INTERNATIONAL JOURNAL OF
NUMERICAL ANALYSIS AND MODELING
Volume 19, Number 4, Pages 587–601

A MESH-LESS, RAY-BASED DEEP NEURAL NETWORK METHOD FOR THE HELMHOLTZ EQUATION WITH HIGH FREQUENCY

ANDY L YANG† AND FENG GU‡

Abstract. This article introduces a mesh-less, ray-based deep neural network method to solve the Helmholtz equation with high frequency. This method does not use an adaptive mesh refinement method, nor does it design a numerical scheme using some specially designed basis function to calculate the numerical solution, but it has the advantages of easy implementation and no mesh. We have carried out various numerical examples to prove the accuracy and efficiency of the proposed numerical method.

Key words. Deep learning, plane wave, deep Neural Network, loss, high frequency, Helmholtz equation.

1. Introduction

In mathematics, the eigenvalue problem of the Laplace operator is called the Helmholtz equation, which has many applications in physics, including the wave equation and diffusive equation. It also has applications in other scientific fields, including electromagnetic radiation [4], acoustics [2], and plasma [16], etc. When the Helmholtz equation is applied to a wave, the eigenvalue value is called the wavenumber. The most obvious feature of the Helmholtz equation is that it is not positive definite, which makes the solution of the equation have strong oscillations when the wavenumber is large. In numerical calculations, the high oscillatory property of the exact solution under high-frequency conditions will cause the approximate solution obtained by the numerical calculation to only have very low accuracy, which is called the "pollution effect", cf. [1]. Therefore, from the perspective of algorithm design, the highly oscillating nature of the solution makes it very challenging to obtain an effective numerical method for this equation, which is also the purpose of this article.

We recall that there exist many available numerical algorithms for the Helmholtz equation with various boundary conditions including, for instance, the finite element method (FEM), Discontinuous Galerkin method, Spectral method, hybridizable discontinuous Galerkin method, weak Galerkin method, etc., see [1, 7, 13, 24, 29, 31] and reference therein. Due to the high oscillating nature of the solution, some commonly-used numerical methods based on low-order polynomials cannot resolve the solution well. Instead, they will produce the so-called pollution effect, that is, for a fixed number of grid points for each wavelength, the numerical error increases with the increase of wavenumbers, see [1]. Therefore, while using the numerical method based on the low-order polynomials, unless a certain number of grid points are used for discretization for each wavelength, the calculation accuracy is relatively poor for high-frequency waves. Therefore, it is natural to use higher-order polynomials or oscillatory non-polynomial basis to replace the low-order polynomials. It has been shown that higher-order polynomials can effectively reduce the pollution effect, see

Received by the editors January 21, 2022 and, in revised form, April 23, 2022.
2000 *Mathematics Subject Classification.* 65L15, 34L16.

[24, 33], however, the computational cost is high due to the increase of degrees of freedom.

In this article, we try to break away from the traditional numerical methods based on variational framework using low/high-order polynomial basis, and use a novel mesh-less deep neural network (DNN) method to solve the high-frequency Helmholtz equation. We recall that the DNN method has attracted many attentions in recent years to many classic problems involved in scientific computing, especially the numerical solution of ordinary or partial differential equations, cf. [23, 5, 3, 8, 10, 12, 11, 18, 19, 21, 28, 30, 32] and references therein. Whether the algorithms of DNN can be applied to the field of scientific computing to obtain effective and accurate numerical algorithms has been confirmed by some recent research works. For example, in [11, 27], the authors give the quantitative relationship between neural network algorithms and low/high-order finite element methods; in [15], the authors discuss the approximate properties of the function classes given by the feedforward neural network using a single hidden layer; and in [17], the authors give the framework of deriving error estimates for a class of neural network algorithms according to the number of neurons. Based on these works, some novel methods on applying the DNN to solving ordinary/partial differential equations are developed, including the so-called PINN (the physics-informed neural network) method given in [21, 18], DGM (Deep-Galerkin method) give in [30] and DRM (Deep-Ritz method) given in [8]. Therefore, inspired by the DLSM (Deep-Least Squares Method) given in [5], in this article, we introduce a mesh-less, ray-based DNN method to solve the Helmholtz equation, and to investigate whether the method can be applied to high-frequency situations well. The mesh-free nature of this method allows us to easily get rid of designing adaptive grids or special spatial discretization methods, so it is very easy to implement. For the large wavenumber case, the obtained numerical results show that the designed DNN method can efficiently and accurately approximate the exact solution of the Helmholtz equations.

The rest of this paper is organized as follows. In Section 2, we review the basic idea of DNNs. In Section 3, the derivation of the methodology for the Helmholtz equation is developed. In Section 4, we present some numerical results to demonstrate the performance of our method. Some concluding remarks are given in Section 5.

2. DNN method

In this section, we briefly discuss the definition and approximation properties of the DNNs.

A DNN is a sequential alternative composition of linear functions and nonlinear activation functions. A n-layer neutral network $\mathcal{N}^n$ can be defined as

- Input layer: $\mathcal{N}^0 = \boldsymbol{x}$,
- Hidden layers: $\mathcal{N}^l = \sigma_l(\mathbf{W}^l\mathcal{N}^{l-1} + \mathbf{b}^l)$, $l = 1, 2, \cdots, n-1$,
- Output layer: $\mathcal{N}^n = \mathbf{W}^n\mathcal{N}^{n-1} + \mathbf{b}^n$,

where σ denotes the activation function, $\mathbf{W}^l$ denote the weights and $\mathbf{b}^l$ denote the biases. The most common used types of activation functions include the sigmoid function $\sigma(t) = (1 + e^{-t})^{-1}$ and the rectified linear unit (ReLU) $\sigma(t) = \max(0, t)$. For simplicity, we denote all the parameters in DNN by a parameter vector $\mathbf{\Theta}$, i.e.,

$$\mathbf{\Theta} = \{\mathbf{W^1}, \cdots, \mathbf{W^n}, \mathbf{b^1}, \mathbf{b^2}, \cdots, \mathbf{b^n}\}.$$

In Fig. 1, we sketch a simple fully connected DNN example with 3 hidden layers and 8 neurons in each hidden layer. The number m_l denotes the number of neurons in the l-th layer.

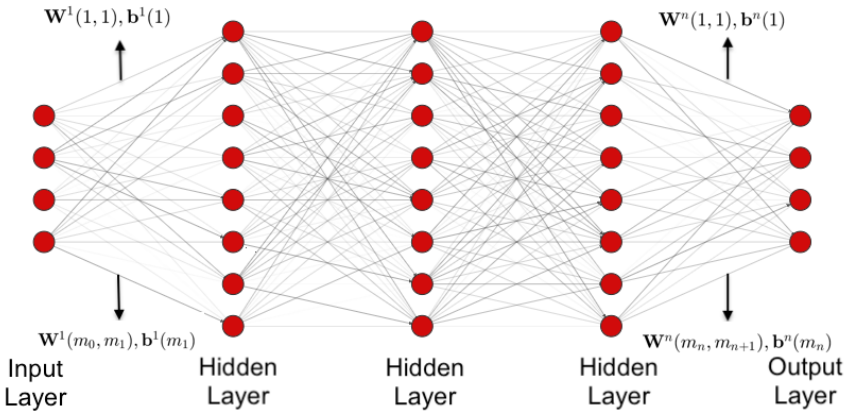


FIGURE 1. An example of a simple fully connected DNN.

We define a function $u_d(\boldsymbol{x}, \boldsymbol{\Theta})$ to be the output of a DNN with a single hidden layer ($n = 1$) of m neurons, that is

$$u_d(\boldsymbol{x}, \boldsymbol{\Theta}) = \sum_{\mathbf{j=1}}^{\mathbf{m}} \mathbf{c_j} \sigma(\mathbf{W_j} \cdot \boldsymbol{x} + \mathbf{b_j}), \boldsymbol{x} \in \boldsymbol{\Omega}, \tag{1}$$

where Ω is the computational domain. The set of all functions formulated as $u_d(\boldsymbol{x}, \boldsymbol{\Theta})$ is denoted as V_m^σ.

The following lemma ensures that any function can be approximated by V_m^σ uniformly to arbitrary precision.

Lemma 2.1 (Universal Approximation, cf. [14]). *Assuming that $1 \le p < \infty, 0 \le s < \infty$ and $\Omega \subset R^d$ is compact, if $\sigma \in C^s(\Omega)$ is non-constant and bounded, then V_L^σ is dense in the Sobolev space*

$$W^{s,p}(\Omega) = \{v \in L^p(\Omega) : D^\alpha v \in L^p(\Omega), \forall |\alpha| \le s\}.$$

Namely, for any function $f \in W^{s,p}(\Omega)$ and $\tau > 0$, there exists $m \in N$ (m depends on τ and f) and $\tilde{f} \in V_m^\sigma$, such that

$$\|f - \tilde{f}\|_{W^{s,p}} < \tau. \tag{2}$$

3. Methodology using DNN for Helmholtz equation

3.1. DNN for Boundary value problem. We consider a boundary value problem that reads as

$$\begin{cases} \mathcal{L}u = f \text{ in } \Omega, \\ Bu = g \text{ on } \partial\Omega, \end{cases} \tag{1}$$

where L and B are two linear operators, $\Omega \in R^d (d = 2, 3)$ denotes the computational domain. When adopting the DNN method, we train a neural network to minimize the following least-square functional

$$L_{least}(u) = \int_\Omega |f - \mathcal{L}u|^2 d\mathbf{x} + \int_{\partial\Omega} |g - Bu|^2 ds. \tag{2}$$

In practice, the integral of the above loss function is usually computed by a numerical way. A commonly-used approach in machine learning method is to use the

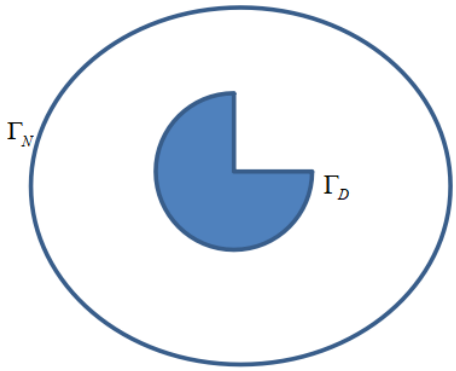


FIGURE 2. The figure illustration of the model problem.

Monte Carlo integration that is given below. For any v, the numerical integral of it reads as

$$\int_\Omega v(\boldsymbol{x})d\boldsymbol{x} = \frac{|\Omega|}{N}\sum_{i=1}^{N} v(\boldsymbol{x}_i), \tag{3}$$

where $|\Omega|$ is the volume of the domain Ω and $\{\boldsymbol{x}_i\}_{i=1}^N$ are the random points in Ω.

When the operators in (1) take some specific forms, we get the high frequency Helmholtz equation, that reads as follows

$$\mathcal{L}u \equiv \Delta u + k^2 u = f(\boldsymbol{x}) \text{ in } \Omega, \tag{4a}$$

$$u = u_D \text{ on } \Gamma_D, \tag{4b}$$

$$\partial_{\mathbf{n}} u + \mathrm{i}ku = g \text{ on } \Gamma_N, \tag{4c}$$

where $\Gamma_D \cup \Gamma_N = \partial\Omega$, k denotes the wavenumber and $\mathbf{n}$ denotes the outward unit normal vector field on Γ_N (cf. Fig. 2). In the next section, we give the ray-based mesh-less DNN method to solve the system (4a)-(4c).

3.2. Asymptotic numerical methods. First, we consider one of the most well-known asymptotic approaches for solving the Helmholtz equation, the so-called LunebergKline expansion [4]. That is, the solution of the Helmholtz problem is expanded to be a series of the following form:

$$u(\boldsymbol{x}) \approx e^{\mathrm{i}k\phi(\boldsymbol{x})}\sum_{n=0}^{\infty}\frac{\tilde{A}_n(\boldsymbol{x})}{kn^n},$$

By taking $k \to \infty$ and considering only the first term one has

$$u(\boldsymbol{x}) = A(\boldsymbol{x})e^{\mathrm{i}k\phi(\boldsymbol{x})} + O(\frac{1}{k}), \tag{5}$$

where $\phi(x)$ denotes the phase function, and A_n means the amplitude function, both of which are unknown.

In the standard geometrical optics approach for solving the Helmholtz equation numerically, cf. [4], the source function f is ignored, i.e., $f = 0$ in (4a). Then, similar as [4], by substituting the expression (5) into (4a), and taking $k \to \infty$, we obtain the following eikonal equation for the phase function, that reads as

$$|\nabla\phi| = 1, \tag{6}$$

and a transport equation for the amplitude function, that reads as

$$2\nabla\phi\cdot\nabla A + A\Delta\phi = 0. \tag{7}$$

However, it is well-known that there exists a major bottleneck that the above asymptotic expansion can only capture single phase wave field. In general, the phase function, ϕ, and the amplitude function, A, are multi-valued functions corresponding to multiple arrivals of wave fronts. Hence, many asymptotic methods consider a more general expansion (cf. [26]) that reads as :

$$u(\boldsymbol{x}) \approx \sum_{j=0}^{p} e^{\mathrm{i}k\phi_j(\boldsymbol{x})} A_j(\boldsymbol{x}, k), \tag{8}$$

where the phase functions ϕ_n are independent from the wavenumber k, while the amplitude functions A_n are dependent on the wavenumber k. The asymptotic expansion (8) is basic for advanced asymptotic methods such as the geometrical theory of diffraction (GTD) [4] and uniform theory of diffraction (UTD) [22]. Similar to the phase function $\phi(\boldsymbol{x})$ used in the Luneberg-Kline expansion (5) that is the solution of eikonal equation (6), the phase functions ϕ_n in (8) are also obtained by solving another similar eikonal equation (6), see [26]. The combination of direct and asymptotic numerical method had been used in the so-called hybrid method given in [26] for solving the system (4a)-(4c). The so-called phase-based IPDG method with spatially varying wavenumber introduces the phase information into the basis of the expansion formula (8) to obtain a more efficient numerical approach, see [20].

3.3. Ray-based mesh-less DNN method. In this paper, by combining the DNN method for the generic boundary value problem with the asymptotic numerical method together, we develop the so-called ray-based mesh-less DNN method to solve the high frequency Helmholtz equation. The key is to combine the DNN method with the plane wave approximation function.

We define a space $P_{k,p}$ that is spanned by the plane wave functions with p different unit direction $\mathbf{d}_l \in R^{N-1}$ with $|\mathbf{d}_l| = 1$ and $l = 1, 2, \cdots, p$, such that

$$P_{k,p}(R^N) = \{u \in C^\infty(R^N) : u(\boldsymbol{x}) = \sum_{j=1}^{p} \alpha_l e^{\mathrm{i}k\boldsymbol{x}\cdot\mathbf{d}_l}\}. \tag{9}$$

Examples of such spaces are the so-called plane-wave DG (discontinuous Galerkin) method, see [13], the ultra weak-variational formulation method, see [6].

We define a coupled DNN as

$$u_d(\boldsymbol{x}, \boldsymbol{\Theta}) = \sum_{j=1}^{p} e^{\mathrm{i}k\boldsymbol{x}\cdot\mathbf{d}_j} \mathcal{N}_j(\boldsymbol{x}, \boldsymbol{\Theta}), \tag{10}$$

to approximate the exact solution of the model problem (4a)–(4c), where $\mathcal{N}_j(\boldsymbol{x}, \boldsymbol{\Theta}) = \mathcal{N}_j^{\mathrm{R}}(\boldsymbol{x}, \boldsymbol{\Theta}) + \mathrm{i}\mathcal{N}_j^{\mathrm{I}}(\boldsymbol{x}, \boldsymbol{\Theta})$ with $\mathcal{N}_j^{\mathrm{R}}(\boldsymbol{x}, \boldsymbol{\Theta})$ and $\mathcal{N}_j^{\mathrm{I}}(\boldsymbol{x}, \boldsymbol{\Theta})$ represent two different DNNs which are independent with each other.

The DNN in (10) is used to compute the solution of Helmholtz equations by minimizing the least squares of the Helmholtz equation's residual, which are given by the loss function as follows:

$$loss(\boldsymbol{\Theta}) = loss_{eq}(\boldsymbol{\Theta}) + \rho_1 loss_{bc}^{D}(\boldsymbol{\Theta}) + \rho_2 loss_{bc}^{N}(\boldsymbol{\Theta}), \tag{11}$$

where ρ_1, ρ_2 denote two penalty parameters and

$$loss_{eq}(\boldsymbol{\Theta}) = \sum_{m=1}^{N} |f(\boldsymbol{x}_m) - \mathcal{L}u_d(\boldsymbol{x}_m, \boldsymbol{\Theta})|^2, \tag{12}$$

$$loss_{bc}^{D}(\boldsymbol{\Theta}) = \sum_{\mathbf{m=1}}^{\mathbf{M_1}} |\mathbf{u_D}(\boldsymbol{x}_\mathbf{m}) - \mathbf{u_d}(\boldsymbol{x}_\mathbf{m}, \boldsymbol{\Theta})|^\mathbf{2}, \tag{13}$$

$$loss_{bc}^{N}(\boldsymbol{\Theta}) = \sum_{m=1}^{M_2} |g(\boldsymbol{x}_m) - \partial_{\mathbf{n}} u_d(\boldsymbol{x}_m, \boldsymbol{\Theta}) - \mathrm{i}k u_d(\boldsymbol{x}_m, \boldsymbol{\Theta})|^2, \tag{14}$$

with N, M_1, M_2 denote the number of choosing points in $\Omega, \Gamma_D, \Gamma_N$.

Remark 3.1. *The approximation property using the coupled DNN formula given in* (10) *can be ensured by the following conclusion. Namely, assuming that* $u \in H^{K+1}(D)$ *be the solution of homogeneous Helmholtz equation, where* D *is a bounded open set and* ∂D *is Lipschitz, then there always exists* $(\alpha_1, \alpha_2, \cdots, \alpha_p) \in \mathbb{C}^p$ *such that, for every* $0 \le j \le K+1$*, the following inequality holds:*

$$\|u - \sum_{j=1}^{p} \alpha_j e^{\mathrm{i}k\boldsymbol{x}\cdot\mathbf{d}_j}\|_{l,D} \le C\|u\|_{K+1,D}, \tag{15}$$

where C *is a constant that is independent with* u*, see* [25].

Remark 3.2. *One of the main advantages of the ray-based DNN method* (10)-(6) *is its mesh-free feature, that is, we do not need to create a mesh that meets certain characteristics like the finite element method or the Spectral method. In practical calculation, we can randomly select the grid points in the computed region, which brings great convenience to calculations. So far, the only existed work of using DNN method to solve the high frequency Helmholtz problem is the so-called phase-DNN method which is developed in* [5]*, in which the high-frequency part is converted into the low-frequency part through phase change technology, so that the convergence speed of the network can be accelerated. Moreover, the main idea of our proposed ray-based DNN method lies on the plane wave approximation property* (15)*, which is significantly different from the so-called phase-DNN method introduced in* [5]*, where the idea of phase shifts in the frequency domain is used.*

Remark 3.3. *In some special case, for example, the direction of wave propagation* $\boldsymbol{x}_0$ *is known, thus one can introduce the so-called phase-shifted (see also in* [20]*) DNN method, namely, we set*

$$u_d(\boldsymbol{x}) = e^{\mathrm{i}k|\boldsymbol{x}-\boldsymbol{x}_0|}\mathcal{N}(\boldsymbol{x}). \tag{16}$$

Numerical experiments in section 4 show that the shifted method can give numerical results with better convergence. But note that in most cases, $\boldsymbol{x}_0$ *is unknown, so this method is not applicable to most problems.*

4. Numerical example

In this section, we implement some numerical examples to verify the proposed mesh-less DNN method (10)-(6) to prove its effectiveness in solving the Helmholtz equation with high wavenumbers.

Example 1 (uni-directional wave). In the first numerical test, we set the computational domain $\Omega = [-1, 1]^2$, $\Gamma_N = \emptyset$ and the wavenumber $k = 100, 500$.

We set the exact solution of the Helmholtz problem as a uni-directional wave that reads as

$$u(\boldsymbol{x}) = H_0^{(1)}(k|\boldsymbol{x} - \boldsymbol{x}_0|), \tag{1}$$

where $H_0^{(1)}$ denotes the zeroth-order Hankel function of the first kind, and $\boldsymbol{x}_0 = (2, 2)$ is the direction of wave propagation.

In our computation, we choose 10 unit direction vectors as

$$\mathbf{d}_j = (\cos\frac{2j\pi}{p}, \sin\frac{2j\pi}{p}),\ p = 10. \tag{2}$$

To show the accuracy and efficiency of our proposed ray-based DNN method, we set each $\mathcal{N}_j^{\mathrm{R}}(\boldsymbol{x}), \mathcal{N}_j^{\mathrm{I}}(\boldsymbol{x})$ with $j = 1, \cdots, 10$ to be 4 hidden layers, each hidden layer has 40 neurons for $k = 10$. We choose 8 hidden layers for $k = 100$, and each hidden layer includes 80 neurons. In the learning process, i.e., the implementation of the Stochastic Gradient Descent (SGD) method, we choose the points in each epoch to be 5000.

In Table 1, we present the error in the maximum norm of the real part and the imaginary part respectively for the two cases of $k = 10$ and $k = 100$. We adopt uniform grid pints for spatial discretization, where $100 * 100$ uniform grid points are used for $k = 10$, and $500 * 500$ uniform points are used for $k = 100$. We see that as the number of epochs increases, the errors of real and imaginary parts both decrease, which shows that the numerical solution of our proposed DNN method converges well to the exact solution. In Fig. 3, we plot the profiles of the real and imaginary parts of the exact solution, the computed solution, as well as their difference. It can be seen that the obtained contours of the numerical solutions are the same as the exact solution.

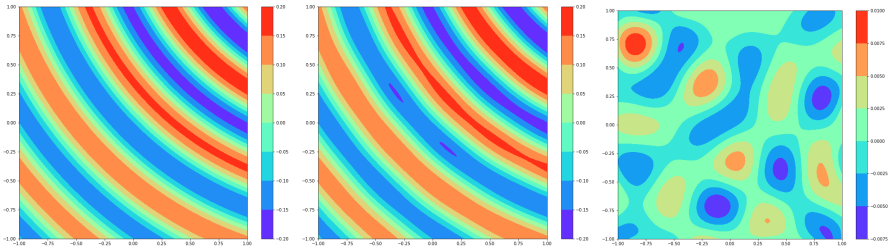
We further test the phase-shifted DNN method (16) since $\boldsymbol{x}_0$ in this particular example is known for the high frequency case with $k = 100$. We take each $\mathcal{N}^{\mathrm{R}}(\boldsymbol{x}), \mathcal{N}^{\mathrm{I}}(\boldsymbol{x})$ to be a 4 hidden layers with 40 neurons in each hidden layer. The computational results are shown in Table 2 where the computed accuracy is better than Table 1 for $k = 100$. However, as mentioned in Remark 3.3, in most cases, the direction of wave propagation $\boldsymbol{x}_0$ is unknown, therefore, the application of the phase-shifted DNN method is very limited.

Example 2 (two-way wave propagation). In this numerical test, we set the exact solution of the system as a two-way wave that reads as

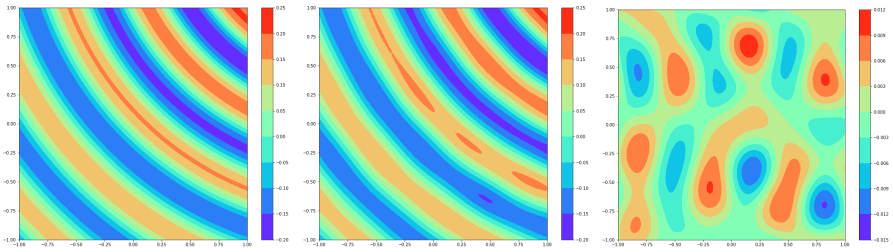
$$u(\boldsymbol{x}) = H_0^{(1)}(k|\boldsymbol{x} - \boldsymbol{x}_1|) + H_0^{(1)}(k|\boldsymbol{x} - \boldsymbol{x}_2|), \tag{3}$$

TABLE 1. Example 1: maximum error with various epochs for $k = 10$ and $k = 100$.

k	No. of points	epoch	Re-Error	ImError
10	100*100	100	3.5e-1	5.0e-1
		5000	1.5e-2	1.7e-2
		10000	9.3e-3	1.2e-2
100	500*500	1000	4.7e-1	5.0e-1
		10000	8.6e-2	8.6e-2
		50000	4.5e-2	4.5e-2

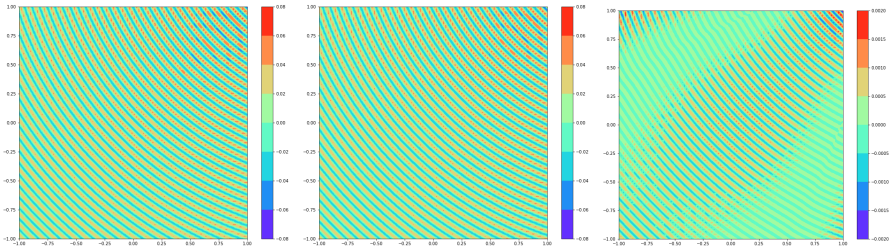


(a) Real part.

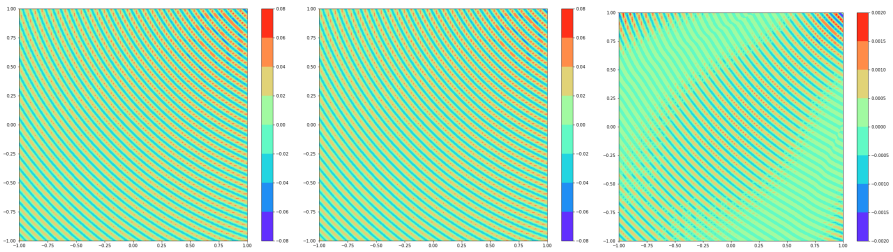


(b) Imaginary part.

FIGURE 3. Example 1 with $k = 10$: (a) the real part, and (b) imaginary part. In each subfigure, the profiles of the exact solution, the numerical solution computed using the DNN method, and the difference between these two are plotted from left to right.



(a) Real part.



(b) Imaginary part.

FIGURE 4. Example 1 with $k = 100$: (a) the real part, and (b) imaginary part. In each subfigure, the profiles of the exact solution, the numerical solution computed using the DNN method, and the difference between these two are plotted from left to right.

TABLE 2. Example 1: maximum error computed by the phase-shifted DNN.

k	No. of points	epoch	Re-Error	ImError
100	500*500	100	1.3e-2	6.3e-3
		500	6.2e-3	6.2e-3
		1000	1.9e-3	1.9e-3

TABLE 3. Example 2: maximum error with various epochs for $k = 10$ and $k = 100$.

k	No. of points	epoch	Re-Error	ImError
10	100*100	100	3.7e-1	3.6e-1
		10000	8.5e-2	7.9e-2
		30000	9.4e-3	9.7e-3
100	500*500	10000	2.7e-1	2.0e-1
		50000	7.1e-2	7.2e-2
		200000	8.6e-3	8.4e-3

where $\boldsymbol{x}_1 = (0.3, -0.1), \boldsymbol{x}_2 = (0.7, -0.1)$ are the two directions of wave propagation. The computational domain is set as $\Omega = [0,1]^2$, $\Gamma_N = \emptyset$, and the wavenumber k are chosen to be 10 and 100, respectively.

In our computation, we choose 30 unit direction vectors as

$$\mathbf{d}_j = (\cos\frac{2j\pi}{p}, \sin\frac{2j\pi}{p}), \quad p = 30. \tag{4}$$

We set 6 hidden layers with 60 neurons in each layer for $k = 10$, and 10 hidden layers with 100 neurons in each layer for $k = 100$. In the learning process, we choose the points in each epoch to be 5000. The maximum errors are shown in Table 3 for each case and various epochs. As the number of epochs increases, the errors of real and imaginary parts both decrease. In Fig. 5, for $k = 100$, we plot the profiles of the real and imaginary parts of the exact solution, the computed solution, as well as their difference. These results display a good correspondence between the numerical and exact solutions.

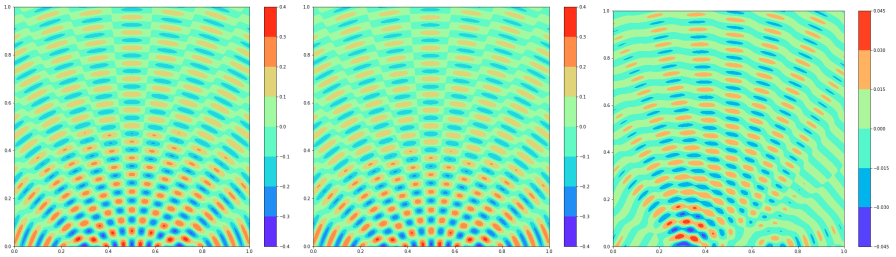
Next, we apply the phase-DNN method developed in [5] to solve the above problem. To do so, we define a function

$$u_d(\boldsymbol{x}, \boldsymbol{\Theta}) = \sum_{\mathbf{j=1}}^{\mathbf{N_c}} \mathbf{e}^{\mathrm{i}\omega_\mathrm{j}|\boldsymbol{x}|} \mathcal{N}_\mathbf{j}(\boldsymbol{x}, \boldsymbol{\Theta}), \tag{5}$$

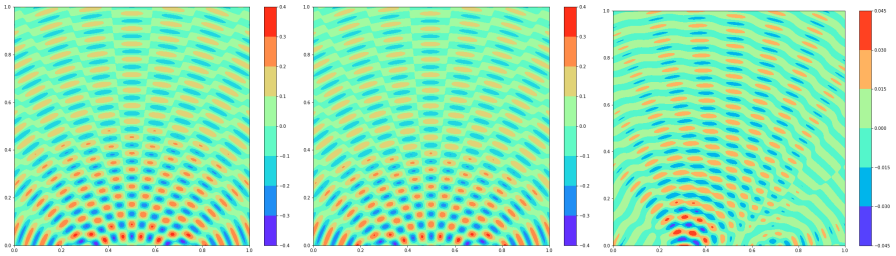
which is used to approximate the exact solution of the model problem (4a)–(4c), where $\mathcal{N}_j(\boldsymbol{x}) = \mathcal{N}_j^{\mathrm{R}}(\boldsymbol{x}, \boldsymbol{\Theta}) + \mathrm{i}\mathcal{N}_\mathbf{j}^{\mathrm{I}}(\boldsymbol{x}, \boldsymbol{\Theta})$ with $\mathcal{N}_j^{\mathrm{R}}(\boldsymbol{x}, \boldsymbol{\Theta})$ and $\mathcal{N}_j^{\mathrm{I}}(\boldsymbol{x}, \boldsymbol{\Theta})$ represent two different DNNs which are independent with each other.

The DNN in (5) is used to compute the solution of Helmholtz equations by minimizing the least squares of the Helmholtz equation's residual, which are given by the loss function as follows:

$$loss(\boldsymbol{\Theta}) = \mathbf{loss_{eq}}(\boldsymbol{\Theta}) + \rho_\mathbf{1}\mathbf{loss_{bc}^D}(\boldsymbol{\Theta}) + \rho_\mathbf{2}\mathbf{loss_{bc}^N}(\boldsymbol{\Theta}), \tag{6}$$



(a) Real part.



(b) Imaginary part.

FIGURE 5. Example 2 with $k = 100$: (a) the real part, and (b) imaginary part. In each subfigure, the profiles of the exact solution, the numerical solution computed using the DNN method, and the difference between these two are plotted from left to right.

where ρ_1, ρ_2 denote two penalty parameters and

$$loss_{eq}(\mathbf{\Theta}) = \sum_{\mathbf{m=1}}^{\mathbf{N}} |\mathbf{f}(\boldsymbol{x}_\mathbf{m}) - \mathcal{L}\mathbf{u_d}(\boldsymbol{x}_\mathbf{m}, \mathbf{\Theta})|^\mathbf{2}, \tag{7}$$

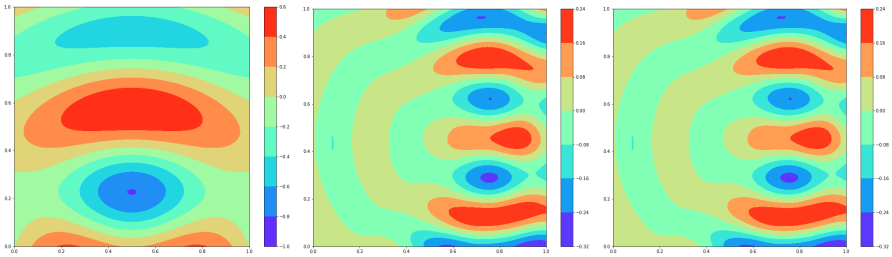
$$loss_{bc}^{D}(\mathbf{\Theta}) = \sum_{\mathbf{m=1}}^{\mathbf{M_1}} |\mathbf{u_D}(\boldsymbol{x}_\mathbf{m},) - \mathbf{u_d}(\boldsymbol{x}_\mathbf{m}, \mathbf{\Theta})|^\mathbf{2}, \tag{8}$$

$$loss_{bc}^{N}(\mathbf{\Theta}) = \sum_{\mathbf{m=1}}^{\mathbf{M_2}} |\mathbf{g}(\boldsymbol{x}_\mathbf{m}) - \partial_\mathbf{n}\mathbf{u_d}(\boldsymbol{x}_\mathbf{m}, \mathbf{\Theta}) - \mathrm{i}\mathbf{k}\mathbf{u_d}(\boldsymbol{x}_\mathbf{m}, \mathbf{\Theta})|^\mathbf{2}, \tag{9}$$

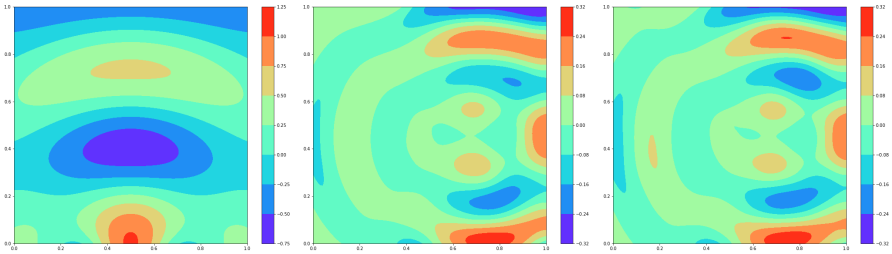
with N, M_1, M_2 denote the number of choosing points in $\Omega, \Gamma_D, \Gamma_N$.

For the low wavenumber of $k = 10$, we set $\omega_j \in \{2, 4, 6, 8, 10, 12\}$, i.e., $N_c = 6$, and take 100 hidden layers with 1000 neurons in each layer. In the learning process, we choose the points in each epoch to be 10000. As the number of epochs increases, the errors of real and imaginary parts are very large, shown in Fig. 7.

Moreover, if we use the method in [5] to compute the case when $k = 100$, we find it does not converge. Compared with the method in [5], our method can compute the high wavenumber case with $k = 100$ show in Fig. 5 or ever lager wavenumber. Therefore, it can be concluded that our ray-based DNN method is much more efficient and accurate in solving Helmholtz equation with unknown wave directions and high wavenumbers.



(a) Real part.



(b) Imaginary part.

FIGURE 6. Example 2 with $k = 10$: (a) the real part, and (b) imaginary part. In each subfigure, the profiles of the exact solution, the two numerical solution computed using the phase-DNN method(50000 epochs and 100000 epochs).

Example 3 (quadratic wave). In this example, we set the exact solution of the system to be the following quadratic form:

$$u = e^{y} e^{\mathrm{i}k\boldsymbol{x}^2}. \tag{10}$$

The computational domain is choosen as $\Omega = [0,1]^2$, $\Gamma_N = \emptyset$ and the wavenumber k are choosen to be 100 and 500, respectively.

We choose the DNN as the following from

$$u_d(\boldsymbol{x}) = e^{\mathrm{i}k\boldsymbol{x}^2} \mathcal{N}_1(\boldsymbol{x}). \tag{11}$$

To show the accuracy and efficiency of the proposed DNN, we set 4 hidden layers with 40 neurons in each layer for $k = 10$, and 8 hidden layers with 100 neurons in each layer for $k = 100$. In the learning process, we choose the points in each epoch to be 5000. The maximum errors are shown in Table 4 for each k and various epochs. In Fig. 7, for $k = 100$, we plot the profiles of the real and imaginary parts of the exact solution, the computed solution, as well as their difference. The computed results show good convergence to the exact solution.

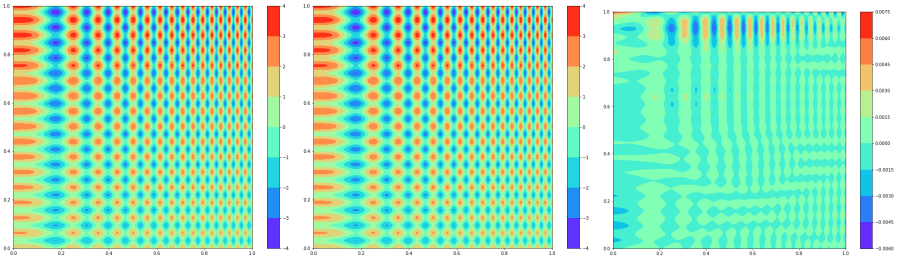
Example 4 (wave scattering). In the final example, we consider the Helmholtz scattering problem that reads as

$$\begin{cases} \Delta u + k^2 u = 0, \text{ in } R^2 \setminus D, \\ u = u_0 \text{ on } \partial D, \\ \sqrt{r}(\dfrac{\partial u}{\partial r} - \mathrm{i}ku) \to 0 \text{ as } r = |\boldsymbol{x}| \to \infty, \end{cases} \tag{12}$$

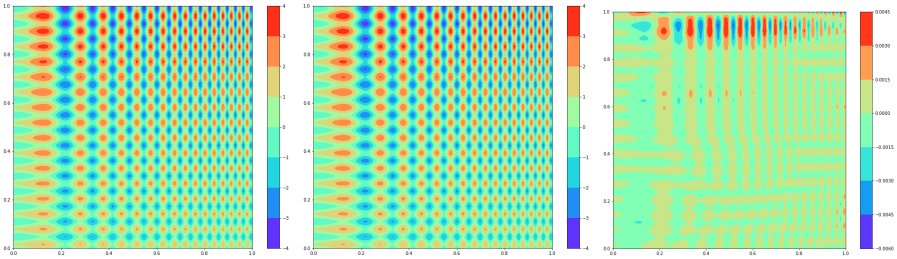
where the domain D is a unit circle. We assume the exact solution as $u = H_0^{(1)}(k\boldsymbol{x})$. To truncate the domain into a bounded computational domain, we follow the crucial

TABLE 4. Example 3: maximum error with various epochs for $k = 10$ and $k = 100$.

k	No. of points	epoch	Re-Error	ImError
10	100*100	100	2.3e-1	2.4e-1
		500	1.1e-2	1.3e-2
		1000	5.6e-3	6.3e-3
100	500*500	100	5.7e-1	5.7e-1
		10000	6.4e-2	6.3e-2
		20000	1.2e-2	1.2e-2



(a) Real part.

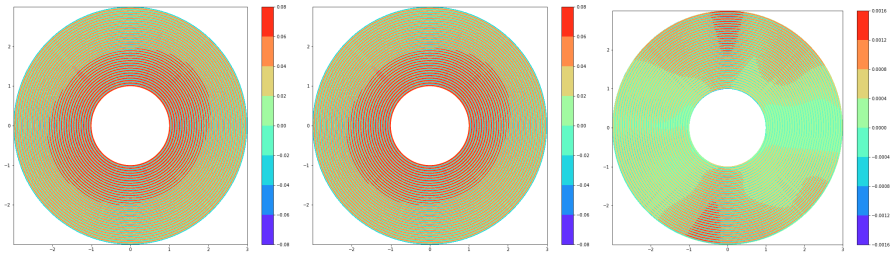


(b) Imaginary part.

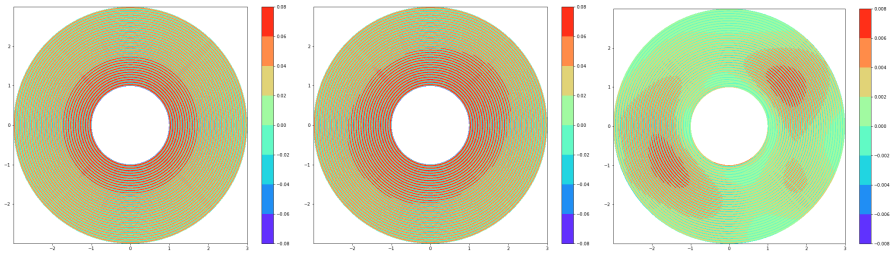
FIGURE 7. Example 3 with $k = 100$: (a) the real part, and (b) imaginary part. In each subfigure, the profiles of the exact solution, the numerical solution computed using the DNN method, and the difference between these two are plotted from left to right.

TABLE 5. Example 4: maximum error with various epochs for $k = 100$ and $k = 500$.

k	No. of points	epoch	Re-Error	Im-Error
100	500*500	100	1.1e-2	1.2e-2
		1000	6.9e-3	6.9e-3
		3000	2.8e-3	2.9e-3
		4000	2.4e-3	2.4e-3
500	2500*2500	100	2.6e-2	2.5e-2
		1000	2.8e-3	2.7e-3
		2000	1.8e-3	1.7e-3
		4000	9.7e-4	9.8e-4



(a) Real part.



(b) Imaginary part.

FIGURE 8. Example 4 with $k = 100$: (a) the real part, and (b) imaginary part. In each subfigure, the profiles of the exact solution, the numerical solution computed using the DNN method, and the difference between these two are plotted from left to right.

step taken in PWDG methods (see [9]). Namely, the first order absorbing boundary condition is adopted:

$$\partial_{\mathbf{n}} u + \mathrm{i}ku = g \text{ on } \Gamma_R. \tag{13}$$

We set $\mathcal{N}^{\mathrm{R}}(\boldsymbol{x}), \mathcal{N}^{\mathrm{I}}(\boldsymbol{x})$ to be 2 hidden layers with 20 neurons in each hidden layer for $k = 100$, and 4 hidden layers with 60 neurons in each hidden layer for $k = 500$. In the learning process, we choose the points in each epoch to be 1000. The maximum errors are shown in Table 5 for each case and various epochs. The profiles of the exact solution, the computed solution using the proposed DNN method, and their errors using 3000 epochs are shown in Fig. 8. These results show that even when k is large, the DNN method can converge to the exact solution well.

5. Concluding remarks

In this paper, in order to solve the high-frequency Helmholtz problem, we use the approximate characteristics of plane waves to propose a ray-based mesh-less DNN method. As shown by various numerical examples, the proposed method improves the ability of DNN as a viable mesh-less tool for solving high-frequency Helmholtz problems. Numerical results prove that the proposed method can approximate the exact solution of the equation well. In our future research, we will explore the generalization ability of DNN method to solve high-frequency Maxwell equations and elastic equations.

Acknowledgements

F. Gu's work was partially supported by the CUNY Interdisciplinary Research Grant.

References

[1] I. Babuska and S. Sauter. Is the pollution effect of the fem avoidable for the helmholtz equation considering high wave numbers? SIAM J. numer. anal., 34(6):2392–2423, 1997.
[2] A. Bayliss, C. Goldstein, and E. Turkel. The numerical solution of the helmholtz equation for wave propagation problems in underwater acoustics. Computers & Mathematics with Applications, 11(7-8):655–665, 1985.
[3] C. Beck, M. Hutzenthaler, A. Jentzen, and B. Kuckuck. An overview on deep learning-based approximation methods for partial differential equations. arXiv preprint arXiv:2012.12348, 2020.
[4] D. Bouche, F. Molinet, and R. Mittra. Asymptotic methods in electromagnetics. Springer Science & Business Media, 2012.
[5] W. Cai, X. Li, and L. Liu. A phase shift deep neural network for high frequency approximation and wave problems. SIAM J. Sci. Comput., 42(5):A3285–A3312, 2020.
[6] O. Cessenat and B. Despres. Application of an ultra weak variational formulation of elliptic pdes to the two-dimensional helmholtz problem. SIAM J. Numer. Anal., 35(1):255–299, 1998.
[7] H. Chen, P. Lu, and X. Xu. A hybridizable discontinuous galerkin method for the helmholtz equation with high wave number. SIAM J. Numer. Anal., 51(4):2166–2188, 2013.
[8] W. E and B. Yu. The deep ritz method: a deep learning-based numerical algorithm for solving variational problems. Commun. Math. Stat., 6(1):1–12, 2018.
[9] C. Gittelson, R. Hiptmair, and I. Perugia. Plane wave discontinuous galerkin methods: analysis of the h-version. ESAIM: Math. Model. Numer. Anal., 43(2):297–331, 2009.
[10] I. Gühring, G. Kutyniok, and P. Petersen. Error bounds for approximations with deep relu neural networks in w s, p norms. Anal. Appl., 18(05):803–859, 2020.
[11] J. He, L. Li, J. Xu, and C. Zheng. Relu deep neural networks and linear finite elements. J. Comput. Math., 38(3):502–527, 2020.
[12] J. He and J. Xu. Mgnet: A unified framework of multigrid and convolutional neural network. Science china mathematics, 62(7):1331–1354, 2019.
[13] R. Hiptmair, A. Moiola, and I. Perugia. Plane wave discontinuous galerkin methods for the 2d helmholtz equation: analysis of the p-version. SIAM J. Numer. Anal., 49(1):264–284, 2011.
[14] K. Hornik. Approximation capabilities of multilayer feedforward networks. Neural networks, 4(2):251–257, 1991.
[15] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. Neural networks, 2(5):359–366, 1989.
[16] L. Imbert-Gérard and B. Després. A generalized plane-wave numerical method for smooth nonconstant coefficients. IMA J. Numer. Anal., 34(3):1072–1103, 2014.
[17] L. Jones. A simple lemma on greedy approximation in hilbert space and convergence rates for projection pursuit regression and neural network training. The annals of Statistics, pages 608–613, 1992.
[18] G. Karniadakis, I. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, and L. Yang. Physics-informed machine learning. Nature Reviews Physics, 3(6):422–440, 2021.
[19] R. Khodayi-Mehr and M. Zavlanos. Varnet: Variational neural networks for the solution of partial differential equations. In Learning for Dynamics and Control, pages 298–307. PMLR, 2020.
[20] C. Lam and C. Shu. A phase-based interior penalty discontinuous galerkin method for the helmholtz equation with spatially varying wavenumber. Computer Methods in Applied Mechanics and Engineering, 318:456–473, 2017.
[21] L. Lu, X. Meng, Z. Mao, and G. Karniadakis. Deepxde: A deep learning library for solving differential equations. SIAM Review, 63(1):208–228, 2021.
[22] D. McNamara, C. Pistorius, and J. Malherbe. Introduction to the uniform geometrical theory of diffraction. Artech House Norwood, MA, 1990.
[23] J. Meade, J. Andrew, and A. Fernandez. The numerical solution of linear ordinary differential equations by feedforward neural networks. Mathematical and Computer Modelling, 19(12):1–25, 1994.

[24] M. Melenk and S. Sauter. Wavenumber explicit convergence analysis for galerkin discretizations of the helmholtz equation. SIAM J. Numer. Anal., 49(3):1210–1243, 2011.
[25] A. Moiola, R. Hiptmair, and I. Perugia. Plane wave approximation of homogeneous helmholtz solutions. Zeitschrift für angewandte Mathematik und Physik, 62(5):809, 2011.
[26] N. 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. Phys., 290:318–335, 2015.
[27] J. Opschoor, P. Petersen, and C. Schwab. Deep relu networks and high-order finite element methods. Anal. Appl., 18(05):715–770, 2020.
[28] E. Samaniego, C. Anitescu, S. Goswami, V. Nguyen-Thanh, H. Guo, K. Hamdia, X. Zhuang, and T. Rabczuk. An energy approach to the solution of partial differential equations in computational mechanics via machine learning: Concepts, implementation and applications. Comput. Methods Appl. Mech. Eng., 362:112790, 2020.
[29] J. Shen and L. Wang. Spectral approximation of the helmholtz equation with high wave numbers. SIAM j. numer. anal., 43(2):623–644, 2005.
[30] J. Sirignano and K. Spiliopoulos. Dgm: A deep learning algorithm for solving partial differential equations. J. comput. phys., 375:1339–1364, 2018.
[31] J. Wang and Z. Zhang. A hybridizable weak galerkin method for the helmholtz equation with large wave number: hp analysis. Int. J. Numer. Anal. Model., 14(4–5):744–761, 2017.
[32] Z. Wang and Z. Zhang. A mesh-free method for interface problems using the deep learning approach. J. Comput. Phys., 400:108963, 2020.
[33] L. Zhu and H. Wu. Preasymptotic error analysis of cip-fem and fem for helmholtz equation with high wave number. part ii: hp version. SIAM J. Numer. Anal., 51(3):1828–1852, 2013.

† Dutchfork High School, Irmo, SC, USA
E-mail: yangandy2006@gmail.com.

‡ Corresponding author. Department of Computer Science, College of Staten Island, The City University of New York, NY, USA.
E-mail: feng.gu@csi.cuny.edu.