Journal of Computational Physics 465 (2022) 111380

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

Learning rays via deep neural network in a ray-based IPDG method for high-frequency Helmholtz equations in inhomogeneous media

Tak Shing Au Yeung [c,b], Ka Chun Cheung [b], Eric T. Chung [a], Shubin Fu [e,*], Jianliang Qian [d]

[a] Department of Mathematics, The Chinese University of Hong Kong, Hong Kong Special Administrative Region
[b] NVIDIA AI Technology Center, NVIDIA, Hong Kong Special Administrative Region
[c] Department of Mathematics, The Chinese University of Hong Kong, Hong Kong Special Administrative Region
[d] Department of Mathematics, Michigan State University, East Lansing, MI 48824, USA
[e] Department of Mathematics, University of Wisconsin-Madison, Madison, WI 53706, USA

ARTICLE INFO

Article history:
Received 16 June 2021
Received in revised form 10 January 2022
Accepted 8 June 2022
Available online 13 June 2022

Keywords:
Deep learning
Ray tracing
Helmholtz equation
Discontinuous Galerkin

ABSTRACT

We develop a deep learning approach to extract ray directions at discrete locations by analyzing highly oscillatory wave fields. A deep neural network is trained on a set of local plane-wave fields to predict ray directions at discrete locations. The resulting deep neural network is then applied to a reduced-frequency Helmholtz solution to extract ray directions, which are further incorporated into a ray-based interior-penalty discontinuous Galerkin (IPDG) method to solve the corresponding Helmholtz equations at higher frequencies. In this way, we observe no apparent pollution effects in the resulting Helmholtz solutions in inhomogeneous media. Our 2D and 3D numerical results show that the proposed scheme is very efficient and yields highly accurate solutions.

1. Introduction

The high-frequency Helmholtz equation is numerically hard to solve. The Shannon's sampling principle [35] states that a necessary condition to solve the high-frequency Helmholtz equation is that the mesh size h and frequency ω satisfies the relationship: $h = \mathcal{O}(\omega^{-1})$. Thus, if the ambient dimension of the Helmholtz equation is d, the degrees of freedom (DOFS) are $\mathcal{O}(\omega^d)$, which means that solving the Helmholtz equation needs a large complexity if the frequency is high. However, this complexity is difficult to achieve numerically. The difficulty is mainly due to the pollution effect in error estimates for finite element methods [3,4,22]. The pollution effect states that the ratio between the numerical error and the best approximation error from a discrete finite element space is ω dependent. This will lead to a difficulty in developing an accurate and stable discretization when the frequency ω is high and the above relation $h = \mathcal{O}(\omega^{-1})$ is maintained. In this paper, inspired by ray theory and related micro-local analysis, we develop a deep learning approach to extract ray directions from a reduced-frequency Helmholtz solution, which are further incorporated into an IPDG method to solve the corresponding high-frequency Helmholtz equation in inhomogeneous media, leading to a new IPDG method with no apparent pollution effect for Helmholtz equations.

* Corresponding author.
E-mail addresses: sfu45@math.wisc.edu, shubinfu89@gmail.com (S. Fu).

https://doi.org/10.1016/j.jcp.2022.111380
0021-9991/

Ray theory provides a powerful asymptotic method for treating high-frequency wave phenomena [28,1,2]. Microlocal analysis is built upon ray theory but is much further developed [39]. In mathematical analysis, microlocal analysis consists of Fourier-transform related techniques for analyzing variable-coefficient partial differential equations, including Fourier integral operators, wavefront sets, and oscillatory integral operators, so that such analysis allows localized scrutiny not only with respect to location in space but also with respect to cotangent directions at a given point [39]. Since wave singularities propagate along characteristics, applying microlocal analysis to spatial wave fields will reveal cotangent-space related characteristic (or ray) information at each spatial point [6,31,32,5]. Moreover, such localized ray information can be incorporated into a finite-element basis so that one can design effective numerical methods to solve wave equations [12,13,10,7,25].

The notion of numerical microlocal analysis (NMLA) method was first proposed in [6]. Assuming that the to-be-processed data are solutions of Helmholtz equations, the authors in [6] designed a Jacobi-Anger expansion and Fourier-transform based plane-wave analysis method to process Dirichlet observables collected on a sphere around each to-be-analyzed point. Later, authors in [27] improved the method in [6] by using L^1 minimization instead of Tikhonov regularization to obtain much less noise-sensitive results. To overcome stability issues and improve accuracy in identified ray directions, the method in [6] was further developed in [5] to analyze impedance observables in a similar setup; to deal with multiple plane waves or point sources arriving at an observation point, the authors of [5] further developed a decomposition filter with Gaussian weights. The NMLA method is used for numerically and locally finding crossing rays and their directions from samples of wave-fields [6,5]. Comparing to other methods, such as the Prony's method [8] and the matrix pencil method [21], that perform similar tasks, the NMLA is simpler and more robust.

In comparison to the approaches in [6,5], the NMLA method in [25] is much straightforward and easy to implement in the sense that fast Gaussian wavepacket transforms are applied directly to the given oscillatory wavefield, where the method neither assumes the underlying model being Helmholtz nor preprocesses the input data into Dirichlet or impedance data on a certain sphere around an observation point, and the relevant ray directions are encoded into cotangent directions in terms of coefficients of Gaussian wavepacket expansions.

In the above works on numerical microlocal analysis [6,5,25], ray directions are extracted via hard-core numerical analysis. Motivated by recent development in deep learning and related computational methodologies [38,37,36,9], we develop a deep learning approach to train a deep neural network (DNN) on a set of local plane-wave fields to predict ray directions at discrete locations, resulting in DNN based microlocal analysis (DNN-MLA) method. Our deep neural network (DNN) based ray-direction extraction method provides a nonlinear parametrized "solution operator" for mapping a highly oscillatory wave field into ray directions, once the DNN is trained on a set of plane waves and corresponding ray directions. We emphasize that our new method of extracting ray directions does not require the input training oscillatory data to be Helmholtz solutions, which is similar to the method in [25]. This original DNN based microlocal analysis method is our first contribution.

To solve high-frequency Helmholtz equations, we further apply the DNN-MLA method to a reduced-frequency Helmholtz solution to extract ray directions, which are further incorporated into an interior penalty discontinuous Galerkin (IPDG) method to solve the high-frequency Helmholtz equation. This is our second contribution.

Our third contribution is to provide an error analysis for the newly developed ray-based IPDG (ray-IPDG) method. The theorem indicates that, in the high-frequency regime, when the frequency parameter ω is large, the L^2 error of the numerical solution is dominated by the mesh size and the approximation error in ray directions.

1.1. The high-frequency Helmholtz problem

Let $\omega > 0$ be the frequency parameter and $\Omega \subset \mathbb{R}^d$ be the computational domain, where $d = 2$ or 3 is the dimension. Our goal is to find the unknown wave field u such that

$$-\nabla^2 u - (\omega/c)^2 u = f \quad \text{in } \Omega, \tag{1.1}$$

where we may impose impedance boundary conditions, Cauchy conditions or perfectly matched layer (PML) boundary conditions. Here the wave speed c is a smooth function with positive lower bound $c_{\min}$ and upper bound $c_{\max}$, and $f \in L^2(\Omega)$ is the source function. We will apply the idea of "probing" from [12] for solving the high-frequency Helmholtz problem. Let $x \in \Omega$ and $f = 0$. We consider the following geometric optics ansatz (cf. [24,29,34,1,33,11]) for the Helmholtz equation

$$u(\mathbf{x}) = \text{ superposition of } \left\{ A_n(\mathbf{x}) e^{i\omega\phi_n(\mathbf{x})} \right\}_{n=1}^{N} + \mathcal{O}\left(\omega^{-1}\right),$$

where N is the number of wavefronts passing through each point, A_n and ϕ_n are respectively the amplitude and phase functions. Note that the phase function satisfies the Eikonal equation $|\nabla\phi_n| = c^{-1}$. Throughout the paper, to simplify the presentation, we will assume that N is the same at all points, so that there are N dominant wavefronts at each point. The functions A_n and ϕ_n are independent of the frequency ω, but depend on the wave speed $c(x)$. We will assume that the functions A_n and ϕ_n are locally smooth. Consider a point $x_0 \in \Omega$ in the computational domain. The Taylor expansion of each ϕ_n for $|\mathrm{x} - \mathrm{x}_0| < h \ll 1$ is given by

$$\phi_n(\mathrm{x}) = \phi_n(\mathrm{x}_0) + |\nabla\phi_n(\mathrm{x}_0)|\,\mathbf{d}_n \cdot (\mathrm{x}-\mathrm{x}_0) + \mathcal{O}\left(h^2\right),$$

where $\mathbf{d}_n := \frac{\nabla\phi_n(\mathrm{x}_0)}{|\nabla\phi_n(\mathrm{x}_0)|}$ is the ray direction. Similarly, the Taylor expansion of A_n gives

$$A_n(\mathrm{x}) = A_n(\mathrm{x}_0) + \nabla A_n(\mathrm{x}_0)\cdot(\mathrm{x}-\mathrm{x}_0) + \mathcal{O}\left(h^2\right).$$

Hence, each wave component in the solution $u(\mathrm{x})$ can be written as

$$A_n(\mathrm{x})e^{i\omega\phi_n(\mathrm{x})} = B_n(\mathrm{x}-\mathrm{x}_0)\,e^{i(\omega/c(\mathrm{x}_0))\mathbf{d}_n\cdot(\mathrm{x}-\mathrm{x}_0)} + \mathcal{O}\left(h^2+\omega h^2+\omega^{-1}\right),$$

where $B_n(\mathrm{x}) = e^{i\omega\phi_n(\mathrm{x}_0)}\left(A(\mathrm{x}_0)+\nabla A_n(\mathrm{x}_0)\cdot\mathrm{x}\right)$ is a linear function. By taking $h\sim\mathcal{O}\left(\omega^{-1}\right)$, we see that $u(\mathrm{x})$ can be approximated by superposition of products of a linear function and a plane wave with an error of $\mathcal{O}\left(\omega^{-1}\right)$. This motivates us to use products of bilinear functions with $e^{i\omega/c(\mathrm{x}_0)\mathbf{d}_n\cdot(\mathrm{x}-\mathrm{x}_0)}$ as local basis.

1.2. Probing of ray directions

To solve the high frequency Helmholtz equation (1.1), the above discussion motivates the use of functions $e^{i\omega/c(\mathrm{x}_0)\mathbf{d}_n\cdot(x-x_0)}$ as local basis. Thus, the ray-based IPDG method [10] will be used for solving the high frequency Helmholtz equation. The most important step is to determine the local ray directions $\mathbf{d}_n$. To do so, we need to compute the solution of a reduced frequency Helmholtz equation

$$-\nabla^2\tilde{u} - (\tilde{\omega}/c)^2\tilde{u} = f \qquad \text{in } \Omega, \tag{1.2}$$

where $\tilde{\omega}<\omega$ is a reduced frequency. After having this reduced frequency solution, we may use the Gaussian wavepacket transform based NMLA method to find the ray directions from the reduced frequency solution as proposed in [25]. But in this paper, we propose a deep learning approach to extract those ray directions. Finally, we use the computed ray directions to form the local basis for the ray-based IPDG method to solve the high frequency Helmholtz equation. We summarize the steps as follows:

1. Use the standard IPDG method to solve the reduced-frequency Helmholtz equation;
2. Use a deep learning or NMLA method to compute ray directions;
3. Use the computed ray directions to form the basis for the Ray-IPDG method;
4. Use the Ray-IPDG method to solve the high-frequency Helmholtz equation.

In order to solve the high frequency Helmholtz equation to a certain accuracy, our goal is to develop a ray-based IPDG method to achieve this, and further more the ray-based IPDG method will use much less computational time and cost than the standard IPDG method does.

1.3. Related works

In a recent survey [20], the authors have given a quite comprehensive review of construction and properties of Trefftz variational methods for the Helmholtz equation. Since such methods use oscillating basis functions in the trial spaces, they may achieve better approximation properties than classical piece-wise polynomial spaces. So far, as stated in [20], it is hard to make unequivocal statements about the merits of *exact Trefftz* methods in that theory developed in the literature such as [18,19,16] fails to provide information about the crucial issue of ω-robust accuracy with ω-independent cost, and these methods provide no escape from the pollution error.

Since Trefftz finite-element methods require test and trial functions to be exact local solutions of the Helmholtz equation, these methods are able to easily deal with discontinuous and piece-wise constant wave speeds. However, when the wave speed is smoothly varying, in general there are no exact analytical solutions for the underlying Helmholtz equation so that no analytical Trefftz functions are available either. Therefore, approximate Trefftz functions are appealing for problems with smoothly varying wave speeds; see [7] for ray-based modulated plane-wave discontinuous Galerkin methods and [23] for generalized plane-wave numerical methods, which are two examples of such approximate Trefftz methods.

As stated in [20], the policy of incorporating local direction of rays is particularly attractive for plane-wave based *approximate Trefftz* methods, since plane-wave basis functions naturally encode a direction of propagation, and overall accuracy may benefit significantly from a priori directional adaptivity [30,26,15]; moreover, the survey [20] also remarks that this strategy appears as the most promising way to achieve ω-uniform accuracy with degrees of freedom that remain ω-uniformly bounded or display only moderate growth as $\omega\to\infty$. On the one hand, the methods in [30,15] are able to incorporate ray directions only when the underlying geometry is simple and the wave speed is constant, in which the resulting ray directions can be computed on the fly; on the other hand, the works in [12,13,10,7,25] have developed such ray-based plane-wave methods for smoothly varying wave speeds, in which ray directions are obtained a priori in some ingenious ways.

From this perspective, the method proposed in this article can be viewed as a plane-wave based approximate Trefftz method as well in that we develop a versatile approach to obtain ray directions from highly oscillatory wave fields by carrying out numerical microlocal analysis via a deep neural network and further incorporate these directions into an IPDG method.

2. The ray-based IPDG method

We will present our ray-based method in this section. In Section 2.1, we will present the variational formulation and the approximation space. In Section 2.2, we will give an error estimate on using our basis functions to approximate the solution.

2.1. Method description

We let $\Omega = [0,1]^d$ be the computational domain. We consider a uniform partition, denoted as $\mathcal{T}_H$, of the domain Ω with mesh size H. For each element $K \in \mathcal{T}_H$, we further consider a set of nodal points $\{\hat{x}_{l,K}\}_{l=1}^L$, where L is the total number of nodal points within K. We will use these L points to define the basis functions for each element K. We define $\mathcal{F}_H, \mathcal{F}_H^I$, and $\mathcal{F}_H^B$ to be respectively the set of all faces, interior faces and boundary faces of the partition $\mathcal{T}_H$. We also define N_E to be the number of coarse elements.

Next, we define the approximation space. Let $K \in \mathcal{T}_H$ be an element. There are 2^d standard Lagrange-type bilinear basis functions on K. Let $x_{i,K}$ be the vertices of K and $\varphi_{j,K}$ be the standard Lagrange-type bilinear basis on K such that $\varphi_{j,K}(\mathrm{x}_{i,K}) = \delta_{ij}$. For each element $K \in \mathcal{T}_H$, we define Θ_K as the set of ray directions in K. In particular, each entry $\mathbf{d} \in \Theta_K$ corresponds to a ray direction at the nodal point $\{\hat{x}_{l,K}\}_{l=1}^L$. To start with, we assume that there is only one ray at each nodal point so that there are L entries in Θ_K; we will deal with the case of multiple rays passing through a nodal point later. For each $\mathbf{d}_{l,K} \in \{\mathbf{d}_{l,K}\}_{l=1}^L = \Theta_K$, we define the phase function $\widehat{\phi}_{l,K} : K \to \mathbb{R}$ by

$$\widehat{\phi}_{l,K}(\mathrm{x}) = 1/c(\hat{\mathrm{x}}_{l,K})\mathbf{d}_{l,K} \cdot (\mathrm{x} - \hat{\mathrm{x}}_{l,K}). \tag{2.1}$$

Given a set of directions Θ_K for K, we define the basis functions by

$$\varphi_{j,K}(\mathrm{x}) e^{i\omega\widehat{\phi}_{l,K}(\mathrm{x})}, \quad \mathbf{d}_{l,K} \in \Theta_K, 1 \le l \le L, 1 \le j \le 2^d. \tag{2.2}$$

Note that there are totally $2^d L$ basis functions for each element K when we assume that there is only one ray passing through each nodal point. We denote the local approximation space by $V_H(\Theta_K)$, which consists of linear combinations of these basis functions over $\mathbb{C}$, and the global approximation space by

$$V_H = \Pi_{K \in \mathcal{T}_H} V_H(\Theta_K).$$

We remark that the choice of Θ_K will be presented in Section 3. Next, we discuss the IPDG formulation for the Helmholtz equation with different boundary conditions.

2.1.1. Impedance boundary condition

Consider the Helmholtz problem (1.1) with an impedance boundary condition:

$$\nabla u \cdot n + i(\omega/c)u = g \quad \text{on } \partial\Omega.$$

Following the derivation of the standard IPDG method [17], we obtain the following scheme for this problem: Find $u_H \in V_H$ such that for any $v_H \in V_H$,

$$a_H(u_H, v_H) - \omega^2 \left(c^{-2}u_H, v_H\right)_{L^2(\Omega)} = (f, v_H)_{L^2(\Omega)} + (g, v_H)_{L^2(\partial\Omega)} \tag{2.3}$$

where a_H is given by

$$a_H(u,v) := \int_\Omega \nabla u \cdot \nabla \bar{v} dx - \int_{\mathcal{F}_H^I} \{\!\{\nabla u\}\!\} \cdot [\![\bar{v}]\!] ds - \int_{F_H^I} [\![u]\!] \cdot \{\!\{\nabla \bar{v}\}\!\} ds + \frac{\mathrm{a}_p}{H} \int_{\mathcal{F}_H^I} [\![u]\!] \cdot [\![\bar{v}]\!] ds + i \int_{\mathcal{F}_H^B} \omega c^{-1} u \bar{v} ds$$

with $\mathrm{a}_p > 0$ serving as the penalty parameter, which is taken to be large enough. Here we have used the usual average and jump operators for discontinuous Galerkin methods. Let $K^\pm \in \mathcal{T}_H$ be two elements sharing a face $F \in \mathcal{F}_H^I$, and $\mathbf{n}^\pm$ be the outward normal of $K^\pm$, which is perpendicular to F. Let u be a smooth scalar function defined on $K^\pm$. Then,

$$\{\!\{\nabla u\}\!\} := \frac{1}{2}\left(\nabla u^+ + \nabla u^-\right), \quad [\![u]\!] := u^+\mathbf{n}^+ + u^-\mathbf{n}^-.$$

2.1.2. Cauchy boundary condition

Let $\partial\Omega = \Gamma_D \cup \Gamma_N$, where Γ_D and Γ_N are disjoint. Consider the Helmholtz problem (1.1) with the Cauchy boundary condition:

$$u = g_1 \quad \text{on } \Gamma_D, \quad \text{and} \quad \nabla u \cdot \mathbf{n} = g_2 \quad \text{on } \Gamma_N,$$

where $\mathbf{n}$ is the unit outward normal vector on $\partial\Omega$. We define the interior local approximation space

$$V_H^\circ(\Theta_K) := \text{span}\left\{\varphi_{j,K}(\mathrm{x}) e^{i\omega\widehat{\phi}_K(\mathrm{x})} : \mathbf{d} \in \Theta_K, \varphi_{j,K}|_{\partial\Omega} \equiv 0, 1 \le j \le 2^d\right\}$$

and the interior global approximation space $V_H^\circ := \Pi_{K\in\mathcal{T}_H} V_H^\circ(\Theta_K)$. Then we solve the following problem: Find $u_H \in V_H(\Theta_K)$ such that for any $v_H \in V_H$

$$\begin{aligned} a_H^C(u_H, v_H) - \omega^2\left(c^{-2}u_H, v_H\right)_{L^2(\Omega)} &= (f, v_H)_{L^2(\Omega)}, \quad \text{for any } v_H \in V_H^\circ, \\ \int_{\Gamma_D} u_H v_H ds &= \int_{\Gamma_D} g_1 v_H ds, \quad \text{for any } v_H \in V_H \backslash V_H^\circ, \\ \int_{\Gamma_N} \nabla u_H \cdot \mathbf{n} v_H ds &= \int_{\Gamma_N} g_2 v_H ds, \quad \text{for any } v_H \in V_H \backslash V_H^\circ, \end{aligned}$$

where a_H^C is given by

$$a_H^C(u, v) := \int_\Omega \nabla u \cdot \nabla \overline{v} dx - \int_{\mathcal{F}_H^I} \{\nabla u\} \cdot [\![\overline{v}]\!] ds - \int_{\mathcal{F}_H^I} [\![u]\!] \cdot \{\!\{\nabla \overline{v}\}\!\} ds + \frac{\mathrm{a}_p}{H} \int_{\mathcal{F}_H^I} [\![u]\!] \cdot [\![\overline{v}]\!] ds.$$

2.1.3. Perfectly matched layer (PML)

Supposing that Ω is embedded in a larger domain $\widetilde{\Omega} := [-\delta, 1+\delta]^d$, where $\delta > 0$ is the width of the PMLs. Typically, we choose δ to be approximately several wavelengths. Furthermore we assume that δ is divisible by the mesh-size H. In this way, we can divide Ω into $(M + 2\delta/H)^d$ cubic elements. We denote the set of all these elements by $\widetilde{\mathcal{T}}_H$. Note that $\mathcal{T}_H \subset \widetilde{\mathcal{T}}_H$. Supposing that Θ_K is defined for all $K \in \widetilde{\mathcal{T}}_H$, we extend the definition of the interior local approximation space $V_H^\circ(\Theta_K)$ and the interior global approximation space V_H° to the mesh $\widetilde{\mathcal{T}}_H$. We denote the extended spaces by $\widetilde{V}_H^\circ(\Theta_K)$ and $\widetilde{V}_H^\circ$, respectively. Then we introduce the PML function $s(x)$:

$$s(x) := \frac{1}{1 + i\gamma(x)/\omega}, \quad \gamma(x) := \frac{A_{pml}}{\delta^2}\left(x^2 \chi_{\{x<0\}}(x) + (x-1)^2 \chi_{\{x>1\}}\right)$$

where χ_S is the characteristic function of set S, A_{pml} controls the magnitude of $s(x)$, and δ is the width of the PML. Define

$$\widetilde{\nabla} := \left(s(x_1)\frac{\partial}{\partial x_1}, \ldots, s(x_d)\frac{\partial}{\partial x_d}\right).$$

The IPDG scheme for the Helmholtz problem with PML boundary conditions is: Find $u_H \in \widetilde{V}_H^\circ$ such that

$$\widetilde{a}_H(u_H, v_H) - \omega^2\left(c^{-2}u_H, v_H\right)_{L^2(\Omega)} = (f, v_H)_{L^2(\Omega)}, \text{ for any } v_H \in \widetilde{V}_H^\circ$$

where

$$\widetilde{a}_H(u, v) := \int_\Omega \widetilde{\nabla} u \cdot \widetilde{\nabla} \overline{v} dx - \int_{\mathcal{F}_H^I} \{\!\{\widetilde{\nabla} u\}\!\} \cdot [\![\overline{v}]\!] ds - \int_{\mathcal{F}_H^I} [\![u]\!] \cdot \{\!\{\widetilde{\nabla} \overline{v}\}\!\} ds + \frac{\mathrm{a}_p}{H} \int_{\mathcal{F}_H^I} [\![u]\!] \cdot [\![\overline{v}]\!] ds$$

with a_p the penalty parameter.

2.2. Error analysis

In this section, we will present an error estimate of approximating the solution u of (1.1) by the global approximation space V_H. We first recall some results from [14]. Let b_{DG} be an auxiliary bilinear form given by

$$b_{DG}(u, v) = a_{DG}(u, v) + 2\omega^2(c^{-2}u, v)_{L^2(\Omega)}, \quad \forall u, v \in V + V_H. \tag{2.4}$$

We define the DG-norm as follows,

$$\|v\|_{DG}^2 = \sum_{K\in\mathcal{T}_H} \|\nabla v\|_{L^2(K)}^2 + H^{-1}\|\mathrm{a}_p^{\frac{1}{2}}[\![v]\!]\|_{\mathcal{F}_H^I}^2 + \omega\|v\|_{\mathcal{F}_H^B}^2 + \omega^2\|v\|_{L^2(\Omega)}^2, \tag{2.5}$$

where

$$\|\mathrm{a}_p^{\frac{1}{2}}[\![v]\!]\|_{\mathcal{F}_H^I}^2 = \sum_{E\in\mathcal{F}_H^I} \|\mathrm{a}_p^{\frac{1}{2}}[\![v]\!]\|_{L^2(E)}^2, \quad \|v\|_{\mathcal{F}_H^B}^2 = \sum_{E\in\mathcal{F}_H^B} \|v\|_{L^2(E)}^2.$$

Then we have the following continuity condition,

$$|a_{DG}(u,v)| \le C_{\mathrm{cont}}\|u\|_{DG}\,\|v\|_{DG}, \quad \forall u, v \in V_H, \tag{2.6}$$

and the following coercivity condition,

$$|b_{DG}(v,v)| \ge C_{\mathrm{coer}}\|v\|_{DG}^2, \quad \forall v \in V_H. \tag{2.7}$$

Notice that we have assumed that the penalty parameter a_p is large enough; see [14]. We remark that the constants C_{cont}, C_{coer} and a_p do not depend on ω. Recall that the following Poincare inequality holds,

$$\|v\|_{L^2(K)} \le C_p H|v|_{H^1(K)}, \quad \forall v \in H_0^1(K), \tag{2.8}$$

where, without loss of generality, we assume that C_p is the same for all elements K. On the other hand, we use the notation $a \lesssim b$ to denote the inequality $a \le Cb$, where C is a constant independent of ω and H.

We will first prove an approximation result. We will assume that the following high-frequency asymptotic approximation [1,2]

$$u(\mathrm{x}) = \sum_{n=1}^{N} A_n(\mathrm{x})e^{i\omega\phi_n(\mathrm{x})} + \mathcal{O}\left(\omega^{-(1-\frac{d-3}{2})}\right) \tag{2.9}$$

holds in each element K, where A_n and ϕ_n are smooth functions. Here we have assumed that, when using the geometric optics ansatz, there are multiple wavefronts passing through the point x, and we also assume that the number of wavefronts, N, is the same for all elements to simplify the discussions. Notice that we skip the dependence of A_n and ϕ_n on K to simplify the notations.

On the other hand, we assume that the ray directions in Θ_K are sufficiently accurate. That is for the set of approximate directions $\Theta_K = \{\widehat{\mathbf{d}}_{n,l,K}\}_{n=1,l=1}^{N,L}$, we have

$$\sup_{K\in\mathcal{T}_H, 1\le n\le N, 1\le l\le L} \left|\widehat{\mathbf{d}}_{n,l,K} - \mathbf{d}_{n,l,K}\right| < \varepsilon, \tag{2.10}$$

where $\varepsilon > 0$ is small. Here, we use $\mathbf{d}_{n,l,K}$ to denote the ray direction of ϕ_n at the point $\hat{\mathrm{x}}_{l,K}$ and use $\widehat{\mathbf{d}}_{n,l,K}$ to denote the corresponding approximate ray direction.

Lemma 2.1. *Let u be the solution of (1.1). Assume that $H = \mathcal{O}(\omega^{-1})$. Assume further that $\omega H C_p/c_{\max} < 1$. Then there exists a function $u_I(\mathrm{x}) \in V_H$ such that*

$$\|u(\mathrm{x}) - u_I(\mathrm{x})\|_{DG} \le C\left(\mathcal{O}\left(\omega^{\frac{d-3}{2}}\right) + \omega H + \omega\varepsilon\right) + C_b H\|f\|_{L^2(\Omega)}, \tag{2.11}$$

where $C_b > 0$ is a constant independent of H and ω.

Proof. We first assume that the source term $f = 0$ in (1.1). Let $K \in \mathcal{T}_H$. Let $\mathbf{d}_{n,l,K} := \nabla\phi_n\left(\hat{\mathrm{x}}_{l,K}\right)/\left|\nabla\phi_n\left(\hat{\mathrm{x}}_{l,K}\right)\right|$ be the ray direction of $A_n(\mathrm{x})e^{i\omega\phi_n(\mathrm{x})}$ at $\hat{\mathrm{x}}_{l,K}$. Next, we define the nodal interpolation of u on K by

$$u_I(\mathrm{x}) := \frac{1}{L}\sum_{n=1}^{N}\sum_{j,l} A_n\left(\mathrm{x}_{j,K}\right)\varphi_{j,K}(\mathrm{x})e^{i\omega\widehat{\phi}_{n,l}(\mathrm{x})},$$

where

$$\widehat{\phi}_{n,l}(\mathrm{x}) := \phi_n\left(\hat{\mathrm{x}}_{l,K}\right) + \left|\nabla\phi_n\left(\hat{\mathrm{x}}_{l,K}\right)\right|\widehat{\mathbf{d}}_{n,l,K}\cdot\left(\mathrm{x} - \hat{\mathrm{x}}_{l,K}\right), \tag{2.12}$$

and $\mathrm{x}_{j,K}$, for $j = 1, 2, \cdots, 2^d$, are the vertices of K. The above summation is over $j = 1, 2, \cdots, 2^d$ and $l = 1, 2, \cdots, L$. We start with estimating the error of using the expansion (2.12) to approximate ϕ_n. Notice that

$$
\begin{aligned}
&\|\phi_n(\mathrm{x}) - \widehat{\phi}_{n,l}(\mathrm{x})\|_{L^2(K)} \\
&\leq \left\|\phi_n(\mathrm{x}) - \phi_n\left(\hat{\mathrm{x}}_{l,K}\right) - \nabla\phi_n\left(\hat{\mathrm{x}}_{l,K}\right)\cdot\left(\mathrm{x}-\hat{\mathrm{x}}_{l,K}\right)\right\|_{L^2(K)} \\
&\quad + \left|\nabla\phi_n\left(\hat{\mathrm{x}}_{l,K}\right)\right| \left\|\left(\mathbf{d}_{n,l,K} - \widehat{\mathbf{d}}_{n,l,K}\right)\cdot\left(\mathrm{x}-\hat{\mathrm{x}}_{l,K}\right)\right\|_{L^2(K)} \\
&\lesssim H^2|\phi_n|_{H^2(K)} + \varepsilon H^{1+\frac{d}{2}}\|\phi_n\|_{W^{1,\infty}(K)}.
\end{aligned}
$$

So, we have

$$
\|\phi_n(\mathrm{x}) - \widehat{\phi}_{n,l}(\mathrm{x})\|_{L^2(K)} \lesssim H^2|\phi_n|_{H^2(K)} + \varepsilon H^{1+\frac{d}{2}}\|\phi_n\|_{W^{1,\infty}(K)}. \tag{2.13}
$$

Similarly, we have

$$
\|\nabla(\phi_n(\mathrm{x}) - \widehat{\phi}_{n,l}(\mathrm{x}))\|_{L^2(K)} \lesssim H|\phi_n|_{H^2(K)} + \varepsilon H^{\frac{d}{2}}\|\phi_n\|_{W^{1,\infty}(K)}. \tag{2.14}
$$

Using the triangle inequality and the assumption (2.9), we obtain

$$
\begin{aligned}
&\|u(\mathrm{x}) - u_I(\mathrm{x})\|_{L^2(K)} \\
&\leq \left\|\sum_{n=1}^{N} A_n(\mathrm{x})e^{i\omega\phi_n(\mathrm{x})} - \sum_{n=1}^{N}\sum_{j=1}^{2^d} A_n\left(\mathrm{x}_{j,K}\right)\varphi_{j,K}(\mathrm{x})e^{i\omega\phi_n(\mathrm{x})}\right\|_{L^2(K)} \\
&\quad + \frac{1}{L}\left\|\sum_{n=1}^{N}\sum_{j,l} A_n\left(\mathrm{x}_{j,K}\right)\varphi_{j,K}(\mathrm{x})\left[e^{i\omega\phi_n(\mathrm{x})} - e^{i\omega\widehat{\phi}_{n,l}(\mathrm{x})}\right]\right\|_{L^2(K)} + \mathcal{O}\left(H^{\frac{d}{2}}\omega^{-(1-\frac{d-3}{2})}\right) \\
&\leq \sum_{n=1}^{N}\left\|A_n(\mathrm{x}) - \sum_{j=1}^{2^d} A_n\left(\mathrm{x}_{j,K}\right)\varphi_{j,K}(\mathrm{x})\right\|_{L^2(K)} \\
&\quad + \frac{1}{L}\sum_{n=1}^{N}\sum_{j,l}\|A_n\|_{L^\infty(\Omega)}\left\|e^{i\omega\phi_n(\mathrm{x})} - e^{i\omega\widehat{\phi}_{n,l}(\mathrm{x})}\right\|_{L^2(K)} + \mathcal{O}\left(H^{\frac{d}{2}}\omega^{-(1-\frac{d-3}{2})}\right) \\
&\lesssim H^2\sum_{n=1}^{N}|A_n|_{H^2(K)} + \omega\sum_{n=1}^{N}\|A_n\|_{L^\infty(K)}\|\phi_n(\mathrm{x}) - \widehat{\phi}_{n,l}(\mathrm{x})\|_{L^2(K)} + \mathcal{O}\left(H^{\frac{d}{2}}\omega^{-(1-\frac{d-3}{2})}\right)
\end{aligned}
$$

Summing this inequality for all $K \in \mathcal{T}_H$ and using (2.13), we have

$$
\begin{aligned}
&\|u(\mathrm{x}) - u_I(\mathrm{x})\|_{L^2(\Omega)} \\
&\lesssim H^2\sum_{n=1}^{N}|A_n|_{H^2(\Omega)} + \omega\sum_{n=1}^{N}\|A_n\|_{L^\infty(\Omega)}\left(H^2|\phi_n|_{H^2(\Omega)} + \varepsilon H\|\phi_n\|_{W^{1,\infty}(\Omega)}\right) + \mathcal{O}\left(\omega^{-(1-\frac{d-3}{2})}\right) \\
&\lesssim H^2 + \omega H^2 + \omega\varepsilon H + \mathcal{O}\left(\omega^{-(1-\frac{d-3}{2})}\right)
\end{aligned}
$$

Using the assumptions on H, we have

$$
\omega\|u(\mathrm{x}) - u_I(\mathrm{x})\|_{L^2(\Omega)} \lesssim \omega^{-1} + \omega^2H^2 + \omega\varepsilon + \mathcal{O}\left(\omega^{\frac{d-3}{2}}\right). \tag{2.15}
$$

Next, we estimate the gradient term. Using similar techniques as above, we have

$$
\begin{aligned}
&\|\nabla(u(\mathrm{x})-u_I(\mathrm{x}))\|_{L^2(K)}\\
\leq&\left\|\nabla\Big(\sum_{n=1}^{N}A_n(\mathrm{x})e^{i\omega\phi_n(\mathrm{x})}-\sum_{n=1}^{N}\sum_{j=1}^{2^d}A_n\left(\mathrm{x}_{j,K}\right)\varphi_{j,K}(\mathrm{x})e^{i\omega\phi_n(\mathrm{x})}\Big)\right\|_{L^2(K)}\\
&+\frac{1}{L}\left\|\nabla\Big(\sum_{n=1}^{N}\sum_{j,l}A_n\left(\mathrm{x}_{j,K}\right)\varphi_{j,K}(\mathrm{x})\left[e^{i\omega\phi_n(\mathrm{x})}-e^{i\omega\widehat{\phi}_{n,l}(\mathrm{x})}\right]\Big)\right\|_{L^2(K)}+\mathcal{O}\left(H^{\frac{d}{2}}\omega^{\frac{d-3}{2}}\right)\\
\lesssim&\sum_{n=1}^{N}\left\|\nabla\Big(A_n(\mathrm{x})-\sum_{j=1}^{2^d}A_n\left(\mathrm{x}_{j,K}\right)\varphi_{j,K}(\mathrm{x})\Big)\right\|_{L^2(K)}\\
&+\omega\|\phi_n\|_{W^{1,\infty}(K)}\sum_{n=1}^{N}\left\|A_n(\mathrm{x})-\sum_{j=1}^{2^d}A_n\left(\mathrm{x}_{j,K}\right)\varphi_{j,K}(\mathrm{x})\right\|_{L^2(K)}\\
&+H^{-1}\sum_{n=1}^{N}\sum_{j,l}\|A_n\|_{L^\infty(\Omega)}\left\|e^{i\omega\phi_n(\mathrm{x})}-e^{i\omega\widehat{\phi}_{n,l}(\mathrm{x})}\right\|_{L^2(K)}\\
&+\sum_{n=1}^{N}\sum_{j,l}\|A_n\|_{L^\infty(\Omega)}\left\|\nabla(e^{i\omega\phi_n(\mathrm{x})}-e^{i\omega\widehat{\phi}_{n,l}(\mathrm{x})})\right\|_{L^2(K)}+\mathcal{O}\left(H^{\frac{d}{2}}\omega^{\frac{d-3}{2}}\right)\\
\lesssim&H\sum_{n=1}^{N}|A_n|_{H^2(K)}+\omega H^2\sum_{n=1}^{N}|A_n|_{H^2(K)}+\omega H^{-1}\sum_{n=1}^{N}\|A_n\|_{L^\infty(K)}\|\phi_n(\mathrm{x})-\widehat{\phi}_{n,l}(\mathrm{x})\|_{L^2(K)}\\
&+\omega^2\sum_{n=1}^{N}\|A_n\|_{L^\infty(K)}\|\phi_n(\mathrm{x})-\widehat{\phi}_{n,l}(\mathrm{x})\|_{L^2(K)}\\
&+\omega\sum_{n=1}^{N}\|A_n\|_{L^\infty(K)}\|\nabla(\phi_n(\mathrm{x})-\widehat{\phi}_{n,l}(\mathrm{x}))\|_{L^2(K)}+\mathcal{O}\left(H^{\frac{d}{2}}\omega^{\frac{d-3}{2}}\right)
\end{aligned}
$$

Summing this inequality for all $K\in\mathcal{T}_H$, using (2.13)-(2.14), and using the assumptions on H, we have

$$
\begin{aligned}
&\|\nabla(u(\mathrm{x})-u_I(\mathrm{x}))\|_{L^2(\Omega)}\\
\lesssim&H+\omega H^2+\omega H^{-1}(H^2+\varepsilon H)+\omega^2(H^2+\varepsilon H)+\omega(H+\varepsilon)+\mathcal{O}\left(\omega^{\frac{d-3}{2}}\right)\\
\lesssim&\omega^{-1}+\omega H+(\omega H)^2+\omega\varepsilon+\mathcal{O}\left(\omega^{\frac{d-3}{2}}\right).
\end{aligned}
\tag{2.16}
$$

We will now estimate the terms involving faces. First, we recall the trace inequality,

$$
\|v\|^2_{L^2(\partial K)}\lesssim H^{-1}\|v\|^2_{L^2(K)}+H|v|^2_{H^1(K)}.
\tag{2.17}
$$

So,

$$
\omega^{\frac{1}{2}}\|u(\mathrm{x})-u_I(\mathrm{x})\|_{\mathcal{F}_H^B}\lesssim\omega^{\frac{1}{2}}H^{-\frac{1}{2}}\|u(\mathrm{x})-u_I(\mathrm{x})\|_{L^2(\Omega)}+\omega^{\frac{1}{2}}H^{\frac{1}{2}}|u(\mathrm{x})-u_I(\mathrm{x})|_{H^1(\Omega)}.
$$

Moreover,

$$
H^{-\frac{1}{2}}\|[\![u(\mathrm{x})-u_I(\mathrm{x})]\!]\|_{\mathcal{F}_H^I}\lesssim H^{-1}\|u(\mathrm{x})-u_I(\mathrm{x})\|_{L^2(\Omega)}+|u(\mathrm{x})-u_I(\mathrm{x})|_{H^1(\Omega)}.
$$

Combining all results, we obtain (2.11) for the case $f=0$.

Now, we consider the general case with non-zero source f. Let $z_K\in H^1_0(K)$ be the solution of

$$
-\nabla^2 z_K-(\omega/c)^2 z_K=f,\quad \text{in } K.
\tag{2.18}
$$

Here we assume that the problem (2.18) is well-posed. Then,

$$
|z_K|^2_{H^1(K)}-(\omega/c)^2\|z_K\|^2_{L^2(K)}=(f,z_K)_{L^2(\Omega)}.
$$

Using the assumption $\omega H C_p / c_{\max} < 1$, there is a constant C_b such that

$$\|z_K\|_{DG} \leq C_b H \|f\|_{L^2(\Omega)}. \tag{2.19}$$

Hence, (2.11) follows. □

Next, we prove a quasi-optimality result.

Lemma 2.2. *Let u be the solution of (1.1) and $u_H \in V_H$ be the solution of (2.3). Assume that $2c_{\min}^{-2}\omega H C_b < C_{\text{coer}}$. We have*

$$\|u - u_H\|_{DG} \lesssim \inf_{v_H \in V_H} \|u - v_H\|_{DG} + \mathcal{O}\left(\omega^{\frac{d-3}{2}}\right) + \omega H + \omega\varepsilon.$$

Proof. By the coercivity condition (2.7) and the Galerkin orthogonality,

$$\begin{aligned} C_{\text{coer}}\|u - u_H\|_{DG}^2 &\leq b_{DG}(u - u_H, u - u_H) \\ &= a_{DG}(u - u_H, u - v_H) + 2\omega^2 (c^{-2}(u - u_H), u - u_H)_{L^2(\Omega)} \end{aligned}$$

for all $v_H \in V_H$. So, we have

$$C_{\text{coer}}\|u - u_H\|_{DG}^2 \leq C_{cont}\|u - u_H\|_{DG}\|u - v_H\|_{DG} + 2\omega^2 c_{\min}^{-2}\|u - u_H\|_{L^2(\Omega)}^2.$$

Let z be the solution of the dual problem with source $\omega^2(u - u_H)$, namely,

$$-\Delta z - (\omega/c)^2 z = \omega^2(u - u_H).$$

Using the same argument as in the proof of Lemma 2.1, we have

$$\|z(\mathrm{x}) - z_I(\mathrm{x})\|_{DG} \leq C\left(\mathcal{O}\left(\omega^{\frac{d-3}{2}}\right) + \omega\varepsilon\right) + \omega^2 H C_b \|u - u_H\|_{L^2(\Omega)}. \tag{2.20}$$

By the definition of z and the consistency of the IPDG scheme, we have

$$\begin{aligned} \omega^2\|u - u_H\|_{L^2(\Omega)}^2 &= a_{DG}(u - u_H, z) \\ &= a_{DG}(u - u_H, z - z_I) \\ &\leq \|u - u_H\|_{DG}\|z - z_I\|_{DG}. \end{aligned}$$

We have

$$\begin{aligned} \omega^2\|u - u_H\|_{L^2(\Omega)}^2 &\leq C\left(\mathcal{O}\left(\omega^{\frac{d-3}{2}}\right) + \omega\varepsilon\right)\|u - u_H\|_{DG} + \omega^2 H C_b \|u - u_H\|_{DG}\|u - u_H\|_{L^2(\Omega)} \\ &\leq C\left(\mathcal{O}\left(\omega^{\frac{d-3}{2}}\right) + \omega\varepsilon\right)\|u - u_H\|_{DG} + \omega H C_b \|u - u_H\|_{DG}^2. \end{aligned}$$

So, if $2c_{\min}^{-2}\omega H C_b < C_{\text{coer}}$, we have

$$\|u - u_H\|_{DG} \lesssim \|u - v_H\|_{DG} + \mathcal{O}\left(\omega^{\frac{d-3}{2}}\right) + \omega H + \omega\varepsilon.$$

This completes the proof. □

Finally, we state the error bound, which proof is based on Lemma 2.1 and Lemma 2.2.

Theorem 2.3. *Let u be the solution of (1.1) and $u_H \in V_H$ be the solution of (2.3). Assume that the conditions on H stated in Lemma 2.1 and Lemma 2.2 hold. We have*

$$\|u - u_H\|_{DG} \lesssim \mathcal{O}\left(\omega^{\frac{d-3}{2}}\right) + \omega H + \omega\varepsilon + H\|f\|_{L^2(\Omega)}, \tag{2.21}$$

and

$$\|u - u_H\|_{L^2(\Omega)} \lesssim \mathcal{O}\left(\omega^{\frac{d-3}{2}-1}\right) + H + \varepsilon + \omega^{-1}H\|f\|_{L^2(\Omega)}. \tag{2.22}$$

We remark that the convergence of the DG-norm (2.21) requires $\omega H \to 0$.

3. Learning ray directions via deep neural networks

In this section, we will present a machine learning approach to determine ray directions required in the ray-based basis functions.

3.1. Deep neural network model

We will use the notation $\mathcal{N}$ to denote a neural network with M layers, where x is the input and y is the corresponding output. We write

$$y = \mathcal{N}(x;\theta) := \sigma\left(W_M \sigma\left(\cdots \sigma\left(W_2 \sigma\left(W_1 x + b_1\right) + b_2\right)\cdots\right) + b_M\right),$$

where $\theta := (W_1, \cdots, W_M, b_1, \cdots, b_M)$, W_i are the weight matrices and b_i are the bias vectors, and σ is the activation function. A neural network describes the connection of a collection of nodes (neurons) sitting in successive layers. The output neurons in each layer are simultaneously the input neurons in the next layer. The data propagate from the input layer to the output layer through hidden layers. The neurons can be switched on or off as the input is propagated forward through the network.

Suppose we are given a collection of sample pairs $\{(x_j, y_j)\}_{j=1}^{N_s}$, and the goal is then to find θ^* by solving an optimization problem

$$\theta^* = \underset{\theta}{\operatorname{argmin}} \sum_{j=1}^{N_s} \operatorname{loss}(y_j, \mathcal{N}(x_j;\theta)),$$

where N_s is the number of samples. Here, the function $\operatorname{loss}(y_j, \mathcal{N}(x_j;\theta))$ is known as the loss function. One needs to select a suitable number of layers, a suitable number of neurons in each layer, a suitable activation function, the loss function, and the optimizers for the network.

We will use a deep neural network $\mathcal{N}$ to model the process of determining ray directions for our basis functions. Let ω be the frequency of the high frequency Helmholtz equation (1.1) to be solved. Let $\widetilde{\omega}$ be the reduced frequency that we use to determine the ray directions as in the NMLA method. Recall that the basis functions in the ray-based IPDG method are defined by (2.2), which will need a set of ray directions Θ_K for each element $K \in \mathcal{T}_H$. We will perform the training process on a generic element K, and apply the result to all elements. The resulting neural network is able to predict ray directions needed by our proposed method. We will use functions of the form $e^{i\sigma \mathbf{d}\cdot(\mathrm{x}-\hat{\mathrm{x}})}$ as the input of the training data.

We will choose the random number σ uniformly from the range $[\widetilde{\omega}-\delta, \widetilde{\omega}+\delta)$, $\delta > 0$, and choose $\hat{\mathrm{x}}$ randomly from the element K. This choice for σ is due to the fact that the wave number for the Helmholtz equation with reduced frequency (1.2) is $\widetilde{\omega}/c$, and we will take $\sigma = \widetilde{\omega}/c(\hat{\mathrm{x}}_{l,K})$ when we apply the neural network. Furthermore, we observe that the outputs that we need are the values of the directions $\mathbf{d}$ which are randomly generated from the unit circle (or sphere).

The following summarizes the training settings of our deep neural network:

- Input: $x = \{\sum_j A_j e^{ik\mathbf{d}_j\cdot(\mathrm{x}-\hat{\mathrm{x}}_{l,K})}\}$, where we notice that x and x are different notations. Here $\hat{\mathrm{x}}_{l,K}$ is a set of randomly chosen points in K and $\mathbf{d}_j$ is a set of randomly chosen directions of unit length. Note that there are multiple ray directions at each point $\hat{\mathrm{x}}_{l,K}$. The functions A_j are bilinear (for the 2D case) or trilinear (for the 3D case) in K with randomly selected nodal point values. The input is therefore a superposition of plane waves in K. We remark that the actual number of random points and ray directions will be specified in each of the examples in Section 4.
- Output: $y = \{\mathbf{d}_j\}$. The output of the network contains the corresponding ray directions $\mathbf{d}_j$ at the point $\hat{\mathrm{x}}_{l,K}$. Note that we assume that the total number of directions N on each element K is fixed.
- Sample pairs: $N_s = 10000$ sample pairs of (x_k, y_k) are used. Note that each x_k is a superposition of plane waves with directions $\{\mathbf{d}_j\}$ and y_k are the corresponding directions $\{\mathbf{d}_j\}$.
- Standard loss function:

$$\frac{1}{N_s}\sum_{k=1}^{N_s} \|y_k - \mathcal{N}(x_k;\theta)\|_2^2 \qquad \text{(MSE)}$$

- Customized loss function:

$$\frac{1}{N_s}\sum_{k=1}^{N_s} \|y_k - \mathcal{N}(x_k;\theta)\|_2^2 + \frac{1}{N_s}\sum_{k=1}^{N_s}\sum_{j} \left| \|y_{k,j}\|_2^2 - \|\mathcal{N}(x_k;\theta)_j\|_2^2 \right| \quad \text{(MSE+norm1)}$$

  In the above $y_{k,j}$ denotes the j-th direction for the k-th training data, and $\mathcal{N}(x_k;\theta)_j$ denotes the j-th predicted direction for the k-th training data. We will use this customized loss function to improve the performance. Notice that we added a term to normalize the length of the predicted ray directions.

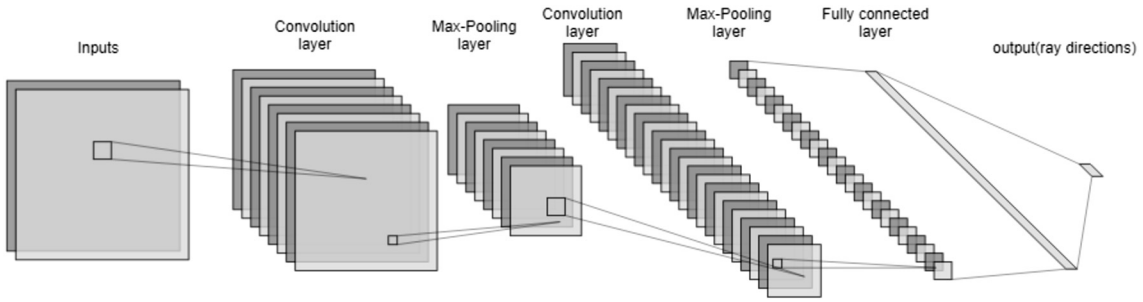

Fig. 3.1.1. A schematic for our deep neural network (Algorithm 1).

- Activation function: The popular ReLU function (the rectified linear unit activation function) is a common choice for activation function in training deep neural network architectures.

In between the input and output layers, we use $M-1$ convolution layers with a constant rate of kernel size 3 with batch-normalization activations on given parameters and pooling operation with Max pooling filter and a constant rate of kernel size 2. The neural network follows with a fully connected hidden layer and finally an output layer. The details of neural network are shown in Algorithm 1, and a schematic is shown in Fig. 3.1.1. As for the training optimizer, we minimize the loss function by using AdaMax, which is a stochastic gradient descent (SGD) type algorithm well suited for high dimensional parameter space.

In Algorithm 1, the neural network takes an input function $\mathbf{u}$, which is defined on an element K. The output of the algorithm is a set of ray directions $\{\mathbf{d}_j\}$. Thus, our neural network will learn the ray directions from the wave field. This is our Algorithm 2. The algorithm takes a global wave field as input. Then the restriction of the wave field in each element K_k, $k = 1, 2, \cdots, N_E$, is entered into the neural network (Algorithm 1), which, in turn, returns the local ray directions. The output of the Algorithm 2 is the set of all ray directions.

Algorithm 1: Neural Network.

Function $\{\mathbf{d}_j\}_{j=1}^{N} = NN(\mathbf{u})$:
  $F_1 = \mathbf{u}$
  for $k = 1 : M-1$ **do**
    $F_{bk} = \text{BatchNorm}(F_k)$ // batch-normalize activations
    $F_{ck} = \text{Conv}(F_{bk})$ // convolution operator
    $F_{k+1} = \text{Pool}(F_{ck})$ // pooling operation with Max pooling filter
  end
  $F_f = \text{flat}(F_{L+1})$ // flatten layer
  $F_{fc} = \text{FConn}(F_f)$ // fully connected layer
  $\{\mathbf{d}_j\}_{j=1}^{N} = \text{output}(F_{fc})$ // output layer

Algorithm 2: Ray Learning.

Function $\left\{\{\mathbf{d}_{j,K_k}\}_{j=1}^{N}\right\}_{k=1}^{N_E} = RAYLEARNING(\mathbf{u})$:
  for $k = 1 : N_E$ **do**
    $\{\mathbf{d}_{j,K_k}\}_{j=1}^{N} = \text{NN}(\mathbf{u}|_{K_k})$ // CNN followed by fully connected hidden layer
  end
  return $\left\{\{\mathbf{d}_{j,N_k}\}_{j=1}^{N}\right\}_{k=1}^{N_\Omega}$

Remark. *In practice, it is desirable to learn the number of ray directions instead of fixing it. To do so, we will set an upper bound of the number of ray directions. That is, N is the upper bound of the number of ray directions. In the case that the solution has fewer ray directions, we will apply the singular value decomposition to remove the redundant directions. We will illustrate this in Section 4.5.*

3.2. The ray-based IPDG method

In order to learn the ray directions, we first solve the Helmholtz equation (1.2) with the reduced frequency $\widetilde{\omega}$. Then we use Algorithm 2 with the reduced frequency solution $\widetilde{u}$ to learn the ray directions for each coarse element. The resulting ray directions will be used in our ray-based IPDG method to solve the high frequency Helmholtz equation (1.1).

First of all, we need the standard IPDG method to solve the reduced frequency Helmholtz equation (1.2). We denote the reduced frequency Helmholtz solution as $\widetilde{u}$ and the detailed implementation of the standard IPDG is shown in Algorithm 3. Here, we use $\mathcal{B}$ to denote the IPDG bilinear form and $\mathcal{F}$ to denote the source terms. In addition, $\{\varphi_j\}$ denotes the standard IPDG basis functions, and N_Ω denotes the number of basis functions.

Algorithm 3: Standard IPDG Helmholtz Solver.

Function $\mathbf{u}_\omega = \textit{S-IPDG}(\omega, h, c, f, g)$**:**
  for $i, j = 1 : N_\Omega$ **do**
    $\mathbf{H}_{ij} = \mathcal{B}(\varphi_i, \varphi_j)$ // Assemble Helmholtz matrix
    $\mathbf{b}_j = \mathcal{F}(\varphi_j)$ // Assemble right-hand side
  end
  $\mathbf{u}_\omega = \mathbf{H}^{-1}\mathbf{b}$ // Solve linear system
  return $\mathbf{u}_\omega$

Once the ray directions for all elements have been computed, we then construct the ray-IPDG space V_H. Next, we will introduce the details of the ray-IPDG method, which is implemented in Algorithm 4. We note that the algorithm takes, among other quantities, the ray directions $\left\{d_{l,K_j}\right\}_{l=1,j=1}^{N_j,N_E}$ as input. Here, we recall again that N_E is the number of elements. To simplify the notations, we use N_j to denote the number of ray directions in the element K_j. Notice that, since each nodal point $\hat{\mathrm{x}}_{l,K_j}$ can have multiple ray directions, $\hat{\mathrm{x}}_{l,K_j}$ and $\hat{\mathrm{x}}_{l',K_j}$ can represent the same nodal point when $l \neq l'$.

Algorithm 4: Ray-IPDG Helmholtz Solver.

Function $\mathbf{u}_{\omega,H} = \textit{RAY-IPDG}(\omega, H, c, f, g, \left\{d_{l,K_j}\right\}_{l=1,j=1}^{N_j,N_E})$**:**
  $N_{\text{dof}} = 0$
  for $j = 1 : N_E, l = 1 : N_j, k = 1 : 2^d$ **do**
    $N_{dof} = N_{dof} + 1, m = N_{dof}$
    $\psi_m(\mathrm{x}) = \varphi_k(\mathrm{x}) e^{i\omega/c\left(\hat{\mathrm{x}}_{l,K_j}\right)\mathbf{d}_{l,K_j}\cdot\mathrm{x}}$ // Construct ray-IPDG basis
    $\widehat{\psi}_m = \psi_m\left(\hat{\mathrm{x}}_{l,K_j}\right)$ // Nodal values of ray-IPDG basis
  end
  for $m, n = 1 : N_{dof}$ **do**
    $\mathbf{H}_{m,n} = \mathcal{B}(\psi_m, \psi_n)$ // Assemble Helmholtz matrix
    $\mathbf{b}_n = \mathcal{F}(\psi_n)$ // Assemble right-hand side
  end
  $\mathbf{v} = \mathbf{H}^{-1}\mathbf{b}$ // Coefficients of ray-IPDG basis
  $\mathbf{u}_{\omega,H} = \mathbf{v} \cdot \widehat{\psi}$ // Ray-IPDG solution on mesh nodes
  return $\mathbf{u}_{\omega,H}$

Finally, our ray-based IPDG high-frequency IPDG Helmholtz solver is formed by the above ray-IPDG method and the deep neural network, which is presented in Algorithm 5. The accuracy of the solution computed by Algorithm 4 using ray-IPDG depends on the accuracy of ray directions computed by the neural network model.

Algorithm 5: Ray-based IPDG High-Frequency Helmholtz Solver.

Function $\mathbf{u}_H = \textit{DEEP-RAY-IPDG}(\omega, H, c, f, g)$**:**
  $\tilde{\omega} \sim \sqrt{\omega}, H \sim \omega^{-1}, h \sim \omega^{-1}$
  $\mathbf{u}_{\widetilde{\omega},} = \text{S-IPDG}(\widetilde{\omega}, h, c, f, g)$ // Low-frequency waves
  $\mathbf{d} := \left\{d_{l,K_j}\right\}_{l=1,j=1}^{N_j,N_E} = \text{RAYLEARNING}\,(\mathbf{u}_{\widetilde{\omega}})$ // Low-freq ray learning
  $\mathbf{u}_{\omega,H} = \text{RAY-IPDG}\,(\omega, H, c, f, g, \mathbf{d})$ // High-frequency waves
  return $\mathbf{u}_{\omega,H}$

4. Numerical examples

In this section, we will present some numerical examples to show the performance of our proposed deep learning based IPDG high-frequency Helmholtz equation solver using ray-based basis functions. In our 2D simulations, we will take

$$\omega = 2^3 \times 10\pi\,, \quad \tilde{\omega} = \sqrt{2^3} \times 10\pi\,.$$

In 3D simulations, we let

$$\omega = 2^2 \times 10\pi\,, \quad \tilde{\omega} = \sqrt{2^2} \times 10\pi\,.$$

We set the computational domain $\Omega = [0,1]^2$ in Examples 1-5. Also, the wave speed for Examples 1-5 is set as $c = 1$ and the source is set as $f = 0$. The impedance boundary condition is used for Examples 1-5:

$$\nabla u \cdot n + i(\omega/c)u = g \quad \text{on } \partial\Omega.$$

For Examples 6 and 7, we solve the problem in the computational domain $\Omega'' = [0.25, 0.75]^2$ supplemented with the Cauchy boundary condition on $x_2 = 0.25$ and PML conditions on the other three sides. We will consider inhomogeneous sound speeds for these two examples. We will also present some 3D test cases in Example 8, where the impedance boundary condition is considered. We will compare the performance by using the NMLA method and our ray-based IPDG method.

In Examples 1-5, the computational time for the NMLA method is 3.265 s and that for our deep learning based method is 0.120 s. By using the NVIDIA TensorRT which optimizes the network by combining layers and optimizing kernel selection for improved latency, throughput, power efficiency, and memory consumption, the time consumed for the deep learning based method will decrease to 0.002 s. We can see an improvement to computational efficiency.

4.1. Example 1

For the first numerical example, we will take the wave field as

$$u_1 = e^{i\omega x_1}.$$

The computational domain is divided into a 40×40 grid, that is, $H = 1/40$. For each element, we consider a finer grid of 4×4. This finer grid is used to define the reduced frequency solution. Fig. 4.1.1 shows a plot of exact directions, approximate directions by the neural network, the reduced frequency IPDG solution, and the high frequency ray-IPDG solution from the approximate directions. Note that we only show the ray directions at some selected points for clarity of presentation.

Table 4.1.1 shows the relative errors of directions and IPDG solutions from the NMLA and the neural network method. Our neural network is trained to learn one ray direction for each element. The relative error of ray directions from the NMLA method is 0.1418. The relative error of ray directions from our neural network method is 0.07605. In terms of the solution, the relative error is 0.03296 for the NMLA method. In addition, the relative error is 0.00967 for the neural network method. We also observe that the loss function including the length of the ray directions gives better results.

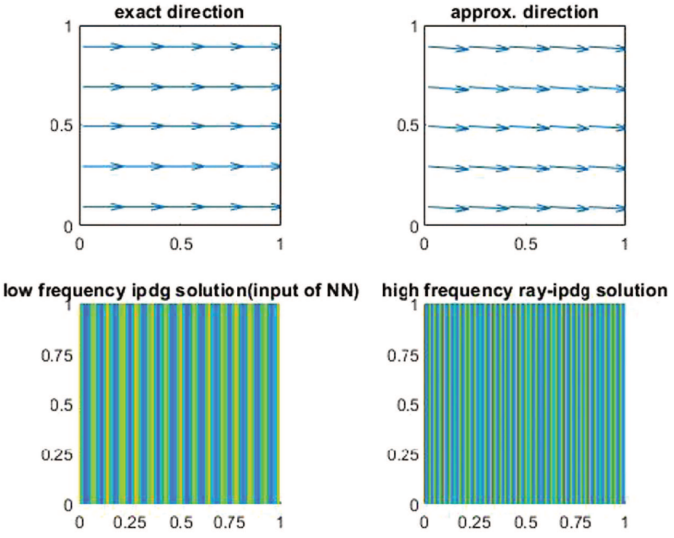

Fig. 4.1.1. Example 1: ray-directions and solutions.

Table 4.1.1
Root mean square error for Example 1.

	relative error of directions	relative error of solutions
NMLA	0.1418	0.03296
Neural Network(mse+norm1)	0.07605	0.00967
Neural Network(mse)	0.08687	0.01693

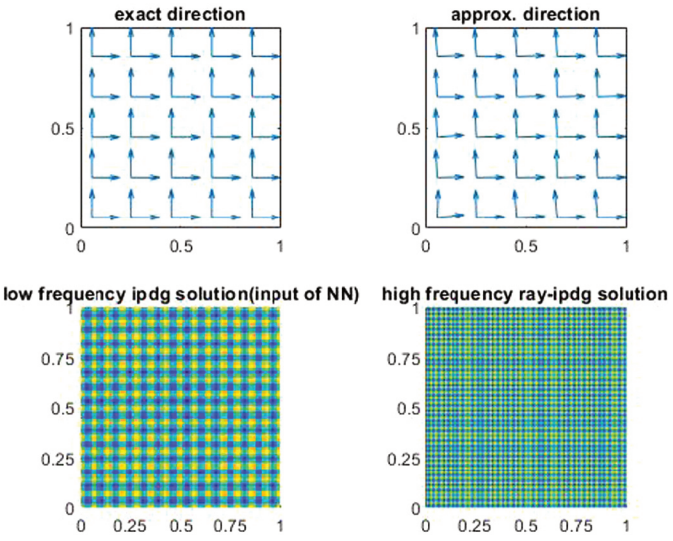

Fig. 4.2.1. Example 2: ray directions and solutions.

Table 4.2.1
Root mean square error for Example 2.

	relative error of directions	relative error of solutions
NMLA	0.1418	0.01979
Neural Network(mse+norm1)	0.05782	0.00369
Neural Network(mse)	0.08257	0.00913

4.2. Example 2

For the second numerical example, we will take the wave field as

$$u_2 = e^{i\omega x_1} + e^{i\omega x_2}.$$

The grid is the same as that of Example 1. Fig. 4.2.1 shows a plot of exact directions, approximate directions by the neural network, the reduced frequency IPDG solution, and the high frequency ray-IPDG solution from the approximate directions.

Table 4.2.1 shows the relative errors of ray directions and solutions from the NMLA and our neural network method. Our neural network is trained to learn two ray directions in each element. The relative error of ray directions from the NMLA method is 0.1418, and the relative error of ray directions from our neural network method is 0.05782. Moreover, the relative error of the solution from the NMLA method is 0.01979, while the relative error from our neural network method is 0.00369. We also observe that the customized loss function gives better performance.

4.3. Example 3

For the third numerical example, we will take the wave field as

$$u_3 = \sqrt{\omega} H_0^{(1)}(\omega |\mathrm{x} - \mathrm{x}_0|)$$

where $\mathrm{x}_0 = (2, 2)$. The grid size is the same as that of Example 1. Fig. 4.3.1 shows a plot of exact directions, approximate directions by the neural network, the reduced frequency IPDG solution, and the high frequency ray-IPDG solution from the approximate directions.

Table 4.3.1 shows the relative errors of ray directions and the solutions from both the NMLA and our method. In particular, the relative errors of ray direction from the NMLA method are 0.07571, while that from our method is 0.04347. Also, the relative error for the solution is 0.01197 and 0.00272 for the NMLA method and our method respectively. We again observe that our method is able to give accurate approximation solution. We remark that our network is trained to learn one ray direction in each element.

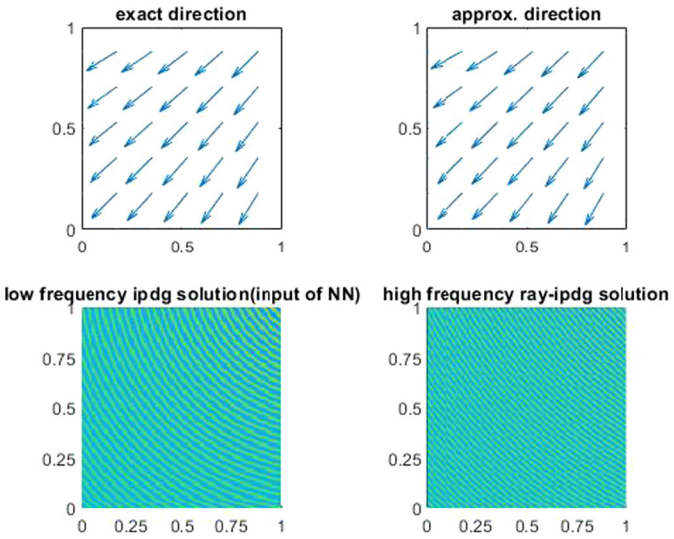

Fig. 4.3.1. Example 3: ray directions and solutions.

Table 4.3.1
Root mean square error for Example 3.

	relative error of directions	relative error of solutions
NMLA	0.07571	0.01197
Neural Network(mse+norm1)	0.04347	0.00272
Neural Network(mse)	0.04996	0.00538

4.4. Example 4

For the fourth numerical example, we will take the wave field as

$$u_3 = \sqrt{\omega} H_0^{(1)}\left(\omega \left|\mathrm{x} - \mathrm{x}_{0,1}\right|\right) + 0.5\sqrt{\omega} H_0^{(1)}\left(\omega \left|\mathrm{x} - \mathrm{x}_{0,2}\right|\right)$$

where $\mathrm{x}_{0,1} = (2, 2)$ and $\mathrm{x}_{0,2} = (-0.5, 2)$. We use the same grid setting as in Example 1. Fig. 4.4.1 shows a plot of exact directions, approximate directions by the neural network, the reduced frequency lPDG solution, and the high frequency ray-lPDG solution from the approximate directions.

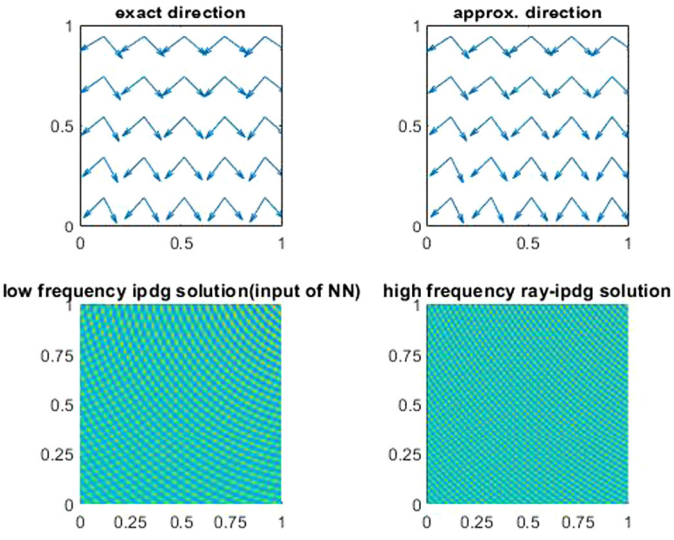

Fig. 4.4.1. Example 4: ray directions and solutions.

Table 4.4.1
Root mean square error for Example 4.

	relative error of directions	relative error of solutions
NMLA	0.1048	0.01183
Neural Network(mse+norm1)	0.07637	0.00333
Neural Network(mse)	0.05549	0.00242

Table 4.4.1 shows the relative error of ray directions and numerical solutions from the NMLA and our method. In particular, the relative error of ray directions from the NMLA method is 0.1048, while the relative error of ray directions from our method is 0.07637. In addition, the relative errors of the approximate solutions from the NMLA method and our method are 0.01183 and 0.00333, respectively. We observe that our method is able to give an accurate solution efficiently. We remark that the neural network is trained to learn two ray directions in each element.

4.5. Example 5

This section is devoted to test our method for predicting the number of ray directions for each element. We will repeat Examples 1 and 3. However, instead of specifying the number of ray directions, we only specify a maximum number of directions. Then we use our deep neural network to predict the directions, and we then use the SVD to determine the number of ray directions by eliminating redundant directions.

We will repeat the Example 1. We will test the two different neural networks. One of them will learn two ray directions in each element, and the other will learn four ray directions in each element. Then the SVD is applied to remove redundant ray directions by considering magnitude of singular values. For the neural network learning two ray directions, the energy of the first singular vector is 98.94%, where the energy is defined using singular values. For the neural network learning

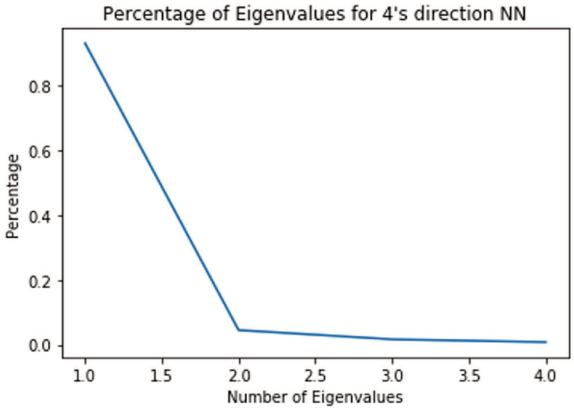

Fig. 4.5.1. Percentage of Eigenvalues for Example 1.

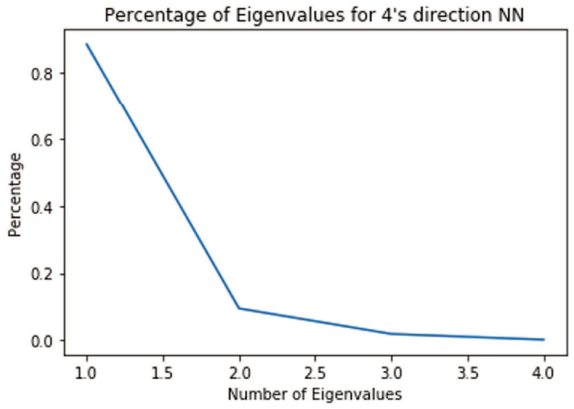

Fig. 4.5.2. Percentage of Eigenvalues for Example 3.

four ray directions, we present the eigenvalues in Fig. 4.5.1. We observe that the first eigenvector carries most of the energy, and it shows that the proposed technique is able to give one ray direction.

We will also repeat Example 3. For the neural network learning two ray directions, the energy of the first singular vector is 96.65%, where the energy is defined using singular values. For the neural network learning four ray directions, we present the eigenvalues in Fig. 4.5.2. We observe that the first eigenvector carries most of the energy, and it shows that the proposed technique is able to give one ray direction.

4.6. Example 6

In our next numerical example, we will take the wave field as $u_6 =$ the free-space solution with wave speed $c_6 = 1 - 0.5e^{-100[(y-0.4)^2+(x+0.5y-0.7)^2)]}$, which is shown in Fig. 4.6.1 and source

$$f_6 := 10^4 e^{-10^4|x-x_0|^2}$$

where $x_0 = (0.5, 0.1)$. The domain is divided into 40×40 coarse grid, that is, $H = 1/40$. For each element, we consider a finer grid of 8×8. Fig. 4.6.1 shows a plot of reference directions predicted by the NMLA method, the approximate ray directions by our neural network, the reduced frequency IPDG solution, and the high frequency ray-IPDG solution using the approximate ray directions. We have shown the reduced frequency solution, which is the input of the neural network. Moreover, we have shown both the reference solution computed by the NMLA method as well as the approximate solution computed by our neural network method. We observe good agreement between these two solutions.

Table 4.6.1 shows the relative error of solutions from the NMLA and our neural network methods. The neural network will learn 4 ray directions in each element. The relative error of the high-frequency ray-IPDG solution from the NMLA method is 0.02723, while that from our neural network method is 0.02540. We observe a comparable performance of these two methods. We remark that our neural network is able to predict ray directions more efficiently.

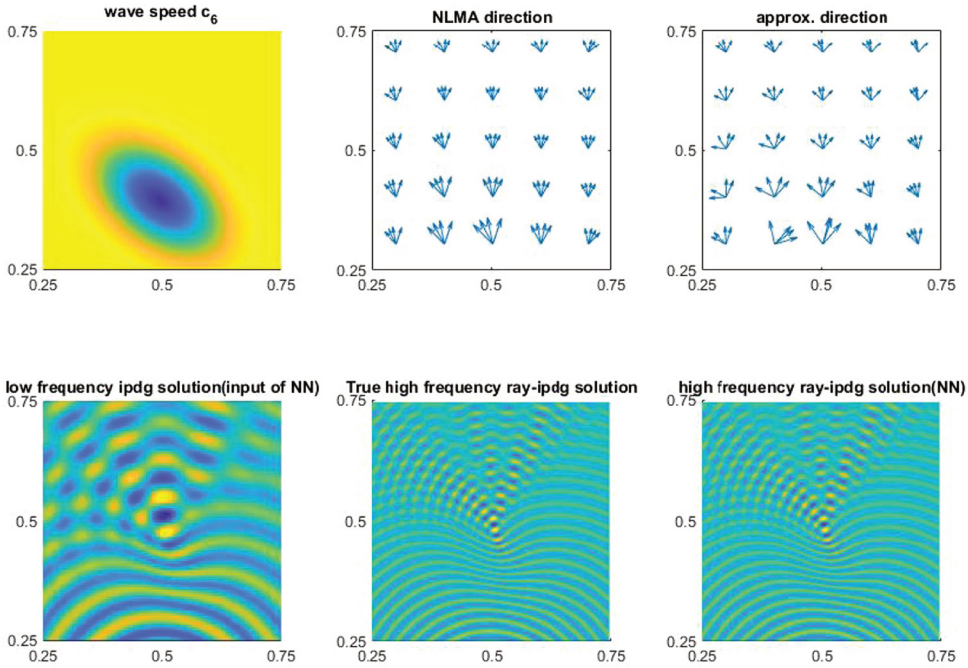

Fig. 4.6.1. Example 6: sound speed, ray directions and solutions.

Table 4.6.1
Root mean square error for Example 6.

	relative error of solutions
NMLA	0.02723
Neural Network (mse+norm1)	0.02540

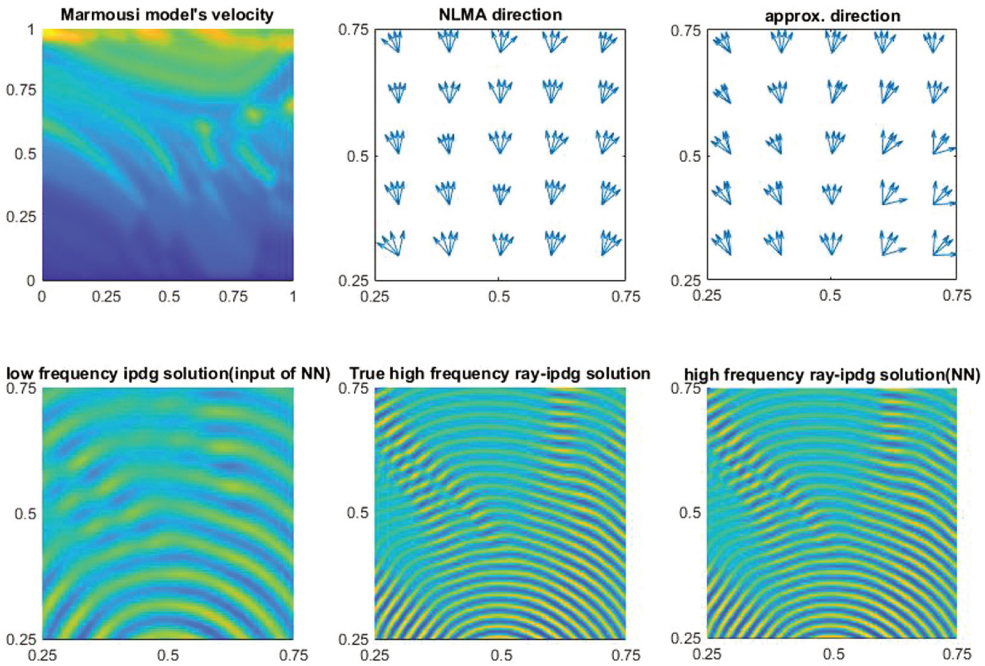

Fig. 4.7.1. Example 7: sound speed, ray directions and solutions.

Table 4.7.1
Root mean square error for Example 7.

	relative error of solutions
NMLA	0.02889
Neural Network (mse+norm1)	0.03018

4.7. Example 7

For the seventh numerical example, we will take the wave field as $u_7 =$ the free space solution with wave speed $c_7 =$ a scaled smooth Marmousi model showed in Fig. 4.7.1. Other settings are the same as Example 6. Fig. 4.7.1 shows a plot of exact directions, approximate directions by the neural network, the reduced frequency IPDG solution, and the high frequency ray-IPDG solution from the approximate directions. We again observe very good agreement between the reference and approximate solutions.

Table 4.7.1 shows the relative errors of the solutions from the NMLA and the neural network method. Our neural network will give 4 ray directions in each element. The relative error of the high frequency ray IPDG solution from the NMLA method is 0.02889, while that from our neural network method is 0.03018. We observe that our neural network method gives a reasonable result. We remark that our method is able to predict the ray directions in a more efficient way.

4.8. 3D examples

Finally, we consider some 3D numerical examples. We will first take the wave field as

$$u_8 = e^{i\omega x_1},$$

the wave speed $c = 1$. The domain $[0, 1] \times [0, 0.2] \times [0, 0.2]$ is divided into $20 \times 4 \times 4$ grid. Our neural network is designed to learn one ray direction in each element.

Fig. 4.8.1 shows the reduced frequency IPDG solution and the high frequency ray-IPDG solution. Note that the reduced frequency solution is used as input of our neural network. The relative error of our approximate solution is 0.0399.

For the second 3D numerical example, we will take the wave field as

$$u_9 = e^{i\omega x_1} + e^{i\omega x_2} + e^{i\omega x_3},$$

the wave speed $c = 1$. The domain and mesh are the same as the previous example. For this example, we take two points in each element, and our neural network will learn three ray directions in each of these two points. Fig. 4.8.2 shows the

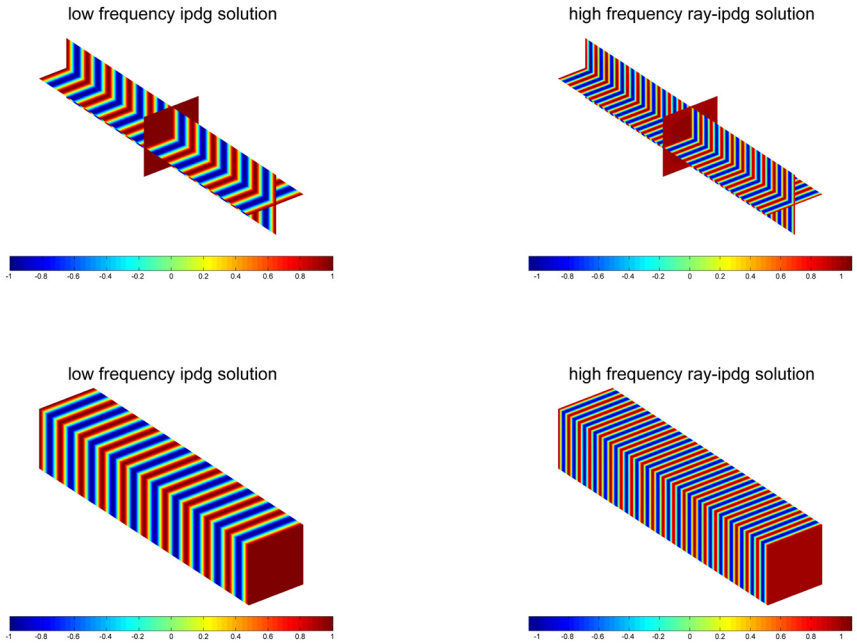

Fig. 4.8.1. 3D example 1, L^2 relative error is 0.0399. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

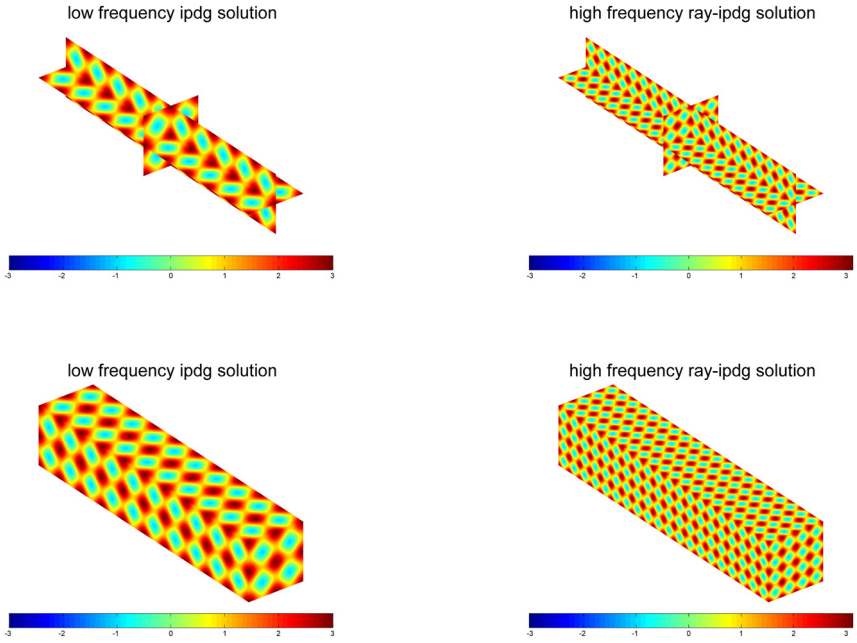

Fig. 4.8.2. 3D example 2, L^2 relative error is 0.0252.

reduced frequency IPDG solution and the high frequency ray-IPDG solution. The relative error of our approximate solution is 0.0252.

For the third numerical example, we will take the wave field as

$$u_{10} = \sqrt{\omega} H_0^{(1)} (\omega |\mathrm{x} - \mathrm{x}_0|)$$

where $\mathrm{x}_0 = (2, 2, 2)$, and the wave speed $c = 1$. The domain and mesh are the same as the previous example. For this example, we take two points in each element, and our neural network will learn one ray direction in each of these two points. Fig. 4.8.3 shows the reduced frequency IPDG solution and high frequency ray-IPDG solution. The relative error of our approximate solution is 0.020104.

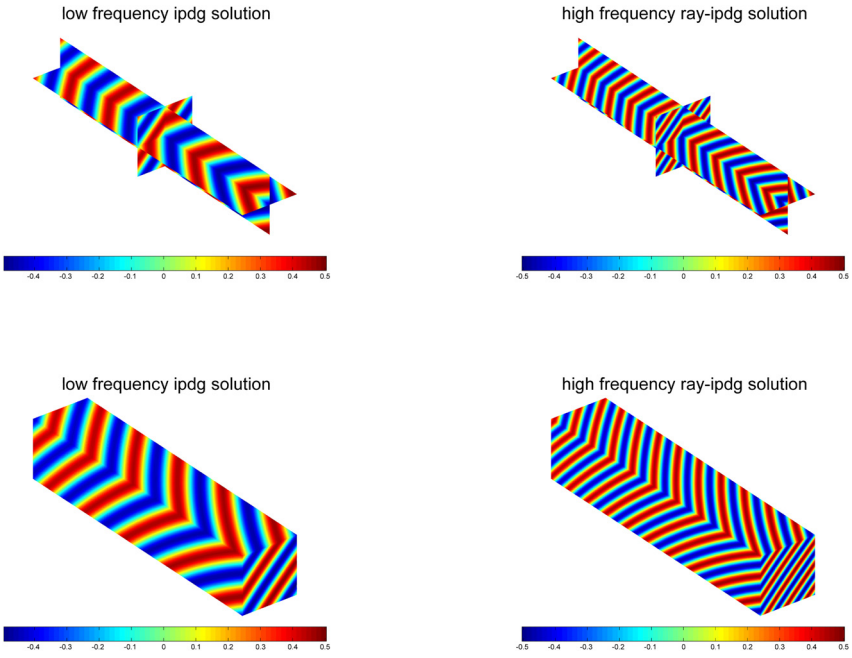

Fig. 4.8.3. 3D example 3, L^2 relative error is 0.020104.

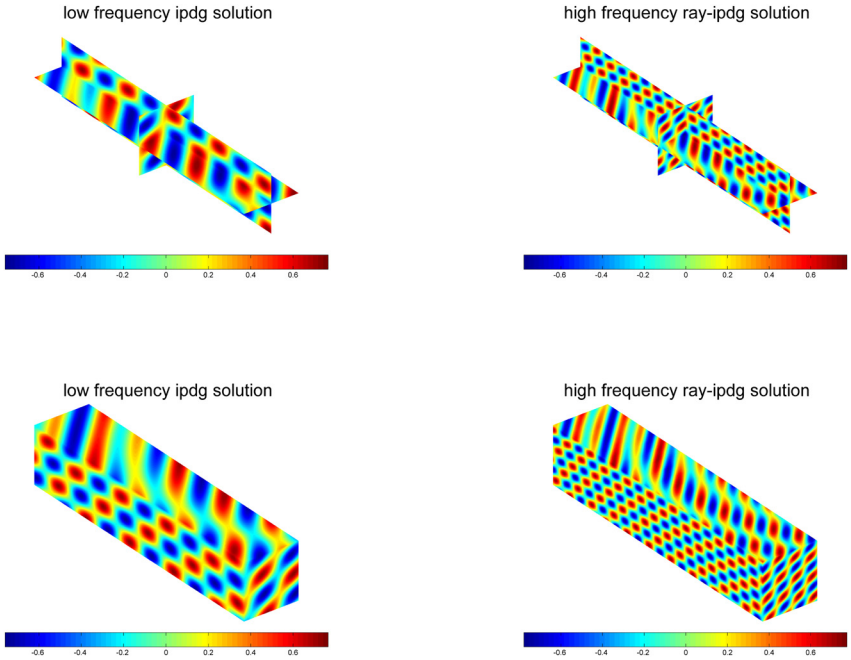

Fig. 4.8.4. 3D example 4, L^2 relative error is 0.024663.

For the fourth 3D numerical example, we will take the wave field as

$$u_{11} = \sqrt{\omega} H_0^{(1)}\left(\omega \left|\mathrm{x} - \mathrm{x}_{0,1}\right|\right) + 0.5\sqrt{\omega} H_0^{(1)}\left(\omega \left|\mathrm{x} - \mathrm{x}_{0,2}\right|\right)$$

where $\mathrm{x}_{0,1} = (2, 2, 2)$ and $\mathrm{x}_{0,2} = (-0.5, -0.5, 2])$. We also let the wave speed $c = 1$. The domain and mesh are the same as the previous example. For this example, we take two points in each element, and our neural network will learn two ray directions in each of these two points. Fig. 4.8.4 shows the reduced frequency IPDG solution and the high frequency ray-IPDG solution. The relative error of our approximate solution is 0.024663.

5. Conclusion

We have developed a deep learning approach to extract ray directions at discrete locations by analyzing highly oscillatory wave fields. A deep neural network is trained on a set of local plane-wave fields to predict ray directions at discrete locations. The resulting deep neural network is then applied to a reduced-frequency Helmholtz solution to extract the directions, which are further incorporated into a ray-based interior-penalty discontinuous Galerkin (IPDG) method to solve the Helmholtz equations at higher frequencies. In this way, we observe no apparent pollution effects in the resulting Helmholtz solutions in inhomogeneous media. Numerical results show that the proposed scheme is very efficient and yields highly accurate solutions.

CRediT authorship contribution statement

TAK SHING AU YEUNG performed 2D numerical examples.
KA CHUN CHEUNG provided computing resources and did some deep learning tests.
ERIC T. CHUNG designed the method and did analysis.
SHUBIN FU performed 3D numerical examples.
JIANLIANG QIAN designed the method.
All authors wrote and reviewed the paper.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

The research of Eric Chung is partially supported by the Hong Kong RGC General Research Fund (Project numbers 14304719 and 14302620) and CUHK Faculty of Science Direct Grant 2020-21, Jianliang Qian's research is partially supported by NSF (grant no. 1614566 and 2012046).

References

[1] G.S. Avila, J.B. Keller, The high-frequency asymptotic field of a point source in an inhomogeneous medium, Commun. Pure Appl. Math. 16 (1963) 363–381.
[2] V.M. Babich, The short wave asymptotic form of the solution for the problem of a point source in an inhomogeneous medium, USSR Comput. Math. Math. Phys. 5 (5) (1965) 247–251.
[3] Ivo Babuška, Frank Ihlenburg, Ellen T. Paik, Stefan A. Sauter, A generalized finite element method for solving the Helmholtz equation in two dimensions with minimal pollution, Comput. Methods Appl. Mech. Eng. 128 (3) (1995) 325–359.
[4] Ivo M. Babuška, Stefan A. Sauter, Is the pollution effect of the fem avoidable for the Helmholtz equation considering high wave numbers?, SIAM J. Numer. Anal. 34 (6) (1997) 2392–2423.
[5] Jean-David Benamou, Francis Collino, Simon Marmorat, Numerical microlocal analysis of 2-D noisy harmonic plane and circular waves, Asymptot. Anal. 83 (1–2) (2013) 157–187.
[6] Jean-David Benamou, Francis Collino, Olof Runborg, Numerical microlocal analysis of harmonic wavefields, J. Comput. Phys. 199 (2) (2004) 717–741.
[7] T. Betcke, J. Phillips, Approximation by dominant wave directions in plane wave methods, 2012.
[8] Rob Carriere, Randolph L. Moses, High resolution radar target modeling using a modified Prony estimator, IEEE Trans. Antennas Propag. 40 (1) (1992) 13–18.
[9] Siu Wun Cheung, Eric T. Chung, Yalchin Efendiev, Eduardo Gildin, Yating Wang, Jingyan Zhang, Deep global model reduction learning in porous media flow simulation, Comput. Geosci. 24 (1) (2020) 261–274.
[10] Eric T. Chung, Chi Yeung Lam, Jianliang Qian, A ray-based IPDG method for high-frequency time-domain acoustic wave propagation in inhomogeneous media, J. Comput. Phys. 348 (2017) 660–682.
[11] Björn Engquist, Olof Runborg, Computational high frequency wave propagation, Acta Numer. 12 (2003) 181–266.
[12] J. Fang, J. Qian, Leonardo Zepeda-Núñez, H. Zhao, Learning dominant wave directions for plane wave methods for high-frequency Helmholtz equations, Res. Math. Sci. 4 (2016) 1–35.
[13] Jun Fang, J. Qian, Leonardo Zepeda-Núñez, H. Zhao, A hybrid approach to solve the high-frequency Helmholtz equation with source singularity in smooth heterogeneous media, J. Comput. Phys. 371 (2018) 261–279.
[14] Shubin Fu, Eric T. Chung, Guanglian Li, An edge multiscale interior penalty discontinuous Galerkin method for heterogeneous Helmholtz problems with large varying wavenumber, J. Comput. Phys. (2021) 110387.
[15] Eldar Giladi, Joseph B. Keller, A hybrid numerical asymptotic method for scattering problems, J. Comput. Phys. 174 (1) (2001) 226–247.
[16] Claude J. Gittelson, Ralf Hiptmair, Ilaria Perugia, Plane wave discontinuous Galerkin methods: analysis of the h-version, ESAIM: Math. Model. Numer. Anal. 43 (02) (2009) 297–331.
[17] Marcus Grote, Anna Schneebeli, Dominik Schoetzau, Discontinuous Galerkin finite element method for the wave equation, SIAM J. Numer. Anal. 44 (01 2006) 2408–2431.
[18] Ralf Hiptmair, Andrea Moiola, Ilaria Perugia, Plane wave discontinuous Galerkin methods for the 2D Helmholtz equation: analysis of the p-version, SIAM J. Numer. Anal. 49 (1) (2011) 264–284.
[19] Ralf Hiptmair, Andrea Moiola, Ilaria Perugia, Plane wave discontinuous Galerkin methods: exponential convergence of the hp-version, Found. Comput. Math. (2015) 1–39.
[20] Ralf Hiptmair, Andrea Moiola, Ilaria Perugia, A survey of Trefftz methods for the Helmholtz equation, arXiv preprint, arXiv:1506.04521, 2015.

[21] Yingbo Hua, Tapan K. Sarkar, Matrix pencil method for estimating parameters of exponentially damped/undamped sinusoids in noise, IEEE Trans. Acoust. Speech Signal Process. 38 (5) (1990) 814–824.
[22] F. Ihlenburg, I. Babuska, Solution of Helmholtz problems by knowledge-based fem, Comput. Assist. Mech. Eng. Sci. 4 (1997) 397–415.
[23] L.M. Imbert-Gerard, B. Despres, A generalized plane-wave numerical method for smooth nonconstant coefficients, IMA J. Numer. Anal. 34 (3) (2014) 1072–1103.
[24] Harold Jeffreys, On certain approximate solutions of linear differential equations of the second order, Proc. Lond. Math. Soc. s2–23 (1) (1925) 428–436.
[25] Chi Yeung Lam, Jianliang Qian, Numerical microlocal analysis by fast Gaussian wave packet transforms and application to high-frequency Helmholtz problems, SIAM J. Sci. Comput. 41 (5) (2019) A2717–A2746.
[26] Chi Yeung Lam, Chi-Wang Shu, A phase-based interior penalty discontinuous Galerkin method for the Helmholtz equation with spatially varying wavenumber, Comput. Methods Appl. Mech. Eng. 318 (2017) 456–473.
[27] Yanina Landa, Nicolay M. Tanushev, Richard Tsai, Discovery of point sources in the Helmholtz equation posed in unknown domains with obstacles, Commun. Math. Sci. 9 (3) (2011) 903–928.
[28] P. Lax, Asymptotic solutions of oscillatory initial value problems, Duke Math. J. 24 (1957) 627–645.
[29] Erich W. Marchand, Electromagnetic Theory and Geometrical Optics (Morris Kline and Irvin W. Kay), 1966.
[30] Ngoc Cuong Nguyen, Jaime Peraire, Fernando Reitich, Bernardo Cockburn, A phase-based hybridizable discontinuous Galerkin method for the numerical solution of the Helmholtz equation, J. Comput. Phys. 290 (2015) 318–335.
[31] J. Qian, L. Ying, Fast Gaussian wavepacket transforms and Gaussian beams for the Schrödinger equation, J. Comput. Phys. 229 (2010) 7848–7873.
[32] J. Qian, L. Ying, Fast multiscale Gaussian wavepacket transforms and multiscale Gaussian beams for the wave equation, SIAM J. Multiscale Model. Simul. 8 (2010) 1803–1837.
[33] Jianliang Qian, William Symes, An adaptive finite-difference method for traveltimes and amplitudes, Geophysics 67 (2001) 08.
[34] Rayleigh Lord, On the propagation of waves through a stratified medium, with special reference to the question of reflection, Proc. R. Soc. Lond. Ser. A, Contain. Pap. Math. Phys. Character 86 (586) (1912) 207–226.
[35] C.E. Shannon, Communication in the presence of noise, Proc. IEEE 86 (2) (1998) 447–457.
[36] Min Wang, Siu Wun Cheung, Wing Tat Leung, Eric T. Chung, Yalchin Efendiev, Mary Wheeler, Reduced-order deep learning for flow dynamics. The interplay between deep learning and model reduction, J. Comput. Phys. 401 (2020) 108939.
[37] Yating Wang, Siu Wun Cheung, Eric T. Chung, Yalchin Efendiev, Min Wang, Deep multiscale model learning, J. Comput. Phys. 406 (2020) 109071.
[38] Tak Shing Au Yeung, Eric T. Chung, Simon See, A deep learning based nonlinear upscaling method for transport equations, arXiv preprint, arXiv: 2007.03432, 2020.
[39] M. Zworski, Semiclassical Analysis, Amer. Math. Soc., 2012.