Journal of Computational Physics 270 (2014) 378–401

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

Fast Huygens sweeping methods for Helmholtz equations in inhomogeneous media in the high frequency regime

Songting Luo [a], Jianliang Qian [b,*], Robert Burridge [c]

[a] Department of Mathematics, Iowa State University, Ames, IA 50011, USA
[b] Department of Mathematics, Michigan State University, East Lansing, MI 48824, USA
[c] Department of Mathematics and Statistics, University of New Mexico, Albuquerque, NM 87131, USA

ARTICLE INFO

Article history:
Received 26 November 2013
Received in revised form 20 February 2014
Accepted 31 March 2014
Available online 8 April 2014

Keywords:
Eikonal
Helmholtz
High-frequency wave
Huygens sweeping

ABSTRACT

In some applications, it is reasonable to assume that geodesics (rays) have a consistent orientation so that the Helmholtz equation may be viewed as an evolution equation in one of the spatial directions. With such applications in mind, we propose a new Eulerian computational geometrical-optics method, dubbed the fast Huygens sweeping method, for computing Green functions of Helmholtz equations in inhomogeneous media in the high-frequency regime and in the presence of caustics. The first novelty of the new method is that the Huygens–Kirchhoff secondary source principle is used to integrate many locally valid asymptotic solutions to yield a globally valid asymptotic solution so that caustics associated with the usual geometrical-optics ansatz can be treated automatically. The second novelty is that a butterfly algorithm is adapted to carry out the matrix–vector products induced by the Huygens–Kirchhoff integration in $O(N \log N)$ operations, where N is the total number of mesh points, and the proportionality constant depends on the desired accuracy and is independent of the frequency parameter. To reduce the storage of the resulting traveltime and amplitude tables, we compress each table into a linear combination of tensor-product based multivariate Chebyshev polynomials so that the information of each table is encoded into a small number of Chebyshev coefficients.

The new method enjoys the following desired features: (1) it precomputes a set of local traveltime and amplitude tables; (2) it automatically takes care of caustics; (3) it constructs Green functions of the Helmholtz equation for arbitrary frequencies and for many point sources; (4) for a specified number of points per wavelength it constructs each Green function in nearly optimal complexity in terms of the total number of mesh points, where the prefactor of the complexity only depends on the specified accuracy and is independent of the frequency parameter.

Both two-dimensional (2-D) and three-dimensional (3-D) numerical experiments are presented to demonstrate the performance and accuracy of the new method.

* Corresponding author.
E-mail addresses: luos@iastate.edu (S. Luo), qian@math.msu.edu (J. Qian), burridge137@gmail.com (R. Burridge).

http://dx.doi.org/10.1016/j.jcp.2014.03.066
0021-9991/

1. Introduction

We consider the Helmholtz equation with a point-source condition,

$$\nabla_{\mathbf{r}}^2 U + \frac{\omega^2}{v^2} U = -\delta(\mathbf{r} - \mathbf{r}_0), \tag{1}$$

where the Sommerfeld radiation condition is imposed at infinity, $\mathbf{r}_0 = (x_0, y_0, z_0)$ is the source point, $U(\mathbf{r}, \omega; \mathbf{r}_0)$ is the wave field, ω is the frequency, $v(\mathbf{r})$ is the wave speed, and $\nabla_{\mathbf{r}}^2$ denotes the Laplacian at $\mathbf{r} = (x, y, z)$. This equation arises in a variety of physical applications, ranging from acoustics, elasticity, electromagnetics to geophysics. Therefore, it is highly desirable to develop efficient and accurate numerical methods for this equation. Because the solution of the Helmholtz equation represents the spatial factor of a time-harmonic solution of the time-dependent wave equation, it is highly oscillatory when the frequency parameter ω is large. Since it is very costly for a direct method to resolve these oscillations, asymptotic methods such as geometrical optics are sought to deal with such difficulties. In this paper, we propose a new Eulerian computational geometrical-optics method, which we call the fast Huygens sweeping method, for the Helmholtz equations in inhomogeneous media in the high-frequency regime and in the presence of caustics. The new method is based on the Huygens secondary source principle and the recent development of butterfly algorithms for constructing low-rank matrix approximations.

To motivate the new method, we apply the geometrical-optics large-ω ansatz to the Helmholtz equation, yielding the eikonal equation for traveltime and the transport equation for amplitude, respectively. These two equations are weakly coupled in the sense that the eikonal equation needs to be solved first to provide necessary coefficients for the transport equation for the amplitude. Because the eikonal equation is a first-order nonlinear partial differential equation (PDE), in general it does not have a globally defined smooth solution. The concept of viscosity solution was invented to single out a uniquely defined weak solution among many possible generalized solutions for such nonlinear first-order PDEs, and the resulting viscosity solution for the eikonal equation is the so-called first-arrival traveltime [26], which is continuous everywhere but not necessarily differentiable everywhere. The difficulty arises exactly when the viscosity solution fails to be differentiable, and the resulting gradient of the traveltime function is discontinuous. Consequently, the linear transport equation for the squared amplitude in the conservation form has discontinuous coefficients; theoretically, such linear transport equations with discontinuous coefficients are not well understood. Computationally, nevertheless, we may design high-order numerical methods to solve the eikonal and transport equations in physical space to obtain the traveltime and amplitude functions; the resulting two functions can be substituted into the geometrical-optics ansatz to obtain an "asymptotic" solution for the Helmholtz equation. But when we compare this asymptotic solution with the direct solution of the Helmholtz equation, we find that these two solutions are different "globally". Mathematically, this difference may be traced back to the multivaluedness of the traveltime function and related caustics [5], which can be detected by applying the method of characteristics to the eikonal equation with the appropriate point-source condition. One of the major observations in [29,31] is that the large-ω ansatz yields faithful asymptotic solutions to the Helmholtz equation before caustics occur; in other words, this ansatz can be used *locally*. Then the natural question is: how to obtain a globally defined, faithful asymptotic solution to the Helmholtz equation by using this large-ω ansatz locally? The answer is provided by the Huygens secondary source principle, which roughly states that the wave field from a source can be replaced by the field radiated by equivalent secondary sources on a surface which encloses the source.

It has been shown in [35,46] that the traveltime function for the eikonal equation with a point-source condition is locally smooth in the neighborhood of the source except at the source point itself; this implies that caustics will not develop right away on the expanding wavefront away from the source. Therefore, in a local neighborhood of the point source, the traveltime and amplitude functions resulting from solving the eikonal and transport equations are smooth except at the point source, and they yield a valid asymptotic Green function in that local neighborhood.

To go beyond caustics, we will make some assumptions for the Helmholtz equation under consideration. For some seismic applications, it is natural to assume that geodesics (rays) have a consistent orientation (such as directed downwards) so that the Helmholtz equation can be viewed as an evolution equation in one of the spatial directions. Consequently, we first partition the computational domain into several subdomains which enclose the primary point source, and we make use of the Huygens principle by setting up secondary sources on the closed boundaries of those subdomains. More importantly, the partitioning should satisfy the requirement that the large-ω ansatz for each secondary source on the subdomain boundaries generates an asymptotic solution which is valid in the corresponding subdomain, and this requirement can be achieved by computing the traveltime function for that secondary source point in a small neighborhood where the resulting traveltime field is smooth. According to this construction each secondary source provides a locally valid Green function. To synthesize information from those secondary sources, we use the Huygens–Kirchhoff integral identity to carry out an integration along enclosed boundaries so that we can compute the global Green function for the primary source at any observation point inside the subdomain under consideration. In this way we sweep through the whole domain to obtain the global Green function for the primary source, and caustics are implicitly and automatically taken care of during this sweeping process.

The issue is now concentrated on how to implement the above sweeping strategy efficiently. To tackle this challenging problem, we must surmount several obstacles. The first obstacle is that the traveltime and amplitude functions for the eikonal and transport equations with point-source conditions have upwind singularities at the source point, making it difficult to compute these two functions with high-order accuracy; moreover, the occurrence of the Laplacian of the traveltime

in the transport equation makes solving the transport equation an even more delicate task. To deal with this obstacle, we use our newly developed high-order schemes for computing the first-arrival traveltimes and amplitudes as illustrated in [29,31].

The second obstacle is how to store the many traveltime and amplitude tables that our sweeping strategy will generate for those specified secondary point sources. This storage issue is critical as we are aiming at solving both the 2-D and 3-D Helmholtz equations in the high frequency regime. To reduce data storage, we propose to compress each table into a linear combination of tensor-product based multivariate Chebyshev polynomials so that the information of each table is encoded into a small number of Chebyshev coefficients. Computationally, such compression leads to a significant storage reduction and efficient memory accesses.

The third obstacle is how to carry out efficiently the dense matrix–vector products induced by the Huygens–Kirchhoff integration. Because we are interested in the asymptotic solution everywhere in the computational domain, the solution at observation points (receivers) on a line in the 2-D case or on a plane in the 3-D case corresponds to the result of two matrix–vector products. Let n be the number of mesh points along each coordinate direction of the computational domain, so that the total number of mesh points is $N = n^d$ in the d-dimensional case. In the 2-D case, straightforward implementation of the above matrix–vector products will lead to $O(N)$ operations for each line of receivers, and the total computation cost will be $O(N^{\frac{3}{2}})$ as we need to carry out such matrix–vector products for roughly $\sqrt{N} = n$ lines of receivers. In the 3-D case, straightforward implementation of the above matrix–vector products will lead to $O(N^{\frac{4}{3}})$ operations for each plane of receivers, and the total computation cost will be $O(N^{\frac{5}{3}})$ as we need to carry out such matrix–vector products for roughly $N^{\frac{1}{3}}$ planes of receivers. Such computational cost is far too expensive to make our strategy practical. Therefore, to tackle this difficulty, we adapt to our application the multilevel matrix decomposition based butterfly algorithm [34,37,53,11,13]. The resulting butterfly algorithm allows us to carry out the required matrix–vector products with the total computational cost of $O(N \log N)$ complexity, where the proportionality constant depends only on the specified accuracy and is independent of frequency ω. Such low-rank rapid matrix–vector products are responsible for the adjective "fast" in the name "fast Huygens-sweeping method" of our method.

The fast Huygens-sweeping method also has two unique merits which may be attributed to the precomputed traveltime and amplitude tables. The first merit is that because the traveltime and amplitude functions are independent of the frequency parameter, those tables can be used to construct asymptotic Green functions at a given primary source for arbitrary frequencies. The second merit is that those tables can be used to construct asymptotic Green functions at many primary sources for arbitrary frequencies as well. These two merits are much desired in many applications, such as seismic imaging and inversion.

1.1. Related work

The high-order schemes for the eikonal and transport equations that we are using here were developed in [29,31], which in turn are based on Lax–Friedrichs sweeping [22,56,49,55,45], weighted essentially non-oscillatory (WENO) finite difference approximation [38,27,20,19], and factorization of the upwind source singularities [39,54,17,29,31,30]. To treat the upwind singularity at the point source, an adaptive method for the eikonal and transport equations has been proposed in [41] as well. See [6,14] for reviews on Eulerian geometrical optics.

The idea of compressing a traveltime table into a small number of coefficients in a certain basis has been used frequently in seismic imaging by the geophysical community. Here we use the tensor-product based Chebyshev polynomials as the basis to compress both the traveltime and amplitude tables, as inspired by the work in [1].

The origin of the multilevel-matrix decomposition based butterfly algorithm can be traced back to the work [34], and it has been further developed in [37,53,11,13]. In this work, we are using the version of the fast butterfly algorithm first developed in [11] and further analyzed in [13].

To construct global asymptotic solutions for the Helmholtz equation even in the presence of caustics, there exist three possible approaches in the literature. The first approach is based on Ludwig's uniform asymptotic expansions at caustics [28,10] which require that the caustic structure is given. The second approach is based on the Maslov canonical operator theory [32]. Although the Maslov theory is beautiful, it is not so useful as it requires identifying where caustics occur first before the theory can be applied; in practice, caustics can occur anywhere along a central ray in an inhomogeneous medium with a high probability as shown in [51]. The third approach is based on Gaussian beam methods [12,40,44,52,25,48,47]. Although Gaussian beam methods can treat caustics automatically along a central ray, the method itself suffers from expensive beam summation and exponential growth of beam width as analyzed and illustrated in [42,43,23,24,36,47], and such shortcomings sometimes have hindered applications of Gaussian beam methods to complicated inhomogeneous media. Our proposed new method is different from the above three approaches.

1.2. Layout

The paper is organized as follows. In Section 2, we present details for constructing the viscosity-solution based, locally valid asymptotic Green functions and the related inherent limitations. In Section 3, we present our fast Huygens-sweeping methods for constructing globally valid asymptotic Green functions, along with detailed implementation and complexity

analysis. In Section 4, both 2-D and 3-D numerical experiments are presented to demonstrate the performance and accuracy of our new method. Concluding remarks are given in Section 5 along with a description of ongoing and future projects.

2. Viscosity-solution based locally valid Green function

Since we are looking for asymptotic solutions for the Helmholtz equation (1) in the high frequency regime in terms of the large parameter ω, we apply the following geometrical-optics large-ω ansatz to the wave field U,

$$U(\mathbf{r},\omega;\mathbf{r}_0) = A(\mathbf{r};\mathbf{r}_0)e^{i\omega\tau(\mathbf{r};\mathbf{r}_0)} + O\left(\frac{1}{\omega}\right), \tag{2}$$

where A is the amplitude and τ is the traveltime function.

Substituting (2) into (1) and equating to zero the coefficients of ω and ω^2, one finds that the traveltime function τ and the amplitude A, respectively, satisfy the eikonal equation,

$$|\nabla_{\mathbf{r}}\tau| = s(\mathbf{r}), \tag{3}$$

and the transport equation,

$$\nabla_{\mathbf{r}}\tau\cdot\nabla_{\mathbf{r}}A + \frac{1}{2}A\nabla_{\mathbf{r}}^2\tau = 0, \tag{4}$$

where $s = 1/v$ is the slowness field.

Since neither the traveltime τ nor the amplitude A depends on frequency ω, upon appending appropriate point-source conditions for τ and A, one can first compute the traveltime function τ by solving the eikonal equation (3) and then the amplitude A by solving the transport equation (4). Afterwards, we assemble τ and A into the ansatz (2) to reconstruct the wave field U.

Specifically, after taking into account the outgoing radiation boundary condition at the point source, we have the following geometrical-optics approximation of the Green functions (See Appendix C in [25]),

$$G(\mathbf{r},\omega;\mathbf{r}') = \frac{1}{\sqrt{\omega}}A(\mathbf{r};\mathbf{r}')e^{i(\omega\tau(\mathbf{r};\mathbf{r}')+\pi/4)}, \quad \text{in 2-D}, \tag{5}$$

$$G(\mathbf{r},\omega;\mathbf{r}') = A(\mathbf{r};\mathbf{r}')e^{i\omega\tau(\mathbf{r};\mathbf{r}')}, \quad \text{in 3-D}, \tag{6}$$

where $\mathbf{r}'$ indicate a generic point-source location. In particular, in a homogeneous medium with $v(\mathbf{r}) = v_0$ being a constant, the traveltime and amplitude functions are given as the following,

$$\tau_0(\mathbf{r};\mathbf{r}') = \frac{|\mathbf{r}-\mathbf{r}'|}{v_0}, \tag{7}$$

$$A_0(\mathbf{r};\mathbf{r}') = \frac{1}{4}\left(\frac{1}{2v_0}\right)^{(d-3)/2}\left(\frac{1}{\pi|\mathbf{r}-\mathbf{r}'|}\right)^{(d-1)/2}, \tag{8}$$

where d denotes the dimension of the space under consideration. These formulas are useful for initializing the computation of the traveltime and amplitude functions.

For a source $\mathbf{r}_0 = (x_0, y_0, z_0)$ in an isotropic solid, we treat the eikonal equation as a first-order nonlinear PDE so that the computed traveltime $\tau(\mathbf{r};\mathbf{r}_0)$ is the viscosity solution of the eikonal equation (3) with the initial condition [26],

$$\lim_{\mathbf{r}\to\mathbf{r}_0}\left(\frac{\tau(\mathbf{r};\mathbf{r}_0)}{|\mathbf{r}-\mathbf{r}_0|} - \frac{1}{v(\mathbf{r})}\right) = 0, \tag{9}$$

which implies that $\tau(\mathbf{r}_0;\mathbf{r}_0) = 0$ at the source $\mathbf{r}_0$.

Because the amplitude function is singular and unbounded at the source which can be seen from the amplitude formula in a homogeneous medium, we cannot start computing the amplitude function right at the source. Instead, based on the computed traveltime field, we solve the transport equation (4) by initializing the amplitude function slightly away from the source.

Eq. (4) is a first-order advection equation for the amplitude A. In order to get a first-order accurate amplitude field, one needs a third-order accurate traveltime field since the Laplacian of the traveltime field is involved [41].

Both the traveltime τ and the amplitude A have upwind singularities at the source $\mathbf{r}_0$ as the distance-like traveltime function is not differentiable at the source. Since the source singularity induces large local truncation errors near the source, these large errors are inherited by the numerical schemes which will propagate these errors to the whole computational domain. Without special treatments of this source singularity, all the first-order or high-order numerical methods for the traveltime or the amplitude can formally have at most first-order convergence and large errors. Therefore, in order to treat these source singularities, we utilize the factorization ideas in [39,54,17,29,31,30] to compute τ and A. We refer to [29,31] for high-order WENO based Lax–Friedrichs schemes to compute τ and A, which in turn allow us to construct the asymptotic Green functions according to formulas (5) and (6).

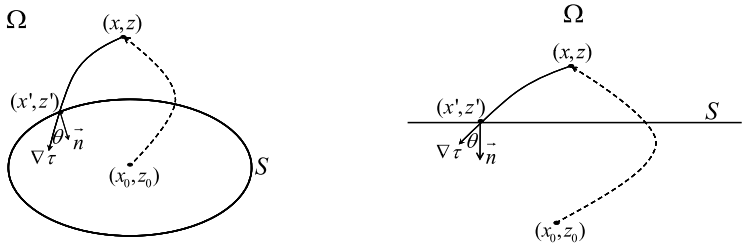

Fig. 1. 2-D demonstration of Huygens–Kirchhoff integral. Left: $\mathcal{S}$ is a closed surface; Right: $\mathcal{S}$ is an infinite plane.

As shown in [29,31], since the viscosity-solution for the first-arrival traveltime field does not contain later-arriving traveltimes, the interference effects due to the multivaluedness of the traveltime function are missing in the constructed asymptotic Green function. Therefore, the question is how to get back the interference effects by using locally valid asymptotic solutions? We utilize the Huygens secondary source principle as the backbone to relay the information carried by locally valid asymptotic solutions, and the Huygens–Kirchhoff integration formula serves as the relay station.

3. Huygens-principle based globally valid Green function

3.1. Huygens–Kirchhoff formula

Mathematically, the Huygens secondary source principle is characterized by the Huygens–Kirchhoff integration formula [9,3]. Consider the Helmholtz equation (1) with the Sommerfeld radiation condition imposed at infinity, which is in fact a special case of the exterior problem for the Helmholtz equation. Assume that a domain Ω is illuminated by the exterior source $\mathbf{r}_0 = (x_0, y_0, z_0) \notin \Omega$, and $\mathcal{S} = \partial\Omega$ is the closed surface enclosing the domain Ω; see Fig. 1. At any observation point $\mathbf{r} = (x, y, z) \in \Omega$, the wave field $U(\mathbf{r}; \mathbf{r}_0)$ excited by the source $\mathbf{r}_0$ can be written as a surface integral over S:

$$U(\mathbf{r}; \mathbf{r}_0) = \int_{\mathcal{S}} \left\{ G(\mathbf{r}'; \mathbf{r}) \nabla_{\mathbf{r}'} U(\mathbf{r}'; \mathbf{r}_0) \cdot \mathbf{n}(\mathbf{r}') - U(\mathbf{r}'; \mathbf{r}_0) \nabla_{\mathbf{r}'} G(\mathbf{r}'; \mathbf{r}) \cdot \mathbf{n}(\mathbf{r}') \right\} dS, \tag{10}$$

where the integration is with respect to $\mathbf{r}'$ over $\mathcal{S}$, $U(\mathbf{r}'; \mathbf{r}_0)$ is the known wave field at the observation point $\mathbf{r}'$ due to the source $\mathbf{r}_0$, $\mathbf{n}(\mathbf{r}')$ is the outward normal to $\mathcal{S}$ at $\mathbf{r}' = (x', y', z')$, and $G(\mathbf{r}'; \mathbf{r})$ is the Green function satisfying the Helmholtz equation (1) with the point source $\mathbf{r}$.

In the Huygens–Kirchhoff integral (10), we assume that $U(\mathbf{r}'; \mathbf{r}_0)$ and $\nabla_{\mathbf{r}'} U(\mathbf{r}'; \mathbf{r}_0) \cdot \mathbf{n}(\mathbf{r}')$ are given on the surface $\mathcal{S}$. In order to determine the wave field $U(\mathbf{r}; \mathbf{r}_0)$ at $\mathbf{r}$, one needs to compute $G(\mathbf{r}'; \mathbf{r})$ and $\nabla_{\mathbf{r}'} G(\mathbf{r}'; \mathbf{r}) \cdot \mathbf{n}(\mathbf{r}')$ on $\mathcal{S}$, which can be approximated with the geometrical-optics ansatz (2). Note that since the Green function has the form,

$$G(\mathbf{r}'; \mathbf{r}) \approx A(\mathbf{r}'; \mathbf{r}) e^{i\omega\tau(\mathbf{r}'; \mathbf{r})}, \tag{11}$$

we have

$$\nabla_{\mathbf{r}'} G(\mathbf{r}'; \mathbf{r}) \approx \nabla_{\mathbf{r}'} A(\mathbf{r}'; \mathbf{r}) e^{i\omega\tau(\mathbf{r}'; \mathbf{r})} + i\omega \nabla_{\mathbf{r}'} \tau(\mathbf{r}'; \mathbf{r}) G(\mathbf{r}'; \mathbf{r}).$$

Therefore, retaining only the leading order term in keeping with the geometrical optics approximation, we have

$$\nabla_{\mathbf{r}'} G(\mathbf{r}'; \mathbf{r}) \cdot \mathbf{n}(\mathbf{r}') \approx i\omega G(\mathbf{r}'; \mathbf{r}) \nabla_{\mathbf{r}'} \tau(\mathbf{r}'; \mathbf{r}) \cdot \mathbf{n}(\mathbf{r}'),$$

so that we can approximate the Huygens–Kirchhoff integral (10) in the following form,

$$U(\mathbf{r}; \mathbf{r}_0) \approx \int_{\mathcal{S}} G(\mathbf{r}'; \mathbf{r}) \left\{ \nabla_{\mathbf{r}'} U(\mathbf{r}'; \mathbf{r}_0) \cdot \mathbf{n}(\mathbf{r}') - i\omega U(\mathbf{r}'; \mathbf{r}_0) \nabla_{\mathbf{r}'} \tau(\mathbf{r}'; \mathbf{r}) \cdot \mathbf{n}(\mathbf{r}') \right\} dS. \tag{12}$$

Since $\tau(\mathbf{r}'; \mathbf{r})$ satisfies the eikonal equation (3) with the point source $\mathbf{r}$, we have

$$\nabla_{\mathbf{r}'} \tau(\mathbf{r}'; \mathbf{r}) \cdot \mathbf{n}(\mathbf{r}') = s(\mathbf{r}') \cos(\theta(\mathbf{r}'; \mathbf{r})),$$

where $\theta(\mathbf{r}'; \mathbf{r})$ is the arrival angle formed by the ray from the source $\mathbf{r}$ to the receiver $\mathbf{r}'$ and the normal direction $\mathbf{n}(\mathbf{r}')$ at $\mathbf{r}'$; see Fig. 1.

Note that in order to determine $U(\mathbf{r}; \mathbf{r}_0)$ from the Huygens–Kirchhoff integral for each point $\mathbf{r}$ in the domain Ω enclosed by $\mathcal{S}$, one needs to compute $\tau(\mathbf{r}'; \mathbf{r})$, $A(\mathbf{r}'; \mathbf{r})$, and $\cos(\theta(\mathbf{r}'; \mathbf{r}))$ by using each $\mathbf{r} \in \Omega$ as a point source, which is numerically expensive since Ω is a manifold of one-dimension higher than S. However, since $G(\mathbf{r}'; \mathbf{r}) = G(\mathbf{r}; \mathbf{r}')$, the following reciprocal relations are satisfied,

$$\tau(\mathbf{r}'; \mathbf{r}) = \tau(\mathbf{r}; \mathbf{r}'), \qquad A(\mathbf{r}'; \mathbf{r}) = A(\mathbf{r}; \mathbf{r}'). \tag{13}$$

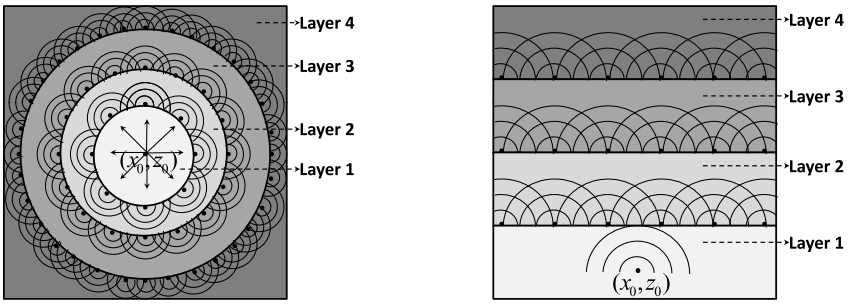

Fig. 2. 2-D demonstration of partition of a domain with 4 layers and 3 lines of secondary sources.

Denoting by $\theta^\star(\mathbf{r};\mathbf{r}')$ the angle formed by the ray direction from source $\mathbf{r}'$ to receiver $\mathbf{r}$ and the outward normal $\mathbf{n}(\mathbf{r}')$, we have $\pi+\theta(\mathbf{r}';\mathbf{r})=\theta^\star(\mathbf{r};\mathbf{r}')$. $\theta^\star(\mathbf{r};\mathbf{r}')$ is the takeoff angle of the ray from the source $\mathbf{r}'$ to the receiver $\mathbf{r}$, which is constant along each ray, that is,

$$\nabla_{\mathbf{r}}\theta^\star(\mathbf{r};\mathbf{r}')\cdot\nabla_{\mathbf{r}}\tau(\mathbf{r};\mathbf{r}')=0; \tag{14}$$

consequently,

$$\nabla_{\mathbf{r}}\cos\big(\theta^\star(\mathbf{r};\mathbf{r}')\big)\cdot\nabla_{\mathbf{r}}\tau(\mathbf{r};\mathbf{r}')=0. \tag{15}$$

Based on Eqs. (13), we need to compute $\tau(\mathbf{r};\mathbf{r}')$, $A(\mathbf{r};\mathbf{r}')$, and $\theta^\star(\mathbf{r};\mathbf{r}')$ (or $\cos(\theta^\star(\mathbf{r};\mathbf{r}'))$) only for each secondary source $\mathbf{r}'$ on $\mathcal{S}$, through the eikonal equation

$$\big|\nabla_{\mathbf{r}}\tau(\mathbf{r};\mathbf{r}')\big|=s(\mathbf{r}), \tag{16}$$

the transport equation

$$\nabla_{\mathbf{r}}\tau(\mathbf{r};\mathbf{r}')\cdot\nabla_{\mathbf{r}}A(\mathbf{r};\mathbf{r}')+\frac{1}{2}A(\mathbf{r};\mathbf{r}')\nabla^2_{\mathbf{r}}\tau(\mathbf{r};\mathbf{r}')=0, \tag{17}$$

and the transport equation (14) (or (15)), respectively.

Therefore, Eq. (12) reduces to

$$\begin{aligned}U(\mathbf{r};\mathbf{r}_0)&\approx\int_{\mathcal{S}}G(\mathbf{r};\mathbf{r}')\big\{\nabla_{\mathbf{r}'}U(\mathbf{r}';\mathbf{r}_0)\cdot\mathbf{n}(\mathbf{r}')-i\omega U(\mathbf{r}';\mathbf{r}_0)\cos\theta^\star(\mathbf{r};\mathbf{r}')s(\mathbf{r}')\big\}\,dS\\&=\int_{\mathcal{S}}\big\{G(\mathbf{r};\mathbf{r}')\big[\nabla_{\mathbf{r}'}U(\mathbf{r}';\mathbf{r}_0)\cdot\mathbf{n}(\mathbf{r}')\big]-G(\mathbf{r};\mathbf{r}')\cos\big(\theta^\star(\mathbf{r};\mathbf{r}')\big)\big[i\omega U(\mathbf{r}';\mathbf{r}_0)s(\mathbf{r}')\big]\big\}\,dS,\end{aligned} \tag{18}$$

where G takes the geometrical-optics form (11).

The validity of the above asymptotic approximation is verified in Appendix B. To compute accurate asymptotic ingredients, such as the traveltime, amplitude, and takeoff angle, we use the factorization idea to remove the upwind singularity at the source and apply the WENO based high-order Lax–Friedrichs scheme to solve the corresponding factored equations as detailed in [29,31] and Appendix A.

3.2. Huygens principle based sweeping method

Because the asymptotic Green function G as constructed by the ansatz (11) is valid only locally in an inhomogeneous medium, the integration formula (18) can only be used in a narrow layer away from the surface $\mathcal{S}$ when the wave field $U(\mathbf{r}';\mathbf{r}_0)$ is available on $\mathcal{S}$. However, since caustics will not develop close to a source in an isotropic medium, the integration formula (18) provides a vehicle for extrapolating the wave field from a source to a global domain in a layer-by-layer fashion.

Suppose that a point source $\mathbf{r}_0$ is given. Then we can construct an asymptotic Green function $U(\mathbf{r}';\mathbf{r}_0)$ by substituting computed geometrical-optics ingredients into the ansatz (2), which is valid in a local neighborhood Ω_1 of the source $\mathbf{r}_0$; consequently, $U(\mathbf{r}';\mathbf{r}_0)$ is available on $\mathcal{S}_1=\partial\Omega_1$. Letting Ω_1 be the first layer, we can set up secondary point sources on $\mathcal{S}_1$ and apply the integration formula (18) in a layer of non-zero thickness around $\mathcal{S}_1$ and outside Ω_1. This layer can be determined according to local neighborhoods of secondary sources in which the asymptotic forms of the secondary Green functions excited on $\mathcal{S}_1$ are valid, and we will provide a method for determining this narrow layer below. Once this narrow layer Ω_2 is determined, we can evaluate the integral (18) in Ω_2, so that $U(\mathbf{r}';\mathbf{r}_0)$ is available on $\mathcal{S}_2=\partial\Omega_2$. This process can be repeated so that we can sweep in a layer-by-layer fashion through the whole computational domain; see Fig. 2

The method that we use to partition the whole computational domain into layers is based on the first-arrival traveltime solution for the eikonal equation with a point-source condition. Suppose that a first-arrival traveltime table is computed for

the primary source in the whole computational domain. We would like to determine a local neighborhood of the primary source where the first-arrival traveltime is smooth; this can be done by setting up the neighborhood small enough. The resulting local neighborhood serves as the first layer. Next, we set up a few secondary sources on the boundary of this first layer, compute first-arrival traveltimes for each individual point source, and determine a local neighborhood of each secondary source where the first-arrival traveltime is smooth; combining the information of those local neighborhoods yields the second narrow layer. This process can be repeated so that all such layers cover the whole computational domain. This domain partitioning needs to be done just once. We remark that it is unnecessary to carry out this partitioning very accurately, and the first-arrival traveltimes can be computed on very coarse meshes without affecting the overall computation.

As a rule of thumb, the asymptotic Green function becomes accurate roughly three wavelengths away from the point source; this will be referred as the 3λ-rule in the sequel, and so we will use the integration formula (18) to compute the wave field at observation points which are at least three wavelengths away from the surface $\mathcal{S}$. By locating the surface $\mathcal{S}$ a little closer to the primary source, the missing portion of the wave field is actually already computed by either using the primary source or using the previous layer.

Consequently, according to the 3λ-rule, the portion of the first-arrival traveltimes corresponding to a small neighborhood of the point source is not used in the integration formula (18); similar considerations apply to the amplitude and takeoff angle. Therefore, asymptotic ingredients used in the formula (18) are smooth inside each layer according to the 3λ-rule.

In practice, however, since we are interested in designing methods with efficiency independent of frequency, we will set a fixed distance $d_f > 0$ to separate the source region from the receiver region so that the 3λ-rule is satisfied for frequency parameter ω large enough.

3.3. Theoretical foundation for Huygens sweeping methods

Concerning the theoretical foundation for the Huygens sweeping method, we have the following observations.

First, the Huygens–Kirchhoff integral is valid globally as long as the radiation boundary condition is imposed in the far field.

Second, since the eikonal equation with a point source condition has a local smooth solution except at the source point itself, the Green function is always locally well-defined in the asymptotic sense according to the geometrical optics ansatz. Moreover, assuming that the computational domain Ω is bounded, there exists a positive number $\epsilon_d > 0$ which may be very small such that the eikonal equation with any point taken from Ω as a source point has a local smooth solution in the neighborhood that is within distance ϵ_d from the point source except at the source point itself. This can be proved by contradiction.

Third, by partitioning the computational domain into circular layers such that secondary sources needed in the Huygens–Kirchhoff integral yield locally well-defined asymptotic Green functions, the proposed Huygens sweeping method will be able to proceed as designed because such circular layers always exist.

Fourth, the question remains to be answered: whether the above sweeping strategy yields a valid asymptotic solution to the Helmholtz equation under the given assumptions. This is an on-going project.

Fifth, in our current implementation, we focus on the planar sweeping strategy. Even in this much simplified setting, our planar sweeping strategy yields "full-aperture" Helmholtz solutions. The usual parabolic-type one-way wave equation cannot yield "full-aperture" Helmholtz solutions.

Sixth, in this paper, our concern is to understand the structure of the Huygens–Kirchhoff integral from an operational viewpoint and exploit it to design efficient numerical algorithms. In fact, we expect this paper to be the first of a projected series which will eventually deal with more complex situations.

3.4. Planar-layer based Huygens sweeping

To better illustrate the overall algorithm, we use the planar-layer based sweeping as a concrete example to summarize the Huygens sweeping method. Referring to Fig. 2, we have the following algorithmic framework.

Algorithm 1 *(Algorithmic sketch for Huygens sweeping method).*

- **Stage 1.** Precomputing asymptotic ingredients.
  - The domain under consideration is partitioned using the above approach.
  - For each line of secondary sources, the tables of the traveltime, amplitude and takeoff angle are computed for each secondary source, which is done on a coarser mesh. These tables need to be computed only in the corresponding layer. In practice, only the tables corresponding to a coarse set of secondary sources are computed. Then the tables on a dense set of secondary sources can be obtained by interpolations along the sources.
  - The tables for the coarse set of secondary sources are stored (on a hard drive) and can be used to construct the wave field for all high frequencies.
- **Stage 2.** Given a frequency ω, constructing the wave field layer by layer.
  - For each line of secondary sources, the tables of the traveltime, amplitude and takeoff angle are loaded from the hard drive to construct the wave field in the corresponding layer.

- For each table, the data is first interpolated onto a finer mesh to resolve the highly oscillatory nature of the wave field, and then the Huygens–Kirchhoff integral is computed with a numerical quadrature rule.
- The tables can be loaded one by one so that the memory requirement is low.
- If the sampling of secondary sources on $\mathcal{S}$ is not dense enough, we will interpolate the tables from given source locations onto the region/line-segment bounded by these source locations. This is feasible because asymptotic ingredients are also continuous functions of the source location.
- If $\mathcal{S}$ is a line, and the tables at two neighboring sources s_1 and s_2 on $\mathcal{S}$ have been precomputed, then after these two tables are loaded and interpolated onto a finer mesh, we interpolate the two tables along the source line $\mathcal{S}$ to get tables for sources on $\mathcal{S}$ between these two points.

To implement the above algorithm, we need to overcome several obstacles. One of these obstacles is how to store many tables of the traveltime and amplitude that our sweeping strategy will generate. This storage issue is critical as we are aiming at solving both the 2-D and 3-D Helmholtz equations in the high frequency regime. Another obstacle is how to carry out efficiently the matrix–vector products induced by the Huygens–Kirchhoff integration, where the matrix is dense. In the following subsections we will address these two critical issues.

3.5. Data tables and compression

To reduce data storage, we propose to compress each table into a linear combination of tensor-product based multivariate Chebyshev polynomials so that the information in each table is encoded into a small number of Chebyshev coefficients. Computationally, such compression leads to a significant storage reduction and efficient memory accesses.

When we need to use several lines of secondary sources, storage and manipulation of a large number of data tables for asymptotic ingredients can be very costly, and this is especially true in 3-D applications. To resolve this issue, we compress the data tables as done routinely in 3-D seismic imaging [1]. Recall that data tables for the asymptotic ingredients are smooth as they are recorded according to the 3λ-rule, and so such data tables are expressible in terms of Chebyshev polynomials, and because of the rapid decay of the coefficients we may disregard those terms corresponding to small spectral coefficients; the truncated Chebyshev expansion for a data table consists of only a small number of terms. As a result, the data tables are compressed.

We detail the compression process only for a 3-D traveltime table as the same can be done for amplitude and takeoff angle tables. Since an arbitrary rectangular domain can be mapped into the cube of $[-1,1]^3$, we assume that the traveltime field $\tau(\mathbf{r};\mathbf{r}_0)$ is defined on this cube for a given source point $\mathbf{r}_0=(x_0,y_0,z_0)$. Because the traveltime field $\tau(\mathbf{r};\mathbf{r}_0)$ is smooth, it can be expanded in terms of Chebyshev polynomials of the first kind [8],

$$\tau(\mathbf{r};\mathbf{r}_0)=\tau(x,y,z;\mathbf{r}_0)=\sum_{m=0}^{\infty}\sum_{n=0}^{\infty}\sum_{k=0}^{\infty}C_{mnk}(\mathbf{r}_0)T_m(x)T_n(y)T_k(z), \tag{19}$$

where T_m is Chebyshev polynomials of the first kind and of order m: $T_m(x)=\cos(m\cos^{-1}x)$; $\{C_{mnk}(\mathbf{r}_0)\}$ are spectral coefficients to be determined.

To determine these spectral coefficients $\{C_{mnk}(\mathbf{r}_0)\}$, we use the computed traveltime table which samples the smooth traveltime field on a finite-difference Cartesian mesh. This implies that we use those mesh points as collocation points in the Chebyshev expansion to determine the spectral coefficients. On the other hand, since a Chebyshev polynomial expansion is merely a Fourier cosine series in disguise [8], the spectral coefficients $\{C_{mnk}(\mathbf{r}_0)\}$ can be computed rapidly by using the fast cosine transform.

Once those spectral coefficients are available, we may truncate the Chebyshev expansion (19). According to the Chebyshev truncation theorem (see [8], page 47), the truncation error in approximating the traveltime field $\tau(\mathbf{r};\mathbf{r}_0)$ by the sum of its first few terms is bounded by the sum of the absolute values of all the neglected coefficients. Therefore, once a certain accuracy threshold is given, we can truncate the Chebyshev expansion (19) accordingly. In numerical applications, it is enough to keep only the first few coefficients among $\{C_{mnk}(\mathbf{r}_0)\}$ (see [1]). Thus,

$$\tau(\mathbf{r};\mathbf{r}_0)=\tau(x,y,z;\mathbf{r}_0)\approx\sum_{m=0}^{C_X-1}\sum_{n=0}^{C_Y-1}\sum_{k=0}^{C_Z-1}C_{mnk}(\mathbf{r}_0)T_m(x)T_n(y)T_k(z), \tag{20}$$

with C_X, C_Y and C_Z being given numbers.

$$\tilde{C}=\left\{C_{mnk}(\mathbf{r}_0):\ 0\leqslant m\leqslant C_X-1,0\leqslant n\leqslant C_Y-1,0\leqslant k\leqslant C_Z-1\right\},$$

the set of significant coefficients will be saved for later use. In practice, it is possible to reconstruct the traveltime τ with high accuracy even with a relatively high compression ratio. For 2-D experiments in Section 4, a compression ratio as high as $100:1$ can provide an accurate reconstruction of the traveltime; namely, if the traveltime is given on a mesh of size $M\times K$, C_X and C_Z can be chosen such that $\frac{M}{C_X}\cdot\frac{K}{C_Z}$ is as high as 100.

Once $\tilde{C}$ is available, the traveltime table for a given source point on any specified computational mesh can be computed by evaluating the formula (20) on that mesh. However, a naive implementation of the summation formula (20) will yield an

inefficient algorithm as it involves a triple summation. Therefore, to have an efficient summation process, we reformulate the summation process on a mesh as a product of low-rank matrices, which is detailed below by using the standard colon notation.

Algorithm 2 *(Efficient Chebyshev summation).*

- In 3-D, assuming that we want to reconstruct the traveltime on a mesh of size $M \times N \times K$ spanned by $\mathbf{X} = [-1 : h_x : 1]$, $\mathbf{Y} = [-1 : h_y : -1]$, and $\mathbf{Z} = [-1 : h_z : 1]$, the traveltime τ on this 3-D mesh is given by

$$\tau = \text{permute}\big(\text{permute}\big(\mathbf{T}_{\mathbf{X}}' C \mathbf{T}_{\mathbf{Z}}, [1\ 3\ 2]\big)\mathbf{T}_{\mathbf{Y}}, [1\ 3\ 2]\big),$$

  where C is of size $C_X \times C_Y \times C_Z$, $\mathbf{T}_{\mathbf{X}}$ of size $C_X \times M$, $\mathbf{T}_{\mathbf{Y}}$ of size $C_Y \times N$, $\mathbf{T}_{\mathbf{Z}}$ of size $C_Z \times K$, and

$$\mathbf{T}_{\mathbf{X}}(m, :) = T_{m-1}(\mathbf{X}) \quad \text{for } m = 1, \ldots, C_X,$$
$$\mathbf{T}_{\mathbf{Y}}(n, :) = T_{n-1}(\mathbf{Y}) \quad \text{for } n = 1, \ldots, C_Y,$$
$$\mathbf{T}_{\mathbf{Z}}(k, :) = T_{k-1}(\mathbf{Z}) \quad \text{for } k = 1, \ldots, C_Z,$$

  where permute(A, order) rearranges the dimensions of array A so that they are in the order specified by the vector 'order'.
- In 2-D, assuming that we want to reconstruct the traveltime τ on a 2-D mesh of size $M \times K$ spanned by $\mathbf{X} = [-1 : h_x : 1]$ and $\mathbf{Z} = [-1 : h_z : 1]$, the traveltime on this mesh is given as

$$\tau = \mathbf{T}_{\mathbf{X}}' C \mathbf{T}_{\mathbf{Z}},$$

  where C is of size $C_X \times C_Z$.

In terms of computational cost, the 2-D case of Algorithm 2 takes $O(C_X \times M \times K + C_X \times C_Z \times K)$ while the 2-D direct summation requires $O(C_X \times C_Z \times M \times K)$ operations. The 3-D case of Algorithm 2 takes $O(C_X \times M \times N \times K + C_X \times C_Y \times N \times K + C_X \times C_Y \times C_Z \times K)$ while the 3-D direct summation requires $O(C_X \times C_Y \times C_Z \times M \times N \times K)$ operations. Apparently, Algorithm 2 is more efficient than the direct summation in the sense that the proportionality constant of Algorithm 2 is smaller. Moreover, when $C_X \approx M$, or $C_Y \approx N$, or $C_Z \approx K$, Algorithm 2 will be much faster than the direct summation. At this point, we mention that Algorithm 2 is in fact equivalent to the Orszag partial summation method [8].

3.6. Discretization of Huygens–Kirchhoff integral

Now we consider how to evaluate the Huygens–Kirchhoff integral (18) efficiently. To simplify the presentation, here we will concentrate on the case that the integration surface $\mathcal{S}$ is planar (a line in 2-D and a plane in 3-D), and the case that $\mathcal{S}$ is a closed surface will be treated in a forthcoming paper.

In numerical computations, we can only handle the integral (18) on a finite, bounded domain; consequently, the integration plane $\mathcal{S}$ will be truncated according to the specified computational domain for the Helmholtz equation. Fortunately, our methodology is based on the geometrical optics, so that the computational domain of dependence is mainly determined by outgoing rays, which satisfy the Sommerfeld radiation boundary condition at infinity automatically. Thus, the truncation of $\mathcal{S}$ may slightly affect the overall accuracy of the solution near the boundary only.

After the integration domain $\mathcal{S}$ is truncated to a bounded domain $\bar{\mathcal{S}}$, we need to discretize $\bar{\mathcal{S}}$ so that a numerical quadrature rule can be applied to evaluate the integral (18). Because we are interested in computing highly oscillatory solutions, we need to specify sufficient mesh points to sample the overall solution. In principle, the minimum number of sampling points is arguably 4 to 6 mesh points per wavelength. However, when the ratio $\frac{\omega}{v}$ is large, direct finite difference methods (FDMs) or (finite element methods) FEMs usually require many more points per wavelength so as to yield accurate numerical solution because of the pollution effect [4,2]. On the other hand, since our method is based on geometrical optics, the phase (traveltime) and amplitude of the solution have been computed independent of ω so that only 4 to 6 mesh points per wavelength are required to resolve the overall solution, and this has been verified numerically.

Since, given the velocity v and the frequency parameter ω, the smallest wavelength can be estimated to be $\lambda_{\min} = 2\pi \frac{v_{\min}}{\omega}$, where $v_{\min}$ is the smallest value of v in the computational domain, we may estimate the number of waves in each direction in the bounded computational domain, and we specify 4 to 6 mesh points per wavelength in each direction accordingly. This is a rule of thumb that we use to construct the overall wave field. We remark that because of the scale separation inherent in the geometrical-optics ansatz, the asymptotic ingredients can be computed on very coarse meshes as they are independent of the frequency parameter ω. Only when we construct the overall wave field do we need to specify enough sampling points accordingly to capture each wave accurately.

Based on the above considerations, we detail the discretization of the integral (18) in the 2-D case, where the infinite line $\mathcal{S}$ is truncated to be a finite interval $\bar{\mathcal{S}}$, and the 3-D case can be dealt with similarly. Since the integral (18) is not absolutely integrable and the integrand decays very slowly, we need to introduce a window function during the truncation process. One option is to use a square window, corresponding to the sinc function in the Fourier domain; another option

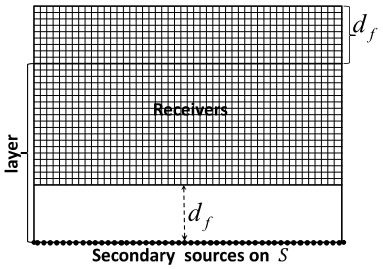

Fig. 3. Receivers and source in a layer: the separation distance d_f between the sources and receivers is independent of ω.

is to use a smooth transition window from zero to one and back to zero, and its Fourier transform will have lower side lobes than that of the square window. In our current implementation, we apply a square window function to truncate the integral, and we have found that this approach works well in practice; consequently, this approach is used throughout the paper.

Assume that the primary source location $\mathbf{r}_0$ is given and the mesh size h is chosen to satisfy the requirement that sampling points are enough to capture each wave. The trapezoidal rule applied to the integral (18) in the 2-D planar case yields

$$U(\mathbf{r};\mathbf{r}_0) \approx h{\sum_{j=1}^{M_S}}'' G(\mathbf{r};\mathbf{s}_j)\big[\nabla_{\mathbf{s}} U(\mathbf{s}_j;\mathbf{r}_0)\cdot\mathbf{n}(\mathbf{s}_j)\big] - h{\sum_{j=1}^{M_S}}'' G(\mathbf{r};\mathbf{s}_j)\cos\big(\theta^\star(\mathbf{r};\mathbf{s}_j)\big)\left[\frac{i\omega U(\mathbf{s}_j;\mathbf{r}_0)}{v(\mathbf{s}_j)}\right], \tag{21}$$

where $\sum_j''$ means that the first and last terms have a factor 1/2. Notice that in the planar case, the normal derivative of the field on the integration surface is computed by taking finite-difference derivatives directly as the field near the integration surface is known already. Here $\{\mathbf{s}_j\}_{j=1}^{M_S}$ are the secondary source points in the interval $\bar{\mathcal{S}}$, where M_S is the number of secondary sources, and they are spaced evenly with distance h on $\bar{\mathcal{S}}$. We are interested in evaluating (21) for all observation points (receivers) which are those mesh points located in a narrow layer near $\bar{\mathcal{S}}$, and we enumerate those receivers as $\{\mathbf{r}_i\}_{i=1}^{N_R}$, where N_R is the number of receivers.

We remark that in the blank region between the sources and the receivers in Fig. 3, the wave field is either due to the primary source or constructed by the Huygens–Kirchhoff integral corresponding to the previous layer. This region is chosen to be about d_f wide so that the source region is well separated from the receiver region. Moreover, although we set up secondary sources along $\bar{\mathcal{S}}$ based on the chosen numerical quadrature, in numerical implementations we will uniformly and coarsely sample these secondary sources so that only the asymptotic ingredients for those coarsely-sampled secondary sources are precomputed and the asymptotic ingredients at other secondary sources will be obtained by interpolation with respect to the source locations if needed.

To simplify the notations, we introduce the following vectors and matrices:

$$\mathbf{f} = (f_1, f_2, \ldots, f_{M_S})^t, \qquad f_j = h\nabla_{\mathbf{s}} U(\mathbf{s}_j;\mathbf{r}_0)\cdot\mathbf{n}(\mathbf{s}_j), \quad 1\leqslant j\leqslant M_S; \tag{22}$$

$$\mathbf{g} = (g_1, g_2, \ldots, g_{M_S})^t, \qquad g_j = -h\, i\omega\frac{U(\mathbf{s}_j;\mathbf{r}_0)}{v(\mathbf{s}_j)}, \quad 1\leqslant j\leqslant M_S; \tag{23}$$

$$\mathcal{G} = \big[G(\mathbf{r}_i;\mathbf{s}_j)\big]_{1\leqslant i\leqslant N_R, 1\leqslant j\leqslant M_S}; \tag{24}$$

$$\mathcal{G}_1 = \big[G(\mathbf{r}_i;\mathbf{s}_j)\cos\big(\theta^\star(\mathbf{r}_i;\mathbf{s}_j)\big)\big]_{1\leqslant i\leqslant N_R, 1\leqslant j\leqslant M_S}; \tag{25}$$

$$\mathbf{U} = \big(U(\mathbf{r}_1;\mathbf{r}_0), U(\mathbf{r}_2;\mathbf{r}_0), \ldots, U(\mathbf{r}_{N_R};\mathbf{r}_0)\big)^t. \tag{26}$$

Consequently, the integration (21) can be written as a sum of two matrix–vector products,

$$\mathbf{U} \approx \mathcal{G}\mathbf{f} + \mathcal{G}_1\mathbf{g}, \tag{27}$$

where $\mathcal{G}$ and $\mathcal{G}_1$ are both of size $N_R \times M_S$. Since the matrices $\mathcal{G}$ and $\mathcal{G}_1$ are highly oscillatory and dense as there are a large number of receivers and secondary sources, direct evaluation of the above matrix–vector products is expensive. To accelerate the evaluation process, we will use a multilevel matrix decomposition based butterfly algorithm originated in [34] and further developed in [37,53,11,13].

3.7. A butterfly algorithm for the Huygens–Kirchhoff integral

3.7.1. Low-rank separation representation of the Huygens–Kirchhoff kernel

To simplify the presentation, we will consider the following typical formulation abstracted from formula (27):

$$u(\mathbf{r}) = \sum_{\mathbf{s}\in\mathbf{X_s}} G(\mathbf{r},\mathbf{s})\mathbf{f}(\mathbf{s}), \quad \mathbf{r}\in\mathbf{X_r}\subset\Omega_{\mathbf{r}}, \tag{28}$$

where the set of receivers $\mathbf{X_r}$ is contained in the domain $\Omega_{\mathbf{r}}$, the set of sources $\mathbf{X_s}$ in the domain $\Omega_{\mathbf{s}}$, and $\mathbf{f}$ is the input; $G(\mathbf{r},\mathbf{s})$ is given as in the WKBJ ansatz,

$$G(\mathbf{r},\mathbf{s}) = A(\mathbf{r},\mathbf{s})e^{\iota\omega\tau(\mathbf{r},\mathbf{s})}, \tag{29}$$

where $A(\mathbf{r},\mathbf{s})$ and $\tau(\mathbf{r},\mathbf{s})$ are the amplitude and traveltime, respectively. To develop a butterfly algorithm for the summation (28), we will follow closely [11,13] by utilizing the low-rank separation property of G under some geometrical separation condition related to the so-called "algorithmic Fresnel number" [13]. Assuming that the sources and receivers are well separated, i.e., $dist(\Omega_{\mathbf{r}},\Omega_{\mathbf{s}}) \geqslant d_f > 0$, where d_f as mentioned above is fixed and is independent of the frequency parameter ω, we consider a pair of square boxes $B_{\mathbf{r}}$ and $B_{\mathbf{s}}$: $B_{\mathbf{r}}\subset\Omega_{\mathbf{r}}$ is centered at $\mathbf{r}_0$ with width $w(B_{\mathbf{r}})$, and $B_{\mathbf{s}}\subset\Omega_{\mathbf{s}}$ is centered at $\mathbf{s}_0$ with width $w(B_{\mathbf{s}})$. Following [11], we introduce the residual phase function $R(\mathbf{r},\mathbf{s})$ defined on $\mathbf{r}\in B_{\mathbf{r}}$ and $\mathbf{s}\in B_{\mathbf{s}}$ as

$$\tau(\mathbf{r},\mathbf{s}) - \tau(\mathbf{r},\mathbf{s}_0) - \tau(\mathbf{r}_0,\mathbf{s}) + \tau(\mathbf{r}_0,\mathbf{s}_0) \equiv R(\mathbf{r},\mathbf{s}), \tag{30}$$

so that

$$e^{\iota\omega\tau(\mathbf{r},\mathbf{s})} = e^{\iota\omega\tau(\mathbf{r},\mathbf{s}_0)}e^{\iota\omega\tau(\mathbf{r}_0,\mathbf{s})}e^{-\iota\omega\tau(\mathbf{r}_0,\mathbf{s}_0)}e^{\iota\omega R(\mathbf{r},\mathbf{s})}. \tag{31}$$

Since $\Omega_{\mathbf{s}}$ and $\Omega_{\mathbf{r}}$ are well separated, $\tau(\mathbf{r},\mathbf{s})$ can be assumed to be smooth. Consequently, applying the mean-value theorem, we have

$$\begin{aligned}\left|R(\mathbf{r},\mathbf{s})\right| &= \left|\tau(\mathbf{r},\mathbf{s}) - \tau(\mathbf{r},\mathbf{s}_0) - \tau(\mathbf{r}_0,\mathbf{s}) + \tau(\mathbf{r}_0,\mathbf{s}_0)\right| \\ &\leqslant \sup_{\mathbf{s}'\in B_{\mathbf{s}}}\sup_{\mathbf{r}'\in B_{\mathbf{r}}}\sum_{|i|=1}\sum_{|j|=1}\left|\partial_{\mathbf{r}}^i\partial_{\mathbf{s}}^j\tau(\mathbf{r}',\mathbf{s}')\right||\mathbf{r}-\mathbf{r}_0|^i|\mathbf{s}-\mathbf{s}_0|^j,\end{aligned} \tag{32}$$

where the standard multi-index notation is used. The inequality (32) implies that if the sizes of $B_{\mathbf{r}}$ and $B_{\mathbf{s}}$ satisfy the assumption that

$$w(B_{\mathbf{r}})w(B_{\mathbf{s}}) = O\left(\frac{1}{\omega}\right), \tag{33}$$

then $\omega R(\mathbf{r},\mathbf{s}) = O(1)$, so that $e^{\iota\omega R(\mathbf{r},\mathbf{s})}$ is non-oscillatory.

As proved in [11], under some mild analyticity conditions on the phase function τ, for any given accuracy requirement $\epsilon > 0$, there exists an ϵ-approximate low-rank separated representation for $e^{\iota\omega R(\mathbf{r},\mathbf{s})}$ over $B_{\mathbf{r}}\times B_{\mathbf{s}}$, where the low rank depends on ϵ only and is independent of the frequency parameter ω. Moreover, it has been shown in [11] that for a given accuracy $\epsilon > 0$, the low-rank separated representation of r_ϵ terms can be constructed by using multi-dimensional Chebyshev interpolants defined by tensor-products of one-dimensional Chebyshev interpolants of order $p_\epsilon \leqslant \log^2(\frac{1}{\epsilon})$, so that the low rank r_ϵ can be taken to be p_ϵ^d, which is independent of ω; it has been further suggested in [11]: when $w(B_{\mathbf{s}}) \leqslant O(\frac{1}{\sqrt{\omega}})$, a low-rank separated representation for $e^{\iota\omega R(\mathbf{r},\mathbf{s})}$ can be constructed by using Chebyshev interpolants for $\mathbf{s}$ in $B_{\mathbf{s}}$; when $w(B_{\mathbf{r}}) \leqslant O(\frac{1}{\sqrt{\omega}})$, a low-rank separated representation for $e^{\iota\omega R(\mathbf{r},\mathbf{s})}$ can be constructed by using Chebyshev interpolants for $\mathbf{r}$ in $B_{\mathbf{r}}$. With slight generalizations [13], such low-rank representation holds also for $A(\mathbf{r},\mathbf{s})e^{\iota\omega R(\mathbf{r},\mathbf{s})}$ as long as the amplitude $A(\mathbf{r},\mathbf{s})$ is sufficiently smooth, which is assumed in our setting.

Therefore, on $B_{\mathbf{r}}\times B_{\mathbf{s}}$, the kernel G can be rewritten as,

$$G(\mathbf{r},\mathbf{s}) = e^{\iota\omega\tau(\mathbf{r},\mathbf{s}_0)}\left[G^R(\mathbf{r},\mathbf{s})\right]e^{\iota\omega\tau(\mathbf{r}_0,\mathbf{s})}e^{-\iota\omega\tau(\mathbf{r}_0,\mathbf{s}_0)}; \quad \text{with } G^R(\mathbf{r},\mathbf{s}) \equiv A(\mathbf{r},\mathbf{s})e^{\iota\omega R(\mathbf{r},\mathbf{s})}, \tag{34}$$

where the term in the square brackets has ϵ-accurate low-rank separated representation for any given accuracy $\epsilon > 0$. Since the other factors depend on at most one variable which are in the form of separated representation already, we conclude that the kernel $G(\mathbf{r},\mathbf{s})$ on $B_{\mathbf{r}}\times B_{\mathbf{s}}$ has an approximate low-rank separated representation for any given accuracy $\epsilon > 0$, which is independent of ω.

To construct low-rank separated representation for $G(\mathbf{r},\mathbf{s})$, we start from the Chebyshev interpolants. Given a fixed positive integer p, the Chebyshev nodes of order p on $[-1,1]$ are defined as

$$\left\{x_i = \cos\left(\frac{(i-1)\pi}{p-1}\right)\right\}_{i=1}^{p},$$

which can be projected to an arbitrary interval $[a, b]$ as

$$\left\{\frac{a+b}{2}+\frac{b-a}{2}x_i\right\}_{i=1}^{p}.$$

Denote the one-dimensional Lagrangian basis function with respect to node x_i as $L(x, x_i)$, for $i = 1, \ldots, p$. Then the d-dimensional Lagrangian basis function is given as a tensor product,

$$L^d\big(\mathbf{x}, \{\mathbf{x}_i\}\big) = L\big(x^1, \{x_{i_1}^1\}\big)\cdots L\big(x^d, \{x_{i_d}^d\}\big),$$

with $\mathbf{x} \equiv (x^1, \ldots, x^d)$, where the Chebyshev nodes $\{\mathbf{x}_i\}$ are obtained as tensor products of one-dimensional nodes: $\{\mathbf{x}_i\}=\{x_{i_1}^1\} \times \{x_{i_2}^2\} \times \cdots \times \{x_{i_d}^d\}$ for $i = (i_1, i_2, \ldots, i_d)$.

Consider two square boxes $B_{\mathbf{r}}$ and $B_{\mathbf{s}}$ satisfying (33) and assume that p Chebyshev nodes are used along each direction in constructing Chebyshev interpolants. When $w(B_{\mathbf{s}}) \leqslant O(\frac{1}{\sqrt{\omega}})$, the Chebyshev interpolants for $G^R(\mathbf{r}, \mathbf{s})$ on $B_{\mathbf{s}}$ is given for $\mathbf{r} \in B_{\mathbf{r}}$,

$$G^R(\mathbf{r}, \mathbf{s}) \approx \sum_{n=1}^{d_p} G^R(\mathbf{r}, \mathbf{s}_n) L^d(\mathbf{s}, \mathbf{s}_n), \tag{35}$$

with $L^d(\mathbf{s}, \mathbf{s}_n)$ the Lagrangian basis function at the Chebyshev nodes $\{\mathbf{s}_n\}_{n=1}^{d_p}$, where $d_p \equiv p^d$; when $w(B_{\mathbf{r}}) \leqslant O(\frac{1}{\sqrt{\omega}})$, the Chebyshev interpolants for $G^R(\mathbf{r}, \mathbf{s})$ on $B_{\mathbf{r}}$ is given for $\mathbf{s} \in B_{\mathbf{s}}$,

$$G^R(\mathbf{r}, \mathbf{s}) \approx \sum_{m=1}^{d_p} L^{\mathcal{D}}(\mathbf{r}, \mathbf{r}_m) G^R(\mathbf{r}_m, \mathbf{s}), \tag{36}$$

with $L^d(\mathbf{r}, \mathbf{r}_m)$ the Lagrangian basis function at the Chebyshev nodes $\{\mathbf{r}_m\}_{m=1}^{d_p}$.

Consequently, when $w(B_{\mathbf{s}}) \leqslant O(\frac{1}{\sqrt{\omega}})$, we have the following partial summation formula for $\mathbf{r} \in B_{\mathbf{r}}$ and $\mathbf{s} \in B_{\mathbf{s}}$,

$$\begin{aligned}
u(\mathbf{r}) &= \sum_{\mathbf{s}\in B_{\mathbf{s}}} G(\mathbf{r}, \mathbf{s})\mathbf{f}(\mathbf{s}) = \sum_{\mathbf{s}\in B_{\mathbf{s}}} e^{\iota\omega\tau(\mathbf{r},\mathbf{s}_0)} G^R(\mathbf{r}, \mathbf{s}) e^{\iota\omega\tau(\mathbf{r}_0,\mathbf{s})} e^{-\iota\omega\tau(\mathbf{r}_0,\mathbf{s}_0)}\mathbf{f}(\mathbf{s}) \\
&\approx \sum_{\mathbf{s}\in B_{\mathbf{s}}} e^{\iota\omega\tau(\mathbf{r},\mathbf{s}_0)} \sum_{n=1}^{d_p} G^R(\mathbf{r}, \mathbf{s}_n) L^d(\mathbf{s}, \mathbf{s}_n) e^{\iota\omega\tau(\mathbf{r}_0,\mathbf{s})} e^{-\iota\omega\tau(\mathbf{r}_0,\mathbf{s}_0)}\mathbf{f}(\mathbf{s}) \\
&= \sum_{\mathbf{s}\in B_{\mathbf{s}}} e^{\iota\omega\tau(\mathbf{r},\mathbf{s}_0)} \sum_{n=1}^{d_p} G^R(\mathbf{r}, \mathbf{s}_n) e^{\iota\omega\tau(\mathbf{r}_0,\mathbf{s}_n)} e^{-\iota\omega\tau(\mathbf{r}_0,\mathbf{s}_0)} e^{-\iota\omega\tau(\mathbf{r}_0,\mathbf{s}_n)} L^d(\mathbf{s}, \mathbf{s}_n) e^{\iota\omega\tau(\mathbf{r}_0,\mathbf{s})}\mathbf{f}(\mathbf{s}) \\
&= \sum_{n=1}^{d_p} G(\mathbf{r}, \mathbf{s}_n) \sum_{\mathbf{s}\in B_{\mathbf{s}}} e^{-\iota\omega\tau(\mathbf{r}_0,\mathbf{s}_n)} L^d(\mathbf{s}, \mathbf{s}_n) e^{\iota\omega\tau(\mathbf{r}_0,\mathbf{s})}\mathbf{f}(\mathbf{s}) \\
&\equiv \sum_{n=1}^{d_p} G(\mathbf{r}, \mathbf{s}_n)\bar{\mathbf{f}}(\mathbf{s}_n)
\end{aligned} \tag{37}$$

with $\bar{\mathbf{f}}(\mathbf{s}_n)$ the equivalent density at equivalent sources $\mathbf{s}_n$,

$$\bar{\mathbf{f}}(\mathbf{s}_n) = \sum_{\mathbf{s}\in B_{\mathbf{s}}} e^{-\iota\omega\tau(\mathbf{r}_0,\mathbf{s}_n)} L^d(\mathbf{s}, \mathbf{s}_n) e^{\iota\omega\tau(\mathbf{r}_0,\mathbf{s})}\mathbf{f}(\mathbf{s}). \qquad \textbf{(M2M)} \tag{38}$$

When $w(B_{\mathbf{r}}) \leqslant O(\frac{1}{\sqrt{\omega}})$, we have the following partial summation formula for $\mathbf{r} \in B_{\mathbf{r}}$ and $\mathbf{s} \in B_{\mathbf{s}}$,

$$\begin{aligned}
u(\mathbf{r}) &= \sum_{\mathbf{s}\in B_{\mathbf{s}}} G(\mathbf{r}, \mathbf{s})\mathbf{f}(\mathbf{s}) = \sum_{\mathbf{s}\in B_{\mathbf{s}}} e^{\iota\omega\tau(\mathbf{r},\mathbf{s}_0)} G^R(\mathbf{r}, \mathbf{s}) e^{\iota\omega\tau(\mathbf{r}_0,\mathbf{s})} e^{-\iota\omega\tau(\mathbf{r}_0,\mathbf{s}_0)}\mathbf{f}(\mathbf{s}) \\
&\approx \sum_{\mathbf{s}\in B_{\mathbf{s}}} e^{\iota\omega\tau(\mathbf{r},\mathbf{s}_0)} \sum_{m=1}^{d_p} L^d(\mathbf{r}, \mathbf{r}_m) G^R(\mathbf{r}_m, \mathbf{s}) e^{\iota\omega\tau(\mathbf{r}_0,\mathbf{s})} e^{-\iota\omega\tau(\mathbf{r}_0,\mathbf{s}_0)}\mathbf{f}(\mathbf{s}) \\
&= \sum_{\mathbf{s}\in B_{\mathbf{s}}} e^{\iota\omega\tau(\mathbf{r},\mathbf{s}_0)} \sum_{m=1}^{d_p} L^d(\mathbf{r}, \mathbf{r}_m) e^{-\iota\omega\tau(\mathbf{r}_m,\mathbf{s}_0)} e^{\iota\omega\tau(\mathbf{r}_m,\mathbf{s}_0)} G^R(\mathbf{r}_m, \mathbf{s}) e^{\iota\omega\tau(\mathbf{r}_0,\mathbf{s})} e^{-\iota\omega\tau(\mathbf{r}_0,\mathbf{s}_0)}\mathbf{f}(\mathbf{s})
\end{aligned}$$

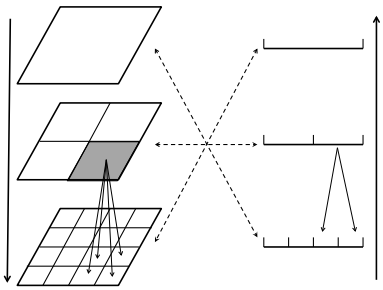

Fig. 4. Construction of cluster trees ($d = 2$) and illustration of the **Upward Pass** and **Downward Pass**. Left: receiver tree; right: source tree. For the receiver tree, each box is equally divided into 4 boxes. For the source tree, each box is equally divided into 2 boxes.

$$= \sum_{\mathbf{s}\in B_{\mathbf{s}}} e^{\iota\omega\tau(\mathbf{r},\mathbf{s}_0)} \sum_{m=1}^{d_p} L^d(\mathbf{r},\mathbf{r}_m) e^{-\iota\omega\tau(\mathbf{r}_m,\mathbf{s}_0)} G(\mathbf{r}_m,\mathbf{s})\mathbf{f}(\mathbf{s})$$

$$\equiv e^{\iota\omega\tau(\mathbf{r},\mathbf{s}_0)} \sum_{m=1}^{d_p} L^d(\mathbf{r},\mathbf{r}_m) e^{-\iota\omega\tau(\mathbf{r}_m,\mathbf{s}_0)} \bar{u}(\mathbf{r}_m), \qquad (\mathbf{L2L}) \tag{39}$$

with $\bar{u}(\mathbf{r}_m)$ being the equivalent field at equivalent receiver $\mathbf{r}_m$,

$$\bar{u}(\mathbf{r}_m) = \sum_{\mathbf{s}\in B_{\mathbf{s}}} G(\mathbf{r}_m,\mathbf{s})\mathbf{f}(\mathbf{s}). \qquad (\mathbf{M2L}) \tag{40}$$

Following the traditional terminology from the fast multipole methods [18,15,33], Eqs. (38), (40), and (39) define the **M2M** (multipole-to-multipole) operator, **M2L** (multipole-to-local) operator, and **L2L** (local-to-local) operator, respectively.

3.7.2. The butterfly algorithm

With the above setup, analogous to the butterfly algorithm in [11], we have the following algorithm which is adapted to our application.

Algorithm 3 *(The butterfly algorithm).*

1. Construct the cluster trees for both receivers and sources. Assume that the domain of receivers is $[\mathcal{L}^{\mathbf{r}}_{\min}, \mathcal{L}^{\mathbf{r}}_{\max}]^d$, and the domain of sources is $[\mathcal{L}^{\mathbf{s}}_{\min}, \mathcal{L}^{\mathbf{s}}_{\max}]^{d-1}$, where d is the dimension of the space. The domains are discretized such that the number of sampling points per wavelength is fixed, such as 4 to 6 points per wave length. For $d = 2$, the cluster tree for the receivers and sources is a quadtree and a binary tree, respectively. For $d = 3$, the cluster tree for the receivers and sources is an octree and a quadtree, respectively.
   At the root level (denoted as level 0), the boxes for both the source and receiver cluster trees are assigned to be the corresponding domain directly. Then the tree construction goes by dyadically subdividing the boxes: for an octree (quadtree, binary tree, respectively), each box is equally divided into 8 (4, 2, respectively) boxes. The construction reaches and stops at the leaf level (denoted as level L) where the size of each box is about 2 wavelengths so that approximately $O(p)$ sampling points are used along each dimension with p the order of the Chebyshev interpolants.
   Hence, except for the leaf level, each box B of an octree (quadtree, binary tree, respectively) has 8 (4, 2, respectively) children boxes, denoted as B^c, and except for the root level, each box B has a parent box, denoted as B^p. We denote the resulting two trees as T_s (the source tree) and T_r (the receiver tree), respectively. Fig. 4 shows an example of constructing the cluster trees with $d = 2$, i.e., a quadtree for the receivers and a binary tree for the sources. From now on, we will use the superscript $(\cdot)^B$ to denote the dependence on the box B.
   The butterfly algorithm traverses through the two cluster trees in the following way: for $\ell = L, \ldots, 0$, visit level ℓ in T_s and level $L - \ell$ in T_r by considering each pair $\{B_{\mathbf{r}}, B_{\mathbf{s}}\}$ with $B_{\mathbf{r}} \in T_r$ and $B_{\mathbf{s}} \in T_s$, $l(B_{\mathbf{s}}) = \ell$ and $l(B_{\mathbf{r}}) = L - \ell$, where $l(B)$ indicates the level of B in a tree.
   For example, at the root level of the receiver tree and the leaf level of the source tree, for each pair $\{B_{\mathbf{r}}, B_{\mathbf{s}}\}$,

$$w(B_{\mathbf{r}})w(B_{\mathbf{s}}) = \left(\mathcal{L}^{\mathbf{r}}_{\max} - \mathcal{L}^{\mathbf{r}}_{\min}\right) O\left(\frac{v_{\min}}{\omega}\right) = \left(\mathcal{L}^{\mathbf{r}}_{\max} - \mathcal{L}^{\mathbf{r}}_{\min}\right) v_{\min} O(1/\omega) = O(1/\omega),$$

   where $v_{\min}$ is the smallest value of the wave speed v in the computational domain. So the geometric separation condition (33) is satisfied.
   As moving downward the receiver tree by one level while simultaneously moving upward the source tree by one level, $w(B_{\mathbf{r}})$ is divided by 2 and $w(B_{\mathbf{s}})$ is doubled by 2, so $w(B_{\mathbf{r}})w(B_{\mathbf{s}}) = O(1/\omega)$ is automatically satisfied.

In the following, the equivalent sources and equivalent points are chosen as the Chebyshev nodes projected onto the corresponding box.

2. The **Upward Pass (M2M)** starts at the leaf level (level L) of the source tree T_s and ends at the level (denoted as $L_{\mathbf{s}}$) where the size of the boxes $w(B_{\mathbf{s}}) \geqslant O(\frac{1}{\sqrt{\omega}})$. Correspondingly, the level of the receiver tree T_r varies from the root level (level 0) to level $L_{\mathbf{r}} \equiv L - L_{\mathbf{s}}$.
   (1) **Initialization:** perform the **M2M** operation for all boxes ($B_{\mathbf{s}}$) at the leaf level of the source tree and all boxes ($B_{\mathbf{r}}$) at the root level of the receiver tree. For each pair $\{B_{\mathbf{r}}, B_{\mathbf{s}}\}$, anterpolate equivalent densities $\{\bar{\mathbf{f}}_n^{B_{\mathbf{r}},B_{\mathbf{s}}}\}$ at equivalent points $\{\mathbf{s}_n^{B_{\mathbf{s}}}\}$ from the given input densities $\{\mathbf{f}(\mathbf{s}^{B_{\mathbf{s}}})\}$ at the sources $\{\mathbf{s}^{B_{\mathbf{s}}}\}$:

$$\bar{\mathbf{f}}_n^{B_{\mathbf{r}},B_{\mathbf{s}}} = \sum_{\mathbf{s}^{B_{\mathbf{s}}} \in B_{\mathbf{s}}} e^{-\iota\omega\tau(\mathbf{r}_0,\mathbf{s}_n^{B_{\mathbf{s}}})} L^d\big(\mathbf{s}^{B_{\mathbf{s}}}, \mathbf{s}_n^{B_{\mathbf{s}}}\big) e^{\iota\omega\tau(\mathbf{r}_0,\mathbf{s}^{B_{\mathbf{s}}})} \mathbf{f}\big(\mathbf{s}^{B_{\mathbf{s}}}\big).$$

   (2) For l from $L-1$ to $L_{\mathbf{s}}$, perform the **M2M** operation for all boxes ($B_{\mathbf{s}}$) at level l of the source tree and all boxes ($B_{\mathbf{r}}$) at level $L-l$ of the receiver tree. For each pair $\{B_{\mathbf{r}}, B_{\mathbf{s}}\}$, anterpolate equivalent densities $\{\bar{\mathbf{f}}_n^{B_{\mathbf{r}},B_{\mathbf{s}}}\}$ at equivalent sources $\{\mathbf{s}_n^{B_{\mathbf{s}}}\}$ from equivalent densities $\{\bar{\mathbf{f}}_j^{B_{\mathbf{r}}^p,B_{\mathbf{s}}^c}\}$ at equivalent sources $\{\mathbf{s}_j^{B_{\mathbf{s}}^c}\}$ of all children clusters of $B_{\mathbf{s}}$ and the parent cluster of $B_{\mathbf{r}}$:

$$\bar{\mathbf{f}}_n^{B_{\mathbf{r}},B_{\mathbf{s}}} = \sum_c \sum_{j=1}^{d_p} e^{-\iota\omega\tau(\mathbf{r}_0,\mathbf{s}_n^{B_{\mathbf{s}}})} L^d\big(\mathbf{s}_j^{B_{\mathbf{s}}^c}, \mathbf{s}_n^{B_{\mathbf{s}}}\big) e^{\iota\omega\tau(\mathbf{r}_0,\mathbf{s}_j^{B_{\mathbf{s}}^c})} \bar{\mathbf{f}}_j^{B_{\mathbf{r}}^p,B_{\mathbf{s}}^c}.$$

3. **Switch (M2L):** at the level where the **Upward Pass** has ended (level $L_{\mathbf{s}}$ of the source tree and level $L_{\mathbf{r}}$ of the receiver tree), perform the **M2L** operation for all the boxes ($B_{\mathbf{s}}$) of the source tree and all boxes ($B_{\mathbf{r}}$) of the receiver tree. For each pair $\{B_{\mathbf{r}}, B_{\mathbf{s}}\}$, compute equivalent fields $\{\bar{u}^{B_{\mathbf{r}},B_{\mathbf{s}}}\}$ at equivalent points $\{\mathbf{r}_m^{B_{\mathbf{r}}}\}$ with equivalent densities $\{\bar{\mathbf{f}}_n^{B_{\mathbf{r}},B_{\mathbf{s}}}\}$ at equivalent sources $\{\mathbf{s}_n^{B_{\mathbf{s}}}\}$:

$$\bar{u}^{B_{\mathbf{r}},B_{\mathbf{s}}}\big(\mathbf{r}_m^{B_{\mathbf{r}}}\big) = \sum_{n=1}^{dp} G\big(\mathbf{r}_m^{B_{\mathbf{r}}}, \mathbf{s}_n^{B_{\mathbf{s}}}\big) \bar{\mathbf{f}}_n^{B_{\mathbf{r}},B_{\mathbf{s}}}.$$

4. The **Downward Pass (L2L)** starts at the level $L_{\mathbf{r}}$ of the receiver tree where the **Upward Pass** has ended and ends at level L of the receiver tree. Meanwhile, the level of the source tree varies from level $L_{\mathbf{s}}$ to level 0.
   (1) For l from $L_{\mathbf{r}}$ to $L-2$, perform the **L2L** operation for all boxes ($B_{\mathbf{r}}$) at level $l+1$ of the receivers tree and all boxes ($B_{\mathbf{s}}$) at level $L-l-1$ of the source tree: for each pair $\{B_{\mathbf{r}}, B_{\mathbf{s}}\}$, interpolate the equivalent fields $\{\bar{u}^{B_{\mathbf{r}},B_{\mathbf{s}}}(\mathbf{r}_m^{B_{\mathbf{r}}})\}$ at equivalent points $\{\mathbf{r}_m^{B_{\mathbf{r}}}\}$ from equivalent fields $\{\bar{u}^{B_{\mathbf{r}}^p,B_{\mathbf{s}}^c}(\mathbf{r}_i^{B_{\mathbf{r}}^p})\}$ at equivalent points $\{\mathbf{r}_i^{B_{\mathbf{r}}^p}\}$ of the parent level l of the receiver tree and the children level $L-l$ of the source tree:

$$\bar{u}^{B_{\mathbf{r}},B_{\mathbf{s}}}\big(\mathbf{r}_m^{B_{\mathbf{r}}}\big) = e^{\iota\omega\tau(\mathbf{r}_m^{B_{\mathbf{r}}},\mathbf{s}_0)} \sum_c \sum_{i=1}^{d_p} L^d\big(\mathbf{r}_m^{B_{\mathbf{r}}}, \mathbf{r}_i^{B_{\mathbf{r}}^p}\big) e^{-\iota\omega\tau(\mathbf{r}_i^{B_{\mathbf{r}}^p},\mathbf{s}_0)} \bar{u}^{B_{\mathbf{r}}^p,B_{\mathbf{s}}^c}\big(\mathbf{r}_i^{B_{\mathbf{r}}^p}\big)$$

   (2) Perform the **L2L** operation for all boxes ($B_{\mathbf{r}}$) at the leaf level of the receivers tree and all boxes ($B_{\mathbf{s}}$) at the root level of the source tree: for each pair $\{B_{\mathbf{r}}, B_{\mathbf{s}}\}$, interpolate the equivalent fields $\{\bar{u}^{B_{\mathbf{r}},B_{\mathbf{s}}}(\mathbf{r}^{B_{\mathbf{r}}})\}$ at all points $\{\mathbf{r}^{B_{\mathbf{r}}}\}$ from equivalent fields $\{\bar{u}^{B_{\mathbf{r}}^p,B_{\mathbf{s}}^c}(\mathbf{r}_i^{B_{\mathbf{r}}^p})\}$ at equivalent points $\{\mathbf{r}_i^{B_{\mathbf{r}}^p}\}$ of the parent level $L-1$ of the receiver tree and the children level 1 of the source tree:

$$\bar{u}^{B_{\mathbf{r}},B_{\mathbf{s}}}\big(\mathbf{r}^{B_{\mathbf{r}}}\big) = e^{\iota\omega\tau(\mathbf{r}^{B_{\mathbf{r}}},\mathbf{s}_0)} \sum_c \sum_{i=1}^{d_p} L^d\big(\mathbf{r}^{B_{\mathbf{r}}}, \mathbf{r}_i^{B_{\mathbf{r}}^p}\big) e^{-\iota\omega\tau(\mathbf{r}_i^{B_{\mathbf{r}}^p},\mathbf{s}_0)} \bar{u}^{B_{\mathbf{r}}^p,B_{\mathbf{s}}^c}\big(\mathbf{r}_i^{B_{\mathbf{r}}^p}\big)$$

5. **Termination:** at the leaf level of the receiver tree, for each box $B_{\mathbf{r}}$, sum up the equivalent fields over all the boxes of the source tree at the root level:

$$u\big(\mathbf{r}^{B_{\mathbf{r}}}\big) = \sum_{B_{\mathbf{s}}} \bar{u}^{B_{\mathbf{r}},B_{\mathbf{s}}}\big(\mathbf{r}^{B_{\mathbf{r}}}\big).$$

Remark 1. In our setting, the domain of the sources is one-dimension lower than the domain of the receivers. In the construction of the cluster trees, if the domain is not a square (or cube), at the root level, the domain is initially partitioned into square boxes (or cubic boxes) approximately. Then as moving down the tree, these boxes are equally subdivided until the leaf level is reached. Fig. 5 shows the illustration.

We analyze the complexity of the butterfly algorithm. Assume that p Chebyshev nodes are chosen in each dimension. Also assume that there are $O(n) = O(2^L)$ points in each dimension. The complexity of constructing the cluster trees is at

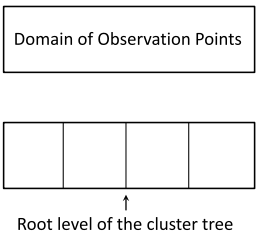

Fig. 5. Initialization of the root level ($d=2$) if the domain is not a square: top: the domain of receivers; bottom: partitioned into squares at the root level initially. Similar partition is initialized at the root level of the source tree.

most $O(n^d \log n)$. This is the worst scenario. In our applications, the domains are sampled uniformly, so the construction of the cluster trees is not time-consuming. At level l of the source tree, there are $O((2^l)^{d-1}(2^{L-l})^d)$ pairs of source-receiver boxes, and for each pair the complexity of **M2M** operator is $O(p^{d+1})$; therefore the total complexity of **Upward Pass** is

$$O\left(\sum_{l=L/2}^{L} p^{d+1}(2^l)^{d-1}(2^{L-l})^d\right) = O\left((2^L)^{d-1}p^{d+1}\sum_{l=L/2}^{L} 2^{L-l}\right) = O\left(p^{d+1}n^{d-1/2}\right),$$

where L is assumed to be even. At level $L/2$, there are $O((2^{L/2})^{d-1}(2^{L/2})^d) = O((2^L)^{d-1/2})$ pairs of source-receiver boxes, and for each pair the **M2L** operator has complexity $O(p^{2d-1})$; hence the total complexity of the **M2L** operator is

$$O\left((2^L)^{d-1/2}\right)O\left(p^{2d-1}\right) = O\left(p^{2d-1}n^{d-1/2}\right).$$

At level l of the receiver tree, there are $O((2^l)^d 2^{L-l^{d-1}}) = O((2^L)^{d-1}2^l)$ pairs of source-receiver boxes, and for each pair the **L2L** operator has complexity $O(p^{d+1})$; therefore the total complexity of **Downward Pass** is

$$O\left(\sum_{l=L/2}^{L}(2^L)^{d-1}2^l p^{d+1}\right) = O\left((2^L)^{d-1}p^{d+1}\sum_{l=L/2}^{L}2^l\right) = O\left((2^L)^{d-1}p^{d+1}2^L\right) = O\left(p^{d+1}n^d\right).$$

So, the total complexity of the butterfly algorithm is $O(p^{d+1}n^d)$ for the computation, and $O(n^d \log n)$ for constructing the cluster trees.

3.8. Complexity analysis of the overall algorithm

The overall algorithm consists of two stages. The first stage is preprocessing in which a set of tables of asymptotic ingredients, such as the traveltime, amplitude, and takeoff angle, are computed and they are further encoded into a set of tables of Chebyshev coefficients. The second stage is postprocessing in which a global wave field is constructed for a given primary source location and an arbitrary frequency ω. We will analyze the computational complexity of the two stages separately as they are independent of each other to some extent and they can be done on different meshes.

In the following analysis, we assume that the whole computational domain is partitioned into P planar layers.

3.8.1. Preprocessing: computing asymptotic ingredients

At first, we notice that since asymptotic ingredients are independent of the frequency parameter ω, these ingredients can be computed on very coarse meshes. Secondly, we notice that those asymptotic ingredients are not only continuous functions of observation points away from the source, but they are also continuous functions of the source itself. Therefore, once a set of asymptotic ingredients are computed, we can not only interpolate a given computed ingredient onto a finer mesh of observation points, but we can also interpolate a given computed ingredient with respect to the source location. These features make the cost of preprocessing step much more reasonable.

Assuming that the d-dimensional computational domain has been partitioned into a finite-difference mesh consisting of m^d sampling points, which amounts to m sampling points in each direction, we may apply higher-order Lax–Friedrichs sweeping methods detailed in Appendix A to compute those asymptotic ingredients.

Because the computational domain has been partitioned into P planar layers, we set up secondary sources on P planes, and each plane consists of m^{d-1} secondary sources. Thus the total number of secondary sources is Pm^{d-1}. Since each secondary source is used only in a local layer, the computational mesh at each secondary source should be restricted to the local layer as well. Thus the computational cost for asymptotic ingredients at each secondary source is $O(\frac{m^d}{P}\log m)$, where we have assumed that the higher-order Lax–Friedrichs sweeping method has a superlinear complexity as it is an iterative method. Furthermore, the computed asymptotic ingredients are compressed into Chebyshev polynomials and the computational cost of the compression for each computed table is $O(\frac{m^d}{P}\log m)$ by using the FFT based fast cosine

transform. Consequently, the overall computational complexity for generating asymptotic ingredients is $O(Pm^{d-1}\frac{m^d}{P}\log m) = O(m^{2d-1}\log m)$, which is analogous to the computational complexity for asymptotic ingredients in phase space [25].

Although this computational complexity seems to be high, these computed asymptotic ingredients can actually be stored and reused for constructing wave fields for many source locations and arbitrary frequencies, and in fact it is these features that make our method appealing in many applications.

On the other hand, to construct wave field in each layer, we need to recover three tables of asymptotic ingredients at some selected secondary sources on a specified mesh of size n in each direction. Recovering each table from the Chebyshev polynomials on that given mesh inside each layer is roughly $O(\frac{n^d}{P})$, where the numbers of compression coefficients are suppressed as they are considered to be small in comparison to n; see Section 3.5.

3.8.2. Postprocessing: constructing global wave fields

Given a source location and a frequency parameter ω, we have to set up a computational mesh which is fine enough to capture the overall oscillatory behavior of the wave field. According to the given smooth velocity v and the frequency parameter ω, we may estimate the smallest wavelength of the overall wave field, which allows us to estimate how many waves there are along each direction; therefore, the total number of mesh points needed to visualize the solution can be estimated accordingly as 4 to 6 mesh points are needed to capture each wavelength. Certainly, it does not hurt to use more mesh points per wavelength to capture one wave. Hence we assume that a computational mesh of n sampling points along each direction has been chosen to satisfy the above consideration, and the total number of mesh points is $N = n^d$.

Once those tabled asymptotic ingredients are available on this specified mesh inside each layer, the wave field can be constructed by the butterfly algorithm. Given accuracy $\epsilon > 0$, according to [11] we may choose $p = p_\epsilon \leqslant O(\log^2(\frac{1}{\epsilon}))$ for the order of one-dimensional Chebyshev interpolants in the butterfly algorithm so that the algorithm computes the summation with accuracy ϵ in $O(N\log N)$, where the prefactor in the complexity depends only on ϵ and is independent of ω.

Therefore, once the tables of the traveltime, amplitude and takeoff angle are available on a given mesh, our approach can be used to construct the global wave field with $O(N\log N)$ complexity for a given primary source location and for an arbitrary given frequency. Moreover, those tables can be reused for many different primary sources.

4. Numerical examples

In the following examples, we use a coarse mesh to compute geometrical-optics (GO) ingredients at the primary source, and the mesh is referred to as the GO coarse mesh. When the corresponding mesh is used for computing GO ingredients at secondary sources, the GO coarse mesh will be restricted to a neighborhood of each secondary source. Those GO ingredients are compressed into tables represented as Chebyshev coefficients. Thus, for all the numerical experiments, the GO ingredients, such as the traveltime, amplitude and takeoff angle, are given as the compressed data, and they are recovered onto the finer mesh by the Chebyshev partial summation method when needed.

Given the velocity v and the frequency parameter ω, the smallest wavelength can be estimated; accordingly we may estimate the number of waves in each direction in the bounded computational domain. In numerical experiments, in principle, 4 to 6 points per wave length for the Huygens sweeping method are sufficient for constructing the wave field accurately.

In the examples we shall where necessary assume that the units of length and of time are kilometers (km) and seconds (sec) respectively, and ω is in radians per second.

Unless otherwise stated, all computations were carried out on a single AMD Opteron core with 8 GB of RAM, which is just one of the eight cores associated with a 2009 Sunfire X4600 M2 node at High Performance Computing Center (HPCC) at MSU. All GO ingredients were computed with C codes, while wave fields were constructed with Matlab codes.

We record the error between the fields computed by the direct sum and the butterfly algorithm inside second layer. The absolute error is defined as

$$E^{abs} = \left(\int_{\Omega} \left| u^{mul}(\mathbf{r}) - u^{dir}(\mathbf{r}) \right|^2 d\mathbf{r} \right)^{1/2},$$

and the relative error is measured as

$$E^{rel} = \frac{(\int_{\Omega} |u^{mul}(\mathbf{r}) - u^{dir}(\mathbf{r})|^2 d\mathbf{r})^{1/2}}{(\int_{\Omega} |u^{dir}(\mathbf{r})|^2 \, d\mathbf{r})^{1/2}},$$

where u^{mul} and u^{dir} are the fields computed by the butterfly-algorithm based Huygens–Kirchhoff summation and the direct Huygens–Kirchhoff summation, respectively. In those tables shown, we also denote the CPU time for the butterfly-algorithm based Huygens–Kirchhoff summation as T_M seconds and the CPU time for the direct Huygens–Kirchhoff summation as T_D seconds.

For the direct finite-difference Helmholtz solver that is used in comparison with our approach, we use a nine-point stencil [21] instead of the usual five-point stencil to reduce dispersion errors and obtain more reliable wave fields; the perfectly-matched-layer (PML) absorbing boundary conditions are also imposed [7,16]; the resulting sparse linear system is solved by the sparse LU solver in the Matlab platform. One may also use newly developed compact sixth order schemes in [50] to calibrate our method, which is left as a future work.

Table 1
Sinusoidal model. CPU time for the butterfly algorithm on the whole domain. NPW: the number of points per wavelength. $p = 7, 9, 11$ Chebyshev nodes are used in each dimension. Source point is $(0.5, 0.2)$ (km).

Mesh	201×401	401×801	801×1601	1601×3201	3201×6401
$\omega/2\pi$	32	64	128	256	512
NPW	5	5	5	5	5
T_M $(p = 7)$	2.78	8.29	27.21	109.91	451.95
T_M $(p = 9)$	3.12	9.89	32.24	124.10	494.64
T_M $(p = 11)$	3.23	10.89	36.30	153.46	595.35

Table 2
Sinusoidal model. Numerical accuracy and comparisons on CPU time between the butterfly-algorithm based Huygens–Kirchhoff summation (denoted as T_M) and the direct Huygens–Kirchhoff summation (denoted as T_D) in the second layer. $p = 7, 9, 11$ Chebyshev nodes are used in each dimension. Source point is $(0.5, 0.2)$ (km).

Mesh	201×61	401×121	801×241	1601×481	3201×961	6401×1921
$\omega/2\pi$	32	64	128	256	512	1024
NPW	5	5	5	5	5	5
T_D	0.42	2.37	26.76	163.94	1544.64	9753.10
$p = 7$						
L_2 (abs)	1.03E−4	7.48E−5	5.25E−5	4.80E−5	3.71E−5	1.88E−5
L_2 (rel)	9.58E−3	9.93E−3	0.99E−2	1.27E−2	1.39E−2	9.23E−3
T_M	0.79	1.94	6.08	23.29	91.51	386.56
T_D/T_M	0.53	1.22	4.40	7.04	16.88	25.23
$p = 9$						
L_2 (abs)	1.46E−5	6.22E−6	4.13E−6	4.13E−6	2.71E−6	1.49E−6
L_2 (rel)	1.36E−3	8.25E−4	0.78E−3	1.10E−3	1.02E−3	7.3E−4
T_M	0.83	2.14	6.39	25.06	94.64	411.73
T_D/T_M	0.51	1.11	4.19	6.54	16.32	23.69
$p = 11$						
L_2 (abs)	1.72E−6	5.09E−7	2.78E−7	2.57E−7	1.65E−7	8.53E−8
L_2 (rel)	1.60E−4	6.76E−5	5.21E−5	6.85E−5	6.22E−5	4.20E−5
T_M	0.88	2.38	7.15	27.70	105.13	453.30
T_D/T_M	0.48	1.00	3.74	5.92	14.69	21.50

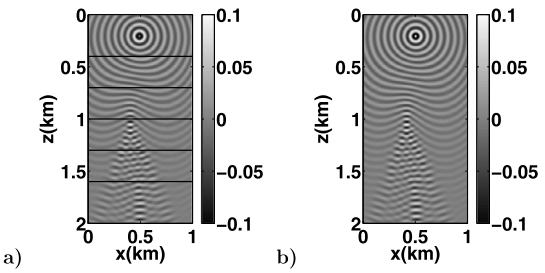

Fig. 6. Sinusoidal model. Real part of the wavefield. (a) obtained by our approach (mesh 101×201, solid black lines indicate the locations of secondary sources for each layer), and (b) obtained by direct Helmholtz solver (mesh 801×1601). $\omega = 32\pi$. Source point is $(0.5, 0.2)$ (km).

Example 1 *(Two-dimensional Sinusoidal velocity model).* In this example we construct the wave field with the following setup:

- $v = 1 + 0.2\sin(3\pi(x + 0.05))\sin(0.5\pi z)$.
- The domain is $[0, 1] \times [0, 2]$, and the GO coarse mesh is 101×201.
- Data compression ratio for asymptotic ingredients: $50 : 1$.
- The separation distance d_f between the sources and receivers is fixed as 0.1.

We use roughly 5 points per wavelength to capture the wave in both the butterfly-algorithm based Huygens–Kirchhoff summation and the direct Huygens–Kirchhoff summation. Table 1 shows the test results of CPU time for the butterfly-algorithm based summation. Table 2 shows the test results on both absolute and relative errors as well the time comparison between the butterfly summation and the direct summation.

Fig. 6 shows the contour plots of the numerical solutions obtained by our approach and the direct Helmholtz solver mentioned above with $\omega = 32\pi$. Our approach uses approximately 5 points per wavelength while the direct Helmholtz solver requires approximately 40 points per wavelength due to the dispersion error. Fig. 7 shows comparisons between the numerical solutions.

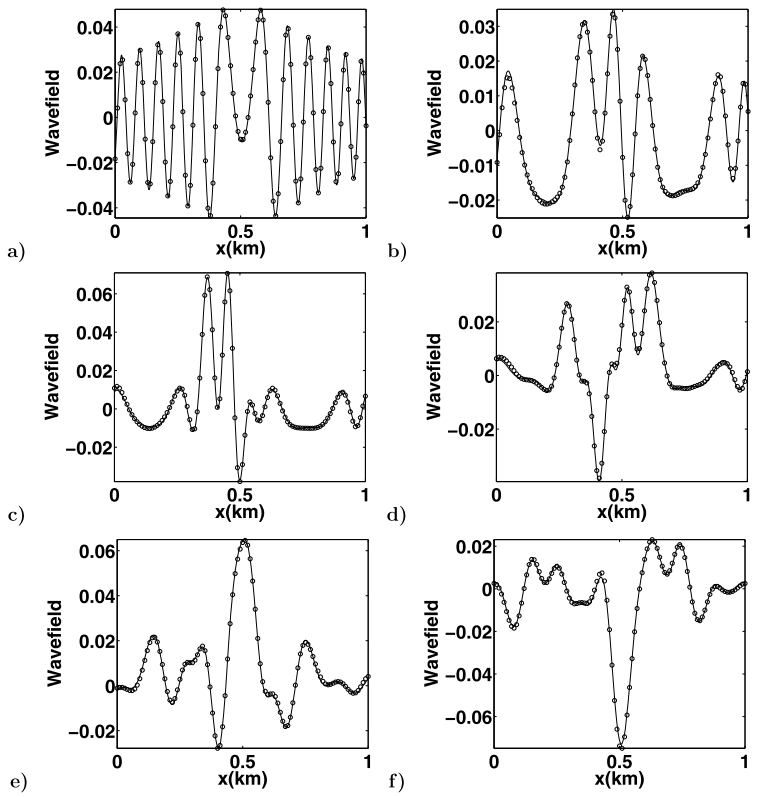

Fig. 7. Sinusoidal model. Real part of the wavefield. (a)–(f) comparisons between the solutions obtained by our approach and the direct Helmholtz solver at $z = 0.35, 0.75, 1.05, 1.35, 1.65, 1.80$ (km). Solid line: direct Helmholtz solver; Circle: our approach ($p = 9$). $\omega = 32\pi$. Source point is $(0.5, 0.2)$ (km).

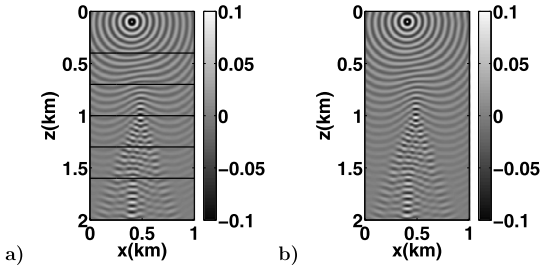

Fig. 8. Sinusoidal model. Real part of the wavefield. (a) obtained by our approach (mesh 101×201, solid black lines indicate the locations of secondary sources for each layer), and (b) obtained by direct Helmholtz solver (mesh 801×1601). $\omega = 32\pi$. Source point is $(0.4, 0.1)$ (km).

Figs. 8 and 9 show more comparisons with different source points. In such cases, the traveltime, amplitude, and takeoff angle need to be recomputed only in the first layer; for the other layers, the tables of the traveltime, amplitude, and takeoff angle corresponding to the secondary sources can be reused. Figs. 10 shows more comparisons.

Example 2 *(Two-dimensional Waveguide velocity model).* In this example we construct the wave field with the following setup:

- $v = 1 - 0.5e^{-(x-1)^2}$.
- The domain is $[0,\ 2] \times [0,\ 2]$, and the GO coarse mesh is 257×257.
- Data compression ratio for asymptotic ingredients: $100 : 1$.
- The separation distance d_f between the sources and receivers is fixed as $\frac{10}{128}$.

We use approximately 4 points per wave length to represent the wave in both the butterfly-algorithm based Huygens–Kirchhoff summation and the direct Huygens–Kirchhoff summation. Table 3 shows the test results of CPU time for the butterfly-algorithm based summation. Table 4 shows the test results on both absolute and relative errors as well the time comparison between the butterfly summation and the direct summation. Table 3 shows the test results of CPU time. Table 4 shows the test results on both absolute and relative errors.

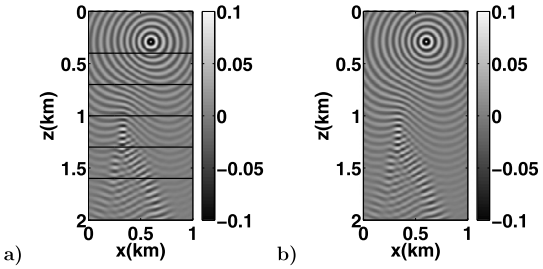

Fig. 9. Sinusoidal model. Real part of the wavefield. (a) obtained by our approach (mesh 101×201, solid black lines indicate the locations of secondary sources for each layer), and (b) obtained by direct Helmholtz solver (mesh 801×1601). $\omega = 32\pi$. Source point is $(0.6, 0.3)$ (km).

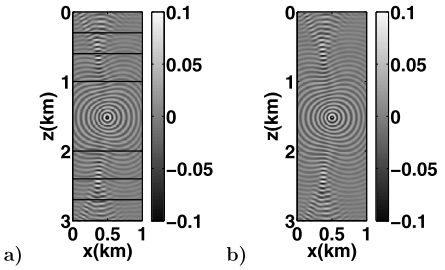

Fig. 10. Sinusoidal model. Real part of the wavefield. (a) obtained by our approach (mesh 101×301, solid black lines indicate the locations of secondary sources for each layer), and (b) obtained by direct Helmholtz solver (mesh 801×2401). $\omega = 32\pi$. Source point is $(0.5, 1.5)$ (km).

Table 3
Waveguide model. NPW: the number of points per wavelength. CPU time for the butterfly algorithm. $p = 7, 9, 11$ Chebyshev nodes are used in each dimension.

Mesh	513×513	1025×1025	2049×2049	4097×4097
$\omega/2\pi$	32	64	128	256
NPW	4	4	4	4
T_M $(p = 7)$	16.21	52.12	202.23	751.99
T_M $(p = 9)$	18.30	59.13	227.26	858.56
T_M $(p = 11)$	24.01	67.69	254.95	978.24

Table 4
Waveguide model. NPW: the number of points per wavelength. Numerical accuracy and comparisons on CPU time between the butterfly-algorithm based Huygens–Kirchhoff summation (denoted as T_M) and the direct Huygens–Kirchhoff summation (denoted as T_D) in the second layer. $p = 7, 9, 11$ Chebyshev nodes are used in each dimension.

Mesh	513×153	1025×305	2049×609	4097×1217	8193×2433
$\omega/2\pi$	32	64	128	256	512
NPW	4	4	4	4	4
T_D	2.76	19.64	175.89	1356.28	10295.74
$p = 7$					
L_2 (abs)	8.35E−4	6.00E−4	5.85E−4	4.34E−4	3.72E−4
L_2 (rel)	7.39E−2	7.54E−2	1.04E−1	1.09E−1	1.33E−1
T_M	3.16	8.72	34.17	119.02	453.21
T_D/T_M	0.87	2.25	5.15	11.40	22.72
$p = 9$					
L_2 (abs)	2.07E−4	9.20E−5	1.11E−4	7.67E−5	7.20E−5
L_2 (rel)	1.83E−2	1.16E−2	1.98E−2	1.93E−2	2.57E−2
T_M	3.63	9.88	38.48	132.11	500.25
T_D/T_M	0.76	2.00	4.57	10.27	20.58
$p = 11$					
L_2 (abs)	4.38E−5	1.33E−5	1.98E−5	1.30E−5	1.18E−5
L_2 (rel)	3.88E−3	1.67E−3	3.52E−3	3.27E−3	4.23E−3
T_M	3.97	10.90	42.70	146.51	567.50
T_D/T_M	0.70	1.80	4.12	9.26	18.14

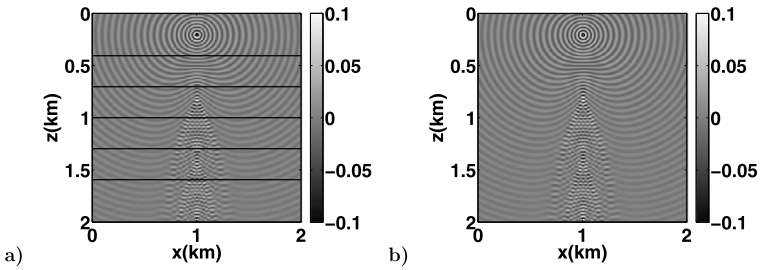

Fig. 11. Waveguide model. Real part of the wavefield. (a) obtained by our approach (mesh 257×257, solid black lines indicate the locations of secondary sources for each layer), and (b) obtained by direct Helmholtz solver (mesh 2561×2561). $\omega = 32\pi$.

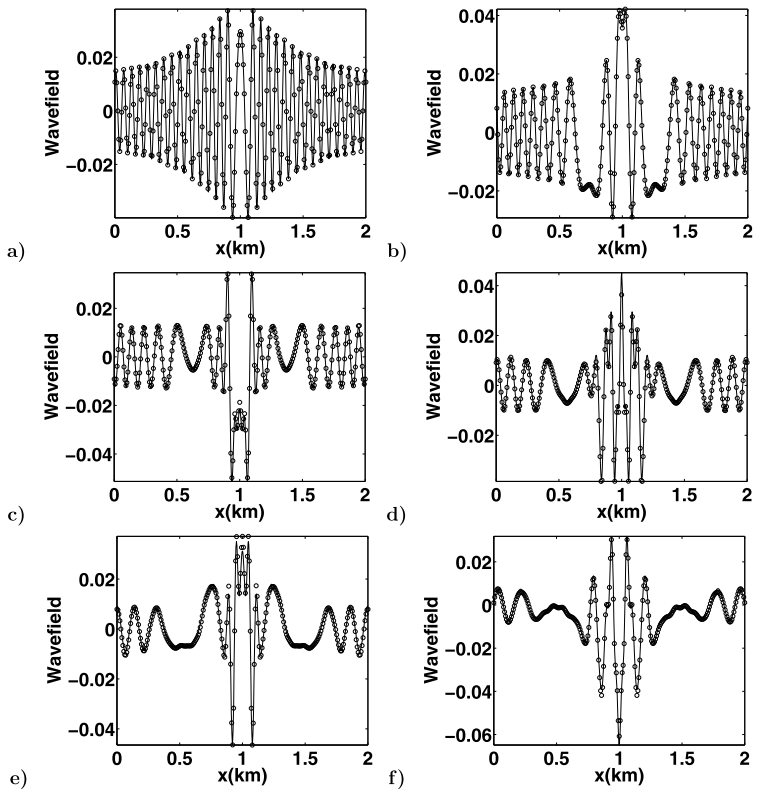

Fig. 12. Waveguide model. Real part of the wavefield. (a)-(f) comparisons between the solutions obtained by our approach and the direct Helmholtz solver at $z = \frac{47}{128}, \frac{77}{128}, \frac{115}{128}, \frac{153}{128}, \frac{191}{128}, \frac{229}{128}$ (km). Solid line: direct Helmholtz solver; Circle: our approach ($p = 9$). $\omega = 32\pi$.

Fig. 11 shows the contour plots of the numerical solutions obtained by our approach and the direct Helmholtz solver mentioned above with $\omega = 32\pi$. Our approach uses approximately 4 points per wave length, while the direct Helmholtz solver needs approximately 40 points per wavelength. Fig. 12 shows comparisons between the numerical solutions. As shown in Figs. 12(c), (e), it seems that the difference between the asymptotic and direct solution is more pronounced at caustics; although the underlying mechanism is not completely clear to us, there are at least two possible reasons: the first one is that the direct solution is not accurate enough due to numerical dispersion at high frequencies, and the second one is that there are an infinite number of rays passing through a caustic while our discretized algorithm is only able to capture a finite number of such rays.

Example 3 *(Three-dimensional Vinje velocity model).* In this 3-D example we construct the wave field with the following setup:

- $v = 3.0 - 1.75e^{(-((x-1)^2+(y-1)^2+(z-1)^2)/0.64)}$.
- The domain is $[0, 2] \times [0, 2] \times [0, 2]$, and the GO coarse mesh is $51 \times 51 \times 51$.
- Data compression ratio for asymptotic ingredients: $27 : 1$.
- The separation distance d between the sources and receivers is fixed as 0.1.

Table 5
Vinje model. NPW: the number of points per wavelength. CPU time for the butterfly-algorithm based Huygens–Kirchhoff summation. $p = 7, 9, 11$ Chebyshev nodes are used in each dimension. Source point is $(1.0, 1.0, 0.2)$ (km).

Mesh	$101 \times 101 \times 101$	$201 \times 201 \times 201$
$\omega/2\pi$	16	32
NPW	5	5
T_M $(p = 7)$	197.10	902.06
T_M $(p = 9)$	338.19	1440.48
T_M $(p = 11)$	558.69	2360.37

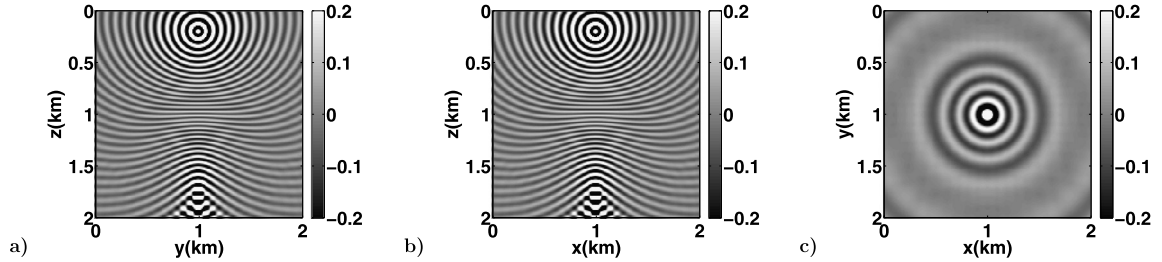

Fig. 13. Vinje model. Real part of the wavefield at $x = 1.0$ (km), $y = 1.0$ (km), and $z = 1.8$ (km). (a)–(c) obtained by our approach. $\omega = 64\pi$. Mesh $201 \times 201 \times 201$. Source point is $(1.0, 1.0, 0.2)$ (km).

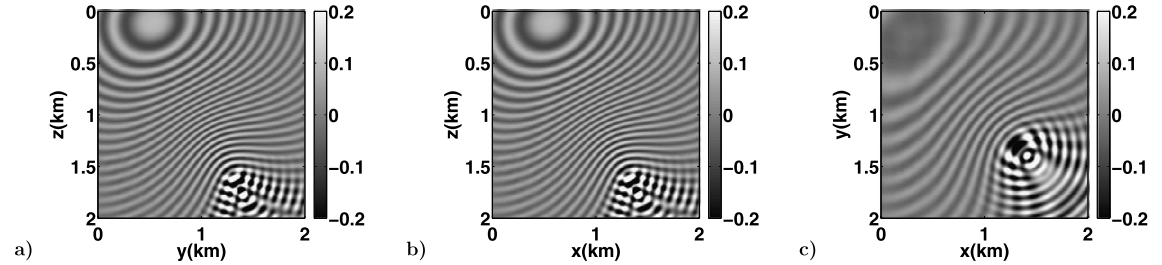

Fig. 14. Vinje model. Real part of the wavefield at $x = 1.4$ (km), $y = 1.4$ (km), and $z = 1.8$ (km). (a)–(c) obtained by our approach. $\omega = 64\pi$. Mesh $201 \times 201 \times 201$. Source point is $(0.6, 0.6, 0.24)$ (km).

We use approximately 4 points per wave length to represent the wave in the butterfly algorithm. The secondary sources are placed at $z = 1.2$ (km). Table 5 shows the test results of CPU time.

Figs. 13 and 14 show the contour plots of the numerical solutions obtained by our approach for different primary sources.

5. Conclusions

We have proposed a new geometrical-optics method, the fast Huygens-sweeping method, for computing the Green functions of the Helmholtz equations in inhomogeneous media in high frequency regime. The new method utilizes the Huygens–Kirchhoff integral to integrate many locally valid asymptotic Green functions into a globally valid asymptotic Green functions. To accelerate the Huygens–Kirchhoff integration process, we have used the butterfly algorithm to speed up the Huygens–Kirchhoff summation. The new method has nearly optimal complexity in constructing high frequency waves for a given source point and a given frequency parameter provided that asymptotic ingredients are precomputed. Numerical examples demonstrate the performance, efficiency, and accuracy of the new method.

We believe that the proposed methodology can be applied to many different problems, such as the Maxwell equations, scattering problems and inverse problems, which constitute ongoing projects.

Acknowledgements

Jianliang Qian is partially supported by NSF 1115363 and 1222368.

Appendix A. Factorization of takeoff angle

Based on the traveltime field computed with the eikonal equation (3) and (9), one can compute the takeoff angle $\theta^\star$ from the transport equation (14). Analogous to the traveltime τ, the takeoff angle $\theta^\star$ also has an upwind singularity at the source $\mathbf{r}_0$. In order to overcome this difficulty, we once again appeal to the factorization idea [29,31,30].

Since we need $\cos(\theta^\star)$ in the Huygens–Kirchhoff integral, we compute $\cos(\theta^\star)$ directly. Therefore, we consider $e^{\cos(\theta^\star)}$ which satisfies the following transport equation,

$$\nabla e^{\cos(\theta^\star)} \cdot \nabla \tau = 0. \tag{A.1}$$

Then $\cos(\theta^\star)$ can be recovered as $\log(e^{\cos(\theta^\star)})$. To remove the upwind singularity at the source, we decompose

$$e^{\cos(\theta^\star)} = e^{\cos(\theta_0^\star)} F,$$

where $\theta_0^\star$ is the takeoff angle corresponding to the constant velocity v_0 and

$$\nabla e^{\cos(\theta_0^\star)} \cdot \nabla \tau_0 = 0.$$

Denoting $E = e^{\cos(\theta^\star)}$ and $E_0 = e^{\cos(\theta_0^\star)}$, we have the factored equation,

$$\nabla \tau_0 \cdot \nabla E_0 F + E_0 (\nabla \tau_0 u + \tau_0 \nabla u) \cdot \nabla F = 0, \tag{A.2}$$

where u is defined in the factorization of $\tau = \tau_0 u$.

With high order accurate traveltime τ, one can compute F with the WENO based high order Lax–Friedrichs scheme [29,31], yielding highly accurate $e^{\cos(\theta^\star)}$ and $\cos(\theta^\star)$.

Appendix B. Verification of Huygens–Kirchhoff integral

We derive the ray-theoretic approximation to the Huygens–Kirchhoff integral in 2-D. Assume that we have

$$U = u e^{i\omega\phi}, \tag{B.1}$$

$$G = g e^{i\omega\tau}, \tag{B.2}$$

where U is the wave field, G is the Green function, u and g are the amplitudes, ϕ and τ are the phase functions.

Then we can rewrite the Huygens–Kirchhoff integral as follows [9],

$$\begin{aligned} U(x,z) \sim & \int_{\mathcal{S}} g(x,z;x',z') e^{i\omega\tau(x,z;x',z')} \vec{\mathbf{n}} \cdot \nabla \left(u(x',z') e^{i\omega\phi(x',z')} \right) \\ & - u(x',z') e^{i\omega\phi(x',z')} \vec{\mathbf{n}} \cdot \nabla \left(g(x,z;x',z') e^{i\omega\tau(x,z;x',z')} \right) dS \\ \sim & \int_{\mathcal{S}} g e^{i\omega\tau} i\omega (\vec{\mathbf{n}} \cdot \nabla\phi) u e^{i\omega\phi} - u e^{i\omega\phi} i\omega (\vec{\mathbf{n}} \cdot \nabla\tau) g e^{i\omega\tau} \, dS \\ = & \int_{\mathcal{S}} i\omega u g e^{i\omega(\tau+\phi)} (\vec{\mathbf{n}} \cdot \nabla\phi - \vec{\mathbf{n}} \cdot \nabla\tau) \, dS, \end{aligned} \tag{B.3}$$

where the integration is with respect to (x', z') over $\mathcal{S}$, and $\mathbf{n}$ is the outward normal to $\mathcal{S}$ which encloses (x, z).

Note that

$$\nabla\phi = \frac{1}{v} \vec{\mathbf{n}} \cdot \vec{t}_U, \tag{B.4}$$

where $\vec{t}_U$ is the unit tangent to the ray at (x', z') belonging to the field U, and that

$$\nabla\tau = -\frac{1}{v} \vec{\mathbf{n}} \cdot \vec{t}_G, \tag{B.5}$$

where $\vec{t}_G$ is the ray at (x', z') belonging to the field G. The minus sign is due to the differentiation at the lower limit of the integration in

$$\tau(x,z;x',z') = \int_{(x',z')}^{(x,z)} d\tau = \int_{(x',z')}^{(x,z)} \frac{1}{v} \, dl. \tag{B.6}$$

Here the integration is along the ray from (x', z') to (x, z).

Now we can rewrite (B.3) as

$$U(x,z) = i\omega \int_{\mathcal{S}} \frac{u(x',z') g(x,z;x',z')}{v(x,z)} (\vec{\mathbf{n}} \cdot \vec{t}_U + \vec{\mathbf{n}} \cdot \vec{t}_G) e^{i\omega(\phi(x',z') + \tau(x,z;x',z'))} \, dS. \tag{B.7}$$

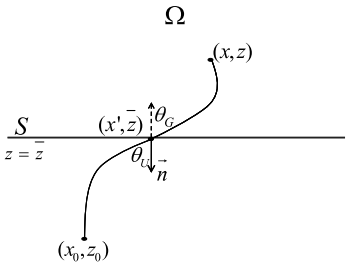

Fig. 15. Points of stationary phase.

Assume that the surface $\mathcal{S}$ is an infinite plane $z = \bar{z}$ (constant) and the outward normal

$$\vec{\mathbf{n}} = (0, -1)^t. \tag{B.8}$$

So,

$$U(x, z) = -i\omega \int_{-\infty}^{\infty} \frac{u(x', \bar{z}) g(x, z; x', \bar{z})}{v(x', \bar{z})} (\cos\theta_U + \cos\theta_G) e^{i\omega(\phi(x', \bar{z}) + \tau(x, z; x', \bar{z}))}\, dx', \tag{B.9}$$

where the angle θ_U is the angle the ray for U through $(x', \bar{z})$ makes with the z-direction and θ_G is the angle the ray for G from $(x', \bar{z})$ to (x, z) makes with the z-direction (see Fig. 15).

If there are any points of stationary phase in (B.9), then the main contribution to the integral will come from the neighborhood of these points.

At a point of stationary phase, we have

$$\frac{\partial}{\partial x'}\left(\phi\left(x', \bar{z}\right) + \tau\left(x, z; x', \bar{z}\right)\right) = 0, \tag{B.10}$$

yielding

$$\hat{\vec{x}} \cdot \vec{t}_U - \hat{\vec{x}} \cdot \vec{t}_G = 0, \tag{B.11}$$

where $\hat{\vec{x}}$ is the unit vector in the x-direction. So,

$$\sin\theta_U = \sin\theta_G. \tag{B.12}$$

This shows that the U-ray and the G-ray join up at points of stationary phase to form one continuous ray. If, however, there are several points of stationary phase close together, we cannot use the usual stationary-phase formula, but the integration is still valid. The integral will be truncated to a finite interval, and this will cause inaccuracy if any stationary-phase points are near an end point.

References

[1] T. Alkhalifah, Efficient traveltime compression for 3D prestack Kirchhoff migration, Geophys. Prospect. 69 (1) (2011) 1–9.
[2] I.M. Babuška, S.A. Sauter, Is the pollution effect of the FEM avoidable for the Helmholtz equation considering high wave numbers?, SIAM Rev. 42 (2000) 451–484.
[3] B. Baker, E.T. Copson, The Mathematical Theory of Huygens' Principle, AMS Chelsea Publishing, 1987.
[4] A. Bayliss, C.I. Goldstein, E. Turkel, On accuracy conditions for the numerical computation of waves, J. Comput. Phys. 59 (1985) 396–404.
[5] J.-D. Benamou, Direct solution of multi-valued phase-space solutions for Hamilton–Jacobi equations, Commun. Pure Appl. Math. 52 (1999) 1443–1475.
[6] J.D. Benamou, An introduction to Eulerian geometrical optics (1992–2002), J. Sci. Comput. 19 (2003) 63–93.
[7] J.-P. Berenger, A perfectly matched layer for the absorption of electromagnetic waves, J. Comput. Phys. 114 (1994) 185–200.
[8] J.P. Boyd, Chebyshev and Fourier Spectral Methods, second edition, Dover, New York, 2001.
[9] R. Burridge, The reflexion of high-frequency sound in a liquid sphere, Proc. R. Soc. Ser. A 270 (1962) 144–154.
[10] R. Burridge, Asymptotic evaluation of integrals related to time-dependent fields near caustics, SIAM J. Appl. Math. 55 (1995) 390–409.
[11] E. Candes, L. Demanet, L. Ying, A fast butterfly algorithm for the computation of Fourier integral operators, Multiscale Model. Simul. 7 (2009) 1727–1750.
[12] V. Cerveny, M. Popov, I. Psencik, Computation of wave fields in inhomogeneous media-Gaussian beam approach, Geophys. J. R. Astron. Soc. 70 (1982) 109–128.
[13] L. Demanet, M. Ferrara, N. Maxwell, J. Poulson, L. Ying, A butterfly algorithm for synthetic aperture radar imaging, SIAM J. Imaging Sci. 5 (2012) 203–243.
[14] B. Engquist, O. Runborg, Computational high frequency wave propagation, Acta Numer. 12 (2003) 181–266.
[15] B. Engquist, L. Ying, Fast directional multilevel algorithms for oscillatory kernels, SIAM J. Sci. Comput. 29 (4) (2007) 1710–1737.
[16] B. Engquist, L. Ying, Sweeping preconditioner for the Helmholtz equation: moving perfectly matched layers, SIAM J. Multiscale Model. Simul. 9 (2) (2011) 686–710.
[17] S. Fomel, S. Luo, H.-K. Zhao, Fast sweeping method for the factored eikonal equation, J. Comput. Phys. 228 (17) (2009) 6440–6455.
[18] L. Greengard, V. Rokhlin, A fast algorithm for particle simulations, J. Comput. Phys. 73 (1987) 325–348.

[19] G.S. Jiang, D. Peng, Weighted ENO schemes for Hamilton–Jacobi equations, SIAM J. Sci. Comput. 21 (2000) 2126–2143.
[20] G.S. Jiang, C.W. Shu, Efficient implementation of weighted ENO schemes, J. Comput. Phys. 126 (1996) 202–228.
[21] C.-H. Jo, C. Shin, J.H. Suh, An optimal 9-point, finite-difference, frequency-space, 2-D scalar wave extrapolator, Geophysics 61 (2) (1996) 529–537.
[22] C.Y. Kao, S. Osher, J. Qian, Lax–Friedrichs sweeping schemes for static Hamilton–Jacobi equations, J. Comput. Phys. 196 (2004) 367–391.
[23] S. Leung, J. Qian, Eulerian Gaussian beam methods for Schrödinger equations in the semi-classical regime, J. Comput. Phys. 228 (2009) 2951–2977.
[24] S. Leung, J. Qian, The backward phase flow and FBI-transform-based Eulerian Gaussian beams for the Schrödinger equation, J. Comput. Phys. 229 (2010) 8888–8917.
[25] S. Leung, J. Qian, R. Burridge, Eulerian Gaussian beams for high-frequency wave propagation, Geophysics 72 (2007) SM61–SM76.
[26] P.-L. Lions, Generalized Solutions of Hamilton–Jacobi Equations, Pitman, Boston, 1982.
[27] X.D. Liu, S.J. Osher, T. Chan, Weighted Essentially NonOscillatory schemes, J. Comput. Phys. 115 (1994) 200–212.
[28] D. Ludwig, Uniform asymptotic expansions at a caustic, Commun. Pure Appl. Math. 19 (1966) 215–250.
[29] S. Luo, J. Qian, Factored singularities and high-order Lax–Friedrichs sweeping schemes for point-source traveltimes and amplitudes, J. Comput. Phys. 230 (2011) 4742–4755.
[30] S. Luo, J. Qian, Fast sweeping methods for factored anisotropic eikonal equations: multiplicative and additive factors, J. Sci. Comput. 52 (2012) 360–382.
[31] S. Luo, J. Qian, H.-K. Zhao, Higher-order schemes for 3-D traveltimes and amplitudes, Geophysics 77 (2012) T47–T56.
[32] V.P. Maslov, M.V. Fedoriuk, Semi-Classical Approximation in Quantum Mechanics, D. Reidel Publishing Company, 1981.
[33] M. Messner, M. Schanz, E. Darve, Fast directional multilevel summation for oscillatory kernels based on Chebyshev interpolation, J. Comput. Phys. 231 (2012) 1175–1196.
[34] E. Michielssen, A. Boag, A multilevel matrix decomposition algorithm for analysing scattering from large structures, IEEE Trans. Antennas Propag. 44 (1996) 1086–1093.
[35] J. Milnor, Morse Theory, Ann. Math., vol. 51, Princeton University Press, 1973.
[36] M. Motamed, O. Runborg, Taylor expansion and discretization errors in Gaussian beam superposition, Wave Motion 47 (2010) 421–439.
[37] M. O'Neil, V. Rokhlin, A New Class of Analysis-Based Fast Transforms, 2007.
[38] S.J. Osher, C.W. Shu, High-order Essentially NonOscillatory schemes for Hamilton–Jacobi equations, SIAM J. Numer. Anal. 28 (1991) 907–922.
[39] A. Pica, Fast and accurate finite-difference solutions of the 3D eikonal equation parametrized in celerity, in: 67th Ann. Int. Mtg. Soc. Expl. Geophys., 1997, pp. 1774–1777.
[40] M.M. Popov, A new method of computation of wave fields using Gaussian beams, Wave Motion 4 (1982) 85–97.
[41] J. Qian, W.W. Symes, An adaptive finite-difference method for traveltimes and amplitudes, Geophysics 67 (2002) 167–176.
[42] J. Qian, L. Ying, Fast Gaussian wavepacket transforms and Gaussian beams for the Schrödinger equation, J. Comput. Phys. 229 (2010) 7848–7873.
[43] J. Qian, L. Ying, Fast multiscale Gaussian wavepacket transforms and multiscale Gaussian beams for the wave equation, Multiscale Model. Simul. 8 (2010) 1803–1837.
[44] J. Ralston, Gaussian beams and the propagation of singularities, in: W. Littman (Ed.), Studies in Partial Differential Equations MAA, in: Stud. Math., vol. 23, 1983, pp. 206–248.
[45] S. Serna, J. Qian, A stopping criterion for higher-order sweeping schemes for static Hamilton–Jacobi equations, J. Comput. Math. 28 (2010) 552–568.
[46] W.W. Symes, J. Qian, A slowness matching Eulerian method for multivalued solutions of eikonal equations, J. Sci. Comput. 19 (2003) 501–526.
[47] N. Tanushev, B. Engquist, R. Tsai, Gaussian beam decomposition of high frequency wave fields, J. Comput. Phys. 228 (2009) 8856–8871.
[48] N. Tanushev, J. Qian, J. Ralston, Mountain waves and Gaussian beams, Multiscale Model. Simul. 6 (2007) 688–709.
[49] R. Tsai, L.-T. Cheng, S.J. Osher, H.K. Zhao, Fast sweeping method for a class of Hamilton–Jacobi equations, SIAM J. Numer. Anal. 41 (2003) 673–694.
[50] E. Turkel, D. Gordon, R. Gordon, S. Tsynkov, Compact 2D and 3D sixth order schemes for the Helmholtz equation with variable wave number, J. Comput. Phys. 232 (2013) 272–287.
[51] B.S. White, The stochastic caustic, SIAM J. Appl. Math. 44 (1984) 127–149.
[52] B.S. White, A. Norris, A. Bayliss, R. Burridge, Some remarks on the Gaussian beam summation method, Geophys. J. R. Astron. Soc. 89 (1987) 579–636.
[53] L. Ying, Sparse Fourier transform via butterfly algorithm, SIAM J. Sci. Comput. 31 (2009) 1678–1694.
[54] L. Zhang, J.W. Rector, G.M. Hoversten, Eikonal solver in the celerity domain, Geophys. J. Int. 162 (2005) 1–8.
[55] 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.
[56] H.K. Zhao, Fast sweeping method for eikonal equations, Math. Comput. 74 (2005) 603–627.