A FAST BUTTERFLY-COMPRESSED HADAMARD-BABICH INTEGRATOR FOR HIGH-FREQUENCY HELMHOLTZ EQUATIONS IN INHOMOGENEOUS MEDIA WITH ARBITRARY SOURCES

YANG LIU*, JIAN SONG †, ROBERT BURRIDGE ‡, AND JIANLIANG QIAN†

Abstract. We present a butterfly-compressed representation of the Hadamard-Babich (HB) ansatz for the Green's function of the high-frequency Helmholtz equation in smooth inhomogeneous media. For a computational domain discretized with N_v discretization cells, the proposed algorithm first solves and tabulates the phase and HB coefficients via eikonal and transport equations with observation points and point sources located at the Chebyshev nodes using a set of much coarser computation grids, and then butterfly compresses the resulting HB interactions from all N_v cell centers to each other. The overall CPU time and memory requirement scale as $O(N_v \log^2 N_v)$ for any bounded 2D domains with arbitrary excitation sources. A direct extension of this scheme to bounded 3D domains yields an $O(N_v^{4/3})$ CPU complexity, which can be further reduced to quasi-linear complexities with proposed remedies. The scheme can also efficiently handle scattering problems involving inclusions in inhomogeneous media. Although the current construction of our HB integrator does not accommodate caustics, the resulting HB integrator itself can be applied to certain sources, such as concave-shaped sources, to produce caustic effects. Compared to finite-difference frequency-domain (FDFD) methods, the proposed HB integrator is free of numerical dispersion and requires fewer discretization points per wavelength. As a result, it can solve wave-propagation problems well beyond the capability of existing solvers. Remarkably, the proposed scheme can accurately model wave propagation in 2D domains with 640 wavelengths per direction and in 3D domains with 54 wavelengths per direction on a state-the-art supercomputer at Lawrence Berkeley National Laboratory.

Key word. butterfly algorithm, high-frequency waves, inhomogeneous Helmholtz equation, Hadamard-Babich ansatz, Lax-Friedrichs weighted non-oscillatory (WENO) schemes, Chebyshev interpolation, fast solvers, scattering problem, caustics, eikonal equation, transport equation, finite-difference frequency domain (FDFD) methods

AMS subject classifications. 15A23, 65F50, 65R10, 65R20

1. Introduction.

We are interested in finding the solution to the high-frequency Helmholtz equation with variable refractive index $n(\mathbf{r})$ subject to an arbitrary source:

$$[\Delta + \omega^2 n^2(\mathbf{r})]u = -s(\mathbf{r}) \quad \text{in} \quad R^d, \tag{1.1}$$

where

$$\Delta = \sum_{i=1}^{d} \frac{\partial^2}{\partial x_i^2}, \quad \mathbf{r} = [x_1, x_2, \dots, x_d]^T, \tag{1.2}$$

$n(\mathbf{r})$ is the index of refraction (or the slowness function), $s(\mathbf{r})$ is a generic source function with compact support in a bounded domain $V \subset R^d$, ω is a large angular frequency, d is the dimension, and the Sommerfeld radiation condition is imposed at infinity. When the source is a point source, $s(\mathbf{r}) = \delta(\mathbf{r}, \mathbf{r}_0)$, the point-source solution of (1.1) is the Green's function $G(\mathbf{r}, \mathbf{r}_0)$ with source location $\mathbf{r}_0$. Assume that we are given a volumetric discretization of the computational domain V with N_v degrees-of-freedom (DOFs) so that the Shannon sampling principle, $\omega h = O(1)$, is obeyed, where

*Applied Mathematics and Computational Research Division, Lawrence Berkeley National Laboratory, Berkeley, CA, USA.
Email: `liuyangzhuan@lbl.gov`

†Department of Mathematics, Michigan State University, East Lansing, MI 48824, USA. Email: `{songji12,jqian}@msu.edu`

‡Department of Mathematics and Statistics, University of New Mexico, Albuquerque, NM 87131, USA. Email: `burridge137@gmail.com`

h is the volumetric mesh size, implying that $N_v = O(\omega^d)$. Accordingly, numerical discretization of the Helmholtz equation (1.1) by a variety of methods, such as finite-difference, finite-element, integral-equation, or hybrid asymptotic finite-element, gives rise to an $N_v \times N_v$ linear system. Ideally, we desire a numerical scheme to have two properties: having almost linear complexity, up to poly-logarithmic factors, in both CPU time and memory storage units (in solving the linear system), and having asymptotically uniform accuracy with respect to $1/\omega$ (at least) as $\omega \to \infty$ while respecting the Shannon sampling principle. However, so far, no method is available enjoying the two properties simultaneously in the literature. Observing that the Helmholtz solution can be written as

$$u(\mathbf{r}) = \iint_V G(\mathbf{r}, \mathbf{r}_0) s(\mathbf{r}_0) d\mathbf{r}_0 \tag{1.3}$$

if the Green's function $G(\mathbf{r}, \mathbf{r}_0)$ is known, we propose to first use the Hadamard-Babich (HB) high-frequency asymptotic ansatz to compute the Green's function, and then use the fast butterfly algorithm to compress the HB integrator, and finally apply the compressed integrator to the source function to obtain the desired Helmholtz solution. As we will see, the resulting new numerical scheme enjoys the two desired properties simultaneously.

Why to use the HB ansatz. High-frequency asymptotics typically assumes an expansion series for the Green's function in terms of the phase (or traveltime) and amplitude functions, which satisfy the eikonal and transport equations, respectively. The Eulerian asymptotics solves these equations with partial differential equation (PDE) solvers and utilizes the resulting asymptotic ingredients to construct the Green's function for each point source. However, the usual geometrical-optics ansatz [3] does not yield uniform accuracy near the source as $\omega \to \infty$ and poses difficulties when initializing the amplitudes. Recently, the HB ansatz [4] based Eulerian asymptotics has been developed in [56] which yields a uniform asymptotic solution in the region of space containing a point source but no other caustics. The eikonal and transport equations for the HB coefficients are solved with high-order Lax-Friedrichs weighted non-oscillatory (WENO) schemes which are initialized near the source point with high-order Taylor expansions [56]. The resulting HB integrator is a highly accurate approximation of the Green's function for media with a smooth and analytic refractive index $n(\mathbf{r})$ that does not introduce caustics. That said, for a computational domain discretized with N_v cells, the Eulerian asymptotics requires solving the eikonal and transport equations N_v times for an arbitrary source function $s(\mathbf{r})$ that can be nonzero across the entire computational domain. Because such an $N_v \times N_v$ discretized HB integrator is still prohibitively expensive to compute, a fast compressed representation is called for. Therefore, we will develop low-rank representations using the Chebyshev interpolation for these HB ingredients.

Why to use the butterfly algorithm. We consider an algebraic compression tool called butterfly [46, 39, 35, 34, 52], a multilevel numerical linear algebra algorithm well-suited for representing highly oscillatory operators such as Fourier transforms and integral operators [10, 67, 66], special function transforms [62, 9, 50], free-space [47, 46, 40], numerical [37], and inverse [25, 38, 26, 27] Green's functions for Helmholtz and Maxwell's equations. As the HB integrator consists of non-oscillatory HB coefficient functions and oscillatory Hankel functions defined via non-oscillatory phase functions, we show that the discretized HB integrator is butterfly compressible. The proposed scheme first constructs low-rank representations for the phase function and HB coefficients via solving the eikonal and transport equations with a set of coarse grids for a constant number of point sources located at the Chebyshev interpolation nodes. Next, it leverages butterfly algorithms and its hierarchical extension, the hierarchical off-diagonal butterfly (HODBF) matrix [38], to compress the HB integrator for cell sizes proportional to the angular frequency, via sampling the phase and HB coefficients in a manageable way. Once compressed, the HB integrator can be applied to any source function as a simple matrix-vector multiplication. This framework is also extended to handle a computational domain with sound-hard inclusion, where an additional surface integral equation using the HB integrator is solved. We analyze our proposed algorithm to validate that the CPU time and memory requirement for most involved discretized integrators scale at most as $O(N_v \log^2 N_v)$. Moreover, this scheme obeys the Shannon sampling principle, is free of dispersion errors due to the asymptotic nature of the method, and requires much smaller numbers of cell points per wavelength than finite-difference solvers, and it further has been distributed-memory parallelized. As a result, it can solve wave propagation problems well beyond the capability of existing FDFD solvers. Remarkably, the proposed scheme can accurately model wave propagation in 2D domains with 640 wavelengths per direction, and 3D domains with 54 wavelengths per direction on a state-the-art supercomputer at Lawrence Berkeley National Lab.

1.1. Related works. To put our work into perspective, let us first point out that in the high-frequency regime, the notion of convergence is different from the standard numerical analysis. Because of pollution errors (numerical dispersions) [6], numerical errors of standard methods for the Helmholtz equation (1.1) do not decay as $\omega \to \infty$ if ωh is fixed, namely, the Shannon sampling principle is respected. Consequently, in the high-frequency asymptotic regime, namely, $\omega \to \infty$, we seek numerical methods which both converge asymptotically with respect to ω and obey the Shannon sampling principle.

There are three popular classes of numerical methods for solving the variable coefficient Helmholtz (or Maxwell's) equations: the differential-equation method such as finite-difference [12, 51] or finite-element [5, 48] method, the volume-integral equation (VIE) method [58, 68, 15], and the hybrid asymptotic finite-element based methods [24, 49, 29, 20, 19, 31]. We consider the following three aspects.

Accuracy. The differential-equation method, for instance the finite-difference frequency-domain (FDFD) method, leverages absorbing boundary conditions and finite-difference stencils to form a sparse $N_v \times N_v$ linear system, whose inverse gives numerical Green's functions. Given a uniform accuracy requirement for all frequencies ω, pollution errors demand that differential-equation methods oversample the numerical solution to mitigate the pollution effects, leading to large-scale systems with sub-optimal DOFs to solve when $\omega \to \infty$, and the resulting CPU time and memory storage units are sub-optimal with respect to the Shannon principle. On the other hand, the VIE method [57] leverages volumetric equivalent sources and the exact, free-space Green's function to form a dense $N_v \times N_v$ linear system. The solution of the linear system yields the equivalent source densities. Although VIE is almost free of numerical dispersion, the system solve is even more expensive than FDFD. The hybrid asymptotic finite-element methods [24, 49, 29, 20, 19, 31] incorporate phase or phase gradient information into the formulation, and the hybrid methods in [20, 19] demonstrate almost linear complexity in CPU time and converge asymptotically as $1/\sqrt{\omega}$ when $\omega \to \infty$ for 2-D problems. We refer to [19] for references on various approaches to eliminate or mitigate pollution effects. Our proposed approach is different from the above three classes, and it is based on the uniform asymptotic HB ansatz and enjoys the following unique feature: given a uniform accuracy requirement for all frequencies ω, the accuracy behaves as $1/\omega$ (at least) asymptotically and shows no apparent dispersion errors when $\omega \to \infty$, as long as ωh is fixed.

Efficiency. Given CPU time and storage requirement scalable with respect to ω by fixing ωh to be a constant, we would like to solve the linear system $N_v \times N_v$ in $O(N_v)$ time and $O(N_v)$ memory storage, up to poly-logarithmic factors. As stated in [19], standard sparse linear algebra algorithms based on nested dissection [22] and multi-frontal methods [14] have a suboptimal complexity, and they are prohibitively expensive memory-wise in dimension greater than two [13, 33, 2, 64, 7, 8]. As a result, quasi-linear-cost preconditioners are developed, such as [16, 17, 59, 11, 63, 69], among many others. In a most recent work [37], a sparse approximate multifrontal factorization with butterfly compression for high-frequency wave equations has been developed, and complexity analysis and numerical experiments demonstrate that it enjoys $O(N_v \log^2 N_v)$ computation and $O(N_v)$ memory complexity when applied to an $N_v \times N_v$ sparse system arising from 3-D high-frequency Helmholtz problems. However, most of these methods use low-order discretizations so that they require oversampling to produce accurate solutions, thus resulting in suboptimal complexities with respect to the frequency ω. Our proposed approach is also based on butterfly compression, but it enjoys the following unique feature: the overall CPU time and memory requirement scale as $O(N_v \log^2 N_v)$ when $\omega \to \infty$ as long as ωh is fixed (no oversampling).

Accuracy and efficiency. As stated in [19], only a few references deal with both accuracy and efficiency simultaneously. [61] develops a hybridizable discontinuous Galerkin method coupled with the method of polarized traces, [70, 36] deal with an integral version of the Helmholtz equation with inclusions by coupling with sparsification and a fast preconditioner, and [20, 19, 31] develop adaptive discretizations for Helmholtz equations by learning the dominant wave directions. In our work, we also consider an integral version of the Helmholtz equation for curved inclusions. Our proposed approach for both cases of inclusion and no-inclusion enjoys both accuracy and efficiency simultaneously in the sense that the overall CPU time and memory requirement scale as $O(N_v \log^2 N_v)$ and the accuracy behaves as $1/\omega$ (at least) asymptotically when $\omega \to \infty$, as long as ωh is fixed (no oversampling).

Finally, we remark that a version of the butterfly algorithm [10] has been implemented in fast Huygens sweeping methods for computing high-frequency Green's functions of point-source Helmholtz equations based on traditional geometrical optics [44] and the HB ansatz [42], respectively. Our butterfly compressed HB integrator proposed here is different from those in [44, 42, 54] in that we are treating arbitrary sources rather than a single point source and we are using the hierarchical off-diagonal butterfly (HODBF) algorithm [38], which is more efficient than the classical butterfly algorithm [10] or hierarchical matrices [26] to compress the high-frequency interaction matrix. We also mention in passing that the necessity of using butterfly compression for fast computing high-frequency waves has been shown in [18].

1.2. Contents.

In Section 2, we introduce the HB ansatz for the point-source problem of Helmholtz equations. Direct computation of the HB integrator for arbitrary sources is detailed in Section 3. Fast computation of the HB integrator for arbitrary sources is described in Section 4. In Section 5, we show numerical results to demonstrate the performance of the proposed scheme.

2. Hadamard-Babich Ansatz for Point Source.

To solve equation (1.1) asymptotically with a point source when $\omega \to \infty$, Babich [4] used Hadamard's method to obtain the following Hankel-based ansatz so as to expand the solution $u(\mathbf{r}) =$

$G(\mathbf{r}, \mathbf{r}_0)$,

$$g_{hb}(\mathbf{r}, \mathbf{r}_0) = \sum_{s=0}^{\infty} v_s(\mathbf{r}, \mathbf{r}_0) f_{s-(d-2)/2}(\omega, \tau), \tag{2.1}$$

where

$$f_q(\omega, \tau) = \mathrm{i}\frac{\sqrt{\pi}}{2} e^{iq\pi} \left(\frac{2\tau}{\omega}\right)^q \mathrm{H}_q^{(1)}(\omega\tau). \tag{2.2}$$

Here $\mathrm{H}_q^{(1)}$ is the q-th Hankel function of the first kind, and the phase τ, more precisely its square τ^2, satisfies the eikonal-squared equation (2.3),

$$|\nabla \tau^2|^2 = 4\tau^2 n^2, \quad \tau^2(\mathbf{r}, \mathbf{r}_0)|_{\mathbf{r}=\mathbf{r}_0} = 0, \tag{2.3}$$

which is equivalent to the usual form of the eikonal equation,

$$|\nabla \tau| = n, \quad \tau(\mathbf{r}, \mathbf{r}_0)|_{\mathbf{r}=\mathbf{r}_0} = 0. \tag{2.4}$$

We remark that two forms of eikonal equations as (2.3) and (2.4) are needed because τ itself as a distance function near the source is not differentiable at the source but τ^2 is, as long as n is locally smooth [3, 60].

We call the above ansatz the Hadamard-Babich ansatz and the coefficients v_s the HB coefficients which are different from amplitude functions in the classical geometrical optics.

The HB coefficients v_{s+1} in expansion (2.1) satisfy the recurrent system

$$4\tau n^2 \frac{\partial v_{s+1}}{\partial \tau} + v_{s+1}\left[\Delta\tau^2 + 2n^2(2s+2-d)\right] = \Delta v_s, \quad s = -1, 0, 1, \ldots, \tag{2.5}$$

and $v_{-1} \equiv 0$, where the differentiation $\frac{\partial}{\partial \tau}$ is performed along the ray departing from $\mathbf{r}_0$. Assuming $v_s(\mathbf{r}, \mathbf{r}_0)$ to be continuous in the neighborhood of $\mathbf{r} = \mathbf{r}_0$, we get the initial conditions for v_0 at $\mathbf{r} = \mathbf{r}_0$,

$$v_0(\mathbf{r}, \mathbf{r}_0)|_{\mathbf{r}=\mathbf{r}_0} = \frac{n_0^{d-2}}{2\pi^{(d-1)/2}}, \quad n_0 = n(\mathbf{r}_0). \tag{2.6}$$

If $n(\mathbf{r})$ is smooth, then τ is smooth in the neighborhood of $\mathbf{r}_0$ except at the source point itself, but τ^2 is smooth in the source neighborhood, including the source itself. If $n(\mathbf{r})$ is analytic, it can be shown [4, 42] that the function $v_0(\mathbf{r}, \mathbf{r}_0)$ will also be analytic in $\mathbf{r}$ when $\mathbf{r}$ is in the neighborhood of $\mathbf{r}_0$; furthermore, $v_{s+1}(\mathbf{r}, \mathbf{r}_0)$ are determined in terms of v_0 and τ so that v_{s+1} are analytic when v_0 are analytic for $s = 0, 1, \ldots$.

2.1. Assumption and essential estimates. In the following, we **assume** that these functions τ^2 and v_s $(s = 0, 1, 2, \cdots)$ are analytic in the computational domain V for all source points $\mathbf{r}_0 \in V$, so that *the HB ansatz is valid for all point sources* $\mathbf{r}_0 \in V$. This means that all these HB ingredients are smooth single-valued functions in V for any point source in V. Although this is a strong assumption, it will serve as a good starting point for many applications.

By the essence of asymptotics [32], the difference between the true Green's function of (1.1) and the HB ansatz (2.1) can be written as

$$G(\mathbf{r}, \mathbf{r}_0) - g_{hb}(\mathbf{r}, \mathbf{r}_0) = O(1/\omega^{\infty}), \tag{2.7}$$

where the "error" term on the right-hand side means that the difference can be made arbitrarily smooth for all $\mathbf{r}$, as long as the HB ingredients τ^2 and v_s $(s = 0, 1, \cdots)$ are analytic.

Moreover, when $|\tau| \leq A \equiv$ constant, the Hankel-based terms f_q with $q = N - (d-2)/2$ have the following asymptotic forms for large ω [4],

(2.8)
$$f_q(\omega,\tau) = \begin{cases} O\left(\left(\frac{\tau}{\omega}\right)^q (\omega\tau)^{-1/2}\right) = O(\omega^{-q-1/2}\tau^{q-1/2}), & \text{if } \omega\tau \geq A_1 \equiv \text{constant}, \\ O\left(\ln(\omega\tau) + 1\right), & \text{if } \omega\tau \leq A_1 \text{ and } q = 0, \\ O\left(\left(\frac{\tau}{\omega}\right)^q (\omega\tau)^{-q}\right) = O(\omega^{-2q}), & \text{if } \omega\tau \leq A_1 \text{ and } q \geq 1, \end{cases}$$

where $N = 0, 1, 2, \ldots$, and both A and A_1 are arbitrarily fixed positive constants.

In this article, we truncate the HB ansatz (2.1) to obtain a $(N+1)$-term expansion,

$$g(\mathbf{r},\mathbf{r}_0) = \sum_{s=0}^{N} v_s(\mathbf{r},\mathbf{r}_0) f_{s-(d-2)/2}(\omega,\tau), \tag{2.9}$$

and we can estimate the truncation error asymptotically in ω by using the asymptotic forms of f_q in (2.8),

$$\|g_{hb}(\cdot,\mathbf{r}_0) - g(\cdot,\mathbf{r}_0)\|_{L^\infty(V)} \leq O\left((1/\omega)^{(N+1-\frac{d-3}{2})}\right); \tag{2.10}$$

see a similar analysis in [19].

Now we estimate the difference of the following solution formulas,

$$u_{true}(\mathbf{r}) = \iint_V G(\mathbf{r},\mathbf{r}_0) s(\mathbf{r}_0) d\mathbf{r}_0 \quad \text{for} \quad \mathbf{r} \in V, \tag{2.11}$$

$$u_{\text{hb}}(\mathbf{r}) = \iint_V g(\mathbf{r},\mathbf{r}_0) s(\mathbf{r}_0) d\mathbf{r}_0 \quad \text{for} \quad \mathbf{r} \in V, \tag{2.12}$$

where we have assumed that the source s is compactly supported and an appropriate absorbing boundary condition has been used to truncate the entire space R^d to the computational domain V. Accordingly, we have

$$\begin{aligned} \|u_{true} - u_{\text{hb}}\|_{L^\infty(V)} &\leq \iint_V \|G(\cdot,\mathbf{r}_0) - g(\cdot,\mathbf{r}_0)\|_{L^\infty(V)} |s(\mathbf{r}_0)| d\mathbf{r}_0 \\ &\leq \iint_V \|G(\cdot,\mathbf{r}_0) - g_{hb}(\cdot,\mathbf{r}_0)\|_{L^\infty(V)} |s(\mathbf{r}_0)| d\mathbf{r}_0 \\ &\quad + \iint_V \|g_{hb}(\cdot,\mathbf{r}_0) - g(\cdot,\mathbf{r}_0)\|_{L^\infty(V)} |s(\mathbf{r}_0)| d\mathbf{r}_0 \\ &\leq O\left((1/\omega)^{(N+1-\frac{d-3}{2})}\right). \end{aligned} \tag{2.13}$$

This estimate is the foundation for our new numerical schemes, and further numerical analysis is an ongoing work.

In what follows, we drop the subscript of u and by default refer to it as the HB ansatz solution, and we will only consider two cases: $N = 0$ and $N = 1$, yielding the first- and second-order asymptotic expansion, respectively.

2.2. Approximations of eikonals and HB coefficients near the source. Evaluating (2.9) requires computation of the squared-phase function τ^2 and the HB coefficient functions v_0 and v_1. In a neighborhood of the point source where the function τ is single-valued, τ^2, v_0, and v_1 can be computed by solving the eikonal equation (2.4) and transport equations (2.5) with initial conditions v_0 in (2.6) and $v_1(\mathbf{r}, \mathbf{r}_0)|_{\mathbf{r}=\mathbf{r}_0}$ to be determined, and the numerical details have been given in [56].

To expedite our presentation, we summarize some numerical aspects in the following subsections. One essential difficulty in computing the eikonal and HB coefficients to high-order accuracy is how to initialize these quantities near the point source for numerical PDE solvers, such as Lax-Friedrichs WENO schemes [30, 73, 72]. Because initial conditions for the eikonal and transport equations are only specified at the source point and high-order schemes need accurate initial values within a small neighborhood of the source to start with, our analyticity assumptions allow us to extract high-order approximations of eikonals and HB coefficients near the source. To do that, we may carry out local Taylor expansions of these functions and further insert these relations into related PDEs so that we may obtain some recursive relations to compute these functions locally. Since such recursive relations actually provide (crude) approximations to the HB ingredients which in turn may be used to construct the Green's function (in a very crude manner) in a large neighborhood, we illustrate how to obtain such recursive relations in the following.

2.2.1. High-order factorization of eikonals. We have the following expansion near the source point $\mathbf{r}_0$ (dropped in the expressions below) for squared τ and n:

$$\tau^2(\mathbf{r}) \approx \sum_{k=0}^{P_T} T_k(\mathbf{r}), \quad n^2(\mathbf{r}) \approx \sum_{k=0}^{P_S} S_k(\mathbf{r}), \tag{2.14}$$

where $T_k(\mathbf{r})$ and $S_k(\mathbf{r})$ are homogeneous polynomials of degree k in $\mathbf{r}$, and P_T and P_S denote the truncation order of τ^2 and n^2, respectively. Upon substituting (2.14) into (2.3), we can determine T_k term-by-term by

$$T_0 = 0, \quad T_1 = 0, \quad T_2(\mathbf{r}) = S_0\mathbf{r}^2, \tag{2.15}$$

$$(P-1)S_0T_P = \sum_{k=1}^{P-2} S_k T_{P-k} - \frac{1}{4}\sum_{k=2}^{P-2} \nabla T_{k+1}\cdot\nabla T_{P-k+1}, \quad P \geq 3. \tag{2.16}$$

Since we will solve for τ in the eikonal equation (2.4) rather than τ^2, we will take the square root of (2.14) to obtain an approximate τ, which will serve as a high-order approximation of the exact τ in numerically solving the eikonal equation (2.4).

2.2.2. High-order factorization of coefficients v_s. Although v_s are assumed to be analytic near the source, we still need to obtain high-order approximations to v_s within a small neighborhood of the source so that high-order numerical schemes can be initialized near the source. Therefore, we will expand the coefficients v_s as homogeneous polynomials of degree k in $\mathbf{r}$ as well.

The coefficient v_0 can be expanded as

$$v_0 = \sum_{k=0}^{P_B} B_k(\mathbf{r}), \tag{2.17}$$

where $B_k(\mathbf{r})$ are homogeneous polynomials of degree k in $\mathbf{r}$. Upon substituting (2.17) into (2.5), we can determine B_k term-by-term by

$$B_0 = \frac{n_0^{d-2}}{2\pi^{(d-1)/2}}, \quad B_1 = \frac{1}{2S_0}\left(-\frac{1}{2}B_0\Delta T_3 + dB_0S_1\right), \tag{2.18}$$

$$2PS_0B_P = -\sum_{k=1}^{P-1} \nabla B_k \cdot \nabla T_{P+2-k} - \frac{1}{2}\sum_{k=0}^{P-1} B_k \Delta T_{P+2-k} + d\sum_{k=0}^{P-1} B_k S_{P-k}, \ P \geq 2. \tag{2.19}$$

Similarly, we expand v_1 as

$$v_1 = \sum_{k=0}^{P_C} C_k(\mathbf{r}), \tag{2.20}$$

and obtain C_0 and C_1,

$$2S_0C_0 = \frac{1}{2}\Delta B_2., \tag{2.21}$$

$$4S_0C_1 = \frac{1}{2}\Delta B_3 - \frac{1}{2}\Delta T_3 \, C_0 - (2-d)S_1 \, C_0, \tag{2.22}$$

and C_P for $P \geq 2$ by

$$2(P+1)S_0C_P = \frac{1}{2}\Delta B_{p+2} - \sum_{k=1}^{P-1} \nabla C_k \cdot \nabla T_{P+2-k} - \frac{1}{2}\sum_{k=0}^{P-1} C_k \Delta T_{P+2-k} - (2-d)\sum_{k=0}^{P-1} C_k S_{P-k}. \tag{2.23}$$

To ensure the same order of accuracy for solving (2.3) and (2.5), the truncation orders are chosen as $P_B = P_S = P_T - 2 = P_C + 2$. If the analytic function n varies very slowly in a certain neighborhood of the point source that we are interested in so that the resulting Taylor expansions for τ^2, v_0, and v_1 are sufficiently accurate, then we may use these Taylor expansions to compute the HB ingredients so as to construct the Green's function in this neighborhood. However, in most of situations, we are interested in wave propagation in large domains, and we will use these local Taylor expansions to initialize high-order Lax-Friedrichs WENO sweeping schemes; the related details have been given in [56] which is briefly summarized here.

2.3. Efficient algorithms for computing eikonals and HB coefficients. According to numerical analysis for PDEs, to obtain v_1 with first-order accuracy, we need a third-order accurate approximation of v_0 and hence a fifth-order accurate approximation of τ. The Lax-Friedrichs WENO schemes as illustrated in [56] can be employed to solve the eikonal equation (2.3). To resolve the singularity at the source, we use the factorization approach [53, 71, 21, 45] so that τ can be factored as

$$\tau = \tilde{\tau}\bar{\tau}. \tag{2.24}$$

Here $\tilde{\tau}$ is pre-determined analytically to capture the source singularity, and for instance, we can choose $\tilde{\tau}(\mathbf{r}, \mathbf{r}_0)$ as $n(\mathbf{r}_0)|\mathbf{r} - \mathbf{r}_0|$ or the square root of the expansion (2.14). Hence $\bar{\tau}$ is the new unknown which is smooth at the source and satisfies the factored eikonal equation

$$|\tilde{\tau}\nabla\bar{\tau} + \bar{\tau}\nabla\tilde{\tau}| = n. \tag{2.25}$$

Accordingly, high-order Lax-Friedrichs WENO schemes [30, 72, 65, 56] can be applied to solve (2.25). In order to obtain a P_T-th order accurate approximation of τ on a mesh of size h_0, $\bar{\tau}$ needs to be initialized in a neighborhood of size $2(P_T-1)h_0$ centered at the source, and these initial values will be fixed during the iterations. In the computation, we will take $P_T = 6$ to obtain τ.

After obtaining a high-order approximation for τ, we apply Lax-Friedrichs WENO schemes to solve transport equations (2.5) as illustrated in [56]. When $s=-1$, v_0 will be initialized as (2.17) in a neighborhood of size $2(P_B-1)h_0$ centered at the source and these values will be fixed during iterations; at other points, high-order Lax-Friedrichs WENO-based schemes are used to update v_0. Similarly, when $s=0$, v_1 is then initialized as (2.20) in a neighborhood of size $2(P_C-1)h_0$ centered at the source and these values will be fixed during iterations; the values at other points for v_1 will be updated using the high-order Lax-Friedrichs WENO-based schemes.

To analyze the complexity of these numerical schemes for computing these HB ingredients, we assume that the d-dimensional computational domain is partitioned into a finite-difference mesh of N_W^d points with N_W sampling points in each direction. Since these high-order Lax-Friedrichs WENO-based schemes are iterative by nature, we may assume that these high-order schemes have a superlinear complexity as analyzed [42], and hence the computational cost for these HB ingredients is $O(N_W^d \log N_W)$ in general.

Nevertheless, we emphasize that since these HB ingredients are independent of the angular frequency ω, the above complexity is for computing the HB ingredients only and consequently is not directly related to computing wave fields, and we can use very coarse meshes to compute these quantities. Moreover, once they are computed, these ingredients can be compressed into low-rank representations and reused for different frequencies as shown in Section 4.1.

3. Direct Computation of Hadamard-Babich Integrator for Arbitrary Source.

Considering equation (1.1) in a spatial domain $V=[0,1]^d$, we can apply the HB ansatz to compute the wave field in V due to an arbitrary source function $s(\mathbf{r})$. We consider two situations: domains without inclusion and domains with inclusion.

3.1. Domain with no inclusion.

When no inclusion is present, the field $u(\mathbf{r})$ can be expressed as

$$u(\mathbf{r}) = \iint_V g(\mathbf{r},\mathbf{r}_0)s(\mathbf{r}_0)d\mathbf{r}_0, \tag{3.1}$$

where the Green's function $g(\mathbf{r},\mathbf{r}_0)$ is given in equation (2.9). We call this the Hadamard-Babich integrator.

To numerically compute (3.1), the domain is discretized into N_v regular cells with cell size h, where h is typically a constant fraction of the ω-dependent wavelength. The source function can be discretized with local volume basis functions $s(\mathbf{r}) = \sum_j s(\mathbf{r}_j)h^d b_j^v(\mathbf{r})$, where the basis function $b_j^v(\mathbf{r})$ is nonzero only inside the source cell c_j with center $\mathbf{r}_j$:

$$b_j^v(\mathbf{r}) = \begin{cases} \frac{1}{h^d} & \text{if } \mathbf{r}\in c_j, \\ 0 & \text{otherwise.}\end{cases} \tag{3.2}$$

Note that for a point source located inside cell c_i, the source function is approximated with $s(\mathbf{r}_j) = \delta_{ij}/h^d$. The field at the center of each cell $u(\mathbf{r}_i)$ can be computed as

$$U = K^{v2v} I, \tag{3.3}$$

where I denotes a vector of length N_v that collects $s(\mathbf{r}_j)h^d$, U denotes a vector that collects $u(\mathbf{r}_i)$, and $K_{ij}^{v2v} = g(\mathbf{r}_i, \mathbf{r}_j)$ for $i \neq j$. The self term K_{ii}^{v2v} can be computed analytically by integrating the free space Green's function over the source cell, where the index of refraction n is taken to be constant; see Appendix A for more details.

The naive computation of all non-diagonal terms of K^{v2v} requires solving the eikonal equation (2.4) and transport equations (2.5) for N_v times to tabulate $\tau^2(\mathbf{r}_i, \mathbf{r}_j)$, $v_0(\mathbf{r}_i, \mathbf{r}_j)$ and $v_1(\mathbf{r}_i, \mathbf{r}_j)$, requiring at least $O(N_v^2)$ CPU time and memory. Moreover, the assembly and application of K^{v2v} also require $O(N_v^2)$ time and memory.

We remark in passing that when the medium is homogeneous, the HB integrator (3.1) is exact since the HB ansatz (2.9) yields the exact Green's function in this case; when the medium is inhomogeneous and smooth without inducing caustics, the HB integrator (3.1) is accurate in the asymptotic order $O(1/\omega^\infty)$ without truncating the ansatz (2.9), and it is accurate in the asymptotic order $O(1/\omega^{(2-\frac{d-3}{2})})$ when truncating the ansatz (2.9) up to the first two terms. Detailed numerical analysis on this is an ongoing work.

3.2. Domain with sound hard inclusion. For simplicity, this subsection only considers 2D domains with curve inclusion; however the proposed scheme can be trivially extended to 3D domains with surface inclusion. Considering a sound hard curve denoted by C, the source function will generate an incident field $u^{\text{inc}}(\mathbf{r})$ that induces an equivalent source $p(\mathbf{r})$ on C, which in turn generates a scattered field $u^{\text{sca}}(\mathbf{r})$. More specifically, we have

$$u^{\text{inc}}(\mathbf{r}) := \iint_V g(\mathbf{r}, \mathbf{r}_0)s(\mathbf{r}_0)d\mathbf{r}_0, \quad \mathbf{r} \in C, \tag{3.4}$$

$$u^{\text{sca}}(\mathbf{r}) := \int_C g(\mathbf{r}, \mathbf{r}_0)p(\mathbf{r}_0)d\mathbf{r}_0, \quad \mathbf{r} \in C \text{ or } V. \tag{3.5}$$

Here the equivalent source p can be solved by the following integral equation:

$$u^{\text{inc}}(\mathbf{r}) = -u^{\text{sca}}(\mathbf{r}), \quad \mathbf{r} \in C. \tag{3.6}$$

To numerically solve (3.6) and compute the total field $u = u^{\text{inc}} + u^{\text{sca}}$ in V, the domain V is discretized with N_v cells with cell size h. After discretization of the inclusion into N_s line segments with length w_i for $i = 1, \dots, N_s$ and enforcement of (3.6) at segment centers, we solve a linear system

$$K^{s2s}P = U^{\text{inc}} \tag{3.7}$$

with

$$K_{ij}^{s2s} = \begin{cases} \frac{-w_i}{4}(1 + \mathrm{i}\frac{2}{\pi}(\log(\frac{\gamma\omega n(\mathbf{r}_i^s)w_i}{4} - 1))) & \text{if } i = j, \\ -g(\mathbf{r}_i^s, \mathbf{r}_j^s)w_j & \text{otherwise,} \end{cases} \tag{3.8}$$

where γ is the Euler constant, $U_i^{\text{inc}} = u^{\text{inc}}(\mathbf{r}_i^s)$ and $P_i = p(\mathbf{r}_i^s)$ with $\mathbf{r}_i^s$ denoting segment centers. The right-hand-side (RHS) in (3.7) is computed by

$$U^{\text{inc}} = K^{v2s}I, \tag{3.9}$$

where I is the same as that in (3.3), and the discretized volume-to-surface operator is $K_{ij}^{v2s} = g(\mathbf{r}_i^s, \mathbf{r}_j)$. Here the source function does not overlap with the curve.

Once the equivalent source $p(\mathbf{r})$ is obtained, the scattered field at any point $\mathbf{r}$ of the computational domain is computed using (3.5),

$$U^{\text{sca}} = K^{s2v}P. \tag{3.10}$$

Here $U_i^{\text{sca}} = u^{\text{sca}}(\mathbf{r}_i)$ with $\mathbf{r}_i$ being the center of cell i, and $K_{ij}^{s2v} = g(\mathbf{r}_i, \mathbf{r}_j^s)w_j$ which can be directly calculated from the transpose of K^{v2s}. Combining (3.3), (3.7), (3.9) and (3.10), the total fields at the cell centers can be expressed as

$$U = (K^{v2v} + K^{s2v}\big(K^{s2s}\big)^{-1}K^{v2s})I. \tag{3.11}$$

As a typical curve in 2D requires $N_s = N_v^{1/2}$ discretization segments, the naive computation of K^{v2s} and K^{s2s} requires solving the eikonal equation (2.4) and transport equations (2.5) for N_s times. In addition, the computation of K^{v2s}, K^{s2s}, and $\big(K^{s2s}\big)^{-1}$ requires $O(N_v^{3/2})$ time and memory. Overall, the computation of (3.11) is still $O(N_v^2)$ dominated by the computation of K^{v2v}.

4. Fast Computation of Hadamard-Babich Integrator for Arbitrary Source.

Here we propose a quasi-linear complexity algorithm for the computation of (3.3) and (3.11). The proposed algorithm leverages the low-rank representation of the squared phase function τ^2 and HB coefficient functions v_0 and v_1 to avoid solving eikonal and transport equations for all point sources. Once these low-rank representations are obtained, the discretized volume-to-volume, volume-to-surface, and surface-to-surface operators are compressed using butterfly algorithms and their hierarchical extensions.

4.1. Low-rank Representation of the Phase and HB Coefficients.

Since we have assumed that τ^2, v_0 and v_1 are analytic, they permit low-rank representations as shown in [42, 41, 43, 55]. Letting $f = \tau^2$, v_0 and v_1, respectively, we consider the following analytical low-rank representation using the Chebyshev interpolation,

$$f(\mathbf{r}, \mathbf{r}_0) \approx \sum_{i=1}^{N_I}\sum_{j=1}^{N_I} T_i(\mathbf{r}) f(\mathbf{r}_i^c, \mathbf{r}_j^c) T_j(\mathbf{r}_0). \tag{4.1}$$

Let n_I denote order of the Chebyshev interpolation and $\mathbf{r}_i^c$ for $i = 1, \dots, N_I$ with $N_I = n_I^d$ be the Chebyshev nodes (i.e., d-dimensional Chebyshev sampling of the domain $V = [0,1]^d$). Accordingly, we define the Lagrange interpolants $T_i(\mathbf{r})$, where

$$T_i(\mathbf{r}) = \prod_{s=1}^{d} l_i(r_s) = \prod_{s=1}^{d} \prod_{\substack{1 \le k \le n_I \\ k \ne i}} \frac{r_s - r_{sk}^c}{r_{si}^c - r_{sk}^c}, \tag{4.2}$$

$\mathbf{r} = [r_1, \dots, r_d]$, and $\mathbf{r}_i^c = [r_{1i}^c, \dots, r_{di}^c]$.

In (4.1), the set of function samples $\{f(\mathbf{r}_i^c, \mathbf{r}_j^c)\}$ of cardinality N_I^2 requires solving the eikonal and transport equations via the Lax-Friedrichs WENO schemes with N_I point sources located at $\mathbf{r}_j^c$, as explained in Section 2. More specifically, for each Chebyshev node $\mathbf{r}_j^c$, we create a grid with mesh size h_0 and grid points $\mathbf{r}_i^0$ for $i = 1, \dots, 1/h_0^d$ that are aligned with the point source at $\mathbf{r}_j^c$. In other words, the grid covers the computation domain V and its ghost regions, and has one grid point collocated with $\mathbf{r}_j^c$. This gives rise to solutions $f(\mathbf{r}_i^0, \mathbf{r}_j^c)$. For each $\mathbf{r}_i^c$, we compute $f(\mathbf{r}_i^c, \mathbf{r}_j^c)$ with

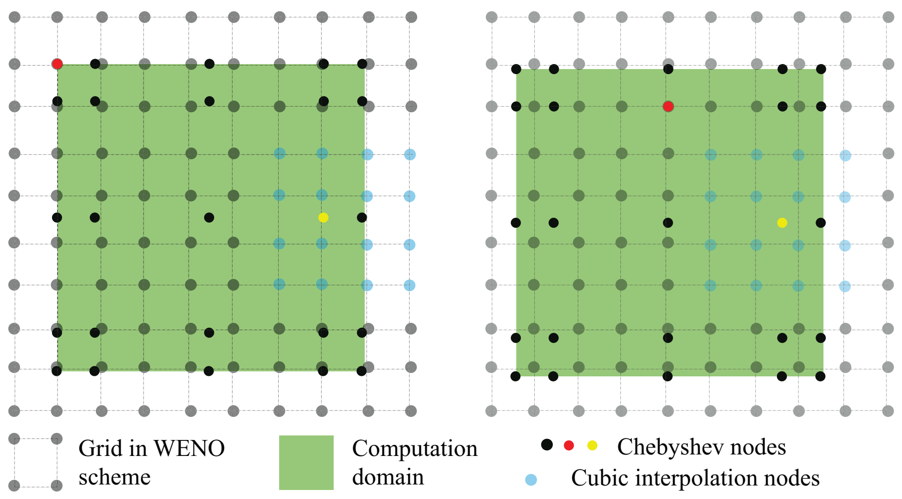

Fig. 4.1: The computation of $f(\mathbf{r}_i^c, \mathbf{r}_j^c)$ in a 2D domain (shown as the green area) with $f = \tau^2, v_0, v_1$. Here r_i^c, $i = 1, \ldots, N_I$ denote the Chebyshev nodes with $N_I = 5^2$ (the black dots). Two instances of $\mathbf{r}_j^c$ are shown: for each instance, a fast sweeping method with a grid size h_0, grid points $\mathbf{r}_i^0$ and a point source collocated at one Chebyshev node (the red dot) is used to compute $f(\mathbf{r}_i^0, \mathbf{r}_j^c)$. For each $\mathbf{r}_i^c$, $i \neq j$ (the yellow dots), $f(\mathbf{r}_i^c, \mathbf{r}_j^c)$ is computed via cubic interpolation from the grid points $\mathbf{r}_i^0$ (the blue dots).

$i \neq j$ from a local cubic interpolation using data points $f(\mathbf{r}_i^0, \mathbf{r}_j^c)$. See Figure 4.1 for a 2D example with two point sources (in red dots).

Once the set $\{f(\mathbf{r}_i^c, \mathbf{r}_j^c)\}$ is obtained, the computation of $f(\mathbf{r}, \mathbf{r}_0)$ for any point pair $(\mathbf{r}, \mathbf{r}_0)$ requires $O(N_I^2) = O(n_I^{2d})$ time. We can assume N_I to be constant as typically $n_I < 15$. In addition, we can leverage a blocked version of (4.1) to further improve its computational efficiency. Consider an $m \times n$ block F with $F_{ij} = f(\mathbf{r}_i, \mathbf{r}_j)$ for arbitrary lists of n source points $\mathbf{r}_j$ and m observation points $\mathbf{r}_i$. The block F can be computed as

$$F \approx T^o F^c T^s. \tag{4.3}$$

Here, $T_{ij}^o = T_j(\mathbf{r}_i)$, $T_{ij}^s = T_i(\mathbf{r}_j)$, and $F_{i,j}^c = f(\mathbf{r}_i^c, \mathbf{r}_j^c)$. By using this blocked form, repetitive computation of the interpolants T^o and T^s is avoided, and high-performance BLAS libraries can be used. As a result, this requires $O(\min(m,n)N_I^2 + mnN_I)$ time using (4.3) as opposed to $O(mnN_I^2)$ using (4.1).

One may attempt to compute F for all entries of K^{v2v} (and similarly for K^{s2v}, K^{v2s} and K^{s2s}), but this leads to $O(N_v^2 N_I)$ computational time. As we will see next, we propose the butterfly algorithm for constructing a compressed representation of K^{v2v} and the other discretized operators, requiring only a total of $mn = O(N_v \log^2 N_v)$ entries in (4.3).

4.2. Butterfly Representation of the Discretized Integral Operators. The butterfly algorithm exploits the fact that judiciously selected submatrices of the discretized operators K^{v2v}, K^{s2v}, K^{v2s} and K^{s2s} are low-rank compressible, despite of the full rankness of these operators.

The algorithm first recursively subdivides the geometry point sets associated with the rows and columns of these operators into two subsets of approximately equal sizes, using such as k-dimensional (K-D) tree clustering algorithms, until the subsets contain a predefined number of points. For the N_v cell centroids in the computational domain, the procedure generates a complete binary tree $\mathcal{T}_{H_v}$ of L_v levels with root level 0 and leaf level L_v. Each node τ at level l is an index set $\tau \subset \{1, \ldots, N_v\}$. Similarly for the N_s segment centers for the inclusion, the procedure generates a complete binary tree $\mathcal{T}_{H_s}$ of L_s levels. In both trees, a non-leaf node τ at level l has two children τ_1 and τ_2, where $\tau = \tau_1 \cup \tau_2$ and $\tau_1 \cap \tau_2 = \emptyset$. For a non-root node τ, its parent is denoted p_τ.

The butterfly representation of a matrix requires binary trees $\mathcal{T}_o$ and $\mathcal{T}_s$ with L levels for the row and column indices, respectively, which are defined for the integral operators as follows:

- K^{v2v}: For any two siblings τ_1 and τ_2 at level l of $\mathcal{T}_{H_v}$, let $o = \tau_1$ and $s = \tau_2$. $K^{v2v}(o, s)$ is compressed as a butterfly with $L = L_v - l$ levels. Let $\mathcal{T}_o$ and $\mathcal{T}_s$ be the subtrees of $\mathcal{T}_{H_v}$ rooted at o and s, respectively. As a result, there are 2^l butterfly representations at each level $l = 1, \ldots, L_v$. The 2^{L_v} blocks $K^{v2v}(\tau, \tau)$ for node τ at level $l = L_v$ are kept as dense blocks. Such a representation is called the hierarchically off-diagonal butterfly (HODBF) representation [38].
- K^{s2s}: For any two siblings τ_1 and τ_2 at level l of $\mathcal{T}_{H_s}$, let $o = \tau_1$ and $s = \tau_2$. $K^{s2s}(o, s)$ is compressed as a butterfly with $L = L_s - l$ levels. Let $\mathcal{T}_o$ and $\mathcal{T}_s$ be the subtrees of $\mathcal{T}_{H_s}$ rooted at o and s, respectively. Similar to K^{v2v}, we seek a HODBF representation of K^{s2s}.
- K^{s2v} (or K^{v2s}): Let o and s be the roots of $\mathcal{T}_{H_v}$ and $\mathcal{T}_{H_s}$, respectively. $K^{s2v}(o, s) = K^{s2v}$ is compressed as a *single* butterfly with $L = \min\{L_v, L_s\}$ levels. Let $\mathcal{T}_o = \mathcal{T}_{H_v}$ and $\mathcal{T}_s = \mathcal{T}_{H_s}$.

4.2.1. Butterfly algorithm. The L-level butterfly representation of the integral operator $K(o, s) \in \mathbb{C}^{m \times n}$ (superscripts of K are dropped) requires the complementary low-rank property: at any level $0 \le l \le L$, for any node τ at level l of $\mathcal{T}_o$ and any node ν at level $L - l$ of $\mathcal{T}_s$, the subblock $K(\tau, \nu)$ is numerically low-rank with rank $r_{\tau,\nu}$ bounded by a small number r called the butterfly rank. We will comment on the butterfly rank for the three integral operators in subsections 4.2.2, 4.2.3, and 4.2.4, respectively.

For any subblock $K(\tau, \nu)$, the complementary low-rank property permits a low-rank representation using for instance interpolative decomposition (ID) as

$$K(\tau, \nu) \approx K(\tau, \bar{\nu}) V_{\tau,\nu}, \tag{4.4}$$

where the skeleton matrix $K(\tau, \bar{\nu})$ contains $r_{\tau,\nu}$ skeleton columns indexed by $\bar{\nu}$, and the interpolation matrix $V_{\tau,\nu}$ has bounded entries. The ID can be computed via for instance rank-revealing QR decomposition with a relative tolerance *tol*. There are several equivalent butterfly representations in literature [39, 34, 35, 52] and here we briefly describe the so-called column-wise butterfly representation [39].

At level $l = 0$, the interpolation matrices $V_{\tau,\nu}$ are explicitly formed. While at level $l > 0$, they are represented in a nested fashion. To see this, consider a node pair (τ, ν) at level $l > 0$ and let ν_1, ν_2 and p_τ be the children and parent of ν and τ, respectively. From (4.4), we have

$$\begin{aligned} K(\tau, \nu) &= \begin{bmatrix} K(\tau, \nu_1) & K(\tau, \nu_2) \end{bmatrix} \\ &\approx \begin{bmatrix} K(\tau, \bar{\nu}_1) & K(\tau, \bar{\nu}_2) \end{bmatrix} \begin{bmatrix} V_{p_\tau,\nu_1} & \\ & V_{p_\tau,\nu_2} \end{bmatrix} \end{aligned} \tag{4.5}$$

$$\approx K(\tau,\bar{\nu})W_{\tau,\nu}\begin{bmatrix}V_{p_\tau,\nu_1} & \\ & V_{p_\tau,\nu_2}\end{bmatrix}. \tag{4.6}$$

Here $W_{\tau,\nu}$ and $\bar{\nu}$ are the interpolation matrix and skeleton columns from the ID of $[K(\tau,\bar{\nu}_1),K(\tau,\bar{\nu}_2)]$, respectively. This allows representing $V_{\tau,\nu}$ as

$$V_{\tau,\nu} = W_{\tau,\nu}\begin{bmatrix}V_{p_\tau,\nu_1} & \\ & V_{p_\tau,\nu_2}\end{bmatrix}. \tag{4.7}$$

We will refer to $W_{\tau,\nu}$ as the transfer matrices in the rest of this paper. We note that the computation of interpolation matrices $V_{\tau,\nu}$ at level $l=0$ and transfer matrices $W_{\tau,\nu}$ at level $0<l<L$ does not require the ID on the full subblocks $K(\tau,\nu)$ and $[K(\tau,\bar{\nu}_1),K(\tau,\bar{\nu}_2)]$, as this immediately leads to an $O(mn)$ compression complexity at level $l=0$.

Instead, we can select a number of $O(r_{\tau,\nu})$ proxy rows $\hat{\tau}\subset\tau$ to compute $V_{\tau,\nu}$ and $W_{\tau,\nu}$ via ID as:

$$K(\hat{\tau},\nu)\approx K(\hat{\tau},\bar{\nu})V_{\tau,\nu},\quad l=0, \tag{4.8}$$

$$\begin{bmatrix}K(\hat{\tau},\bar{\nu}_1) & K(\hat{\tau},\bar{\nu}_2)\end{bmatrix}\approx K(\hat{\tau},\bar{\nu})W_{\tau,\nu},\quad l>0. \tag{4.9}$$

When $l=L$ and $\hat{\tau}=\tau$, no proxy rows are needed. We will discuss the choice of the proxy rows in more details in subsections 4.2.2, 4.2.3, and 4.2.4.

With all the interpolation and transfer matrices computed, the butterfly representation of $K(o,s)$ is:

$$K(o,s)\approx K^L W^L W^{L-1}\dots W^1 V^0. \tag{4.10}$$

Let $\nu_1,\nu_2,\dots,\nu_{2^{L-l}}$ denote the nodes at level $L-l$ of $\mathcal{T}_s$, and $\tau_1,\tau_2,\dots,\tau_{2^l}$ denote the nodes at level l of $\mathcal{T}_o$. The interpolation factor V^0, the transfer factors W^l for $l=1,\dots,L$, and the skeleton factor K^L are:

$$V^0 = \mathrm{diag}(V_{\tau,\nu_1},\dots,V_{\tau,\nu_{2^L}}),\quad (\tau,\nu_i)\text{ at level } l=0, \tag{4.11}$$

$$K^L = \mathrm{diag}(K(\tau_1,\bar{\nu}),\dots,K(\tau_{2^L},\bar{\nu})),\quad (\tau_i,\nu)\text{ at level } l=L, \tag{4.12}$$

$$W^l = \mathrm{diag}(W\tau_1,\dots,W\tau_{2^{l-1}}),\quad l=1,\dots,L, \tag{4.13}$$

$$W_{\tau_i} = \begin{bmatrix}\mathrm{diag}(W_{\tau_i^1,\nu_1},\dots,W_{\tau_i^1,\nu_{2^{L-l}}})\\ \mathrm{diag}(W_{\tau_i^2,\nu_1},\dots,W_{\tau_i^2,\nu_{2^{L-l}}})\end{bmatrix},\quad (\tau_i^{\{1,2\}},\nu_i)\text{ at level } l, \tag{4.14}$$

where τ_i^1 and τ_i^2 denote the children of τ_i. Note that V^0 and K^L contain 2^L diagonal blocks each with $O(r_{\tau\nu})$ nonzeros, and W^l contains 2^L blocks $W_{\tau,\nu}$ each with $O(r_{\tau\nu}^2)$ nonzeros. The construction of these blocks via (4.8) at each level l requires the computation of $O(n)$ submatrices (i.e., the left-hand side (LHS) of (4.8)) of sizes $O(r_{\tau,\nu})\times O(r_{\tau,\nu})$. From the discussion of (4.3) in subsection 4.1, each submatrix can be computed in $O(r_{\tau,\nu}N_I^2 + r_{\tau,\nu}^2 N_I)$ time, which is time-wise optimal assuming N_I constant. If $\max_{\tau,\nu} r_{\tau\nu}$ is $O(1)$, it is immediately clear that the butterfly representation (IDs and matrix entry computation) requires $O(n\log n)$ memory and CPU time. We will see that this is not the case for any of K^{v2v}, K^{s2v} and K^{s2s}, but quasi-linear complexities can still be attained for most of these operators.

In what follows, we discuss the rank estimate, proxy row selection, and computational complexity for each of the three discretized integral operators K^{v2v}, K^{s2v} and K^{s2s}.

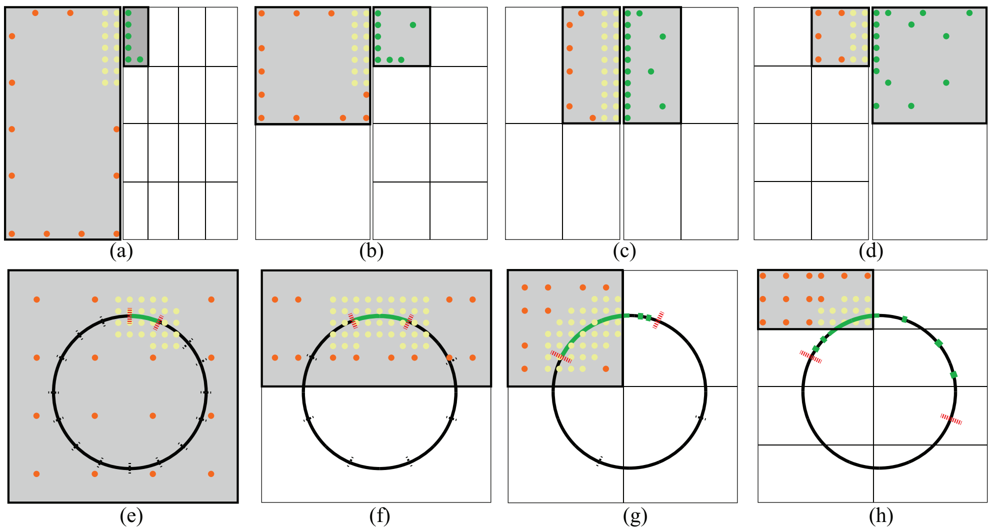

Fig. 4.2: Top: illustration of the butterfly compression of 4 levels for one top-level off-diagonal block of HODBF representation of K^{v2v} in a 2D computational domain. The subdomains at levels $l = 0, 1, 2, 3$ ((a)-(d)) are denoted by the vertical and horizontal lines. One subdomain pair (τ, ν) at each level with non-constant rank $r_{\tau,\nu}$ is shown in the shaded areas (left: τ, right: ν). The green dots correspond to skeleton columns (cell centroids) for the ID computation, and the red and yellow dots correspond to uniformly sampled and nearest neighboring proxy rows (cell centroids) used to compute the ID. Note that the unused cell centroids at each stage are not plotted. Bottom: illustration of the butterfly compression with 5 levels for K^{s2v} involving a circular inclusion in a 2D computational domain. The subdomains and arcs at levels $l = 0, 1, 2, 3$ ((e)-(h)) are denoted by the vertical/horizontal lines and dashed lines, respectively. One subdomain-arc pair (τ, ν) at each level with non-constant rank $r_{\tau,\nu}$ is shown with the shaded areas for the subdomain and curves ending with red dashed lines for the arc. The green dots correspond to skeleton columns (line segments) for the ID computation, and the red and yellow dots correspond to uniformly sampled and nearest neighboring proxy rows (cell centroids) used to compute the ID.

4.2.2. Computation of K^{v2v}. Consider the top level off-diagonal block of its HODBF representation, $K^{v2v}(o, s)$ with o and s being the children of the root of $\mathcal{T}_{H_v}$, and sizes $m = n = N_v/2$. This permits a butterfly representation with $L = L_v - 1$ levels. Letting $l_m = L/2$ denote the middle butterfly level, we can show that among the $O(N_v)$ subblocks $K(\tau, \nu)$ at each level l, there are $O(2^{|l-l_m|/d} N_v^{(d-1)/2d})$ subblocks representing interactions between adjacent or close-by geometry subdomains, where $d = 2$ or 3 denotes the problem dimension. Their ranks scale as $r_{\tau,\nu} = O(2^{-|l-l_m|/d} N_v^{(d-1)/2d})$ dominated by the interface DOFs between the two computational subdomains. Each of these non-constant rank subblocks requires $O(r_{\tau,\nu}^2)$ storage and matrix entry computation, and $O(r_{\tau,\nu}^3)$ ID cost. Thus, they require

$$\sum_l r_{\tau,\nu}^2 O(2^{|l-l_m|/d} N_v^{(d-1)/2d}) = O(N_v^{3(d-1)/2d}) \leq O(N_v) \quad \text{storage}$$

and

$$\sum_{l} r^3_{\tau,\nu} O(2^{|l-l_m|/d} N_v^{(d-1)/2d}) = O(N_v^{2(d-1)/d}) \quad \text{CPU time}$$

for the interpolation and transfer matrices. Specifically, the CPU time is $O(N_v)$ when $d = 2$ and $O(N_v^{4/3})$ when $d = 3$. The rest of the subblocks essentially has $r_{\tau,\nu} = O(1)$ and requires $O(N_v \log N_v)$ CPU time and memory based on the DOF analysis in [47]. Note that the sub-optimal CPU complexity for $d = 3$ can be improved by considering strong-admissible hierarchical matrices [26] to keep butterfly rank constant, or considering alternatives to ID, e.g., analytical interpolation schemes such as (4.1) or the one in [10], which brings the $O(r^3_{\tau,\nu})$ CPU cost per block back to $O(r^2_{\tau,\nu})$. However, we will not implement the analytical interpolation-based butterfly representation in this article due to the large prefactors in these schemes and will leave that as a future work.

As an example, Figure 4.2 (top) shows a 2D computational domain with a partitioning tree $\mathcal{T}_{H_v}$ with $L_v = 5$ levels. $K^{v2v}(o, s)$ with o and s being the children of the root $\mathcal{T}_{H_v}$ (i.e., the left and right half of the domain) is compressed as a 4-level butterfly. Figure 4.2(a)-(d) shows the subdomain pairs at levels $l = 0, 1, 2, 3$, respectively. For each level, one subdomain pair with non-constant interaction rank is shown in grey. The cell centroids in green represent the skeleton columns $\bar{\nu}$, which mainly lie on the subdomain interface.

As mentioned above, the computation of $\bar{\nu}$ is performed via (4.8) with proxy rows. Specifically, consider (τ, ν) at level $0 < l < L$ (with $l = 0$, L being similar). Let n_i denote the nearest neighboring centroids i (e.g., all centroids that are within a $10h$ distance of r_i), and let f_τ denote the $\chi|\bar{\nu}_1 \cup \bar{\nu}_2|$ uniformly selected centroids near the boundary of subdomain τ with an oversampling parameter χ. We choose the proxy rows as

$$\hat{\tau} = (\cup_{i \in \bar{\nu}_1 \cup \bar{\nu}_2} n_i) \cap \tau \cup f_\tau. \tag{4.15}$$

For example, Figure 4.2(a)-(d) shows the nearest neighboring proxy rows in yellow and the uniform proxy rows in red.

Because of the above rank estimate and proxy row selection scheme, $O(N_v \log N_v)$ CPU time and memory complexities can be achieved for the top-level off-diagonal block of the HODBF. This leads to an $O(N_v \log^2 N_v)$ complexity for the overall HODBF construction.

4.2.3. Computation of K^{s2v}. Consider K^{s2v} between a 2D computational domain with N_v cells and a curve inclusion with N_s discretization segments with typically $N_s = N_v^{1/2}$. K^{s2v} is compressed as a butterfly with $L = \min\{L_v, L_s\}$ levels. For simplicity we assume that $L_s = L_v$, each leaf node in $\mathcal{T}_s$ contains $O(1)$ columns, and each leaf node in $\mathcal{T}_v$ contains $O(N_v/N_s) = O(N_s)$ rows. Just like the K^{v2v} operator, we need to identify the subblocks with non-constant ranks $r_{\tau,\nu}$. We first identify a level l_m at which the side length of τ is on the same order as the length of ν. From $N_s^{1/2} 2^{(L-l_m)/2} = N_s 2^{l_m}$, where the LHS and right-hand-side (RHS) represent lengths of τ and ν at level l_m, respectively, we have $l_m = O(\frac{2L}{3})$. For each level $l \leq l_m$, we can show that there are $O(2^{-l} N_s)$ subblocks, out of the total of $O(N_s)$ subblocks, representing interactions between a node ν of $\mathcal{T}_s$ fully contained in or close to some node τ of $\mathcal{T}_o$. These subblocks have ranks at most $r_{\tau,\nu} = O(2^l)$ (i.e., the length of ν). These non-constant rank subblocks require $\sum_{l \leq l_m} r^2_{\tau,\nu} O(2^{-l} N_s) = O(N_s^{5/3}) = O(N_v^{5/6})$ storage and matrix entry computation, and $\sum_{l \leq l_m} r^3_{\tau,\nu} O(2^{-l} N_s) = O(N_s^{7/3}) = O(N_v^{7/6})$

CPU time for the IDs. On the other hand, for each level $l > l_m$, we can show that there are $O(2^{l/2})$ subblocks, out of the total of $O(N_s)$ subblocks, representing interactions between a node τ of $\mathcal{T}_o$ intersecting with some node ν of $\mathcal{T}_s$. These subblocks have ranks at most $r_{\tau,\nu} = O(2^{l/2} N_s)$ (i.e., the side length of τ). These non-constant rank subblocks again require $O(N_v^{5/6})$ storage and matrix entry computation, and $O(N_v^{7/6})$ CPU time for the IDs. Just like K^{v2v} in 3D, we can use analytical interpolation schemes to reduce $O(N_v^{7/6})$ to $O(N_v^{5/6})$. The rest of the subblocks essentially have $r_{\tau,\nu} = O(1)$ based on the DOF analysis in [47], and their CPU and memory complexities are dominated by $l = L$, which scale as $O(N_v)$.

As an example, Figure 4.2 (bottom) shows the compression of K^{s2v} representing interaction between a 2D computational domain and a circle inclusion, with $L = L_s = 5$ levels. Figure 4.2(e)-(h) shows the subdomain-arc pairs at levels $l = 0, 1, 2, 3$ respectively. For each level, one subdomain-arc pair with non-constant interaction rank is shown in grey for the subdomain and red for the arc. The discretization line segments in green on the curve represent the skeleton columns $\bar{\nu}$, which mainly lie on the intersection of the arc with the subdomain. Note that for $l = 0, 1$, the arc is fully contained in the subdomain, and the green points overlap with the red arc; for $l = 2, 3$, the subdomain intersects with the arc. Just like the K^{v2v} operator, proxy rows in the subdomains are chosen by (4.15), except that f_τ is a set of $\chi|\bar{\nu}_1 \cup \bar{\nu}_2|$ uniformly distributed centroids in the subdomains. In Figure 4.2(e)-(h), the nearest neighboring and uniform proxy rows (centroids) are shown in yellow and red, respectively.

4.2.4. Computation of K^{s2s} and its inverse. Just like K^{v2v}, we seek a HODBF representation of K^{s2s} for $d = 2$. Considering a top-level off-diagonal block $K^{s2s}(o, s)$, it has been shown in [40] that its butterfly rank scales as $O(\log N_s)$ and its CPU time and memory scales as $O(N_s \log N_s)$. As a result, the HODBF representation of K^{s2s} requires $O(N_s \log^2 N_s) = O(N_v^{1/2} \log^2 N_v)$ CPU time and storage units.

Once constructed, the inverse of the HODBF compressed K^{s2s} can be computed using algorithms described in [38, 37] leveraging sketching-based butterfly construction algorithms [39]. For K^{s2s}, the HODBF inversion requires $O(N_s^{3/2} \log N_s) = O(N_v^{3/4} \log N_v)$ based on the analysis in [38].

We summarize the algorithmic complexities in subsections 4.2.2, 4.2.3, and 4.2.4. The computation of K^{v2v} requires $O(N_v \log^2 N_v)$ memory for $d = 2, 3$, $O(N_v \log^2 N_v)$ time for $d = 2$, and $O(N_v^{4/3})$ time for $d = 3$ (the latter becoming quasi-linear if analytical interpolation rather than ID-based butterfly algorithms are used or strong-admissible hierarchical matrices are used). The computation of K^{s2v} for $d = 2$ requires $O(N_v)$ memory and $O(N_v^{7/6})$ CPU time ($O(N_v)$ attainable if the analytical interpolation is used). The computation and inversion of K^{s2s} for $d = 2$ require $O(N_v^{1/2} \log^2 N_v)$ memory and $O(N_v^{3/4} \log N_v)$ CPU time. Therefore, the overall complexities of the proposed butterfly compressed HB integrator scale as $O(N_v \log^2 N_v)$.

4.3. Expected convergence behavior. Although a complete numerical analysis of our new method is an ongoing work, we sketch the expected convergence behavior of the algorithm.

Our new algorithm has three principal sources of numerical errors: the first source is due to numerical computation of HB ingredients, such as τ, v_0 and v_1; the second one is due to truncating the infinite asymptotic series to keep only the leading or the first two terms: $N = 0$ or $N = 1$ in (2.9); the third one is due to the butterfly compression.

Therefore, the overall error of our HB integrator will be roughly controlled by the above three sources of errors.

To start with, we consider the first source of errors. According to equation (2.5), v_1 depends on Δv_0 (the Laplacian of v_0) and $\Delta\tau$. To have first-order accurate v_1 (which is a minimum requirement for our current setup of the new algorithm), we need at least first-order accurate Δv_0 and $\Delta\tau$. To have first-order Δv_0, v_0 itself must be computed to third-order accuracy so that it can be numerically differentiated twice to yield first-order accurate Laplacian Δv_0. To have third-order accurate v_0, $\Delta\tau$ (the Laplacian of τ) must be computed to third-order accuracy according to equation (2.5) again, which in turn implies that τ itself must be computed to fifth-order accuracy. Consequently, we have chosen to apply the fifth-order Lax-Friedrichs WENO scheme to compute τ, v_0 and v_1 in our current implementation so that the computed τ, v_0 and v_1 will have fifth-, third-, and first-order accuracy, respectively. We denote the computational errors of these ingredients as $O(h_0^\beta)$, where h_0 is the mesh size, and β is the order of accuracy: $\beta = 1, 3, 5$, respectively. Since these ingredients are independent of the frequency parameter ω, they can be computed just once and re-used for many different frequencies as explained below.

Now we consider the second source of errors. According to equation (2.9), when we keep the leading-order term of the HB ansatz, we are expecting the first-order asymptotic convergence in the form of $O\left((1/\omega)^{(N+1-\frac{d-3}{2})}\right)$, where $N = 0$; when we keep the first two terms of the HB ansatz, we are expecting the second-order asymptotic convergence in the form of $O\left((1/\omega)^{(N+1-\frac{d-3}{2})}\right)$, where $N = 1$.

In addition, we consider the error due to the butterfly compression. Since a detailed analysis of the butterfly compression algorithm with prescribed proxy rows is beyond the scope of the current work, we assume that the error is simply represented as $O(e_{\mathrm{bf}})$ to simplify the matter, where e_{bf} is a small positive constant. That said, one can refer to Section 5 of [39] for a detailed analysis of SVD-based butterfly algorithms.

Finally, when $d = 3$, by using some estimates provided in Appendix B, we can write the overall error of our algorithm roughly as

$$E_{\mathrm{total}} = O\left((1/\omega)^{(N+1-\frac{d-3}{2})}\right) + O(h_0^3) + O(\omega h_0^5) + O(e_{\mathrm{bf}}), \tag{4.16}$$

for $N = 0$, and

$$E_{\mathrm{total}} = O\left((1/\omega)^{(N+1-\frac{d-3}{2})}\right) + O(h_0^3) + O(\omega h_0^5) + O\left(\frac{h_0}{\omega}\right) + O(e_{\mathrm{bf}}), \tag{4.17}$$

for $N = 1$. When $d = 2$, we may use the analysis in [19] for the H-B ansatz to obtain analogous estimates. Note that here we ignore the errors of Chebyshev and cubic interpolations assuming that they do not dominate over those induced by the high-order LxF-WENO schemes.

We remark that the term $O(\omega h_0^5)$ in (4.16) captures the magnification of numerical phase errors by the frequency ω in the Hankel-based H-B ansatz, in which the phase function τ appears together with ω in the form of $\omega\tau$; this implies two things: first, given a set of computed H-B ingredients, we can reuse these ingredients for many different frequencies as long as the sum of the error terms $O(\omega h_0^5)$ and $O(h_0^3)$ is not dominant over the other errors; second, if it happens that the frequency ω is so large that the error from $O(\omega h_0^5) + O(h_0^3)$ is dominant over other errors for a given set of computed H-B ingredients, then we can always compute more accurate H-B ingredients on finer meshes so that the resulting error from $O(\omega h_0^5) + O(h_0^3)$ is not dominant. Note that such computation tabulates the H-B ingredients on the fixed Chebyshev nodes in the *off-line* stage and does not affect the computation time for the wave function in the *online* stage. Similar observations apply to the estimate (4.17).

Our numerical results demonstrate that the above error estimates are sound.

5. Numerical Results. This section provides several numerical examples to demonstrate the accuracy and efficiency of the proposed butterfly-compressed HB integrator when applied to 2D and 3D computational domains with both homogeneous and inhomogeneous media. The low-rank compression of the phase/HB coefficients in subsection 4.1 and the butterfly compression of K^{v2v}, K^{s2v}, K^{s2s} and $(K^{s2s})^{-1}$ in subsection 4.2 have been implemented with distributed-memory parallelism. Most of the experiments are performed on the Haswell nodes of the Cori machine, a Cray XC40, at NERSC in Berkeley, where each of the $2,388$ Haswell nodes has two 16-core Intel Xeon E5-2698v3 processors and 128GB of 2133MHz DDR4 memory. For most experiments we use 64 Haswell nodes for both the proposed algorithm and the reference FDFD solver. Part of the experiments are performed on development nodes at the High Performance Computing Center (HPCC) of MSU, where each node has two 2.4Ghz 20-core Intel Xeon Gold 6148 CPU and 377GB of RAM.

5.1. Accuracy comparison with FDFD. We first compare the performance of the proposed algorithm with that of state-of-the-art FDFD solvers for both 2D and 3D computational domains.

5.1.1. 2D domains without and with inclusion. We consider the following examples of homogeneous and inhomogeneous media:

- **Constant media**: the computational domain is $V = [0, 2]^2$ with $n(\mathbf{r}) = n(x, y) = 2$. The phase has an analytical form $\tau(\mathbf{r}, \mathbf{r}_0) = n(\mathbf{r}_0)|\mathbf{r}-\mathbf{r}_0|$. The HB coefficients have analytical formulas $v_0(\mathbf{r}, \mathbf{r}_0) = 1/(2\sqrt{\pi})$ and $v_1(\mathbf{r}, \mathbf{r}_0) = 0$. Therefore, the HB integrator becomes $g(\mathbf{r}, \mathbf{r}_0) = \frac{\mathrm{i}}{4}H_0^{(1)}(\omega n(\mathbf{r}_0)|\mathbf{r} - \mathbf{r}_0|)$, i.e., the well-known form of the free-space Green's function. We use high-order Lax-Friedrichs WENO methods with $h_0 = 0.01$ to solve equations (2.4) and (2.5) with point sources, construct their low-rank representation with an order of $n_I = 3$ for the Chebyshev interpolation, and compare the results with these exact formulas.
- **Constant-gradient media**: the computational domain is $V = [0, 1]^2$ with $n(\mathbf{r}) \equiv n(x, y) = \frac{1}{0.5-0.25(y-0.5)}$. Note that $1/n$ has a nonzero constant partial derivative in y. Let $\mathbf{r}_c = (0.5, 0.5)$. When the point source $\mathbf{r}_0$ is in the interior of V, the phase function has an analytical formula $\tau(\mathbf{r}, \mathbf{r}_0) = \frac{1}{|\mathbf{G_0}|}\operatorname{arccosh}\Big(1 + \frac{1}{2}n(\mathbf{r})n_0|\mathbf{G}_0|^2|\mathbf{r} - \mathbf{r}_0|^2\Big)$, where $\mathbf{G}_0 = [0, -0.25]$ and $n_0 = \frac{n(\mathbf{r}_c)}{1+n(\mathbf{r}_c)\mathbf{G}_0\cdot(\mathbf{r}_0-\mathbf{r}_c)}$. The HB coefficients have no known analytical expressions. We use the high-order Lax-Friedrichs(LxF)-WENO method with $h_0 = 0.0025$ to solve equation (2.4) and (2.5) with point sources, and construct the their low-rank representation with an order of $n_I = 13$ for the Chebyshev interpolation.
- **Sinusoidal media**: the computational domain is $V = [0, 1]^2$ with $n(\mathbf{r}) = n(x, y) = \frac{1}{1+0.2\sin(\pi(x+0.05))\sin(0.5\pi y)}$. In this case, both the phase and the HB coefficients have no known analytical expressions. We use the high-order LxF-WENO method with $h_0 = 0.01$ to solve (2.4) and (2.5) with point sources, and construct their low-rank representation with an order of $n_I = 13$ for the Chebyshev interpolation. Since the exact solutions are not available, we use FDFD solutions as references and compare the corresponding results

accordingly.

- **Waveguide media**: the computational domain is $V = [0,1]^2$ with $n(\mathbf{r}) = n(x,y) = \frac{1}{1-0.5e^{-2.0(x-0.5)^2}}$. In this case, both the phase and the HB coefficients have no known analytical expressions. We first use the high-order LxF-WENO method with $h_0 = 0.01$ to solve (2.4) and (2.5) with point sources, and construct the low-rank representation with an order of $n_I = 13$ for the Chebyshev interpolation. We then compare the results with the FDFD solutions.

For the inclusion (if present), we consider a semi-circle of radius 0.5 and an open square of side length 0.8 centered at the domain center. When computing K^{v2v}, K^{s2v}, and K^{s2s}, let n_p denote the number of points per wavelength (PPW) for the discretization of the computational domain or the curve inclusion. The computational domain is discretized with $h = 2\pi/(\omega n_{\max} n_p)$ with $n_p = 10$ and $n_{\max}$ being the maximum refractive index over the domain. Similarly, the curve inclusion is discretized with $w_i = 2\pi/(\omega n_{\max} n_p)$ with $n_p = 100 \sim 500$ to ensure highly accurate approximation for K^{s2s}. The butterfly and HODBF compression of K^{v2v}, K^{s2v} and K^{s2s} are computed with tolerance $\text{tol} = 10^{-8}$ in (4.4) and oversampling factor $\chi = 20$ in (4.15).

Once the discretized integral operators are computed, we apply (3.3) and (3.11) to the following source functions (i.e., RHSs) centered at $\mathbf{r}_c = (x_c, y_c)$ (the domain center):

- **Point source:** $s(\mathbf{r}) = 1/h^2$ if $\mathbf{r}$ is inside the source cell centered at $\mathbf{r}_c$.
- **Gaussian wavepacket source:** $s(\mathbf{r}) = \exp(-|\mathbf{r}-\mathbf{r}_c|^2/(2\sigma^2))\exp(i\omega_0(\mathbf{r}\cdot\mathbf{d}))t(|\mathbf{r}-\mathbf{r}_c|, w_1, w_2)$ with $\sigma = 0.15$, $w_1 = 0.3$, $w_2 = 0.1$, $\omega_0 = 0.9\omega$, and $\mathbf{d} = \frac{1}{\sqrt{2}}[1,1]$. Here $t(x, w_1, w_2)$ is the cosine tapering function:

$$t(x, w_1, w_2) = 0.5(1 + \cos((x - w_1)\pi/w_2)) \text{ if } w_1 < x < w_2, \tag{5.1}$$

  and $t(x, w_1, w_2) = 1$ if $x \le w_1$, and $t(x, w_1, w_2) = 0$ if $x \ge w_2$.
- **Concave kite-shaped source:** Let a kite-shaped curve be $\partial\Omega = \{(x,y) : x(t) = b(\cos(t) + 0.65\cos(2t) - 0.65) + x_c,\ y(t) = 1.5b\sin(t) + y_c \text{ for } 0 \le t \le 2\pi\}$ with a scaling factor $b = 0.2$. $s(\mathbf{r}) = 1$ if $\mathbf{r} \in \Omega$ and $s(\mathbf{r}) = t(|\mathbf{r}-\mathbf{r}_c|, w_1, w_2)$ if $\mathbf{r} \notin \Omega$ with the tapering function in (5.1). Here $w_2 = 0.1$, $w_1 = |\mathbf{r}_{\min}|$, and $\mathbf{r}_{\min}$ is the point on $\partial\Omega$ closest to the line $\mathbf{r} - \mathbf{r}_c$.

Note that although the current form of HB ansatz (2.9) needs further modification to handle media permitting presence of caustics, we can still model caustics induced by the interaction of the RHS (the source) with the medium, as shown in the two examples: the concave kite-shaped source and the open cavity inclusion; both cases will induce caustics in the wave field as we will see.

As for the reference FDFD solver for computing wavefields $u_{\text{fd}}(\mathbf{r})$, we use the 9-point staggered grid scheme in [12]. The computational domain is extended in each direction with a perfectly matched layer (PML) of thickness $8\pi/(\omega n_{\max})$ (i.e., 4 wavelengths). The extended domain is discretized with $h = 2\pi/(\omega n_{\max} n_p)$ with PPW $n_p = 10 \sim 50$. The resulting sparse linear system is solved with a multi-frontal sparse direct solver STRUMPACK [23, 37]. When inclusion is present, the rows and columns of the system which represent grid cells overlapping with the inclusion are removed from the system. This can introduce significant staircase approximation errors to the inclusion, unless more sophisticated subgridding techniques are used. Nevertheless, by careful implementation, we still manage to produce good FDFD results in the examples that we are going to show.

For the constant medium, we first consider $\omega = 80\pi$, which amounts to 160 wavelengths in each direction. The fields computed by the proposed scheme (PPW=10) and FDFD (PPW=10, 50) and their differences are shown in Figure 5.1. Note that for the point source without inclusion (Row 1 in Figure 5.1), the exact solution is also plotted (in dashed green) in the third column. One can clearly see that the solution by the proposed scheme matches well with the exact solution, while it requires PPW=50 or higher for FDFD to achieve a similar order of accuracy. For the kite-shaped source (Row 2 in Figure 5.1), the concave shape can induce caustics, which are well-captured by the proposed scheme. FDFD matches better with the proposed scheme if PPW=50 other than PPW=10 is used. For the Gaussian wavepacket source with the semi-circle inclusion (Row 3 in Figure 5.1), FDFD results match poorly with the proposed scheme even using PPW=50, particularly near the inclusion, due to the stair-case error in FDFD (this is the case for all the source functions considered). As a workaround, we consider the open square inclusion (Row 4 in Figure 5.1) for which FDFD introduces no staircase error. Again, FDFD requires PPW=50 or higher to achieve a similar accuracy as the proposed scheme. Note that the square inclusion permits a hierarchical matrix representation of K^{s2s} as an alternative to HODBF, but this is not considered in this paper.

Based on these experiments, we further consider $\omega = 320\pi$, which amounts to 640 wavelengths in each direction. The fields computed by the proposed scheme (PPW=10) and FDFD (PPW=10) and their differences are shown in Figure 5.2. We note that the FDFD solver with PPW=10 already results in a sparse system of dimension $387,223,684$, and denser discretization causes memory crashes. As a comparison, the proposed scheme results in a dense, butterfly compressed K^{v2v} matrix of dimension $N_v = 40,972,801$. One can see from the point source case (Row 1 in Figure 5.2) that the proposed scheme is still very accurate when changing from $\omega = 80\pi$ to $\omega = 320\pi$, but FDFD suffers from dispersion errors. From the Gaussian wavepacket source without and with inclusion (Row 2-3 in Figure 5.2), it is clear that PPW=10 for FDFD does not give satisfactory results.

For the constant-gradient medium, we consider $\omega = 50\pi$ and $\omega = 100\pi$, which amount to 100 and 200 wavelengths in each direction, respectively. The results are shown in Figure 5.3 and Figure 5.4. It is not hard to see that for the point source, kite source, and Gaussian wavepacket source without inclusion (Row 1 and 2 in Figure 5.3 and Figure 5.4), FDFD requires at least PPW=50 to achieve a similar accuracy as the proposed scheme. However for $\omega = 100\pi$, FDFD with PPW=20 already leads to a sparse system of dimension $193,710,724$. In contrast, the proposed scheme leads to a dense, butterfly compressed K^{v2v} matrix of dimension $N_v = 16,008,001$. Similar conclusions can be drawn for the Gaussian wavepacket source with inclusions (Row 3 and 4 in Figure 5.3 and Figure 5.4).

To see that our method can be applied to generic inhomogeneous media, we test two more models: the sinusoidal model and the waveguide model.

For the sinusoidal model, we first consider $\omega = 80\pi$, which amounts to 50 wavelengths in each direction, and the results are shown in the first three rows of Figure 5.5. It is not hard to see that for the point source and Gaussian wavepacket source without inclusion (Row 1 and 2 in Figure 5.5), the proposed scheme can achieve a similar accuracy as FDFD with PPW=50; similar conclusions can be drawn for the Gaussian wavepacket source with inclusions (Row 3 in Figure 5.5). In addition, we also show the result in Row 4 of Figure 5.5 when $\omega = 40\pi$ for the concave kite-shaped source, and we have chosen this particular frequency so that caustic effects are apparent near the concave region.

For the waveguide model, we consider $\omega = 40\pi$, which amounts to 40 wavelengths in each direction. We compute wavefields for three different types of sources: the point source, the Gaussian wavepacket source, and the concave kite-shaped source, and the results are shown in Row 1 to 3 in Figure 5.6. The FDTD solutions are computed as references. It can be seen that the proposed scheme can achieve a similar accuracy as FDFD with PPW=50.

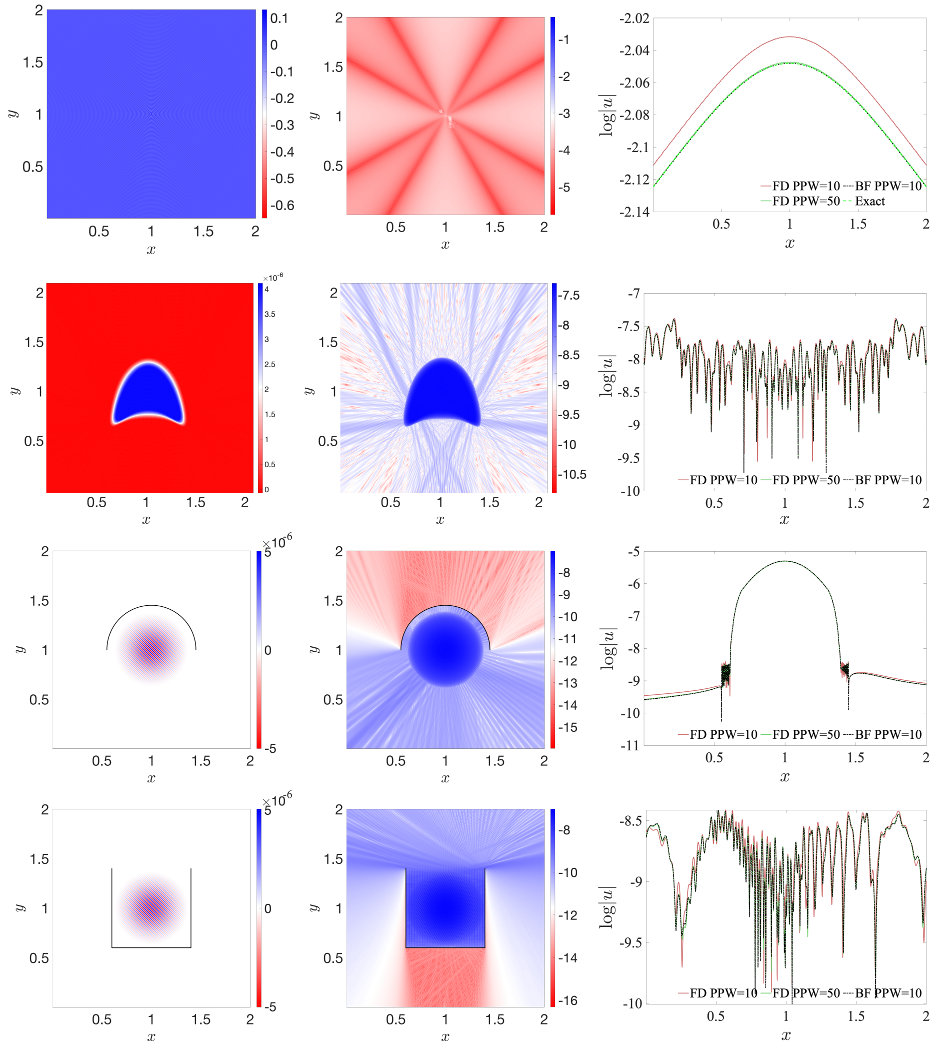

Fig. 5.1: Constant media. $\omega = 80\pi$ (160 wavelengths each direction). Left column: the field $\mathrm{Re}(u_{\mathrm{hb}})$ (in linear scale) computed by the proposed scheme. Middle column: difference $|u_{\mathrm{hb}} - u_{\mathrm{fd}}|$ (in log scale) between the fields computed by the proposed scheme (PPW=10) and FDFD (PPW=50). Right column: the fields $|u_{\mathrm{hb}}|, |u_{\mathrm{fd}}|, |u_{true}|$ (in log scale) drawn along the line $y = y_{\mathrm{post}}$. Row 1: point source with $y_{\mathrm{post}} = 2 - 10h$ and h corresponding to PPW=10. Row 2: kite-shaped source with $y_{\mathrm{post}} = 2 - 10h$. Row 3: Gaussian packet source with $y_{\mathrm{post}} = 1 - 10h$ and a semi-circle inclusion. Row 4: Gaussian packet source with $y_{\mathrm{post}} = 2 - 10h$ and an open-square inclusion.

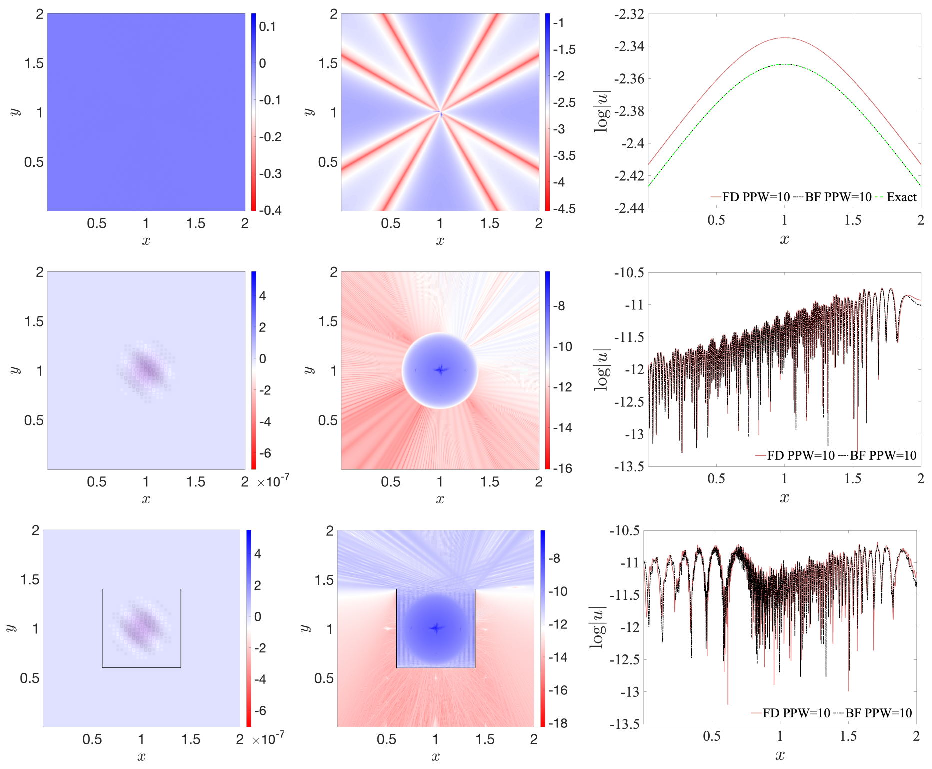

Fig. 5.2: Constant media. $\omega = 320\pi$ (640 wavelengths each direction). Left column: the field $\mathrm{Re}(u_{\mathrm{hb}})$ (in linear scale) computed by the proposed scheme. Middle column: difference $|u_{\mathrm{hb}} - u_{\mathrm{fd}}|$ (in log scale) between the fields computed by the proposed scheme (PPW=10) and FDFD (PPW=10). Right column: the fields $|u_{\mathrm{hb}}|, |u_{\mathrm{fd}}|, |u_{true}|$ (in log scale) drawn along the line $y = y_{\mathrm{post}}$. Row 1: point source with $y_{\mathrm{post}} = 2 - 10h$ and h corresponding to PPW=10. Row 2: Gaussian packet source with $y_{\mathrm{post}} = 2 - 10h$. Row 3: Gaussian packet source with $y_{\mathrm{post}} = 2 - 10h$ and an open-square inclusion.

5.1.2. 3D domain without inclusion. We consider the following examples of homogeneous and inhomogeneous media for $d = 3$:

- **Constant media**: the computational domain is $V = [0, 0.5]^3$ with $n(\mathbf{r}) = n(x, y, z) = 2$. The phase function has an analytical formula $\tau(\mathbf{r}, \mathbf{r}_0) = n(\mathbf{r}_0)|\mathbf{r} - \mathbf{r}_0|$. The HB coefficients have analytical form $v_0(\mathbf{r}, \mathbf{r}_0) = n(\mathbf{r}_0)/(2\pi)$ and $v_1(\mathbf{r}, \mathbf{r}_0) = 0$. The HB integrator becomes $g(\mathbf{r}, \mathbf{r}_0) = \exp(\mathrm{i}\omega n(\mathbf{r}_0)|\mathbf{r} - \mathbf{r}_0|)/(4\pi|\mathbf{r} - \mathbf{r}_0|)$, i.e., the well-known form of the free-space Green's function. We use the Lax-Friedrichs WENO method with $h_0 = 0.01$ to solve (2.4) and (2.5) with point sources, construct the low-rank representation with an order of $n_I = 3$ for the Chebyshev interpolation, and compare the results with these exact formulas.
- **Constant-gradient media**: the computational domain is $V = [0, 0.52]^3$ with $n(\mathbf{r}) = n(x, y, z) = \frac{1}{0.5 - 0.8(y - 0.26)} = \frac{-1.25}{y - 0.885}$. Note that $1/n$ has a nonzero constant partial derivative in y. The phase function has an analytical form $\tau(\mathbf{r}, \mathbf{r}_0) = \frac{1}{|\mathbf{G}_0|}\mathrm{arccosh}\left(1 + \frac{1}{2}n(\mathbf{r})n(\mathbf{r}_0)|\mathbf{G}_0|^2|\mathbf{r} - \mathbf{r}_0|^2\right)$ with $\mathbf{G}_0 =$

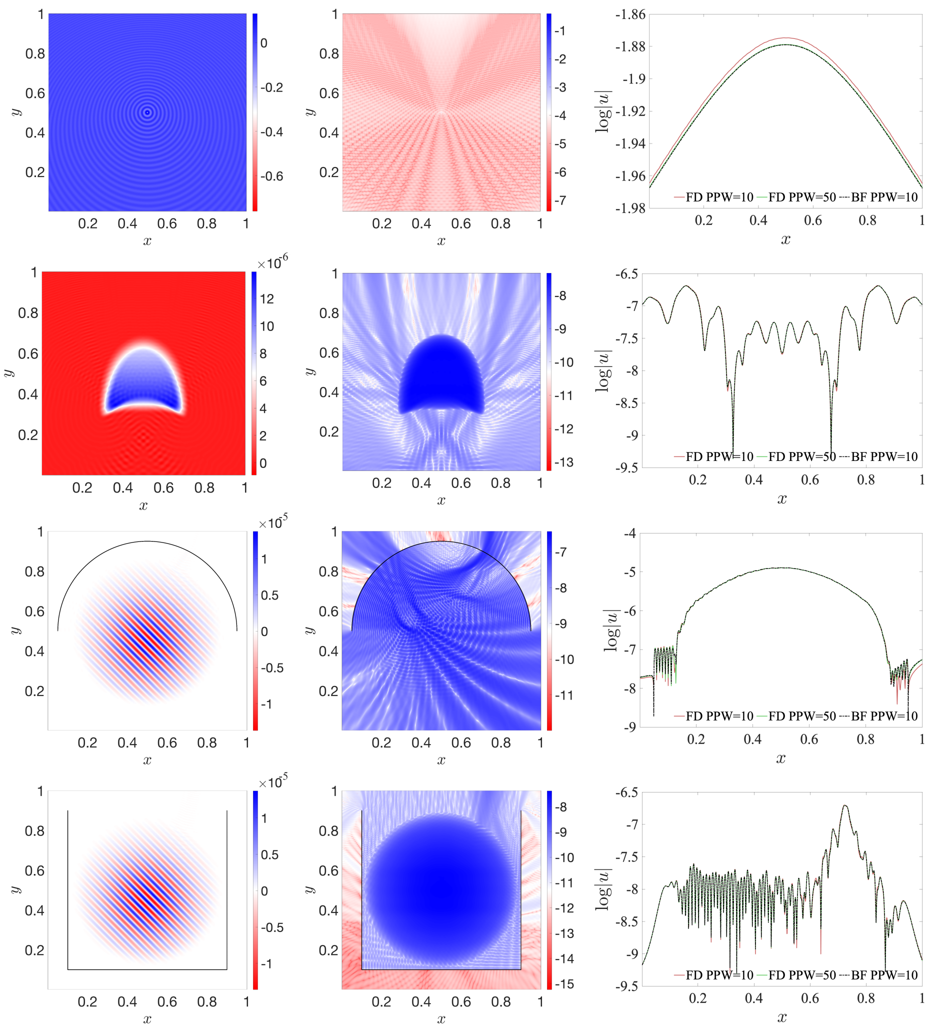

Fig. 5.3: Constant-gradient media. $\omega = 50\pi$ (100 wavelengths each direction). Left column: the field $\mathrm{Re}(u_{\mathrm{hb}})$ (in linear scale) computed by the proposed scheme. Middle column: difference $|u_{\mathrm{hb}} - u_{\mathrm{fd}}|$ (in log scale) between the fields computed by the proposed scheme (PPW=10) and FDFD (PPW=50). Right column: the fields $|u_{\mathrm{hb}}|, |u_{\mathrm{fd}}|$ (in log scale) drawn along the line $y = y_{\mathrm{post}}$. Row 1: point source with $y_{\mathrm{post}} = 1 - 10h$ and h corresponding to PPW=10. Row 2: kite-shaped source with $y_{\mathrm{post}} = 1 - 10h$. Row 3: Gaussian packet source with $y_{\mathrm{post}} = 0.5 - 10h$ and a semi-circle inclusion. Row 4: Gaussian packet source with $y_{\mathrm{post}} = 1 - 10h$ and an open-square inclusion.

$[0, -0.8, 0]$. The HB coefficients have no known analytical expressions. We use the fifth-order LxF WENO method with $h_0 = 0.01$ to solve (2.4) and (2.5) with point sources, and construct the low-rank representation with an order of $n_I = 9$ for the Chebyshev interpolation. It is worth mentioning that the exact Green's function [28] exists as $g(\mathbf{r}, \mathbf{r}_0) = \frac{(|y-c||y_0-c|)^{1/2}}{2\pi RR'} \exp\left(2\mathrm{i}(a^2\omega^2 - 1/4)^{1/2}\mathrm{arctanh}(R/R')\right)$. Here $a = 1.25$, $c = 0.885$, $R = |\mathbf{r} - \mathbf{r}_0|$, and

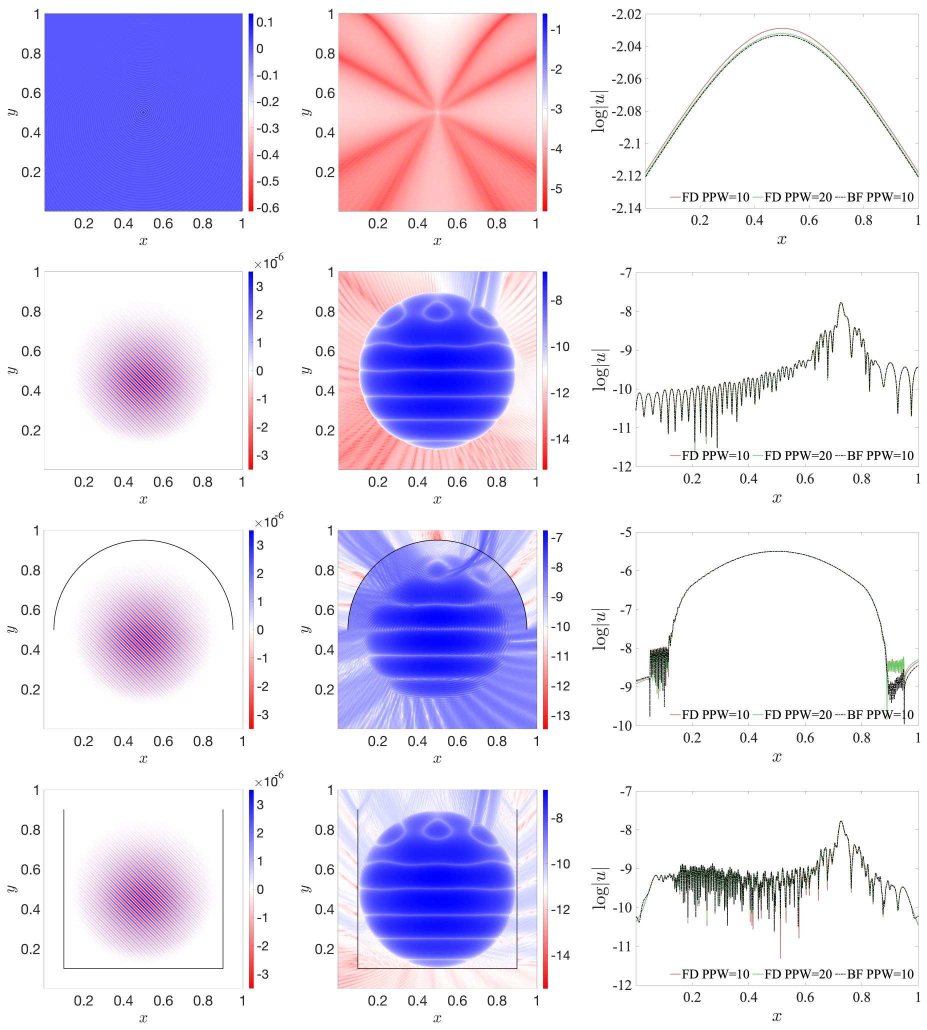

Fig. 5.4: Constant-gradient media. $\omega = 100\pi$ (200 wavelengths each direction). Left column: the field $\mathrm{Re}(u_{\mathrm{hb}})$ (in linear scale) computed by the proposed scheme. Middle column: difference $|u_{\mathrm{hb}} - u_{\mathrm{fd}}|$ (in log scale) between the fields computed by the proposed scheme (PPW=10) and FDFD (PPW=20). Right column: the field $|u_{\mathrm{hb}}|, |u_{\mathrm{fd}}|$ (in log scale) drawn along the line $y = y_{\mathrm{post}}$. Row 1: point source with $y_{\mathrm{post}} = 1 - 10h$ and h corresponding to PPW=10. Row 2: Gaussian packet source with $y_{\mathrm{post}} = 1 - 10h$. Row 3: Gaussian packet source with $y_{\mathrm{post}} = 0.5 - 10h$ and a semi-circle inclusion. Row 4: Gaussian packet source with $y_{\mathrm{post}} = 1 - 10h$ and an open-square inclusion.

$$R' = \sqrt{(x-x_0)^2 + (y+y_0-2c)^2 + (z-z_0)^2}.$$

To compute K^{v2v}, we discretize the domain with $h = 2\pi/(\omega n_{\max} n_p)$ with $n_p \sim 5$ and $n_{\max}$ being the maximum refractive index over the domain. The HODBF compression of K^{v2v} is computed with tolerance tol $= 10^{-6}$ in (4.4) and oversampling factor $\chi = 4$ in (4.15). We apply (3.3) to the following RHSs centered at $\mathbf{r}_c$ (the domain center):

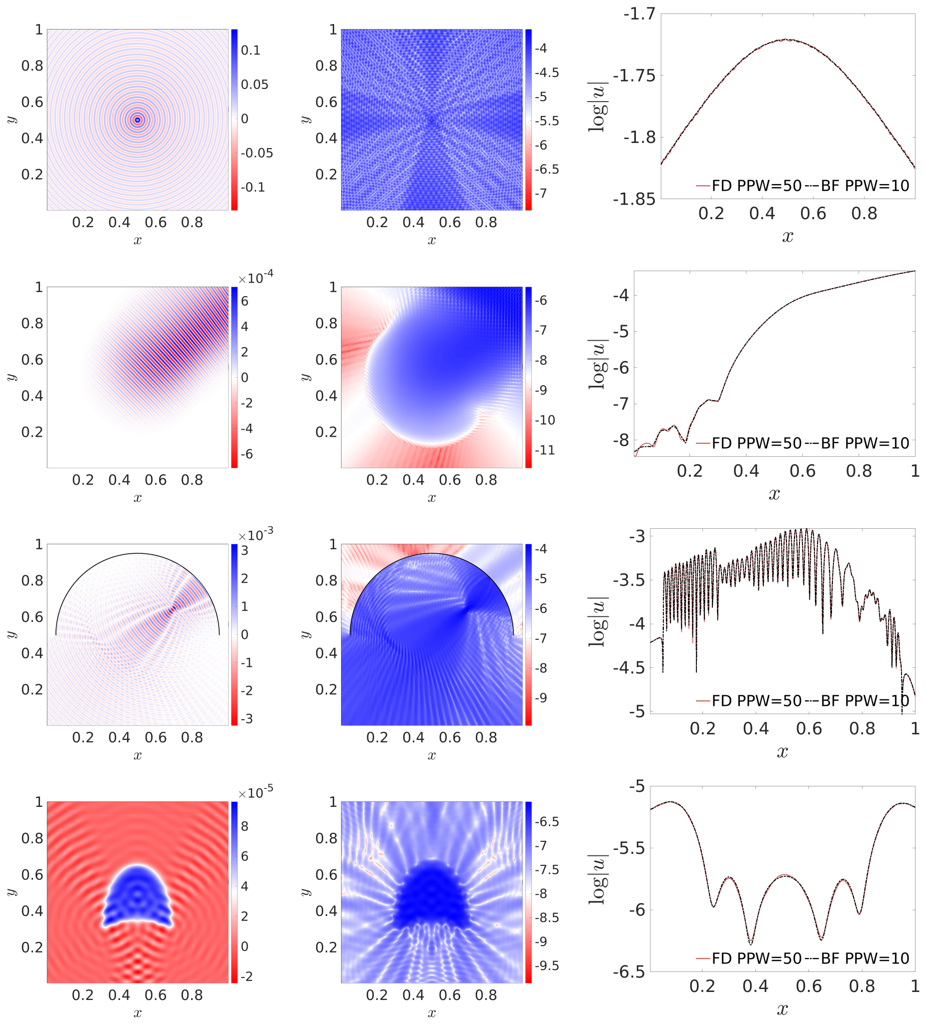
Fig. 5.5: Sinusoidal media. $\omega = 80\pi$ (50 wavelengths each direction) for the first three rows and $\omega = 40\pi$ (25 wavelengths each direction) for the last row. Left column: the field $\mathrm{Re}(u_{\mathrm{hb}})$ (in the linear scale) computed by the proposed scheme. Middle column: difference $|u_{\mathrm{hb}} - u_{\mathrm{fd}}|$ (in the log scale) between the fields computed by the proposed scheme (PPW=10) and FDFD (PPW=50). Right column: the fields $|u_{\mathrm{hb}}|, |u_{\mathrm{fd}}|$ (in the log scale) drawn along the line $y = y_{\mathrm{post}}$. Row 1: point source with $y_{\mathrm{post}} = 1-10h$ and h corresponding to PPW=10. Row 2: Gaussian packet source with $y_{\mathrm{post}} = 1 - 10h$. Row 3: Gaussian packet source with $y_{\mathrm{post}} = 0.5$ and a semi-circle inclusion. Row 4: concave kite-shaped source with $y_{\mathrm{post}} = 1 - 10h$.

- **Point source:** $s(\mathbf{r}) = 1/h^3$ if $\mathbf{r}$ is inside the source cell centered at $\mathbf{r}_c$.
- **Gaussian wavepacket source:** $s(\mathbf{r}) = \exp(-|\mathbf{r} - \mathbf{r}_c|^2/(2\sigma^2))\exp(i\omega_0(\mathbf{r} \cdot \mathbf{d}))t(|\mathbf{r} - \mathbf{r}_c|, w_1, w_2)$ with $\sigma = 0.15$, $w_1 = 0.05$, $w_2 = 0.05$, $\omega_0 = 0.9\omega$, $\mathbf{d} = \frac{1}{\sqrt{3}}[1, 1, 1]$, and t is the tapering function in (5.1).

As for the reference FDFD solver, we use the 27-point staggered grid scheme in [51]. The computational domain is extended in each direction with a perfectly matched

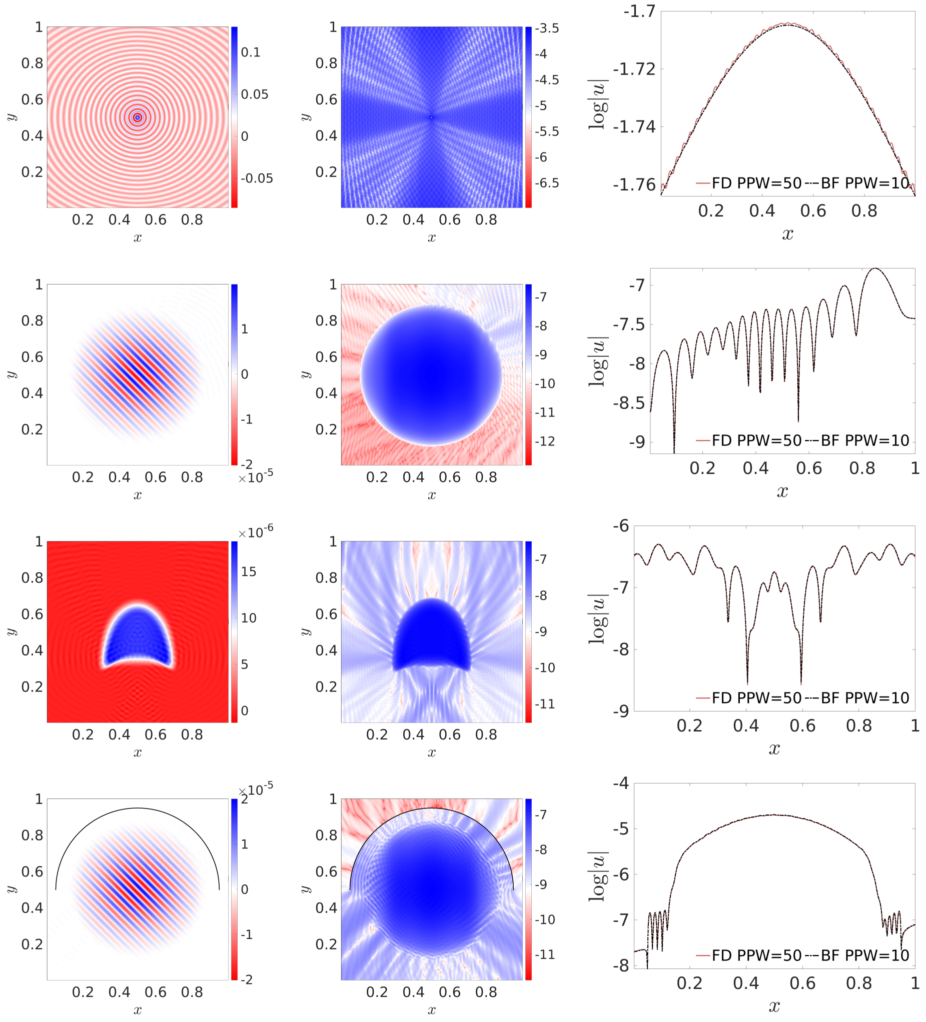

Fig. 5.6: Waveguide media. $\omega = 40\pi$ (40 wavelengths each direction). Left column: the field $\mathrm{Re}(u_{\mathrm{hb}})$ (in the linear scale) computed by the proposed scheme. Middle column: difference $|u_{\mathrm{hb}} - u_{\mathrm{fd}}|$ (in the log scale) between the fields computed by the proposed scheme (PPW=10) and FDFD (PPW=50). Right column: the fields $|u_{\mathrm{hb}}|, |u_{\mathrm{fd}}|$ (in the log scale) drawn along the line $y = y_{\mathrm{post}}$. Row 1: point source with $y_{\mathrm{post}} = 1 - 10h$ and h corresponding to PPW=10. Row 2: Gaussian packet source with $y_{\mathrm{post}} = 1 - 10h$. Row 3: concave kite-shaped source with $y_{\mathrm{post}} = 1 - 10h$. Row 4: Gaussian packet source with $y_{\mathrm{post}} = 0.5$ and a semi-circle inclusion.

layer (PML). The resulting sparse linear system is solved with STRUMPACK [23, 37].

For the constant medium, we use PMLs of thickness $3\pi/(\omega n_{\max})$ (1.5 wavelengths). We consider $\omega = 40\pi$ (20 wavelengths in each direction). The fields computed by the proposed scheme (PPW=5) and FDFD (PPW=10) and their differences are shown in Figure 5.7. For the point source (Row 1 in Figure 5.7), the exact solution is also plotted (in dashed green) in the third column. We can see that for both the point source and Gaussian wavepacket source, the results from FDFD with PPW=10 are not satisfactory. The FDFD scheme with PPW=10 results in a sparse system of dimension $240^3 = 13,824,000$, and PPW=20 will lead to $480^3 = 110,592,000$. In contrast, the proposed scheme results in a dense, compressed matrix of dimension $N_v = 1,030,301$.

For the constant-gradient medium, we use PMLs of thickness $10\pi/(\omega n_{\max})$ (5 wavelengths). We first consider $\omega = 32\pi$ (27 wavelengths in each direction). The fields computed by the proposed scheme (PPW=4.56) and FDFD (PPW=9.12) and their differences are shown in Figure 5.8. For the point source (Row 1 in Figure 5.8), the exact solution is also plotted (in dashed green) in the third column. The FDFD scheme with PPW=9.12 results in a sparse system of dimension $352^3 = 43,614,208$. The proposed scheme results in a dense, compressed matrix of dimension $N_v = 2,248,091$. Next, we consider $\omega = 64\pi$ (54 wavelengths in each direction). The fields computed by the proposed scheme (PPW=4.56) and FDFD (PPW=4.56) and their differences are shown in Figure 5.8. For the point source (Row 3 in Figure 5.8), the exact solution is also plotted (in dashed green) in the third column. Clearly, the FDFD results are not reliable. The FDFD scheme with PPW=4.56 results in a sparse system of dimension $305^3 = 28,372,625$ and even higher PPWs for the FDFD scheme lead to large-scale linear systems of dimensions that no existing sparse direct solvers can deal with. In contrast, the proposed scheme results in a dense, compressed matrix of dimension $N_v = 17,779,581$.

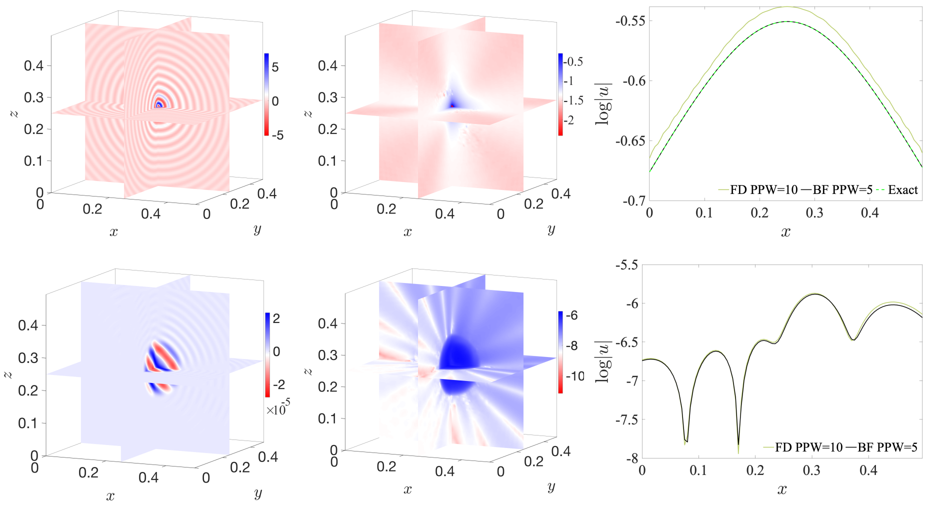

Fig. 5.7: Constant media in $d = 3$. $\omega = 40\pi$ (20 wavelengths each direction). Left column: the field $\mathrm{Re}(u_{\mathrm{hb}})$ (in linear scale) computed by the proposed scheme. Middle column: difference $|u_{\mathrm{hb}} - u_{\mathrm{fd}}|$ (in log scale) between the fields computed by the proposed scheme (PPW=5) and FDFD (PPW=10). Right column: the field $|u_{\mathrm{hb}}|, |u_{\mathrm{fd}}|, |u_{true}|$ (in log scale) drawn along the line $y = 0.5-10h$ and $z = 0.5-10h$ with h corresponding to PPW=5. Row 1: point source. Row 2: Gaussian packet source.

5.2. Convergence test. Next, we validate the convergence of high-order Lax-Friedrichs WENO schemes and the overall error estimates (4.16) and (4.17) of the HB ansatz using the 3-D constant-gradient model with a point source excitation, where $d = 3$. As mentioned in subsection 5.1.2, both the phase τ and the Green's function in such a medium have exact formulas.

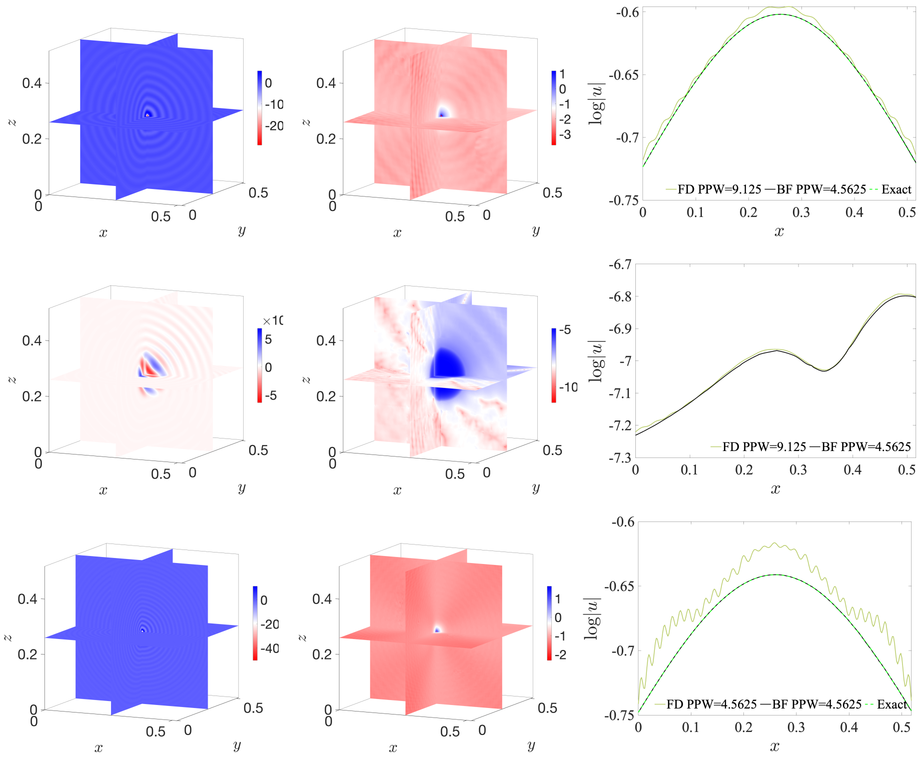

Fig. 5.8: Constant-gradient media in $d=3$. Left column: the field $\mathrm{Re}(u_{\mathrm{hb}})$ (in linear scale) computed by the proposed scheme. Middle column: difference $|u_{\mathrm{hb}}-u_{\mathrm{fd}}|$ (in log scale) between the fields computed by the proposed scheme and FDFD. Right column: the field $|u_{\mathrm{hb}}|, |u_{\mathrm{fd}}|, |u_{true}|$ (in log scale) drawn along the line $y=10h$ and $z=10h$ with h corresponding to PPW=4.56. Row 1: $\omega=32\pi$ (27 wavelengths each direction) with point source. Row 2: $\omega=32\pi$ (27 wavelengths each direction) with Gaussian packet source. Row 3: $\omega=64\pi$ (54 wavelengths each direction) with point source.

First, errors of the phase computed by the first-, third- and fifth-order Lax-Friedrichs WENO schemes with varying h_0 are shown in Figure 5.9 (left), which behave as $O(h_0^{\beta})$ with higher convergence order β for higher order WENO schemes. When we apply the fifth-order Lax-Friedrichs WENO scheme to compute τ, v_0 and v_1 are expected to have third-order and first-order accuracy, respectively, since v_0 and v_1 are computed from τ. Because exact solutions of v_0 and v_1 are unknown, we will use the third- and first-order accuracy of τ as the reference accuracy for v_0 and v_1, respectively.

Second, overall errors of wavefields (Green's functions) using the one-term ($N=0$) or two-term ($N=1$) HB ansatz (2.9) with the HB ingredients computed by the fifth-order Lax-Friedrichs WENO scheme are shown in Figure 5.9 (right), which behave as $O(\omega^{-1})$ and $O(\omega^{-2})$ as estimated by (4.16) and (4.17), respectively, for the one-term and two-term expansions.

When $N=1$, the HB coefficient v_1 only has first-order accuracy $O(h_0)$ which is dominant over the accuracy of τ and v_1, and thus the overall error E_{total} in (4.17)

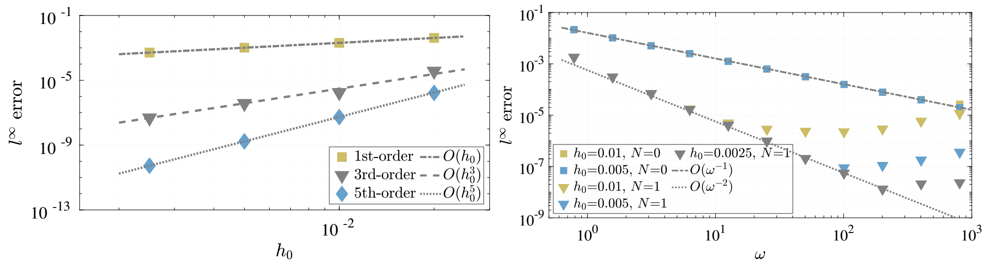

Fig. 5.9: Convergence test for a constant-gradient model in $d = 3$ with a point source. Left: Errors (w.r.t. the exact solution) of the phase function computed by the first-, third-, and fifth-order Lax-Friedrichs WENO schemes with varying h_0. Right: Errors (w.r.t. the exact solution; see (2.10)) of the wavefield using the one-term ($N = 0$) or two-term ($N = 1$) HB ansatz (2.9), where the phase function is computed by the fifth-order Lax-Friedrichs WENO scheme.

reduces to

$$E_{\text{total}} = O\left((1/\omega)^2\right) + O(h_0^3) + O(\omega h_0^5) + O\left(\frac{h_0}{\omega}\right), \tag{5.2}$$

where the butterfly compression is not used and thus its error does not appear in the above. When the $O((1/\omega)^2)$ term is larger than the other terms combined, the first error term dominates so that we can observe the second-order asymptotic convergence in $1/\omega$ clearly; however, once ω is so large that the sum of three terms, $O(h_0^3) + O(\omega h_0^5) + O(\frac{h_0}{\omega})$, dominates, the overall error nearly saturates since the $O(\omega h_0^5)$ term increases slowly as ω does. Such convergence behavior can be seen clearly in Figure 5.9.

When $N = 0$, v_1 disappears in the HB expansion and v_0 has third-order accuracy $O(h_0^3)$ which is dominant over that of τ; hence, the overall error E_{total} in (4.16) reduces to

$$E_{\text{total}} = O(1/\omega) + O(h_0^3) + O(\omega h_0^5), \tag{5.3}$$

where the butterfly compression is not used and thus its error does not appear in the above. When the $O(1/\omega)$ term is larger than the other terms combined, then the first error term dominates so that we can observe the first-order asymptotic convergence in $1/\omega$ clearly for a much broader band of ω. Such convergence behavior can be seen clearly in Figure 5.9.

5.3. Complexity validation.

In this subsection, we validate the CPU and memory complexities of the proposed scheme claimed in subsections 4.2.2, 4.2.3, and 4.2.4 using the constant and constant-gradient medium for $d = 2$ and $d = 3$.

5.3.1. 2D domains.

For the constant medium, we consider the computational domain $[0, 2]^2$ with an open square inclusion of side length $0.8m$, as described in subsection 5.1.1. The domain and the inclusion are discretized with PPW $n_p = 10$ and $n_p = 500$, respectively. We vary the frequency and cell count from $\omega = 10\pi$ and $N_v = 40,401$ to $\omega = 320\pi$ and $N_v = 40,972,801$, respectively. Note that $\omega = 320\pi$ corresponds to 640 wavelengths per direction. Each simulation uses 64 Cori Haswell nodes. The CPU time and memory requirement for computing K^{v2v}, K^{s2v} and K^{s2s} (and its inverse) are plotted in Figure 5.10 (top). Note that the value of N_s for each sample of N_v is not shown here. For the memory requirement, all discretized integral operators scale at most as $O(N_v \log^2 N_v)$ as expected. For the computational time, K^{v2v} and K^{s2s} scale at most as $O(N_v \log^2 N_v)$, and K^{s2v} scales as $O(N_v^{7/6})$, which can be further improved by additional matrix partitioning or analytical interpolation-based compression. We note that the time for K^{s2v} is about 8 times faster than K^{v2v} when $\omega = 320\pi$.

For the constant-gradient medium, we consider the computational domain $[0, 1]^2$ with an open square inclusion of side length 0.8, as described in subsection 5.1.1. The domain and the inclusion are discretized with PPW $n_p = 10$ and $n_p = 500$, respectively. We vary the frequency and cell count from $\omega = 12.5\pi$ and $N_v = 63,001$ to $\omega = 200\pi$ and $N_v = 16,008,001$, respectively. Note that $\omega = 200\pi$ corresponds to 400 wavelengths per direction. Each simulation uses 64 Cori Haswell nodes. The CPU time and memory requirement for computing K^{v2v}, K^{s2v} and K^{s2s} (and its inverse) are plotted in Figure 5.10 (bottom). Just like the constant medium, the memory requirement and CPU time mostly scale as at most $O(N_v \log^2 N_v)$.

5.3.2. 3D domains. For the constant medium, we consider the computational domain $[0, 0.5]^3$ as described in subsection 5.1.2. The domain is discretized with PPW $n_p = 5$. We vary the frequency and cell count from $\omega = 10\pi$ and $N_v = 26^3 = 17576$ to $\omega = 80\pi$ and $N_v = 201^3 = 8,120,601$, respectively. Note that $\omega = 80\pi$ corresponds to 40 wavelengths per direction. Each simulation uses 64 Cori Haswell nodes. The CPU time and memory requirement for computing K^{v2v} are plotted in Figure 5.11 (top). As estimated in subsection 4.2.2, the memory requirement scales as $O(N_v \log^2 N_v)$, and the CPU time scales as $O(N_v^{4/3})$. From subsection 4.2.2, the matrix entry computation requires $O(N_v \log^2 N_v)$ and the IDs require $O(N_v^{4/3})$. Both theoretical curves are plotted in Figure 5.11 (top). This sub-optimal CPU complexity for the IDs can be further improved via analytical interpolation schemes.

For the constant-gradient medium, we consider the domain $[0, 0.52]^3$ as described in subsection 5.1.2. The domain is discretized with PPW $n_p = 4.56$. We vary the frequency and cell count from $\omega = 8\pi$ and $N_v = 36^3 = 46656$ to $\omega = 64\pi$ and $N_v = 262^3 = 17,984,728$, respectively. Note that $\omega = 64\pi$ corresponds to 54 wavelengths per direction. Each simulation uses 64 Cori Haswell nodes. The CPU time and memory requirement for computing K^{v2v} are plotted in Figure 5.11 (top). The conclusion is very similar to the case of constant media.

6. Conclusion.

We present a fast and accurate scheme based on the Hadamard-Babich integrator for solving high-frequency Helmholtz equations in smooth, inhomogeneous media with arbitrary sources. The scheme low-rank compresses the phase and HB coefficients in the HB integrator with Chebyshev interpolation by solving their governing equations via Lax-Friedrichs WENO schemes with point sources located at the Chebyshev nodes. Once compressed, the phase and HB coefficients are used in the butterfly and HODBF compression of the resulting HB integrator discretized using N_v cells. Construction and application of the HB integrator require $O(N_v \log^2 N_v)$ CPU time and storage units. The scheme can also handle scattering problems involving sound-hard inclusion in the computational domain. In addition, the new scheme requires a much smaller number of discretization points per wavelength compared to finite-difference solvers. As a result, the scheme can model wave propagation for inhomogeneous media in so-far the largest 2D and 3D domains in terms of wavelength volume on a state-the-art supercomputer at Lawrence Berkeley National Laboratory. Future research direction includes extension of the proposed

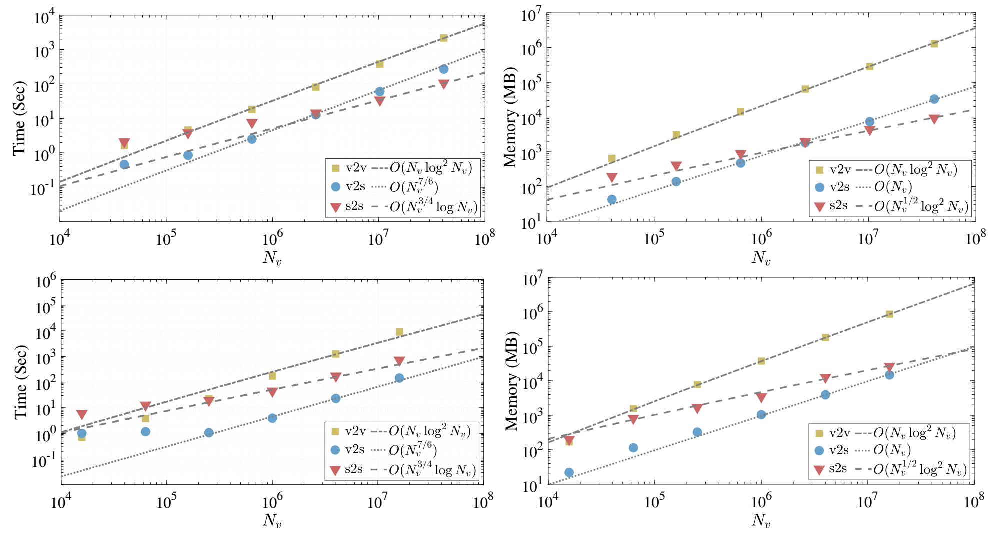

Fig. 5.10: CPU time (left) and storage units (right) for computing K^{v2v}, K^{s2v} and K^{s2s} (and its inverse) with problem dimension $d = 2$. (Top): constant media with an open square inclusion. The largest data point corresponds to 640 wavelengths per direction. (Bottom): constant-gradient media with an open square inclusion. The largest data point corresponds to 400 wavelengths per direction.

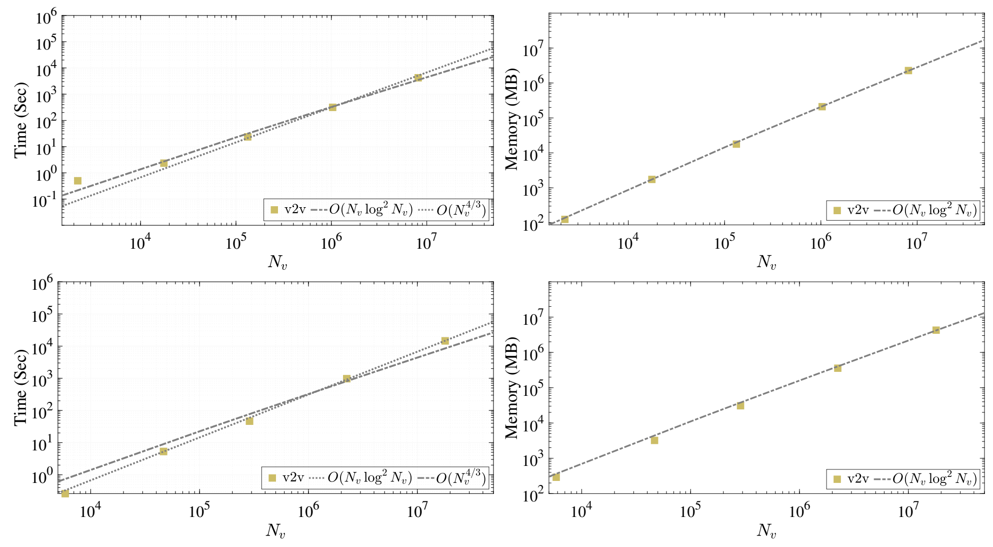

Fig. 5.11: CPU time (left) and storage units (right) for computing K^{v2v} with problem dimension $d = 3$. (Top): constant media. The largest data point corresponds to 40 wavelengths per direction. (Bottom): constant-gradient media. The largest data point corresponds to 54 wavelengths per direction.

scheme to non-smooth media or those media permitting presence of caustics, as well as to Maxwell's equations.

Acknowledgements. This research was supported in part by the Exascale Computing Project (17-SC-20-SC), a collaborative effort of the U.S. Department of Energy Office of Science and the National Nuclear Security Administration, and in part by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, Scientific Discovery through Advanced Computing (SciDAC) program through the FASTMath Institute under Contract No. DE-AC02-05CH11231 at Lawrence Berkeley National Laboratory. This research used resources of the National Energy Research Scientific Computing Center (NERSC), a U.S. Department of Energy Office of Science User Facility operated under Contract No. DE-AC02-05CH11231. Qian is partially supported by NSF (grants #2012046 and #2152011). Qian is grateful to Lexing Ying for his insightful comments and suggestions on this project. We are also grateful to anonymous reviewers for constructive comments and suggestions.

Appendix A. Computation of Self-interaction Terms.

A.1. 2-D self-interaction terms. By the H-B ansatz (2.9), we need to integrate the leading order term, which reduces to integrating

$$\begin{aligned} f_0(\omega,\tau) &= i\frac{\sqrt{\pi}}{2}H_0^{(1)}(\omega\tau(\mathbf{r},\mathbf{r}_0)) \\ &= i\frac{\sqrt{\pi}}{2}H_0^{(1)}(\omega n(\mathbf{r})|\mathbf{r}-\mathbf{r}_0|) \end{aligned} \tag{A.1}$$

over the cell c_j of size h with respect to $\mathbf{r}_0$, where $\mathbf{r}$ is the center of c_j. Here the leading HB coefficient is constant and is suppressed for now.

Lemma A.1. *The integral*

$$I(\mathbf{r}) = \int_{c_j} H_0^{(1)}(\omega n(\mathbf{r})|\mathbf{r}-\mathbf{r}_0|)d\mathbf{r}_0$$

can be reduced to evaluating

$$I(\mathbf{r}) = \frac{1}{(n_0\omega)^2}\left[8\int_0^{\frac{\pi}{4}} \frac{hn_0\omega}{2\cos\theta}H_1^{(1)}\left(\frac{hn_0\omega}{2\cos\theta}\right)d\theta + 4i\right], \tag{A.2}$$

where $n_0 = n(\mathbf{r})$, and we need to use the Bessel function to evaluate the above integral.

Proof. The integral I can be reduced to the integration over a cell of size h centered at the origin. We further partition this cell into eight equal triangles. By using the geodesic polar coordinates centered at the origin, the integral I can be reduced to evaluating the following integral over one triangle,

$$I(\mathbf{r}) = 8\int_0^{\frac{\pi}{4}} d\theta \int_0^{\frac{h}{2\cos\theta}} H_0^{(1)}(\omega n_0 r) r dr. \tag{A.3}$$

Using the following relation from [1], formula 9.1.30,

$$\frac{d}{dt}\left(t\, H_1^{(1)}(t)\right) = t\, H_0^{(1)}(t),$$

we consider the integral

$$\int_0^{\frac{h}{2\cos\theta}} H_0^{(1)}(\omega n_0 r)\; r\; dr = \frac{1}{(n_0\omega)^2}\int_0^{\frac{hn_0\omega}{2\cos\theta}} H_0^{(1)}(t)\; t\; dt$$

$$
\begin{aligned}
&= \frac{1}{(n_0\omega)^2}\int_0^{\frac{hn_0\omega}{2\cos\theta}} \frac{d}{dt}(tH_1^{(1)}(t))dt \\
&= \frac{1}{(n_0\omega)^2}\left[\frac{hn_0\omega}{2\cos\theta}H_1^{(1)}\left(\frac{hn_0\omega}{2\cos\theta}\right)+\frac{2i}{\pi}\right], \qquad \text{(A.4)}
\end{aligned}
$$

where we have used the formula 9.1.9 of [1] to obtain the constant term.

Now the integral I can be further reduced to

$$
I(\mathbf{r}) = \frac{1}{(n_0\omega)^2}\left[8\int_0^{\frac{\pi}{4}} \frac{hn_0\omega}{2\cos\theta}H_1^{(1)}\left(\frac{hn_0\omega}{2\cos\theta}\right)d\theta + 4i\right], \qquad \text{(A.5)}
$$

where we need to use the Bessel function to evaluate the above integral. This yields the formula (A.2). □

Therefore, we approximate the self term as

$$
\begin{aligned}
\int_{c_j} g(\mathbf{r},\mathbf{r}_0)d\mathbf{r}_0 &\approx \int_{c_j}\left[v_0(\mathbf{r},\mathbf{r}_0)\frac{i\sqrt{\pi}}{2}H_0^{(1)}(\omega n(\mathbf{r})|\mathbf{r}-\mathbf{r}_0|)\right]d\mathbf{r}_0 \\
&\approx v_0(\mathbf{r},\mathbf{r})\frac{i\sqrt{\pi}}{2}I(\mathbf{r}) = \frac{1}{2\sqrt{\pi}}\frac{i\sqrt{\pi}}{2}I(\mathbf{r}) = \frac{i}{4}I(\mathbf{r}), \qquad \text{(A.6)}
\end{aligned}
$$

where I is defined in (A.2).

A.2. 3-D self-interaction terms. Near the source in the 3-D case, from formulas (2.9) and (2.2) we have by keeping the leading-order term,

$$
g(\mathbf{r},\mathbf{r}_0) = n_0\frac{e^{i\omega\tau}}{4\pi\tau}, \qquad \text{(A.7)}
$$

where n_0 is the slowness at the source, and we refer to [56] for the reduction process.

Formula (A.7) represents the 3-D Green's function near the source. What we need is its integration over the 3-D cube of side h. We first consider a special case.

A.2.1. Integration of Hankel in 3-D: a special case. The basic idea of the calculation is to write the integral of (A.7) over a cube of side h with source point at the center, taken as the origin. The faces of the cube are the planes $x = \pm\frac{h}{2}$, $y = \pm\frac{h}{2}$, and $z = \pm\frac{h}{2}$.

Since we have

$$
\begin{aligned}
h_0^{(1)}(r) &= j_0(r) + iy_0(r) && [[1], 10.1.1] \\
&= \frac{\sin r - i\cos r}{r} && [[1], 10.1.11, 10.1.12] \\
&= -i\,\frac{e^{i\,r}}{r}, && \text{(A.8)}
\end{aligned}
$$

which is a scaled form of (A.7), we start with the integral of $h_0^{(1)}(r)$, where r is the spherical polar radius.

We need to integrate this function over the cube of side h, so we need

$$
Q = \int_{S_2}\int_0^{r_1(\mathbf{k})} h_0^{(1)}(r)r^2dr\,d\mathbf{k}, \qquad \text{(A.9)}
$$

where $\mathbf{k}$ is the unit vector direction of $\mathbf{x} = (x, y, z)$ and $r_1(\mathbf{k})$ is the value of r where the ray $\mathbf{k}$ pierces through the surface of the cube of "radius" $\frac{h}{2}$, and S_2 is the unit spherical surface in R^3.

We will integrate this over the tetrahedron,

$$0 < z < \frac{h}{2}, \quad 0 < x < z, \quad 0 < y < x.$$

So now let us consider the parameterization of $\mathbf{k}$. Let the ray in direction $\mathbf{k}$ cut the plane $z = 1$ in $(\xi, \eta, 1)$ or $(\rho\cos\phi, \rho\sin\phi, 1)$, where ρ is the cylindrical polar radius and ϕ is the azimuthal angle, varying from 0 to $\frac{\pi}{4}$. We will integrate over the region $0 < z < \frac{h}{2}, 0 < x < y$, drawn as quite a small tetrahedron near the origin $\mathbf{0}$. The whole cube of side h contains 48 of these small tetrahedra.

Equation (A.9) may be rewritten

$$Q = -48i \int_{S_2'} \int_0^{r_1(\mathbf{k})} e^{i\ r} r dr\ d\mathbf{k}, \tag{A.10}$$

where $d\mathbf{k}$ is the surface element of the unit sphere on which $\mathbf{k}$ lies, and S_2' is a part of the unit spherical surface to be made precise below.

We parameterize $\mathbf{k}$ in the first instance by (ξ, η), which are x and y coordinates in the plane $z = 1$. Thus, we have

$$\mathbf{k} = \frac{(\xi, \eta, 1)}{\sqrt{1 + \xi^2 + \eta^2}} = \frac{(\rho\cos\phi, \rho\sin\phi, 1)}{\sqrt{1 + \rho^2}}. \tag{A.11}$$

Let us radically project the element $d\xi d\eta = \rho\ d\rho\ d\phi$ onto the unit sphere. Thus,

$$d\mathbf{k} = \frac{1}{1 + \rho^2} \cos\psi\ d\xi\ d\eta. \tag{A.12}$$

Here ψ is the angle between the normal to the plane $z = 1$ and $\mathbf{k}$, i.e.,

$$\cos\psi = (0, 0, 1)^T \cdot (\xi, \eta, 1)^T \frac{1}{\sqrt{1 + \rho^2}} = \frac{1}{\sqrt{1 + \rho^2}}. \tag{A.13}$$

Thus,

$$d\mathbf{k} = \frac{d\xi\ d\eta}{(1 + \rho^2)^{\frac{3}{2}}} = \frac{\rho\ d\rho\ d\phi}{(1 + \rho^2)^{\frac{3}{2}}}. \tag{A.14}$$

From (A.9) and (A.14), we get

$$Q = -48i \int_0^{\frac{\pi}{4}} d\phi \int_0^{r_1(\rho)} r e^{ir} dr \frac{\rho\ d\rho}{(1 + \rho^2)^{\frac{3}{2}}}, \tag{A.15}$$

$$r_1(\rho) = \frac{h}{2}(1 + \rho^2)^{\frac{1}{2}}. \tag{A.16}$$

The r integral can be done easily using integration by parts:

$$\begin{aligned}\int_0^{r_1} r e^{ir} dr &= [\frac{1}{i} r e^{ir}]_0^{r_1} - \int_0^{r_1} \frac{1}{i} e^{ir} dr \\ &= \frac{1}{i} r_1 e^{ir_1} + [e^{ir}]_0^{r_1}\end{aligned}$$

$$\begin{aligned} &= -ir_1e^{ir_1} + e^{ir_1} - 1 \\ &= e^{ir_1}(1 - ir_1) - 1. \end{aligned} \tag{A.17}$$

So, from (A.10),

$$Q = -48i\int_0^{\pi/4} d\phi \int_0^{\frac{1}{\cos\phi}} (e^{ir_1}(1-ir_1)-1)\frac{\rho\, d\rho}{(1+\rho^2)^{\frac{3}{2}}}. \tag{A.18}$$

Set

$$\rho^{'} = \rho\cos\phi, \qquad d\rho = \frac{1}{\cos\phi}d\rho^{'}. \tag{A.19}$$

Then

$$Q = -48i\int_0^{\pi/4} d\phi \int_0^{1} d\rho'[e^{ir_1(\rho)}(1-ir_1(\rho))-1]\frac{\rho}{(1+\rho^2)^{\frac{3}{2}}}\frac{1}{\cos\phi}. \tag{A.20}$$

The function $f(\rho', \phi)$ is given by

$$f(\rho', \phi) = [e^{ir_1(\rho)}(1-ir_1(\rho))-1]\frac{\rho}{(1+\rho^2)^{\frac{3}{2}}}\frac{1}{\cos\phi}, \tag{A.21}$$

where

$$\begin{aligned} \rho &= \frac{\rho'}{\cos\phi}, \\ r_1(\rho) &= \frac{h}{2}(1+\rho^2)^{\frac{1}{2}} \end{aligned} \tag{A.22}$$

by equations (A.16) and (A.19).

A.2.2. Integration of Hankel in 3-D: generic case. Formula (A.7) represents the 3-D Green's function near the source. What we need is its integration over the 3-D cube of side h. To do that, we just need to carry out a coordinate transformation to transfer the integral of $h_0^{(1)}$ derived in Appendix A.2.1 to our current setting.

We have near the source

$$\begin{aligned} g(\mathbf{r}, \mathbf{r}_0) &= n_0\frac{e^{i\omega\tau}}{4\pi\tau} \\ &= \frac{in_0\omega}{4\pi}(-i)\frac{e^{i\omega\tau}}{\omega\tau} \\ &= \frac{in_0\omega}{4\pi}h_0^{(1)}(\omega\tau) \\ &= \frac{in_0\omega}{4\pi}h_0^{(1)}(\omega n_0|\mathbf{r}-\mathbf{r}_0|) \\ &= \frac{in_0\omega}{4\pi}h_0^{(1)}(\omega n_0 r), \end{aligned} \tag{A.23}$$

where $r = |\mathbf{r} - \mathbf{r}_0|$.

The integration of the above Green's function in the cell centered at the source will be

$$I = \frac{in_0\omega}{4\pi}\int_{S_2}\int_0^{r_1(\mathbf{k})} h_0^{(1)}(\omega n_0 r)r^2 dr\; d\mathbf{k}$$

$$
\begin{aligned}
&= \frac{in_0\omega}{4\pi}\int_{S_2}\int_0^{\omega n_0 r_1(\mathbf{k})} h_0^{(1)}(t)\frac{t^2}{\omega^2 n_0^2}\frac{dt}{\omega n_0}\; d\mathbf{k} \\
&= \frac{i}{4\pi\omega^2 n_0^2}\int_{S_2}\int_0^{\omega n_0 r_1(\mathbf{k})} h_0^{(1)}(t)\; t^2\; dt\; d\mathbf{k} \\
&= \frac{i}{4\pi\omega^2 n_0^2} Q_s, \qquad \text{(A.24)}
\end{aligned}
$$

where Q_s is the scaled integral of Q as defined in (A.20) and hence is defined by the following integration,

$$
\begin{aligned}
Q_s &= -48i\int_0^{\pi/4} d\phi \int_0^1 d\rho' \left[e^{ir_1(\rho)}(1-ir_1(\rho))-1\right]\frac{\rho}{(1+\rho^2)^{\frac{3}{2}}}\frac{1}{\cos\phi} \\
&= -48i\int_0^{\pi/4} d\phi \int_0^1 d\rho' f(\rho',\phi). \qquad \text{(A.25)}
\end{aligned}
$$

The function $f(\rho',\phi)$ is given by

$$
f(\rho',\phi) = \left[e^{ir_1(\rho)}(1-ir_1(\rho))-1\right]\frac{\rho}{(1+\rho^2)^{\frac{3}{2}}}\frac{1}{\cos\phi}, \qquad \text{(A.26)}
$$

where

$$
\rho = \frac{\rho'}{\cos\phi} \quad \text{and} \quad r_1(\rho) = \omega n_0 \frac{h}{2}(1+\rho^2)^{\frac{1}{2}} \qquad \text{(A.27)}
$$

by equation (A.22). Here $r_1(\rho)$ is scaled by the factor ωn_0.

Appendix B. Numerical accuracy of the truncated 3-D H-B ansatz. Since the H-B ansatz (2.1) is based on Hankel functions, we may directly analyze the effect of accuracy of the H-B ingredients, such as the phase and H-B coefficients, on wave solution. However, since, *away from the point source*, the H-B ansatz is equivalent to the traditional geometrical optics [56, 42], to avoid some technical details we will consider the following truncated geometrical-optics ansatz (valid for 3-D wave motion) as a proxy for the truncated H-B ansatz away from the point source:

$$
g_{\mathrm{GO}}(\mathbf{r},\mathbf{r}_0) = \sum_{s=0}^{N}\frac{A_s(\mathbf{r},\mathbf{r}_0)}{(i\omega)^s}e^{i\omega\tau(\mathbf{r},\mathbf{r}_0)}, \qquad \text{(B.1)}
$$

where $\mathbf{r}_0$ is the source, $\mathbf{r}$ is the observation point, N is an integer, τ is the phase satisfying the eikonal equation, and $\{A_s\}_{s=0}^N$ are amplitudes satisfying transport equations [3, 4, 56, 42].

Since these A_s functions are directly linked to the H-B coefficients v_s [4, 56, 42], we can assume that these A_s functions are computed to the same orders of accuracy as v_s away from the point source. Starting from this assumption, we briefly analyze the error between $g_{\mathrm{GO}}(\mathbf{r},\mathbf{r}_0)$ and its numerical solution $g^h_{\mathrm{GO}}(\mathbf{r},\mathbf{r}_0)$ for any point $\mathbf{r}$ away from the source $\mathbf{r}_0$.

When $N=0$, we have

$$
\begin{aligned}
|g_{\mathrm{GO}}(\mathbf{r},\mathbf{r}_0) - g^h_{\mathrm{GO}}(\mathbf{r},\mathbf{r}_0)| &= |A_0(\mathbf{r},\mathbf{r}_0)e^{i\omega\tau(\mathbf{r},\mathbf{r}_0)} - A_0^h(\mathbf{r},\mathbf{r}_0)e^{i\omega\tau^h(\mathbf{r},\mathbf{r}_0)}| \\
&\le O(h_0^3) + O(\omega h_0^5). \qquad \text{(B.2)}
\end{aligned}
$$

When $N = 1$, we have

$$
\begin{aligned}
|g_{\mathrm{GO}}(\mathbf{r},\mathbf{r}_0) - g^h_{\mathrm{GO}}(\mathbf{r},\mathbf{r}_0)| &\le |A_0(\mathbf{r},\mathbf{r}_0)e^{i\omega\tau(\mathbf{r},\mathbf{r}_0)} - A^h_0(\mathbf{r},\mathbf{r}_0)e^{i\omega\tau^h(\mathbf{r},\mathbf{r}_0)}| \\
&\quad + \frac{1}{\omega}|A_1(\mathbf{r},\mathbf{r}_0)e^{i\omega\tau(\mathbf{r},\mathbf{r}_0)} - A^h_1(\mathbf{r},\mathbf{r}_0)e^{i\omega\tau^h(\mathbf{r},\mathbf{r}_0)}| \\
&\le O(h_0^3) + O(\omega h_0^5) + O\left(\frac{h_0}{\omega}\right) + O(h_0^5) \\
&= O(h_0^3) + O(\omega h_0^5) + O\left(\frac{h_0}{\omega}\right). \qquad \text{(B.3)}
\end{aligned}
$$

Therefore, *away from the point source* we will use the above geometrical-optics estimates as the proxy for the truncated H-B estimates in the total error estimates. On the other hand, *near the source but excluding the source*, such truncated H-B estimates also hold since we have the following two observations: (1) the H-B ansatz is an uniformly asymptotic solution to the point-source Helmholtz equation so that it can be treated as the exact solution of the point-source equation, and (2) the computed H-B ingredients in the truncated H-B expansion are initialized near the point source according to specified orders of accuracy.

REFERENCES

[1] M. Abramowitz and I. A. Stegun. *Handbook of mathematical functions.* Dover Publications, Inc., New York., 1965.

[2] P. Amestoy, C. Ashcraft, O. Boiteau, A. Buttari, J.-Y. L'Excellent, and C. Weisbecker. Improving multifrontal methods by means of block low-rank representations. *SIAM Journal on Scientific Computing*, 37(3):A1451–A1474, 2015.

[3] G. S. Avila and J. B. Keller. The high-frequency asymptotic field of a point source in an inhomogeneous medium. *Commun. Pure Appl. Math.*, 16:363–381, 1963.

[4] V. M. Babich. The short wave asymptotic form of the solution for the problem of a point source in an inhomogeneous medium. *USSR Computational Mathematics and Mathematical Physics*, 5(5):247–251, 1965.

[5] I. Babuska, F. Ihlenburg, E. T. Paik, and S. A. Sauter. A generalized finite element method for solving the Helmholtz equation in two dimensions with minimal pollution. *Computer Methods in Applied Mechanics and Engineering*, 128(3-4):325–359, 1995.

[6] I. M. Babuška and S. A. Sauter. Is the pollution effect of the FEM avoidable for the Helmholtz equation considering high wave numbers? *SIAM Review*, 42:451–484, 2000.

[7] M. Bebendorf. *Hierarchical Matrices: A Means to Efficiently Solve Elliptic Boundary Value Problems*, volume 63 of *Lecture Notes in Computational Science and Engineering (LNCSE).* Springer-Verlag, 2008. ISBN 978-3-540-77146-3.

[8] S. Boerm, L. Grasedyck, and W. Hackbusch. *Hierarchical matrices.* Max-Planck- Institute Lecture Notes, 2006.

[9] James Bremer, Ze Chen, and Haizhao Yang. Rapid Application of the Spherical Harmonic Transform via Interpolative Decomposition Butterfly Factorization. *arXiv preprint arXiv:2004.11346*, 2020.

[10] Emmanuel Candès, Laurent Demanet, and Lexing Ying. A fast butterfly algorithm for the computation of Fourier integral operators. *Multiscale Model. Sim.*, 7(4):1727–1750, 2009.

[11] Z. Chen and X. Xiang. A source transfer domain decomposition method for Helmholtz equations in unbounded domain. *SIAM Journal on Numerical Analysis*, 51(4):2331–2356, 2013.

[12] Zhongying Chen, Dongsheng Cheng, Wei Feng, and Tingting Wu. An optimal 9-point finite difference scheme for the helmholtz equation with pml. *International Journal of Numerical Analysis & Modeling*, 10(2), 2013.

[13] T. A. Davis. Algorithm 832: UMFPACK v4.3—an unsymmetric-pattern multifrontal method. *ACM Transactions on Mathematical Software*, 30(2):196–199, June 2004.

[14] I. S. Duff and J. K. Reid. The multifrontal solution of indefinite sparse symmetric linear. *ACM Trans. Math. Softw.*, 9(3):302–325, September 1983.

[15] Kjersti Solberg Eikrem, Geir Nævdal, and Morten Jakobsen. Iterative solution of the Lippmann–Schwinger equation in strongly scattering acoustic media by randomized construction of preconditioners. *Geophysical Journal International*, 224(3):2121–2130, 10 2020.

[16] B. Engquist and L. Ying. Sweeping preconditioner for the Helmholtz equation: hierarchical matrix representation. *Communications on Pure and Applied Mathematics*, 64(5):697–735, 2011.

[17] B. Engquist and L. Ying. Sweeping preconditioner for the Helmholtz equation: moving perfectly matched layers. *Multiscale Modeling & Simulation*, 9(2):686–710, 2011.

[18] B. Engquist and H.-K. Zhao. Approximate separability of the Green's function of the Helmholtz equation in the high-frequency limit. *Comm. Pure Appl. Math.*, 71:2220–2274, 2018.

[19] J. Fang, J. Qian, L. Zepeda-Nunez, and H. Zhao. A hybrid approach to solve the high-frequency Helmholtz equation with source singularity in inhomogeneous media. *J. Comp. Phys.*, 371:261–279, 2018.

[20] J. Fang, J. Qian, L. Zepeda-Nunez, and H.-K Zhao. Learning dominant wave directions for plane wave methods for high-frequency Helmholtz equations. *Research in the Mathematical Sciences*, 4:9–39, 2017.

[21] S. Fomel, S. Luo, and H. K. Zhao. Fast sweeping method for the factored eikonal equation. *J. Comput. Phys.*, 228:6440–6455, 2009.

[22] A. George. Nested dissection of a regular finite element mesh. *SIAM Journal on Numerical Analysis*, 10:345–363, 1973.

[23] Pieter Ghysels, Xiaoye Sherry Li, Christopher Gorman, and François-Henry Rouet. A robust parallel preconditioner for indefinite systems using hierarchical matrices and randomized sampling. In *2017 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pages 897–906. IEEE, 2017.

[24] E. Giladi and Keller. J. B. A hybrid numerical asymptotic method for scattering problems. *Journal of Computational Physics*, 174(1):226–247, 2001.

[25] Han Guo, Jun Hu, and Eric Michielssen. On MLMDA/butterfly compressibility of inverse integral operators. *IEEE Antennas Wirel. Propag. Lett.*, 12:31–34, 2013.

[26] Han Guo, Yang Liu, Jun Hu, and Eric Michielssen. A butterfly-based direct integral-equation solver using hierarchical LU factorization for analyzing scattering from electrically large conducting objects. *IEEE Trans. Antennas Propag.*, 65(9):4742–4750, 2017.

[27] Han Guo, Yang Liu, Jun Hu, and Eric Michielssen. A butterfly-based direct solver using hierarchical LU factorization for Poggio-Miller-Chang-Harrington-Wu-Tsai equations. *Microw Opt Technol Lett.*, 60:1381–1387, 2018.

[28] R. L. Holford. Elementary source-type solutions of the reduced wave equation. *The Journal of the Acoustical Society of America*, 70(5):1427–1436, 1981.

[29] C. Howarth. *New generation finite element methods for forward seismic modelling.* PhD thesis, University of Reading, 2014.

[30] C. Y. Kao, S. J. Osher, and J. Qian. Lax-Friedrichs sweeping schemes for static Hamilton-Jacobi equations. *J. Comput. Phys.*, 196:367–391, 2004.

[31] C. Y. Lam and J. Qian. Numerical microlocal analysis by fast Gaussian wavepacket transforms and application to high-frequency Helmholtz problems. *SIAM J. Sci. Comput.*, 41:A2717–A2746, 2019.

[32] P. Lax. Asymptotic solutions of oscillatory initial value problems. *Duke Math. J.*, 24:627–645, 1957.

[33] X. S. Li and J. W. Demmel. SuperLU DIST: A scalable distributed-memory sparse direct solver for unsymmetric linear systems. *ACM Trans. Mathematical Software*, 29(2):110–140, June 2003.

[34] Yingzhou Li and Haizhao Yang. Interpolative butterfly factorization. *SIAM J. Sci. Comput.*, 39(2):A503–A531, 2017.

[35] Yingzhou Li, Haizhao Yang, Eileen R Martin, Kenneth L Ho, and Lexing Ying. Butterfly factorization. *Multiscale Model. Sim.*, 13(2):714–732, 2015.

[36] F. Liu and L. Ying. Sparsify and sweep: an efficient preconditioner for the lippmann-schwinger equation. *ArXiv e-prints*, [math.NA] 1705.09443, 2017.

[37] Yang Liu, Pieter Ghysels, Lisa Claus, and Xiaoye Sherry Li. Sparse approximate multifrontal factorization with butterfly compression for high-frequency wave equations. *SIAM Journal on Scientific Computing*, 0(0):S367–S391, 2021.

[38] Yang Liu, Han Guo, and Eric Michielssen. An HSS matrix-inspired butterfly-based direct solver for analyzing scattering from two-dimensional objects. *IEEE Antennas Wirel. Propag. Lett.*, 16:1179–1183, 2017.

[39] Yang Liu, Xin Xing, Han Guo, Eric Michielssen, Pieter Ghysels, and Xiaoye Sherry Li. Butterfly factorization via randomized matrix-vector multiplications. *SIAM Journal on Scientific Computing*, 43(2):A883–A907, 2021.

[40] Yang Liu and Haizhao Yang. A hierarchical butterfly LU preconditioner for two-dimensional electromagnetic scattering problems involving open surfaces. *J. Comput. Phys.*,

401:109014, 2020.

[41] W. Lu, J. Qian, and R. Burridge. Babich-like ansatz for three-dimensional point-source Maxwell's equations in an inhomogeneous medium at high frequencies. *SIAM J. Multiscale Model. Simul.*, 14(3):1089–1122, 2016.

[42] W. Lu, J. Qian, and R. Burridge. Babich's expansion and the fast Huygens sweeping method for the Helmholtz wave equation at high frequencies. *J. Comput. Phys.*, 313:478–510, 2016.

[43] W. Lu, J. Qian, and R. Burridge. Extending Babich's ansatz for point-source Maxwell's equations using Hadamard's method. *SIAM J. Multiscale Model. Simul.*, 16:727–751, 2018.

[44] S. Luo, J. Qian, and R. Burridge. Fast Huygens sweeping methods for Helmholtz equations in inhomogeneous media in the high frequency regime. *J. Comput. Phys.*, 270:378–401, 2014.

[45] S. Luo, J. Qian, and R. Burridge. High-order factorization based high-order hybrid fast sweeping methods for point-source eikonal equations. *SIAM J. Numer. Analy.*, 52:23–44, 2014.

[46] Eric Michielssen and Amir Boag. Multilevel evaluation of electromagnetic fields for the rapid solution of scattering problems. *Microw Opt Technol Lett.*, 7(17):790–795, 1994.

[47] Eric Michielssen and Amir Boag. A multilevel matrix decomposition algorithm for analyzing scattering from large structures. *IEEE Trans. Antennas Propag.*, 44(8):1086–1093, 1996.

[48] P. Monk and D.-Q. Wang. A least-squares method for the Helmholtz equation. *Computer Methods in Applied Mechanics and Engineering*, 175(1–2):121–136, 1999.

[49] N. C. Nguyen, J. Peraire, F. Reitich, and B. Cockburn. A phase-based hybridizable discontinuous Galerkin method for the numerical solution of the Helmholtz equation. *J. Comput. Physics*, 290:318–335, 2015.

[50] Michael O'Neil, Franco Woolfe, and Vladimir Rokhlin. An algorithm for the rapid evaluation of special function transforms. *Appl. Comput. Harmon. A.*, 28(2):203 – 226, 2010. Special Issue on Continuous Wavelet Transform in Memory of Jean Morlet, Part I.

[51] Stéphane Operto, Jean Virieux, Patrick Amestoy, Jean-Yves L'Excellent, Luc Giraud, and Hafedh Ben Hadj Ali. 3D finite-difference frequency-domain modeling of visco-acoustic wave propagation using a massively parallel direct solver: A feasibility study. *Geophysics*, 72(5):SM195–SM211, 2007.

[52] Qiyuan Pang, Kenneth L. Ho, and Haizhao Yang. Interpolative decomposition butterfly factorization. *SIAM J. Sci. Comput.*, 42(2):A1097–A1115, 2020.

[53] A. Pica. Fast and accurate finite-difference solutions of the 3-D eikonal equation parameterized in celerity. In *Expanded Abstracts*, pages 1774–1777. Soc. Expl. Geophys., Tulsa, OK, 1997.

[54] J. Qian, W. Lu, L. Yuan, S. Luo, and R. Burridge. Eulerian geometrical optics and fast Huygens sweeping methods for three-dimensional time-harmonic high-frequency Maxwell's equations in inhomogeneous media. *SIAM J. Multiscale Modeling and Simulation*, 16:595–636, 2016.

[55] J. Qian, J. Song, W. Lu, and R. Burridge. Hadamard-Babich ansatz for point-source elastic wave equations in variable media at high frequencies. *SIAM Multiscale Model. Simul.*, 19:46–86, 2021.

[56] J. Qian, L. Yuan, Y. Liu, S. Luo, and R. Burridge. Babich's expansion and high-order Eulerian asymptotics for point-source Helmholtz equations. *Journal of Scientific Computing*, 67:883–908, 2016.

[57] Sadeed Bin Sayed, Yang Liu, Luis J. Gomez, and Abdulkadir C. Yucel. A butterfly-accelerated volume integral equation solver for broad permittivity and large-scale electromagnetic analysis. *IEEE Transactions on Antennas and Propagation*, 70(5):3549–3559, 2022.

[58] D Schaubert, D Wilton, and A Glisson. A tetrahedral modeling method for electromagnetic scattering by arbitrarily shaped inhomogeneous dielectric bodies. *IEEE Trans. Antennas Propag.*, 32(1):77–85, 1984.

[59] C. C. Stolk. A rapidly converging domain decomposition method for the Helmholtz equation. *Journal of Computational Physics*, 241(0):240–252, 2013.

[60] W. W. Symes and J. Qian. A slowness matching Eulerian method for multivalued solutions of eikonal equations. *J. Sci. Comp.*, 19:501–526, 2003.

[61] M. Taus, L. Demanet, and L. Zepeda-Núñez. A short note on a fast and high-order hybridizable discontinuous Galerkin solver for the 2D high-frequency Helmholtz equation. In *SEG Technical Program Expanded Abstracts 2016*, pages 3835–3840, 2016.

[62] Mark Tygert. Fast algorithms for spherical harmonic expansions, III. *J. Comput. Phys.*, 229(18):6181 – 6192, 2010.

[63] A. Vion and C. Geuzaine. Double sweep preconditioner for optimized Schwarz methods applied to the Helmholtz problem. *Journal of Computational Physics*, 266(0):171–190, 2014.

[64] S. Wang, X. S. Li, Xia J., Y. Situ, and M. V. de Hoop. Efficient scalable algorithms for solving dense linear systems with hierarchically semiseparable structures. *SIAM Journal on Scientific Computing*, 35(6):C519–C544, 2013.

[65] T. Xiong, M. Zhang, Y. T. Zhang, and C.-W. Shu. Fast sweeping fifth order weno scheme

for static hamilton-jacobi equations with accurate boundary treatment. *J. Sci. Comput.*, 45:514–536, 2010.

[66] Haizhao Yang. A unified framework for oscillatory integral transforms: When to use NUFFT or butterfly factorization? *J. Comput. Phys.*, 388:103 – 122, 2019.

[67] Lexing Ying. Sparse Fourier Transform via Butterfly Algorithm. *SIAM J. Sci. Comput.*, 31(3):1678–1694, 2009.

[68] M.A. Yurkin and A.G. Hoekstra. The discrete dipole approximation: An overview and recent developments. *Journal of Quantitative Spectroscopy and Radiative Transfer*, 106(1):558–589, 2007. IX Conference on Electromagnetic and Light Scattering by Non-Spherical Particles.

[69] L. Zepeda-Núñez and L. Demanet. The method of polarized traces for the 2D Helmholtz equation. *Journal of Computational Physics*, 308:347–388, 2016.

[70] L. Zepeda-Núñez and H. Zhao. Fast alternating bidirectional preconditioner for the 2D high-frequency Lippmann–Schwinger equation. *SIAM Journal on Scientific Computing*, 38(5):B866–B888, 2016.

[71] L. Zhang, J. W. Rector, and G. M. Hoversten. Eikonal solver in the celerity domain. *Geophys. J. Internat.*, 162:1–8, 2005.

[72] Y. T. Zhang, H. K. Zhao, and J. Qian. High order fast sweeping methods for static Hamilton-Jacobi equations. *J. Sci. Comp.*, 29:25–56, 2006.

[73] H. K. Zhao. Fast sweeping method for eikonal equations. *Math. Comp.*, 74:603–627, 2005.