_0 = 2$, $\mathbf{v} = (0.5, 0)$. The thick black line in Fig. 2b corresponds to solving (2) on Ω and clearly shows the sublinear convergence. (Throughout this paper, all logarithms of errors and grid sizes are reported and plotted in base e.)

To alleviate this issue, one could simply "enlarge the exit set" by choosing some (h-independent) constant radius $r > 0$, initializing $u = 0$ on the disk $B = B_r(\mathbf{x}_0) = \{\mathbf{x} \in \Omega \mid |\mathbf{x} - \mathbf{x}_0| \le r\}$, and solving (2) on the rest of the grid. This avoids the rarefaction fan (since only one characteristic stems from each point on ∂B), but introduces a $O(r)$ difference compared to the solution of the original point-source-based problem. One can also use a better approximation of u on B; e.g., using $T(\mathbf{x}) = \frac{|\mathbf{x}-\mathbf{x}_0|}{F(\mathbf{x}_0)}$, we already reduce this additional error to $O(r^2)$. The latter approach is based on assuming that F (rather than u) is constant on B, in which case the characteristics are straight lines. Of course, one could also take a truly Lagrangian approach, employing ray-tracing to compute a highly accurate value of u at all gridpoints on B, but this becomes increasingly expensive as $h \to 0$. Instead, we evaluate the feasibility of an "approximate Lagrangian" initialization technique, where the characteristics are still assumed to be straight on B, but $u(x)$ is approximated more accurately by integrating $1/F$ on the line segment from $\mathbf{x}_0$ to $\mathbf{x}$. In all the figures of this section, we slightly abuse the

[1] This formula for $u(\mathbf{x})$ is derived on the unbounded domain $\Omega_\infty = \{\mathbf{x} \in R^2 \mid F(\mathbf{x}) > 0\}$ but remains valid on $\Omega = [0, 1] \times [0, 1]$ as long as Ω_∞-optimal trajectories from every $\mathbf{x} \in \Omega$ to $\mathbf{x}_0$ stay entirely inside Ω, which is the case for all examples considered in this section. The linearity of $F(\mathbf{x})$ can be used to show that all optimal paths are circular arcs, whose radii are monotone decreasing in $|\mathbf{v}|$. (When $\mathbf{v} = 0$, these radii are infinite; i.e., all optimal paths are straight lines and $u(\mathbf{x}) = s_0|\mathbf{x} - \mathbf{x}_0|$.)



(a) level sets **(b)** L^∞ error **(c)** L^1 error

Fig. 3 (Left) level curves of the value function $u(\mathbf{x})$ computed from the formula (4) with $\mathbf{x}_0 = (0, 0)$, $s_0 = 0.5$, $\mathbf{v} = (12, 0)$.(Center and right) the corresponding convergence plot for several discretization approaches (based on the L^∞ & L^1 errors respectively) (Color figure online)

notation and call this approach Lagrangian, reporting the results for $r = 0.1$ in all the figures of this section. Figures 3 and 4 show that, for general F, the error due to this "characteristics are straight" assumption prevents the overall first order convergence on the entire grid.

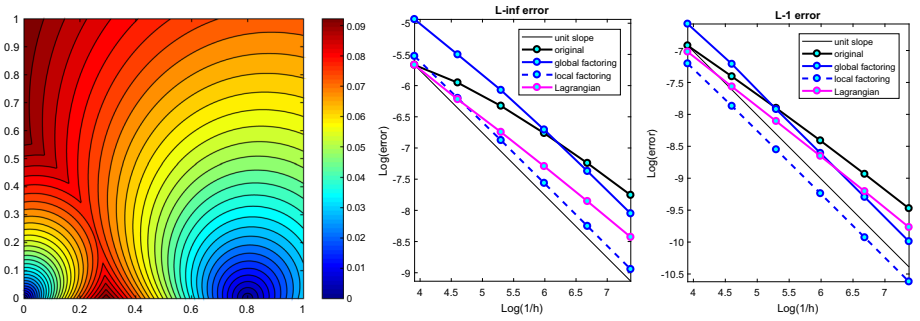
A factored Eikonal equation was proposed in [10] as a method for dealing with point-source rarefaction fans without introducing any special approximations on B and recovering the first order of accuracy on the entire domain. The main idea is to split the original value function $u(\mathbf{x})$ into two functions: one of them (T, defined above) encodes the right type of singularity at the point source, while the other ($\tau(\mathbf{x})$, our new unknown) will be smooth near $\mathbf{x}_0$. In addition to point-sources, [10] also used factoring to treat "plane-wave sources" (i.e., constant Dirichlet boundary conditions specified on a straight line in 2D). In the latter case, there is no singularities in the solutions at the boundary (so, the numerics for the original/unfactored version is still first-order accurate), but a factored version still yields lower errors in many examples.

The original factoring in [10] used an ansatz $u(\mathbf{x}) = T(\mathbf{x})\tau(\mathbf{x})$ to derive a new factored PDE for τ, which was then solved with the boundary condition $\tau(\mathbf{x}_0) = 1$. In this paper we rely on a slightly simpler additive splitting[2] introduced in [16] : assuming that $u(\mathbf{x}) = T(\mathbf{x}) + \tau(\mathbf{x})$, we find τ by solving

$$|\nabla T(\mathbf{x}) + \nabla\tau(\mathbf{x})|\, F(\mathbf{x}) \;=\; 1, \tag{5}$$

with the boundary condition $\tau(\mathbf{x}_0) = 0$. Similarly to (1) this can be discretized using upwind finite differences and then solved on Ω (see Sect. 2.2). In all our convergence figures we refer to this approach as the "global factoring" (shown by a solid blue line). In Figs. 2, 3 and 4 it is clear that global factoring has a clean linear convergence. Moreover, in Fig. 2 it exhibits smaller errors than either the original/"unfactored" discretization or even the Lagrangian-initialized version regardless of the grid resolution. However, as examples in Figs. 3 and 4 show, on relatively coarse grids it can be actually less accurate than the unfactored discretization—particularly when the characteristics are far from straight lines. This phenomenon has not been examined in prior literature, but it is hardly surprising: farther from the point source, u and T can be quite different, and if the second derivatives of u are smaller, this will result in larger local truncation errors when computing τ.

[2] Throughout the paper we refer to this approach as "additive factoring" to stay consistent with the terminology used in prior literature.



(a) level sets **(b)** L^∞ error **(c)** L^1 error

Fig. 4 (Left) level curves of the value function $u(\mathbf{x})$ for $s_0 = 0.5$, $\mathbf{v} = (5, 20)$ and two point sources at $\mathbf{x}_0 = (0, 0)$ and $\mathbf{x}_1 = (0.8, 0)$.(Center and right) the corresponding convergence plot for several discretization approaches (based on the L^∞ & L^1 errors respectively) (Color figure online)

We further examine a "localized factoring" version of this idea. For small r, this could be posed as a 2-stage process: first solve (5) on B and then switch to solving (1) on $\Omega \backslash B$. However, we have found that another interpretation is more suitable, particularly when characteristics are highly curved: solve (5) on the entire Ω but defining $T(\mathbf{x}) = 0$ on $\Omega \backslash B$. We note that several recent papers have already considered such hybrid/localized factoring motivated by decreasing the computational cost (since $\nabla T = 0$ on most of the domain) [20] and by the need for additional properties of T when pursuing a higher-order accurate discretization [17]. Here, however, we show that the localized factoring (shown by a blue dashed lines on convergence plots) has some accuracy advantages even with the first-order upwind discretization. In our first example (Fig. 2), $u \approx T$ remains true on the whole Ω and characteristics are fairly close to straight lines; so, the global factoring is more accurate. But in Fig. 3 this is no longer the case, and the localized factoring is clearly preferable.

Localized factoring is also often advantageous (and more natural) when dealing with multiple point sources. Consider, for example, the speed function specified in (4) with $s_0 = 0.5$, $\mathbf{v} = (5, 20)$ and two point sources: $\mathbf{x}_0 = (0, 0)$ and $\mathbf{x}_1 = (0.8, 0)$. (See Fig. 4.) If $Q = \{\mathbf{x}_0\}$ or $Q = \{\mathbf{x}_1\}$, the respective value functions u_0 and u_1 are specified by formula (4). For the two point-sources case, $Q = \{\mathbf{x}_0, \mathbf{x}_1\}$, the value function $u(\mathbf{x}) = \min(u_0(\mathbf{x}), u_1(\mathbf{x}))$ is no longer smooth: ∇u is undefined at the points from which the optimal paths to $\mathbf{x}_0$ and $\mathbf{x}_1$ are equally good. Since the characteristics run into the shockline rather than originate from it, this does not degrade the rate of convergence, but the rarefaction fans remain a challenge. In global factoring, there is a number of choices to capture in T the singularities at both point sources. We use

$$T_0(\mathbf{x}) = \frac{|\mathbf{x} - \mathbf{x}_0|}{F(\mathbf{x}_0)}, \qquad T_1(\mathbf{x}) = \frac{|\mathbf{x} - \mathbf{x}_1|}{F(\mathbf{x}_1)}, \qquad T(\mathbf{x}) = \min\big(T_0(\mathbf{x}), T_1(\mathbf{x})\big),$$

but the convergence results based on $T = T_0 + T_1$ are qualitatively similar (though the L^1 error becomes significantly larger). For the localized factoring, we simply set $T(\mathbf{x}) = T_i(\mathbf{x})$ on $B_r(x_i)$ and $T(\mathbf{x}) = 0$ everywhere else. Figure 4 shows that both versions of factoring exhibit linear convergence, but the localized factoring yields significantly smaller errors both in L^∞ and L^1 norms.

2.1 Fast Methods for an (Unfactored) Eikonal

Since an Eikonal equation arises in so many applications, there has been a number of fast numerical methods developed for it in the last 20 years. Most of these methods mirror the logic of classical label-setting and label-correcting algorithms for finding shortest paths on graphs [4]. In this discrete setting, an equation for the min-time-to-exit U_i starting from each node $\mathbf{x}_i$ is posed using the min-time-to-exit values (U_j's) at its neighboring nodes ($\mathbf{x}_j$'s). Dijkstra's method [9] is perhaps the most famous of label-setting algorithms for graphs with positive edge-weights. It is based on the idea of *monotone causality*: an optimal path from x_i starts with a transition to some neighboring node $\mathbf{x}_{j^*}$ and, since all edge weights are positive, this implies $U_i > U_{j^*}$. Thus, even if we don't know $\mathbf{x}_{j^*}$ a priori, U_i can be still computed based on the set of all *smaller* neighboring values. Dijkstra's method exploits this observation to dynamically uncover the correct node-ordering, decoupling the system of equations for U_i's. The nodes are split into three sets: Accepted (with permanently fixed U values), Considered (with tentatively computed U values) and Far (with U values assumed to be $+\infty$). At each stage of the algorithm, the smallest of Considered U values is declared Accepted and the values at its immediate not-yet-Accepted neighbors are recomputed. On a graph with M nodes and bounded node connectivity, this yields the overall complexity of $O(M \log M)$ due to the use of a heap to sort the Considered values. An additional useful feature of this approach is that the U values are fixed/accepted after a small number of updates on an (incrementally growing) part of the graph. Many label-correcting algorithms aim to mimic this property, but without using expensive data structures to sort any of the values. Unlike in label-setting algorithms, they cannot provide an a priori upper bound on the number of times each U_i might be updated. As a result, their worst-case complexity is $O(M^2)$, but on many types of graphs their average-case behavior has been observed to be at least as good as that of label-setting techniques [4].

For Eikonal PDEs discretized on Cartesian grids, the Dijkstra-like approach was first introduced in Tsitsiklis's Algorithm [29] and Sethian's Fast Marching Method (FMM) [21]. The latter was further extended to simplicial mesh discretizations in R^n and on manifolds [12, 24], to higher-order accurate numerical schemes [22,24], and to Hamilton–Jacobi–Bellman equations, which can be viewed as anisotropic generalizations of the Eikonal [1,8,18,25,26]. The major difficulty in applying Dijkstra-like ideas in continuous setting is that, unlike in problems on graphs, a value of u at a gridpoint $\mathbf{x}_{i,j}$ will depend on several neighboring values used to approximate $\nabla u(\mathbf{x}_{i,j})$. To obtain the same monotone causality, this $u(\mathbf{x}_{i,j})$ has to be always larger than *all* of such contributing u values adjacent to $\mathbf{x}_{i,j}$. This property is enjoyed by only some discretiations of (1), including the first-order accurate upwind scheme (2): a direct verification shows that u_x and u_y are never approximated using any neighbors larger that $u_{i,j}$.

Another popular class of efficient Eikonal solvers is Fast Sweeping Methods (FSM) [32], which solve the system (2) by Gauss–Seidel iterations, but changing the order in which the gridpoints are updated from iteration to iteration. When the direction of the current sweep is aligned with the general direction of characteristics, many gridpoints will receive correct values in a single "sweep". If marching-type methods attempt to uncover the correct gridpoint-ordering dynamically, in FSM the idea is to alternate through a number of geometrically motivated orderings. (In 2D problems: from northeast, from northwest, from southwest, and from southeast.) The resulting Eikonal solvers have $O(M)$ complexity, but with a hidden constant factor, which depends on F and the grid orientation, and cannot be bound a priori. These techniques have also been extended to anisotropic (and even non-convex) Hamilton–Jacobi equations [11,28], as well as higher-order finite-difference (e.g., [31]) and

discontinuous Galerkin (e.g., [30]) discretizations. Hybrid two-scale methods, combining the best features of marching and sweeping, were more recently introduced in [5,6]. We also refer to [5] for a comprehensive review of other fast solvers inspired by label-correcting algorithms.

2.2 Modified Fast Marching for Factored Eikonal

We start by simplifying the original upwind discretization scheme (2) for the unfactored Eikonal. Focusing on a gridpoint $\mathbf{x}_{i,j}$ we define its smallest horizontal and vertical neighboring values: $u_H = \min(u_{i-1,j}, u_{i+1,j})$ and $u_V = \min(u_{i,j-1}, u_{i,j+1})$. If both of these values are needed to compute $u_{i,j}$, then (2) becomes equivalent to a quadratic equation $(u_{i,j} - u_H)^2 + (u_{i,j} - u_V)^2 = h^2/F_{i,j}^2$. We are interested in its smallest real root satisfying an *upwinding condition* $u_{i,j} \geq \max(u_H, u_V)$. If there is no root satisfying it, then $u_{i,j}$ should instead be computed from a *one-sided-update* $u_{i,j} = \min(u_H, u_V) + h/F_{i,j}$, which corresponds to the case where either $\max\{D_{i,j}^{-x}u, -D_{i,j}^{+x}u, 0\}$ or $\max\{D_{i,j}^{-y}u, -D_{i,j}^{+y}u, 0\}$ evaluates to zero. This procedure is monotone causal by construction and its equivalence to (2) was demonstrated in [21], making a Dijkstra-like computational approach suitable.

Recalling the "additive factoring" ansatz $u(\mathbf{x}) = T(\mathbf{x}) + \tau(\mathbf{x})$, we now define the upwind vertical and horizontal neighboring values for the new unknown $\tau_{i,j}$ but basing the comparison on u rather than on τ itself and using the flags $k_H, k_V \in \{-1, 1\}$ to identify the selected neighbors. More specifically,

$$\begin{cases} \tau_H = \tau_{i-1,j} \text{ and } k_H = 1, & \text{if } (T_{i-1,j} + \tau_{i-1,j}) < (T_{i+1,j} + \tau_{i+1,j}); \\ \tau_H = \tau_{i+1,j} \text{ and } k_H = -1, & \text{otherwise;} \end{cases}$$

with (τ_V, k_V) similarly defined based on the vertical neighbors. Since the partial derivatives of T are known, the corresponding quadratic equation is

$$\left(k_H \frac{\partial T_{i,j}}{\partial x} + \frac{\tau_{i,j} - \tau_H}{h}\right)^2 + \left(k_V \frac{\partial T_{i,j}}{\partial y} + \frac{\tau_{i,j} - \tau_V}{h}\right)^2 = \frac{1}{F_{i,j}^2}. \tag{6}$$

We are interested in its smallest real root satisfying a similarly modified *upwinding condition*

$$T_{i,j} + \tau_{i,j} \geq \max\left(\min_{k=\pm 1}\left\{T_{i+k,j} + \tau_{i+k,j}\right\}, \min_{k=\pm 1}\left\{T_{i,j+k} + \tau_{i,j+k}\right\}\right). \tag{7}$$

If there is no such root, then $\tau_{i,j}$ should instead be computed from a *one-sided-update* as the smaller of the two values corresponding to

$$k_H \frac{\partial T_{i,j}}{\partial x} + \frac{\tau_{i,j} - \tau_H}{h} = \frac{1}{F_{i,j}}, \qquad \text{and} \qquad k_V \frac{\partial T_{i,j}}{\partial y} + \frac{\tau_{i,j} - \tau_V}{h} = \frac{1}{F_{i,j}}.$$

In other words,

$$\tau_{i,j} = \min\left\{\tau_H - hk_H \frac{\partial T_{i,j}}{\partial x}, \tau_V - hk_V \frac{\partial T_{i,j}}{\partial y}\right\} + \frac{h}{F_{i,j}}. \tag{8}$$

The above is a full recipe for computing $\tau_{i,j}$ if all of the neighboring grid values are already known. But since τ is a priori known only on Q, this yields a large coupled system of discretized equations (one per each gridpoint in $\Omega \setminus Q$). This system was first treated iteratively via Fast Sweeping [16], but the monotone causality makes a Dijkstra-like approach applicable as well. As in the original Fast Marching, the Considered gridpoints are sorted using a binary

heap with a $O(M \log M)$ computational complexity; however, the sorting criterion is based on u rather than on τ values. The resulting method is summarized in Algorithm 1. It is very similar to a modified FMM recently introduced for the case of "multiplicative factoring" in [27].

Algorithm 1: Modified Fast Marching Method

Input: source point $\mathbf{x}_0$, speed function $F(\mathbf{x})$

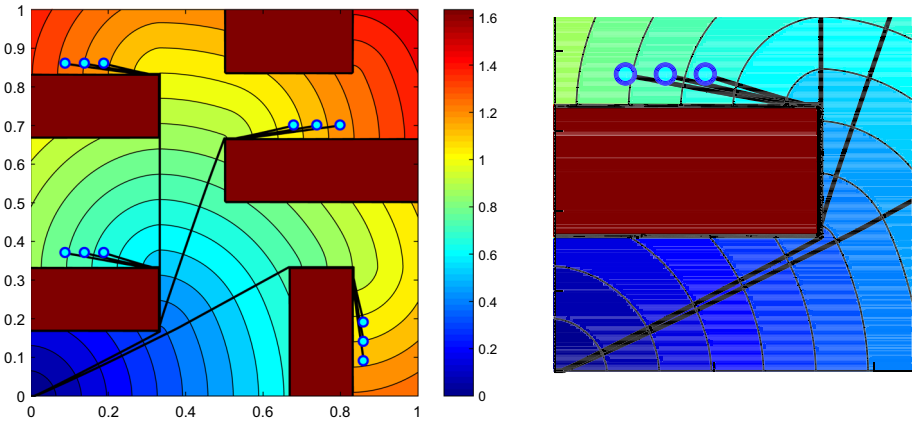
Initialize $\tau(\mathbf{x}_0) := 0$ and $\tau(\mathbf{x}) := +\infty$ for all gridpoints $\mathbf{x} \neq \mathbf{x}_0$;
Initialize Considered $:= \{\mathbf{x}_0\}$ and Accepted $:= \emptyset$;

while *Considered* $\neq \emptyset$ **do**
  Find a Considered gridpoint $\hat{\mathbf{x}}$ whose $(T + \tau)$ value is the smallest;
  Move $\hat{\mathbf{x}}$ to Accepted list;

  for *all not-yet-Accepted neighbors* $\boldsymbol{x}_{i,j}$ *of* $\hat{\boldsymbol{x}}$ **do**
    Find τ_H, τ_V and solve the quadratic equation (6) for $\tau_{i,j}^{new}$;
    if $\tau_{i,j}^{new}$ *does not satisfy the upwinding condition* (7) **then**
      Use a one-sided update formula (8) to (re-)compute $\tau_{i,j}^{new}$;
    end
    if $\tau_{i,j}^{new} < \tau_{i,j}$ **then**
      Set $\tau_{i,j} := \tau_{i,j}^{new}$;
    end
  end
end

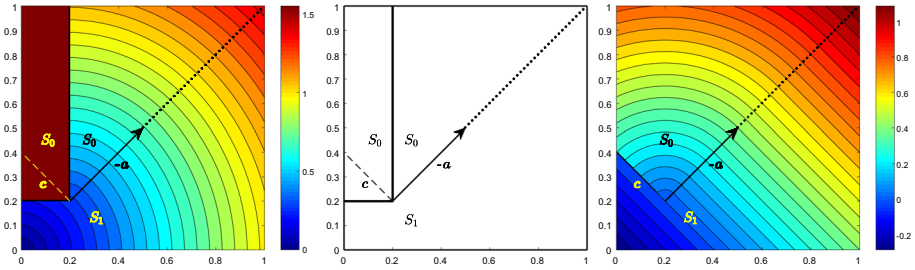
3 Rarefaction Fans at Obstacle Corners

Even though all prior work on factored Eikonal equation was focused on isolated point sources, there are other well-known situations where rarefaction fans can arise. As a simple example in Fig. 1 shows, they can easily develop at the corners of obstacles (which are viewed as a part of $R^2 \backslash \Omega$) or, more generally at any points on $\partial\Omega$ where the boundary is non-smooth and the interior angle is larger than π. These non-point-source rarefaction fans result in a similar degradation of convergence rate for standard numerical methods and also lead to unpleasant artifacts in optimal trajectory approximations obtained by following $(-\nabla u)$ to the target set Q. Figure 5 shows several such trajectories in a "maze navigation" problem. All of these trajectories should be piecewise-linear, with their directions only changing at obstacle corners. A zoomed version in Fig. 5b clearly shows that they often approach an obstacle too early, following its boundary to the corner and yielding longer paths. Similar artifacts are common in determining parts of the domain visible by an observer [23] and in multiobjective path-planning [13,19]. A natural question (and the focus of this paper) is whether factoring techniques can be used to alleviate this problem. In Sect. 3.1 we demonstrate experimentally that the "global factoring" is not suitable for this task. On the other hand, the localized factoring works, but adopting it to corner-induced rarefaction fans presents two new challenges. First of all, not all obstacle corners produce this effect; e.g., see the lower left corner in Fig. 1a.



(a) multiple obstacles **(b)** distorted trajectories

Fig. 5 A maze navigation example: non-permiable obstacles with $F = 1$ on the rest of Ω.(Left) the level sets of the value function u computed by the Fast Marching Method on a 240×240 grid and approximate optimal trajectories to the origin from 12 starting locations. (Right) a zoomed version to highlight the incorrect direction of "optimal" trajectories in the rarefaction fans at obstacle corners (Color figure online)



(a) level sets of u **(b)** domain splitting **(c)** level sets of T

Fig. 6 A simple example with one rarefying corner. (Left) level curves of the domain-restricted distance to a point source. (Center) a dynamic domain splitting based on the rarefaction fan. (Right) the level sets of a "cone + plane" function T capturing the correct rarefaction behavior (Color figure online)

Definition 3.1 An obstacle corner $\tilde{\mathbf{x}}$ is "regular" if the characteristic leading to it from Q points *into* that obstacle. (I.e., if an optimal trajectory starts from $\tilde{\mathbf{x}}$ in the direction $\mathbf{a}$, then $(-\mathbf{a})$ should point into the obstacle.) An obstacle corner is "rarefying" if it is not regular.

So, even though the rarefying corners are not known in advance, we can identify them dynamically, checking the above condition when the corresponding corner $\tilde{\mathbf{x}}$ becomes Accepted in Fast Marching Method and approximating the optimal $\mathbf{a} = \frac{-\nabla u(\tilde{\mathbf{x}})}{|\nabla u(\tilde{\mathbf{x}})|}$ using $\tilde{\mathbf{x}}$'s previously Accepted upwind neighbors. The resulting *"just-in-time localized factoring"* method is detailed in Algorithm 2. It maintains a list of identified rarefaction fans, with each entry containing the center of the fan (either a point source or a rarefying corner) and the corresponding localized function T. The algorithm is formulated in terms of u, with the corresponding τ values computed on the fly once the appropriate localized T is selected.

The second difficulty is to define a suitable T that will be used for factoring when updating all not-yet-Accepted gridpoints in $B_r(\tilde{\mathbf{x}})$. Intuitively, it might seem that a cone-like $T = \frac{|\mathbf{x}-\tilde{\mathbf{x}}|}{F(\tilde{\mathbf{x}})}$ is the right choice, similarly to our handling of point sources. However, as we show in Sect. 3.1,

Algorithm 2: Just-in-time Localized Factoring

Input: source point $\mathbf{x}_0$, speed function $F(\mathbf{x})$, fixed radius r

Initialize $u(\mathbf{x}_0) := 0$ and $u(\mathbf{x}) := +\infty$ for all gridpoints $\mathbf{x} \neq \mathbf{x}_0$;
Initialize Considered $:= \{\mathbf{x}_0\}$ and Accepted $:= \emptyset$;
Initialize FanList $:= \left\{\left(\mathbf{x}_0,\ T^{\mathbf{x}_0} = |\mathbf{x} - \mathbf{x}_0|/F(\mathbf{x}_0)\right)\right\}$;

while *Considered* $\neq \emptyset$ **do**
  Find a Considered gridpoint $\hat{\mathbf{x}}$ whose u value is the smallest;
  Move $\hat{\mathbf{x}}$ to Accepted list;
  if $\hat{\boldsymbol{x}}$ *is a rarefying corner* **then**
    Build a suitable $T^{\hat{\mathbf{x}}}$ using formula (9);
    Add an entry $\left(\hat{\mathbf{x}}, T^{\hat{\mathbf{x}}}\right)$ to FanList;
  end

  for *all not-yet-Accepted neighbors* $\boldsymbol{x}_{i,j}$ *of* $\hat{\boldsymbol{x}}$ **do**
    Check if $\mathbf{x}_{i,j}$ is within distance r from any $\tilde{\mathbf{x}}$ on FanList
    and identify the appropriate $T = T^{\tilde{\mathbf{x}}}$ (use $T = 0$ by default);
    Given the current u values at $\mathbf{x}_{i,j}$ & its neighbors,
    define their τ values as $\tau = (u - T)$ based on T selected for $\mathbf{x}_{i,j}$;

    Find τ_H, τ_V and solve the quadratic equation (6) for $\tau_{i,j}^{new}$;
    if $\tau_{i,j}^{new}$ *does not satisfy the upwinding condition* (7) **then**
      Use a one-sided update formula (8) to (re-)compute $\tau_{i,j}^{new}$;
    end
    $u_{i,j}^{new} := \tau_{i,j}^{new} + T_{i,j}$;
    if $u^{new} < u_{i,j}$ **then**
      Set $u_{i,j} := u_{i,j}^{new}$;
    end
  end
end

this choice does not yield the desired rate of convergence. This is due to the fact that such corner-born rarefaction fans are not radially symmetric. They only exist for $u > u(\tilde{\mathbf{x}})$ in the sector between a part of obstacle boundary and the characteristic passing through $\tilde{\mathbf{x}}$; see Fig. 1 and an even simpler example in Fig. 6. Note that, outside of that rarefaction sector, the second derivatives of u are bounded. But using the ansatz $u = T + \tau$ with the above cone-like T would introduce unbounded second derivatives in τ on non-rarefying parts of $B_r(\tilde{\mathbf{x}})$, thus degrading the rate of convergence. Therefore, we need to construct T which is cone-like only in the correct sector and remains smooth on the entire not-yet-Accepted portion of the domain. This yields a "cone+plane" version of T shown in Fig. 6c. Assuming that $\mathbf{c}$ is a unit vector bisecting the obstacle corner at $\tilde{\mathbf{x}}$, we can now split the plane into two sets

$$S_0 = \{\mathbf{x} \in \Omega \mid (\mathbf{x} - \tilde{\mathbf{x}}) \text{ is between } \mathbf{c} \text{ and } (-\mathbf{a})\}; \qquad S_1 = \Omega \backslash S_0.$$

Analytically, we can define T as follows

$$T(\mathbf{x}) = \begin{cases} \dfrac{|\mathbf{x} - \tilde{\mathbf{x}}|}{F(\tilde{\mathbf{x}})}, & \mathbf{x} \in S_0 \\ \dfrac{-\mathbf{a} \cdot (\mathbf{x} - \tilde{\mathbf{x}})}{F(\tilde{\mathbf{x}})}, & \mathbf{x} \in S_1. \end{cases} \tag{9}$$

The resulting T is not continuous along $\mathbf{c}$, but this will not matter since the discontinuity is hidden within the obstacle. The gradient of T is also continuous wherever T is, though the second derivatives are bounded but discontinuous along $(-\mathbf{a})$. Our numerical results show that this T fully recovers the first-order convergence of the numerical solution.

Remark 1 The bisector of obstacle corner is not the only choice for $\mathbf{c}$. Any directions falling inside the obstacle will work just as well since the idea is to "hide" the discontinuity line of T. For rectangular obstacles, it might feel more natural to choose $\mathbf{c}$ orthogonal to the characteristic direction $\mathbf{a}$. However, we prefer the bisector simply because it is a safe choice for arbitrary polygonal obstacles, which can be handled by a version of Algorithm 2 on (obstacle-fitted) triangulated meshes.

Another possibility is to hide the T's discontinuity in the part of Ω already Accepted (e.g., along $\mathbf{a}$) by the time this rarefying corner is identified. This is the approach we use in Sect. 4, when dealing with "slowly permeable obstacles".

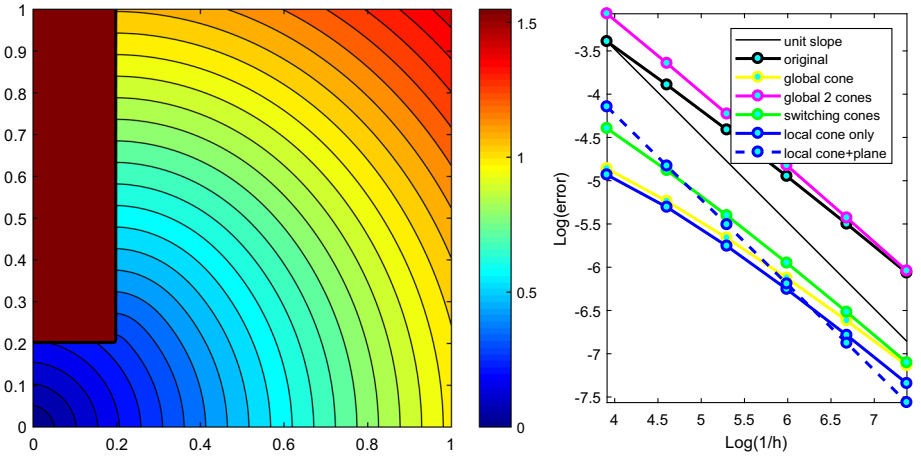
Remark 2 Our approach uses local factoring with a continuous T on the entire $B_r(\tilde{\mathbf{x}}) \cap \Omega$. A variant of the same idea is to employ local factoring with a standard cone-like $T = \frac{|\mathbf{x}-\tilde{\mathbf{x}}|}{F(\tilde{\mathbf{x}})}$ but only when updating gridpoints from a subset $B_r(\tilde{\mathbf{x}}) \cap \Omega \cap S_0$. (The difference is that one would use $T = 0$ when updating gridpoints in $B_r(\tilde{\mathbf{x}}) \cap \Omega \cap S_1$.) While we do not formally include this variant in our convergence studies in Sect. 3.1, its performance appears to be quite similar. E.g., for the above example from Fig. 6, the alternative version also shows the first-order convergence, but with L^∞ errors $\approx 10\%$ larger than those resulting from the "cone+plane" formula (9).

We close this section by discussing a subtle property implicitly used in our approach. The above construction relies on having a sufficiently accurate representation of the characteristic direction $\mathbf{a}$ at each rarefying corner. This might seem unreasonable: if our numerical approximation of u is only $O(h)$ accurate [as is the case in formulas (2) and (6–8)], then one could expect the resulting finite difference approximation of ∇u to be completely inaccurate. The same argument would imply that optimal trajectories also cannot be reliably approximated based on any first-order accurate representation of the value function. However, there is ample experimental evidence that such trajectory approximation works in practice. See, for example, the optimal trajectories in Figs. 5, 8, 9, 13 and in many optimal control and seismic imaging publications with and without factoring. The fact that this gradient approximation is in fact $O(h)$ accurate is also confirmed by a numerical study in [3] and is instrumental for constructing other (compact stencil, second-order) schemes for the Eikonal [3,24].

A plausible explanation for this "superconvergence" phenomenon is that the error in u-approximation is sufficiently "smooth", resulting in a convergent ∇u–approximation despite the use of divided differences. To the best of our knowledge, this property has not been proven for general Eikonal PDEs, though it has been rigorously demonstrated for the distance-to-a-point computations and for constant coefficient linear advection equations [14, Appendix B]. In our current context, we use the same idea to conjecture that the $\mathbf{a}$-dependent approximation of the localized T is sufficiently accurate to recover the full first-order accuracy in u computations with dynamic factoring. This conjecture appears to be fully confirmed by the convergence rates observed in our numerical experiments throughout this paper.

3.1 Numerical Examples

As a first numerical test, we consider a simple example from Fig. 6a: a domain-constrained distance u to the origin on $\Omega = [0, 1] \times [0, 1] \backslash \Omega_{ob}$, with a single rectangular obstacle



(a) level sets **(b)** L^∞ error

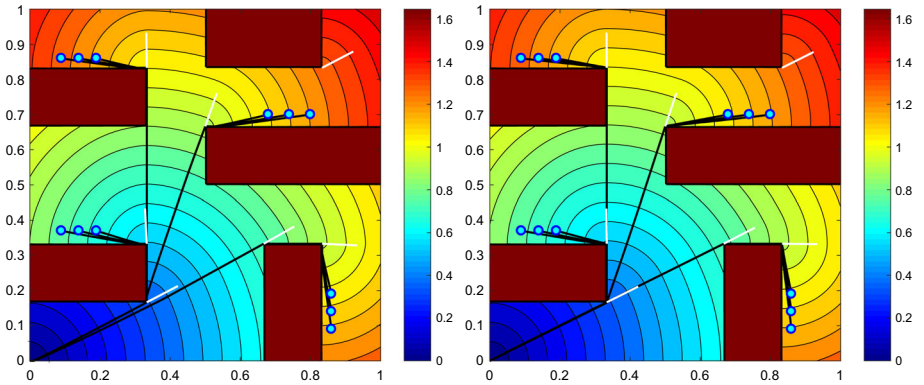
Fig. 7 Convergence of several methods for a simple obstacle case of Fig. 6a (Color figure online)

$\Omega_{ob} = (0, 0.2) \times (0.2, 1)$. Since $F = 1$, all optimal paths are piecewise-linear. According to Definition 3.1, we find that the corner at $\tilde{\mathbf{x}} = (0.2, 0.2)$ is rarefying. As a result, the problem contains two rarefaction fans: one at this corner and the other at the point source $\mathbf{x}_0 = (0, 0)$. We test the accuracy of several methods:

1. *Original* Original (un-factored) Eikonal solved on the entire Ω with the original Fast Marching Method.
2. *Global cone* Global factoring using $T(\mathbf{x}) = |\mathbf{x} - \mathbf{x}_0| / F(\mathbf{x}_0)$ on the entire Ω with Algorithm 1.
3. *Global 2 cones* Global factoring using $T(\mathbf{x}) = |\mathbf{x} - \mathbf{x}_0| / F(\mathbf{x}_0) + |\mathbf{x} - \tilde{\mathbf{x}}| / F(\tilde{\mathbf{x}})$ on the entire Ω with Algorithm 1.
4. *Switching cones* Global factoring using $T(\mathbf{x}) = |\mathbf{x} - \mathbf{x}_0| / F(\mathbf{x}_0)$ until $\tilde{\mathbf{x}}$ is accepted and then switch to global factoring using $T(\mathbf{x}) = |\mathbf{x} - \tilde{\mathbf{x}}| / F(\tilde{\mathbf{x}})$ on the rest of Ω.
5. *Localized cone only* Just-in-time localized factoring Algorithm 2 with $T(\mathbf{x}) = |\mathbf{x} - \mathbf{x}_0| / F(\mathbf{x}_0)$ on $B_r(\mathbf{x}_0)$ and another cone-like $T(\mathbf{x}) = |\mathbf{x} - \tilde{\mathbf{x}}| / F(\tilde{\mathbf{x}})$ on $B_r(\tilde{\mathbf{x}})$.
6. *Localized cone+plane* Just-in-time localized factoring Algorithm 2 with $T(\mathbf{x}) = |\mathbf{x} - \mathbf{x}_0| / F(\mathbf{x}_0)$ on $B_r(\mathbf{x}_0)$ and a dynamically defined "cone+plane" $T(\mathbf{x})$ specified by formula (9) on $B_r(\tilde{\mathbf{x}})$.

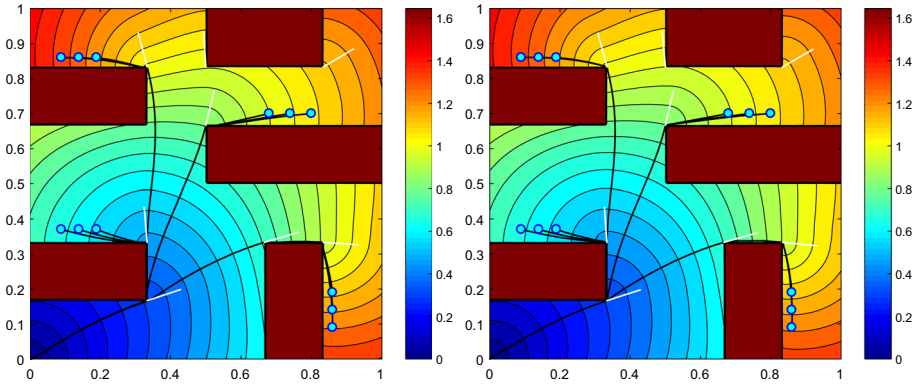
The first four of these are included to show that the corner-induced rarefaction fans do indeed degrade the rate of convergence and the issue cannot be addressed by global factoring. Accuracy of all methods is tested using a range of gridsizes ($h = \frac{1}{50} 2^{-k}$, where $k = 0, \ldots, 5$). The localized factoring is based on $r = 0.18$. As Fig. 7b clearly shows, only the last method actually exhibits the first-order of convergence. Even though the usual global factoring (method 2) starts out with smaller errors on coarser meshes, it becomes worse than our preferred approach (method 6) for smaller values of h. The fact that method 5 has a similar performance degradation proves the importance of choosing the correct localized factoring function T.

Since we do not rely on knowing ahead of time which corners are rarefying, the just-in-time localized factoring algorithm is excellent for "maze-navigation problems" with numerous obstacles and possibly inhomogeneous speed function F. Before running the algorithm, we



(a) maze: original **(b)** maze: localized factoring

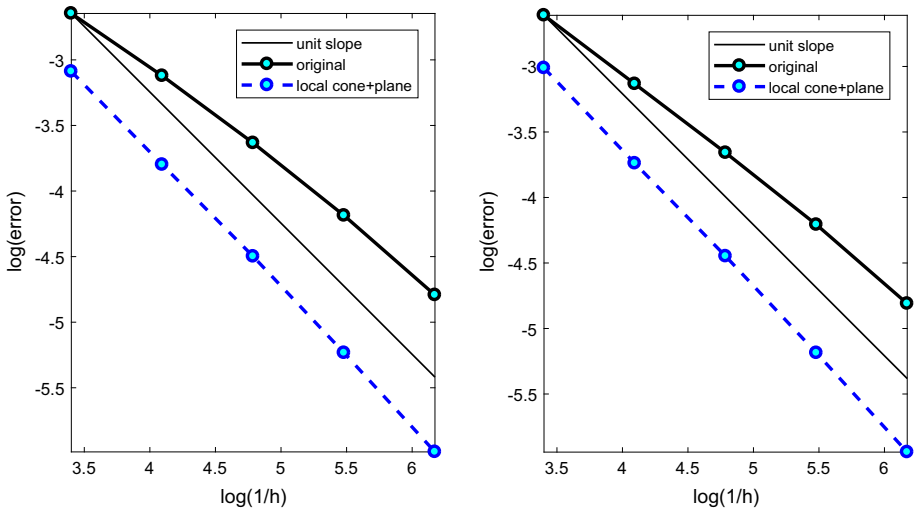
Fig. 8 Navigating a maze with $F = 1$. Level sets of u and representative optimal trajectories computed by the original FMM (left) and by Algorithm 2 (right). The latter avoids obvious numerical artifacts near rarefying corners (Color figure online)



(a) maze:original **(b)** maze: localized factoring

Fig. 9 Navigating a maze with $F(x, y) = 1 + 0.3 \sin(2\pi x) \sin(2\pi y)$.Level sets of u and representative optimal trajectories computed by the original FMM (left) and by Algorithm 2 (right). The latter avoids obvious numerical artifacts near rarefying corners (Color figure online)

create a binary array to identify all gridpoints contained inside obstacles. This array is then used to identify obstacle corners in Algorithm 2, and whenever a corner is found to be rarefying by Definition 3.1, a factoring procedure is locally applied with an appropriately chosen "additive factor" $T(\mathbf{x})$. Below we show two examples based on a "maze" from Fig. 5. In Fig. 8 we explore the version with $F = 1$. In Fig. 9 we use an inhomogeneous speed $F(x, y) = 1 + 0.3 \sin(2\pi x) \sin(2\pi y)$. In both cases, the white lines are used to indicate the (local) boundaries of corner-induced rarefaction fans. Twelve sample trajectories are shown to demonstrate that the trajectory distortions near the corners are avoided by just-in-time factoring. The convergence is tested using the gridsizes $h = \frac{1}{30} 2^{-k}$, where $k = 0, \ldots, 4$, and the "ground truth" is computed on a much finer grid with $h = 1/4800$. Figure 10 shows that, unlike the Fast Marching Method for the original Eikonal, our approach is globally first-order accurate in both examples.



(a) maze with constant F **(b)** maze with variable F

Fig. 10 L^∞ error for maze navigation examples (Color figure online)

4 Discontinuous Speed Function

Rarefaction fans can also arise due to discontinuities in F. Here we consider a simple subclass of such problems: a generalization of maze navigation examples from the previous section to account for "slowly permeable obstacles". We will assume that obstacles are described by an open set $\Omega_{ob} \subset (\Omega \backslash Q)$ and the speed F is lower inside them. In the simplest setting, F is piecewise constant with a discontinuity on $\partial\Omega_{ob}$ and $0 < F_{ob} < F_{free}$. We will use $\Upsilon = F_{free}/F_{ob}$ to measure the severity of obstacle slowdown.

The following properties are relatively easy to prove for this simple type of discontinuous F in 2D problems:

1. Rarefaction fans can only arise at point sources or at rarefying corners of slowly permeable obstacles. (E.g., there are no fans arising on non-corner parts of obstacle boundaries.)
2. When a rarefaction fan arises at an obstacle corner $\tilde{\mathbf{x}}$, it does not propagate into that obstacle.
3. Such rarefaction fans are always confined to a sector between the characteristic direction $(-\mathbf{a})$ and another vector $(-\mathbf{b})$ found from Snell's law.
4. Suppose $\mathbf{a}$ makes an angle $\alpha \in (0, \pi/2)$ with a normal to one side of a rectangular obstacle at $\tilde{\mathbf{x}}$ and $(-\mathbf{b})$ makes an angle $\beta \in (0, \pi/2]$ with a normal to the other side of that obstacle; see Fig. 11. Then these angles must satisfy

$$\sin\beta \;=\; \min\left(\sqrt{\Upsilon^2 - \sin^2\alpha},\; 1\right). \tag{10}$$

and the rarefaction fan takes place in a sector of angle $\delta = (\alpha + \beta - \frac{\pi}{2})$.

For the sake of brevity, we sketch the proof of the last of these only.

Proof We can reinterpret the characteristics as light rays traveling from a point source and refracted at the boundary of slowly permeable obstacles. We then use the Snell's Law to determine their changing directions.

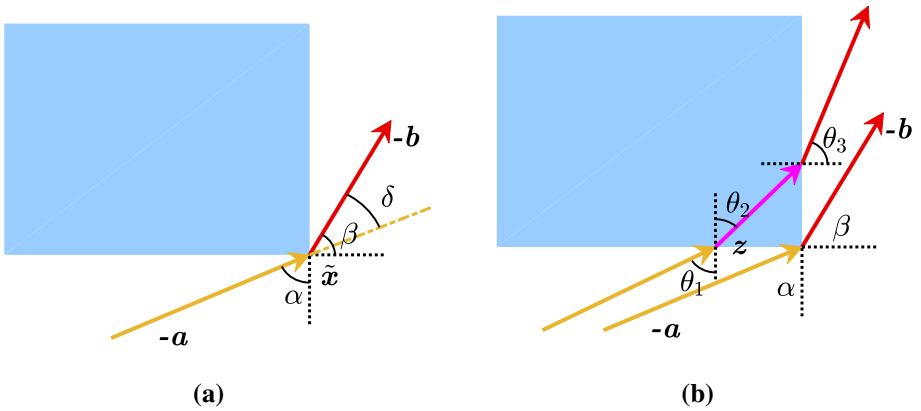


Fig. 11 A rarefaction fan at the corner of a slowly permeable obstacle. (Left) α is the incidence angle of a ray from the point source to the rarefying corner $\tilde{\mathbf{x}}$ and β is the "refracted" angle from the normal on the other side of the obstacle. The rarefaction fan appears in a sector corresponding to the angle δ between $(-\mathbf{b})$ shown in red and the yellow dash-dotted line corresponding to $(-\mathbf{a})$. (Right) a ray refraction happening at a point $\mathbf{z}$ close to $\tilde{\mathbf{x}}$. As $\mathbf{z} \to \tilde{\mathbf{x}}$, θ_1 and θ_3 will tend to α and β respectively, with the purple segment disappearing, and yielding the $(-\mathbf{a}, -\mathbf{b})$ path through $\tilde{\mathbf{x}}$ in the limit (Color figure online)

Consider a point $\mathbf{z}$ close to the corner $\tilde{\mathbf{x}}$, whose characteristic has an incidence angle θ_1 and refraction angle θ_2; see Fig. 11b. The incidence angle of the second refraction is $\frac{\pi}{2} - \theta_2$ and the second refraction angle is θ_3. If $\theta_3 < \frac{\pi}{2}$, by Snell's Law these three angles must satisfy

$$\frac{\sin\theta_1}{F_{free}} = \frac{\sin\theta_2}{F_{ob}}, \qquad \frac{\cos\theta_2}{F_{ob}} = \frac{\sin\theta_3}{F_{free}}. \tag{11}$$

Eliminating θ_2, we obtain

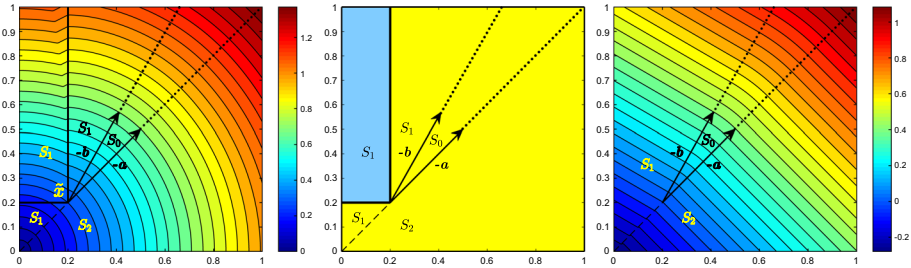
$$\sin\theta_3 = \sqrt{\left(\frac{F_{free}}{F_{ob}}\right)^2 - \sin^2\theta_1} = \sqrt{\Upsilon^2 - \sin^2\theta_1}$$

We note that this equality only makes sense if $\sqrt{\Upsilon^2 - \sin^2\theta_1} \le 1$. Otherwise, we will observe the "total internal reflection" with $\theta_3 = \frac{\pi}{2}$ and Snell's Law not holding for the $\theta_2 - \theta_3$ transition. So, the more accurate version of this relationship is $\sin\theta_3 = \min\left(\sqrt{\Upsilon^2 - \sin^2\theta_1},\ 1\right)$. As $\mathbf{z} \to \tilde{\mathbf{x}}$, we have $\theta_1 \to \alpha$, $\theta_3 \to \beta$, and we recover (10) in the limit. □

Remark 3 It is easy to provide a sufficient condition for the rarefaction fan filling the whole region between $(-\mathbf{a})$ and the obstacle boundary (exactly as we saw in the non-permeable case). Whenever $\Upsilon \ge \sqrt{2}$, we have $\sin\beta = 1$ and hence $\beta = \frac{\pi}{2}$; so, optimal trajectories from all starting positions in $\Omega \backslash \Omega_{ob}$ reach the exit set Q without passing through Ω_{ob}. On the other hand, in the continuous case ($\Upsilon = 1$), formula (10) implies that $\beta = \frac{\pi}{2} - \alpha$, $\delta = 0$ and no rarefaction fan is present.

Using $\mathbf{a}$ and $\mathbf{b}$ defined at a rarefying corner $\tilde{\mathbf{x}}$ in the above properties, it is natural to split $B_r(\tilde{\mathbf{x}})$ into three regions:

$$S_0 = \{\mathbf{x} \in \Omega \mid (\mathbf{x} - \tilde{\mathbf{x}}) \text{ is between } (-\mathbf{b}) \text{ and } (-\mathbf{a})\};$$
$$S_1 = \{\mathbf{x} \in \Omega \mid (\mathbf{x} - \tilde{\mathbf{x}}) \text{ is between } (-\mathbf{b}) \text{ and } \mathbf{a}\}; \quad S_2 = \Omega \backslash (S_0 \cup S_1).$$



(a) level sets of u **(b)** domain splitting **(c)** level sets of T

Fig. 12 A simple example with one "permeable obstacle". (Left) level curves of the "partially refracted" distance to a point source. (Center) a dynamic domain splitting based on the rarefaction fan. (Right) the level sets of a "cone + 2 planes" function T capturing the correct rarefaction behavior. The function has a small discontinuous jump along the ray parallel to $\mathbf{a}$ through $\tilde{\mathbf{x}}$ (Color figure online)

We can now build a suitable (localized) factoring function T as follows:

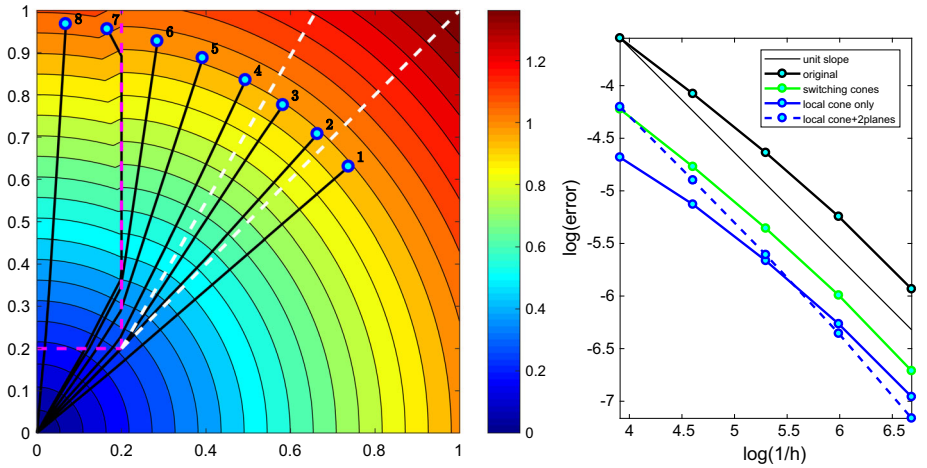
$$T(\mathbf{x}) = \begin{cases} \dfrac{|\mathbf{x}-\tilde{\mathbf{x}}|}{F(\tilde{\mathbf{x}})}, & \mathbf{x} \in S_0 \\ \dfrac{-\mathbf{b}\cdot(\mathbf{x}-\tilde{\mathbf{x}})}{F(\tilde{\mathbf{x}})}, & \mathbf{x} \in S_1 \\ \dfrac{-\mathbf{a}\cdot(\mathbf{x}-\tilde{\mathbf{x}})}{F(\tilde{\mathbf{x}})}, & \mathbf{x} \in S_2. \end{cases} \tag{12}$$

Based on the shape of the graph, we refer to this function as a "cone+2 planes"; see Fig. 12c. This formulation makes T discontinuous along a ray parallel to $\mathbf{a} = -\nabla u(\tilde{\mathbf{x}})/|\nabla u(\tilde{\mathbf{x}})|$, but for a sufficiently small r all gridpoints close to this ray in $B_r(\tilde{\mathbf{x}})$ will be already Accepted by the time we start this factoring. Both T and ∇T are continuous along $(-\mathbf{a})$ and $(-\mathbf{b})$.

4.1 Numerical Examples

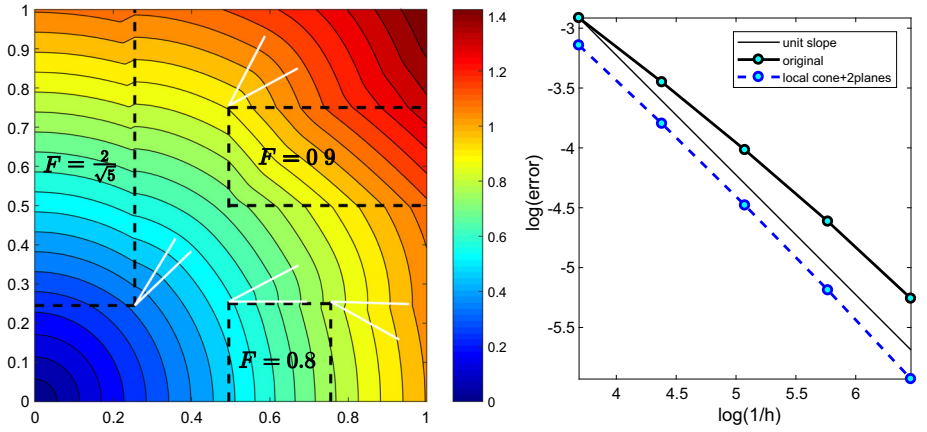
Returning to the example in Fig. 12, we choose $F_{ob} = \frac{2}{\sqrt{5}} \approx 0.894$ inside the obstacle $\Omega_{ob} = (0, 0.2) \times (0.2, 1)$ and $F_{free} = 1$ on $\Omega \backslash \Omega_{ob}$. Based on the properties discussed above, this will result in a rarefaction fan spreading in a sector of angle $\delta = \frac{\pi}{12}$ between the two white dashed lines in Fig. 13a. We test the convergence of several methods described in Sect. 3.1 and report the results in Fig. 13b. The numerical experiments are conducted using gridsizes $h = \frac{1}{50}2^{-k}$, where $k = 0, \ldots, 4$ and the "ground truth" is computed on a much finer grid with $h = 1/4000$. Unsurprisingly, the "original" (unfactored) method results in the largest errors and only the "localized cone + 2 planes" method exhibits the first-order convergence.

Our final example in Fig. 14 has multiple slowly permeable obstacles with each having a different F_{ob} (indicated in Fig. 14a) and $F_{free} = 1$ in the complement. At each corner, we use equation (10) to find $(-\mathbf{b})$ and use two white line segments to indicate the rarefying region. The "ground truth" is computed using $h = 1/6400$ and the convergence is tested using gridsizes $h = \frac{1}{40}2^{-k}, k = 0, \ldots, 4$. Figure 14b demonstrates that our method reduces the errors and recovers the first-order convergence.



(a) level sets of $u(\mathbf{x})$ **(b)** L^∞ error

Fig. 13 Optimal trajectories and convergence for a "single permeable obstacle" example introduced in Fig. 12. (Left) The level sets of u, with purple dashed lines showing the obstacle boundaries and white dashed lines showing the rarefaction fan boundaries. Eight representative optimal trajectories shown in black: (1) outside any region influenced by the obstacle, taking a straight line to the point source; (2, 3) starting within the rarefaction fan, coinciding after reaching the rarefying corner; (4–6) experiencing a double refraction; (7) starting inside the obstacle and experiencing the "total internal reflection" described in the proof, with two different segments inside the obstacle; (8) starting inside the obstacle and experiencing a single refraction. [*Note* the "light rays" (5–8) enter the obstacle with small incidence angles, resulting in barely changed refracted angles, so the first refraction is difficult to identify visually.] (Color figure online)



(a) level sets of $u(\mathbf{x})$ **(b)** L^∞ error

Fig. 14 A maze with several slowly permeable obstacles. (Left) the level sets of u with dashed lines showing the obstacle boundaries and white line segments showing the rarefaction fan boundaries. The speed F_{ob} is shown inside each obstacle (Color figure online)

5 Conclusions

We have introduced a new just-in-time factoring algorithm for Eikonal equations to reduce the numerical errors due to rarefaction fans. Prior (global and localized) factoring algorithms were meant to deal with rarefactions arising at point sources and we have carefully compared their accuracy in that setting. However, our main focus has been on rarefactions arising in 2D due to nonsmothness of $\partial\Omega$ (e.g., corners of non-permeable obstacles) or discontinuities in the speed function (e.g., corners of "slowly-permeable" obstacles). The locations and the geometry of such rarefaction fans are a priori unknown. Our algorithm uncovers them dynamically and adoptively applies the localized factoring. This dynamic aspect makes our approach natural in the Fast Marching framework. (With Fast Sweeping, one could in principle solve the original Eikonal on the entire domain, then identify all rarefaction fans in post-processing and re-solve the correctly factored equation on Ω.) Numerical tests confirm that our method restores the full linear convergence and prevents numerical artifacts in approximating optimal trajectories once the value function is already computed. While we have only implemented and tested the "additive" dynamic factoring, we expect that in the "multiplicative" case the results would be qualitatively similar. All presented examples were in the context of time-optimal path planning, but other optimization criteria (e.g., a cumulative exposure to an enemy observer) would also lead to a similarly factored Eikonal equation as long as the running cost remains isotropic. Even though our focus so far has been on applications in robotic navigation and computational geometry, we hope that the same general approach might also be useful in seismic imaging problems, which motivated much of the prior work on Eikonal factoring.

Several straightforward generalizations will make our method more useful in practice.

1. We can easily treat general polygonal obstacles by adding dynamic factoring to prior Fast Marching techniques on (obstacle-fitted) triangulated meshes [12,24]. The definition of our "additive factor" T will stay exactly the same; see also Remark 1 in Sect. 3.
2. The examples presented in Sect. 4 are based on rectangular "slowly permeable obstacles" with a piecewise constant speed function. However, the same approach is also applicable for the general discontinuous speed functions as long as the discontinuity lines are polygonal and aligned with the discretization mesh. The rarefaction fans can be determined based on a local information only (i.e., the directional limits of the speed function at a rarefying corner of the discontinuity line), and the definition of T in dynamic factoring will remain the same even when F is not piecewise constant.
3. If the speed of motion is anisotropic (i.e., dependent on the direction of motion rather than just the current location), the value function satisfies a more general Hamilton–Jacobi–Bellman PDE. Point-source-based factoring for the latter has already been developed (e.g., by Fast Sweeping in [16]). Marching-type techniques for anisotropic problems (e.g., [26] or [18]) can be similarly modified to handle the corner-induced rarefactions.
4. Another easy extension is to treat rarefaction fans due to more general boundary conditions (e.g., fast-varying $u = g$ specified on $\partial\Omega$ can result in rarefactions even if the boundary is smooth).

It will be more difficult to move to factoring suitable for higher-order accurate discretizations. For point-source-induced fans, this has been addressed in [15] and [17]. Similar ideas might work in our context, but higher derivatives will need to be estimated at rarefying corners and one would need to construct a smoother T than the version used in this paper.

Finally, the obvious limitation of our current approach is that $\Omega \subset R^2$. We expect that Eikonal problems in higher dimension will be much harder to factor dynamically. Even with

$F = 1$ and simple non-permeable box-obstacles in 3D, one would already need to deal with *rarefying edges* rather than corners.

Acknowledgements The authors are grateful to anonymous reviewers for their suggestions on improving this paper.

References

1. Alton, K., Mitchell, I.M.: An ordered upwind method with precomputed stencil and monotone node acceptance for solving static convex Hamilton–Jacobi equations. J. Sci. Comput. **51**(2), 313–348 (2012)
2. Bardi, M., Italo, C.D.: Optimal Control and Viscosity Solutions of Hamilton–Jacobi–Bellman Equations. Springer, Berlin (1997)
3. Benamou, J.D., Luo, S., Zhao, H.: A compact upwind second order scheme for the Eikonal equation. J. Comput. Math., pp. 489–516 (2010)
4. Bertsekas, D.P.: Network Optimization: Continuous and Discrete Models. Athena Scientific, Belmont (1998)
5. Chacon, A., Vladimirsky, A.: Fast two-scale methods for Eikonal equations. SIAM J. Sci. Comput. **34**(2), A547–A578 (2012)
6. Chacon, A., Vladimirsky, A.: A parallel two-scale method for eikonal equations. SIAM J. Sci. Comput. **37**(1), A156–A180 (2015)
7. Crandall, M.G., Lions, P.L.: Viscosity solutions of Hamilton–Jacobi equations. Trans. Am. Math. Soc. **277**(1), 1–42 (1983)
8. Dahiya, D., Cameron, M.: Ordered line integral methods for computing the quasi-potential. J. Sci. Comput. (2017). In revision; arXiv:1706.07509
9. Dijkstra, E.: A note on two problems in connexion with graphs. Numer. Math. **1**, 269–271 (1959)
10. Fomel, S., Luo, S., Zhao, H.: Fast sweeping method for the factored Eikonal equation. J. Comput. Phys. **228**(17), 6440–6455 (2009)
11. Kao, C.Y., Osher, S., Qian, J.: Lax–Friedrichs sweeping scheme for static Hamilton–Jacobi equations. J. Comput. Phys. **196**(1), 367–391 (2004)
12. Kimmel, R., Sethian, J.A.: Fast marching methods on triangulated domains. Proc. Natl. Acad. Sci. **95**, 8341–8435 (1998)
13. Kumar, A., Vladimirsky, A.: An efficient method for multiobjective optimal control and optimal control subject to integral constraints. J. Comput. Math. **28**(4), 517–551 (2010)
14. Luo, S.: Numerical Methods for Static Hamilton–Jacobi Equations. Ph.D. thesis, University of California at Irvine (2009)
15. Luo, S., Qian, J.: Factored singularities and high-order Lax–Friedrichs sweeping schemes for point-source traveltimes and amplitudes. J. Comput. Phys. **230**(12), 4742–4755 (2011)
16. Luo, S., Qian, J.: Fast sweeping methods for factored anisotropic eikonal equations: multiplicative and additive factors. J. Sci. Comput. **52**(2), 360–382 (2012)
17. Luo, S., Qian, J., Burridge, R.: High-order factorization based high-order hybrid fast sweeping methods for point-source Eikonal equations. SIAM J. Numer. Anal. **52**(1), 23–44 (2014)
18. Mirebeau, J.M.: Efficient fast marching with Finsler metrics. Numer. Math. **126**(3), 515–557 (2014)
19. Mitchell, I.M., Sastry, S.: Continuous path planning with multiple constraints. In: 42nd IEEE International Conference on Decision and Control (IEEE Cat. No.03CH37475), vol. 5, pp. 5502–5507 (2003)
20. Noble, M., Gesret, A., Belayouni, N.: Accurate 3-d finite difference computation of traveltimes in strongly heterogeneous media. Geophys. J. Int. **199**(3), 1572–1585 (2014)
21. Sethian, J.A.: A fast marching level set method for monotonically advancing fronts. Proc. Natl. Acad. Sci. **93**(4), 1591–1595 (1996)
22. Sethian, J.A.: Fast marching methods. SIAM Rev. **41**(2), 199–235 (1999)
23. Sethian, J.A., Adalsteinsson, D.: An overview of level set methods for etching, deposition, and lithography development. IEEE Trans. Semicond. Manuf. **10**(1), 167–184 (1997)
24. Sethian, J.A., Vladimirsky, A.: Fast methods for the Eikonal and related Hamilton–Jacobi equations on unstructured meshes. Proc. Natl. Acad. Sci. **97**(11), 5699–5703 (2000)
25. Sethian, J.A., Vladimirsky, A.: Ordered upwind methods for static Hamilton–Jacobi equations. Proc. Natl. Acad. Sci. **98**(20), 11069–11074 (2001)
26. Sethian, J.A., Vladimirsky, A.: Ordered upwind methods for static Hamilton–Jacobi equations: theory and algorithms. SIAM J. Numer. Anal. **41**(1), 325–363 (2003)

27. Treister, E., Haber, E.: A fast marching algorithm for the factored Eikonal equation. J. Comput. Phys. **324**, 210–225 (2016)
28. Tsai, Y.H.R., Cheng, L.T., Osher, S., Zhao, H.K.: Fast sweeping algorithms for a class of Hamilton–Jacobi equations. SIAM J. Numer. Anal. **41**(2), 673–694 (2003)
29. Tsitsiklis, J.N.: Efficient algorithms for globally optimal trajectories. IEEE Trans. Autom. Control **40**(9), 1528–1538 (1995)
30. Zhang, Y.T., Chen, S., Li, F., Zhao, H., Shu, C.W.: Uniformly accurate discontinuous galerkin fast sweeping methods for Eikonal equations. SIAM J. Sci. Comput. **33**(4), 1873–1896 (2011)
31. Zhang, Y.T., Zhao, H.K., Qian, J.: High order fast sweeping methods for static Hamilton–Jacobi equations. J. Sci. Comput. **29**(1), 25–56 (2006)
32. Zhao, H.: A fast sweeping method for Eikonal equations. Math. Comput. **74**(250), 603–627 (2005)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.