Hierarchical Learning to Solve PDEs Using Physics-Informed Neural Networks

Jihun Han[1[0000−0001−9300−0541]] and Yoonsang Lee[1[0000−0002−5573−9414]]

Department of Mathematics, Dartmouth College, Hanover, NH 03755 USA
{jihun.han, yoonsang.lee}@dartmouth.edu

Abstract. The neural network-based approach to solving partial differential equations has attracted considerable attention. In training a neural network, the network learns global features corresponding to low-frequency components while high-frequency components are approximated at a much slower rate. For a class of equations in which the solution contains a wide range of scales, the network training process can suffer from slow convergence and low accuracy due to its inability to capture the high-frequency components. In this work, we propose a sequential training based on a hierarchy of networks to improve the convergence rate and accuracy of the neural network solution to partial differential equations. The proposed method comprises multi-training levels in which a newly introduced neural network is guided to learn the residual of the previous level approximation. We validate the efficiency and robustness of the proposed hierarchical approach through a suite of partial differential equations.

Keywords: hierarchical learning · scientific machine learning · physics-informed neural networks

1 Introduction

Many research efforts have focused on well-designed objective or loss functions to guide a neural network to approximate the solution of a PDE. An objective function measures how well a neural network satisfies the PDE, typically defined as the empirical mean of the residual by a neural network. Physics-informed neural networks (PINN) [10], and DGM [11] consider the direct PDE residual as the loss function so that the neural network satisfies the PDE in the domain.

In particular, PINN has flexibility in informing physical laws described in differential equations, and thus it has been employed in solving a wide range of PDEs. Despite its successful results in many applications, PINN suffers from a slow convergence rate and accuracy degradation for a certain class of PDEs. Such computational challenges are often inherent from the characteristics of the solution of a PDE, in particular when the solution involves a wide range of scales. The multiscale PDE problems arise in various scientific domains, such as fluid dynamics, quantum mechanics, or molecular dynamics. Standard methods, such as finite difference methods (FDM) or finite element methods (FEM), encounter

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36024-4_42

an intractable computational complexity in resolving all relevant scales, numerical instabilities, or slow convergence in general. There have been significant efforts in developing efficient discretization methods for multiscale problems. As a representative example, the multigrid (MG) method [2] addresses the disparate convergence rates of different scale components through a hierarchical design of discretizations. The MG method captures the diverse target scale components of the solution from the collaboration of scale-corresponding grid approximations. The MG method achieves fast convergence as the method approximates all scale components corresponding to the grids. A hierarchical approach for multiscale problems has also been discussed in [6] for turbulent diffusion. Instead of using a fine resolution grid for whole domain at each level, the approach in [6] uses a local spatiotemporal domain. By designing a hierarchy that captures all possible scale ranges of the solution, the approach can capture the effective macroscopic behavior by a significant computational gain.

Neural network-based methods also face hurdles in approximating the multiscale solution of a PDE in that a neural network prefers low frequencies (F-Principle) [17]. Therefore, a standard network design can be ineffective in learning the high-frequency components. There are several recent research efforts to address the limitation in training high-frequencies by modifying the architecture or the ingredients of neural networks [3, 12, 15]. In particular, [15] proposed a neural network structure with Fourier feature embeddings to learn the multiscale solution efficiently. The embedding allows one to specify target characteristic frequencies of the neural network. The authors consider the multiple embeddings of inputs to simultaneously learn the diverse range of frequencies in the solution.

This work proposes hierarchical learning for solving PDEs to expedite the convergence through a sequential training of neural networks based on a hierarchy of networks. Using a set of networks of different target scales, where it is assumed that the sum of the networks can represent the solution of a PDE solution, we separate the training process so that each network can learn its corresponding scales without the training interruption from other networks. Once the training for a network finished, the training switches to the next network to correct the residual of the approximation up to the previous level. Among other methods to impose different target scales for networks, we test 1) standard MLPs of different complexity (number of layers and neurons) and 2) Fourier feature embedding. We emphasize that our proposed approach differs from other network design-based methods, such as [3, 15, 7], in that the focus of the hierarchical learning is the sequential training process. In our numerical experiments, it is shown that a sequential training process of networks with different target scales performs better than the training of the complex network that combines all networks of different levels at once. Thus, separating the training process for different target scales becomes a key ingredient of the proposed hierarchical learning. We believe the proposed hierarchical learning can apply to many neural network-based methods. In this study, we investigate the efficacy of the proposed hierarchical learning method in the framework of PINN along with other tech-

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36024-4_42

niques to improve the convergence of PINN, such as the aforementioned adaptive weighting algorithms.

The rest of the paper is organized as follows. Section 2 reviews the PINN method and discusses the previous efforts to overcome the spectral barriers in training neural networks to solve PDEs. In section 3, we propose the hierarchical learning methodology, while section 4 provides numerical experiments validating the efficacy of the proposed method. We conclude this paper with discussions about the limitation and future directions of the current study in section 5.

2 Physics-informed neural networks

In this section, we summarize the standard Physics-informed Neural Networks (PINN) for a boundary value problem and discuss its variants to address the limitations of PINN. We consider the partial differential equation of unknown real-valued function u in a bounded domain $\Omega \subset \mathbb{R}^n$

$$\mathcal{N}[u](\boldsymbol{x}) = f(\boldsymbol{x}), \text{ and } \mathcal{B}[u](\boldsymbol{x}) = g(\boldsymbol{x}), \ \boldsymbol{x} \in \partial\Omega, \tag{1}$$

where $\mathcal{N}$ is a differential operator and $\mathcal{B}$ represents a boundary condition operator. General deep learning-based methods to solve Eq. (1) employ a neural network, $u(\boldsymbol{x}; \boldsymbol{\theta})$, to approximate the solution, and train the parameters $\boldsymbol{\theta}$ under the guidance of a loss function leading the neural network to satisfy Eq. (1). The PINN measures the direct PDE residuals in the loss function

$$\mathcal{L}(\boldsymbol{\theta}) = \lambda_\Omega \mathcal{L}_\Omega(\boldsymbol{\theta}) + \lambda_{\partial\Omega} \mathcal{L}_{\partial\Omega}(\boldsymbol{\theta}), \tag{2}$$

which consists of the interior and boundary loss terms

$$\mathcal{L}_\Omega(\boldsymbol{\theta}) = \sum_{i=1}^{N_r} \frac{\left|\mathcal{N}[u(\cdot;\boldsymbol{\theta})](\boldsymbol{x}_r^i) - f(\boldsymbol{x}_r^i)\right|^2}{N_r}, \quad \mathcal{L}_{\partial\Omega}(\boldsymbol{\theta}) = \sum_{i=1}^{N_b} \frac{\left|\mathcal{B}[u(\cdot;\boldsymbol{\theta})](\boldsymbol{x}_b^i) - g(\boldsymbol{x}_b^i)\right|^2}{N_b} \tag{3}$$

respectively. Here $\{\boldsymbol{x}_r^i\}_{i=1}^{N_r}$ and $\{\boldsymbol{x}_b^i\}_{i=1}^{N_b}$ are sampling points in the interior, and the boundary of the domain, respectively.

Despite the remarkable achievement in many applications, the PINN often struggles to learn the solutions of PDEs with either slow convergence or degraded accuracy. Recent works have endeavored to understand unfavorable training scenarios of neural networks and proposed alternative methodologies to overcome the limitations. One of the methods includes balancing different terms of the loss function in the context of multi-objective optimization discussed in section 1 [14, 16, 8]. Another direction addresses the intrinsic behavior of training neural networks, which is specifically disadvantageous to learn functions involving diverse frequency spectrum [1, 9, 17, 18].

The general learning process of neural networks has been studied from spectral analysis [1, 9, 17, 18]. The F-principle [17] shows that the gradient-based

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36024-4_42

training process has spectral bias as the neural networks tend to learn low frequencies while it requires a longer time to fit high frequencies. This phenomenon is a challenge in neural network-based methods to solve multiscale PDE problems that suffer from slow convergence or low accuracy. The networks miss the high-frequency components unless the training process is sufficiently long to learn high-frequency components. [3] proposed a neural network architecture with an input scaling treatment for converting the high-frequency components to low-frequency ones preferable to learning. The network is installed with a compact supported activation function and is effectively applied to multiscale applications [13, 7].

Another work in [12] showed that a simple random Fourier feature embedding of inputs enables a standard MLP to learn high-frequency components more efficiently in applications of computer vision and graphics. Namely, the embedding corresponds to a map from the input $\boldsymbol{x} \in \mathbb{R}^n$ to the $2m$-dimensional frequency domain as

$$\boldsymbol{x} \in \mathbb{R}^n \quad \mapsto \quad \begin{bmatrix} \boldsymbol{a} \odot \cos(\boldsymbol{B}_\sigma \boldsymbol{x}) \\ \boldsymbol{a} \odot \sin(\boldsymbol{B}_\sigma \boldsymbol{x}) \end{bmatrix} \in \mathbb{R}^{2m}, \tag{4}$$

where $\boldsymbol{B}_\sigma \in \mathbb{R}^{n \times m}$ is a random wave number matrix sampled from the Gaussian distribution $\mathcal{N}(0, \sigma^2)$ and $\boldsymbol{a} \in \mathbb{R}^m$ is a scaling vector. The authors analyzed the effect of the embedding on the neural tangent kernel (NTK) of the standard MLP to attenuate the spectral bias using appropriate σ and $\boldsymbol{a}$.

3 Hierarchical PINN

The methods discussed in the previous section focus on various strategies to improve the capability of a single neural network to learn a wide range of scales in the solution of a PDE. In the current study, we propose a hierarchical learning procedure of neural networks to represent the multiscale solution. The proposed method, which we call 'hierarchical Physics-informed neural network' (HiPINN), uses a sequence of neural networks to represent the multiscale solution of a PDE and trains them sequentially rather than simultaneously. Our hierarchical approach is motivated by the multigrid method that uses a hierarchy of different grid sizes to expedite the convergence of an iterative method to solve PDEs [2]. The rationale of the multigrid method is that a grid size has its characteristic scale with its corresponding convergence rate. The multigrid method achieves a fast convergence rate by capturing different scale components through variable grid sizes. The idea of the proposed hierarchical approach for PINN is to impose a hierarchy in training so that each network can capture its corresponding scales without the interruption in training other networks, which enables us to capture uniformly all possible ranges of scales.

3.1 Hierarchical design of networks

HiPINN employs a set of M neural networks $\{v_m(\boldsymbol{x}; \boldsymbol{\theta}_m)\}_{m=1}^M$ with a hierarchy to represent the PDE solution where M represents the number of levels for different characteristic scales. To mimic the hierarchy of the multigrid method, we

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36024-4_42

consider two approaches in the current study. The first approach is the standard multiplayer perceptron (MLP) with various network sizes and complexity. We expect that a simple network will be enough to approximate for low variability components of the unknown solution, while high variability components require a more complicated network. Following this intuitive argument, we increase the complexity of networks by increasing the depth and width of each network. One issue of this approach is that it is unclear to cover a specific range of scales. Suppose two networks are significantly different in terms of complexity. In that case, we expect that the two networks will represent different scale components, but it is not clear whether there is a gap between them. In the study of the spectral bias of neural networks [9], it is shown that higher frequencies are significantly less robust than lower ones in the perturbation of the neural network parameters. This observation indicates that a limited volume in the parameter space is involved in expressing the high-frequency components. With the support of this observation, we consider the complexity of networks in composing the hierarchy. The schematic diagram of the hierarchical employment of the MLPs is displayed in Fig. 1.

Another approach to impose a hierarchy in the network is the Fourier feature embedding [12]. The structure of each network is identical through levels with the input embedding as Eq. (4). However, we vary them by increasing σ as the level (m) increases so that a high-level network represents high frequency or wavenumber behaviors compared to the ones captured by the low-level networks. As the network size does not change through the hierarchy, the Fourier embedding-based approach does not provide any computational efficiency in solving a low-level network compared to the hierarchy using the network complexity of MLP. However, the Fourier embedded hierarchy can specify the target characteristic scales through σ. In the multiscale approach using the Fourier embedding for PINN [15], a various range of σ values is incorporated to design a single network to target all possible ranges of scales in the solution. In terms of the network complexity, HiPINN does not necessarily use a network more complicated than the one used in [15]. The goal of HiPINN is to expedite the training process by dividing the training into specific scales instead of training all possible scales simultaneously. The schematic of a hierarchical neural network design using the Fourier feature embedding is shown in Fig. 2.

3.2 HiPINN algorithm

The benefit of the proposed hierarchical learning method comes from the sequential training based on a hierarchy of networks. Using the neural networks with a hierarchy, the M-level HiPINN representation of the PDE solution is the sum of all neural networks, which is given as

$$u_M(\boldsymbol{x}) = \sum_{m=1}^{M} v_m(\boldsymbol{x}; \boldsymbol{\theta}_m). \tag{5}$$

Under this structure, the training of each level network is on the correction of the residual of the previous level solution representation. To add the $(M+1)$-th

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36024-4_42

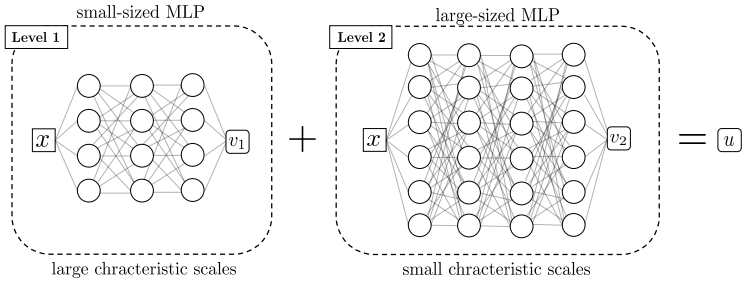

Fig. 1. Hierarchical composition of standard MLPs; small-sized MLP for capturing low variability components and large-sized MLP for high variability components.

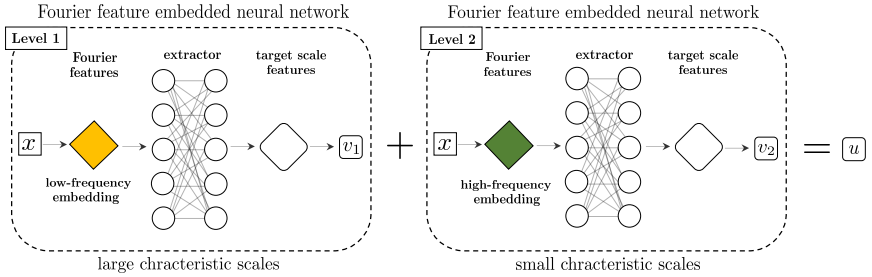

Fig. 2. Hierarchical composition of Fourier feature embedded neural networks; With the same network architecture, the target characteristic frequency is controlled by the Fourier feature embedding of inputs as Eq. (4).

level to u_M using v_{M+1}, the loss function $\mathcal{L}^{(M+1)}$ is

$$\mathcal{L}^{(M+1)}(\boldsymbol{\theta}_M) = \lambda_\Omega \mathcal{L}_\Omega^{(M+1)}(\boldsymbol{\theta}_{M+1}) + \lambda_{\partial\Omega} \mathcal{L}_{\partial\Omega}^{(M+1)}(\boldsymbol{\theta}_{M+1}), \tag{6}$$

$$\mathcal{L}_\Omega^{(M+1)}(\boldsymbol{\theta}_{M+1}) = \frac{1}{N_r} \sum_{i=1}^{N_r} \left| \mathcal{N}[u_M + v_{M+1}(\cdot;\boldsymbol{\theta}_{M+1})](\boldsymbol{x}_r^i) - f(\boldsymbol{x}_r^i) \right|^2, \tag{7}$$

$$\mathcal{L}_{\partial\Omega}^{(M+1)}(\boldsymbol{\theta}_{M+1}) = \frac{1}{N_b} \sum_{i=1}^{N_b} \left| \mathcal{B}[u_M + v_{M+1}(\cdot;\boldsymbol{\theta}_{M+1})](\boldsymbol{x}_b^i) - g(\boldsymbol{x}_b^i) \right|^2. \tag{8}$$

We note that u_M is already approximated, and thus the training variable related to $\mathcal{L}^{(M+1)}$ is $\boldsymbol{\theta}_{M+1}$. If the differential operator and the boundary operator are linear, the $(M+1)$-th level training is equivalent to solving the original PDE operator using v_{M+1} for modified $f^{(M+1)}(\boldsymbol{x})$ and $g^{(M+1)}(\boldsymbol{x})$, which are given by

$$f^{(M+1)}(\boldsymbol{x}) = f(\boldsymbol{x}) - \mathcal{N}[u_M](\boldsymbol{x}) \text{ and } g^{(M+1)}(\boldsymbol{x}) = g(\boldsymbol{x}) - \mathcal{B}[u_M](\boldsymbol{x}), \tag{9}$$

respectively. Therefore, the implementation for the linear case involves only marginal modification of the standard PINN method. When the differential operator $\mathcal{N}$ is nonlinear, the differential operator on v_{M+1} at the $(M+1)$-th level

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36024-4_42

will be different from the original operator $\mathcal{N}$. However, the structure of the operator does not change over the level; it remains at minimizing the loss related to $\mathcal{N}$['approximation up to the previous level' + 'current level network'] over the current level network. Thus HiPINN for a nonlinear problem requires only one implementation of a solver and uses it repeatedly for all levels.

We also note that HiPINN does not require any projection or interpolation operations between different level solutions, which are crucial in the algebraic multigrid method. In HiPINN, each level approximate solution uses the same sampling points $\{\boldsymbol{x}_r^i\}_{i=1}^{N_r}$ and $\{\boldsymbol{x}_b^i\}_{i=1}^{N_b}$. From the homogeneity of the problem to be solved at each level, it is straightforward to implement various types of cycles to iterate over different levels, such as V and W cycles [2]. The V cycle starts from a low resolution to a high resolution and iterates back to a low resolution. The W cycle repeats the V cycle to approximate scale components that are not sufficiently captured at the corresponding level.

4 Numerical experiments

In this section, we validate the robustness and effectiveness of the proposed hierarchical learning methodology to solve PDEs through a suite of test problems. In all numerical experiments, we use the standard multilayer perceptrons (MLPs) and Fourier feature embedded neural networks [15] with the tanh activation function. In the Fourier feature embedding case, the architecture of each network is designed as follows in sequence; 1) multiple Fourier feature embeddings of input, each of embedding corresponding to the map in Eq. (4) with scaling vector $\boldsymbol{a} = \mathbf{1}$, 2) a multiscale feature extractor MLP common for each embedded feature, 3) a final linear layer passed by concatenated features extracted. In our numerical experiments, we consider the dimension of a Fourier feature embedding the same as that of the first hidden layer of the multiscale feature extractor. Moreover, we include a dense layer to pass the concatenated features, which performs better than direct linear mapping to the output in our experiments.

We train each neural network using the Adam optimizer [5] with $\beta_1 = 0.95$ and $\beta_2 = 0.95$, and all the trainable parameters are initialized from Glorot normal distribution [4]. Moreover, we employ the adaptive weights algorithm [16] in all experiments, updating the weights in every 100 gradient descent steps for computational efficiency. To validate the performance of the proposed method, we consider the standard training procedure using networks with and without a hierarchy. We note that the standard training using networks with a hierarchy uses the same overall network structure as in the proposed learning method. The difference is in the training process; the proposed method uses sequential training while the standard training uses simultaneous training (that is, the networks of different scales are trained at the same time). Except for the first test in which an exact solution is available, we obtain reference solutions using the FEM method with sufficiently large mesh sizes. We measure the accuracy of the network-based solutions $\tilde{u}$ using the relative $\mathcal{L}^2$-error, $\frac{\|\tilde{u}-u\|_{2,\Omega}}{\|u\|_{2,\Omega}}$. All benchmark

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36024-4_42

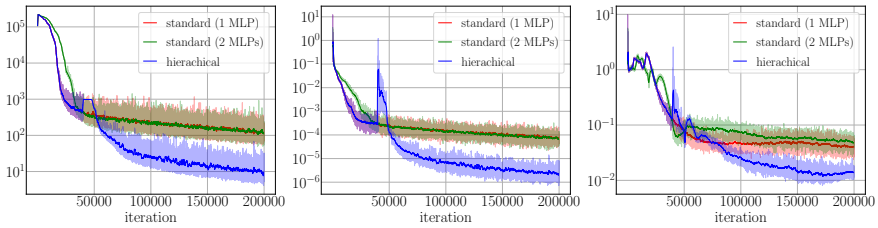

Fig. 3. Training procedures of MLPs in solving Eq. (10) by standard (single MLP: 3 hidden layers, 200 units, two MLPs: 2, 3 hidden layers, 200 units) and hierarchical (first level: 3 hidden layers, 200 units, second level: 5 hidden layers, 200 units) learning. (left) interior losses, (middle) boundary losses, (right) relative $\mathcal{L}^2$-errors. The hierarchical learning corresponds to second level initiation at 4×10^4 iterations.

losses are referred to the test losses computed on the grid points. We want to note that the reference FEM simulation is much more efficient than the neural network-based methods in the tests we consider here, which are PDEs in the 2D space domain. The computational efficiency of the neural network-based methods comes in when the domain is in a high-dimensional space. Note that in a high-dimensional case, the mesh generation and solving its corresponding (nonlinear) system can be extremely slow compared to the Monte-Carlo-based training in the neural network approach [10].

4.1 Poisson equation

As the first example, we consider the Poisson equation in the unit square $\Omega = [0,1]^2$ with a Dirichlet boundary condition,

$$\Delta u = f \quad \text{in} \quad \Omega, \quad u = g \quad \text{on} \quad \partial\Omega. \tag{10}$$

Here, we choose the force term f and the boundary value g such that Eq. (10) has the exact solution $u(\boldsymbol{x}) = \sin(8\pi x_1^2 + 4\pi x_2)\sin(8\pi x_2^2 + 4\pi x_1)$. We consider two-level hierarchical learning with neural networks, $v(\boldsymbol{x};\boldsymbol{\theta}_1)$ and $v(\boldsymbol{x};\boldsymbol{\theta}_2)$, which are sequentially trained using the corresponding loss functions,

$$\mathcal{L}^{(1)}(\boldsymbol{\theta}_1) = \frac{\lambda_\Omega}{N_r}\sum_{i=1}^{N_r}\left|\Delta v(\boldsymbol{x}_r^i;\boldsymbol{\theta}_1) - f(\boldsymbol{x}_r^i)\right|^2 + \frac{\lambda_{\partial\Omega}}{N_b}\sum_{i=1}^{N_b}\left|v(\boldsymbol{x}_b^i;\boldsymbol{\theta}_1) - g(\boldsymbol{x}_b^i)\right|^2, \tag{11}$$

$$\begin{aligned}\mathcal{L}^{(2)}(\boldsymbol{\theta}_2) = \frac{\lambda_\Omega}{N_r}\sum_{i=1}^{N_r}\left|\left(\Delta v(\boldsymbol{x}_r^i;\boldsymbol{\theta}_1^*) + \Delta v(\boldsymbol{x}_r^i;\boldsymbol{\theta}_2)\right) - f(\boldsymbol{x}_r^i)\right|^2 \\ + \frac{\lambda_{\partial\Omega}}{N_b}\sum_{i=1}^{N_b}\left|\left(v(\boldsymbol{x}_b^i;\boldsymbol{\theta}_1^*) + v(\boldsymbol{x}_b^i;\boldsymbol{\theta}_2)\right) - g(\boldsymbol{x}_b^i)\right|^2,\end{aligned} \tag{12}$$

respectively. Here, $\boldsymbol{\theta}_1^*$ in Eq. (12) is the updated $\boldsymbol{\theta}_1$ at level 1 and is fixed at level 2. We note that as the differential (i.e., Laplacian) and boundary operators are

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36024-4_42

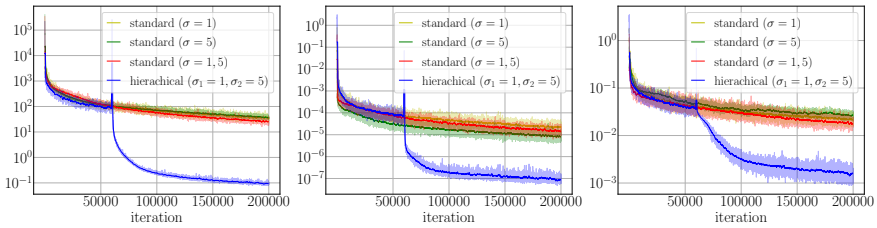

Fig. 4. Training procedures for Eq. (10) by standard and hierarchical learning. (left) interior losses, (middle) boundary losses, (right) relative $\mathcal{L}^2$-errors. The standard learning corresponds to single Fourier feature embedding with $\sigma = 1$, $\sigma = 5$, separately, and multiple embeddings $\sigma = 1, 5$. The hierarchical learning runs with single embedding $\sigma = 1$ at the first level, and $\sigma = 5$ at the second level, in sequence. The second level training is initiated at 6×10^4 iterations.

linear, the second level PDE for the neural network $v(\boldsymbol{x}; \boldsymbol{\theta}_2)$ is also a Poisson equation with a Dirichlet boundary condition with shifted force and boundary functions. We test our method using the standard MLPs, and Fourier feature embedded neural networks with training sample sizes $N_r = 400$ and $N_b = 400$.

First, we use the MLP with three hidden layers of dimension 200 at the first level and five hidden layers of dimension 200 at the second level. The hierarchical learning is compared with the standard learning (i.e., single-level hierarchy) with different sizes of MLPs, H numbers of hidden layers of dimension 200 for $H = 2, 3, \cdots, 8$, among which the MLP with $H = 3$ achieves the best performance in approximating the solution. We also test the standard learning with the sum of two MLPs with H_1 and H_2 numbers of hidden layers of dimension 200 for $H_1, H_2 = 2, 3, \cdots, 8$, $H_1 < H_2$, in which $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$ are trained simultaneously. Among the combination of two MLPs, the networks with $H_1 = 2$ and $H_2 = 3$ perform the best approximation of the solution. Fig. 3 shows the training procedures for 2×10^5 iterations. We observe that the correction of the hierarchical learning at the second level properly works to accelerate the convergence in two losses and achieve better approximation accuracy than standard learning with the single network or the sum of two networks. We want to emphasize that the importance of the sequential training process of the different level networks. In comparison with the proposed method and the two MLP case in which the total networks are the same, the hierarchical training (blue curve) shows better performance than the combined network case (green curve). This result shows that the training of each target scale network can be hindered by the training of other scale networks. For the case of the Fourier feature embedded neural networks, we use the same size neural networks at both levels, where each network has a different Fourier embedding. We use single Fourier feature embedding at each level with $\sigma = 1$ and $\sigma = 5$, respectively, in considering low target frequencies at the first level and relatively high frequencies at the next level. The rest of the network consists of the feature extractor with three hidden layers of dimension 200 followed by the last dense layer of dimension 200. To demonstrate

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36024-4_42

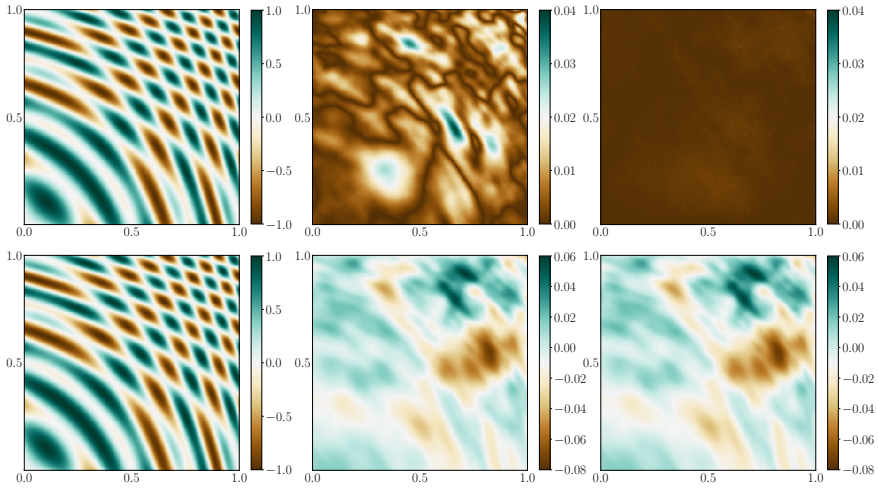

Fig. 5. first row: The numerical solutions of Eq. (10) by standard (multiple embeddings $\sigma = 1, 5$) and hierarchical (single embedding $\sigma_1 = 1$, $\sigma_2 = 5$ in sequence) learning. (left) the exact solution, (middle) the pointwise error corresponding to standard learning, (right) the pointwise error corresponing to proposed hierarchical learning. second row: the approximations at each level in the hierarchical learning. (left) the approximation $v(\cdot; \boldsymbol{\theta}_1^*)$ at the first level, (middle) the approximation $v(\cdot; \boldsymbol{\theta}_2^*)$ at the second level, (right) the target function for $v(\cdot; \boldsymbol{\theta}_2)$ at the second level, which is equal to $(u_{\text{exact}} - v(\cdot; \boldsymbol{\theta}_1^*))$.

the effectiveness of learning the diverse frequencies from low to high in sequence, we compare our method with the standard learning with the single embedding ($\sigma = 1$ and $\sigma = 5$) and multiple embeddings ($\sigma = 1, 5$) aiming to learn various frequencies simultaneously. As shown in Fig. 4, our method accelerates the convergence at the second level and achieves an accurate approximation (relative $\mathcal{L}^2$-error 1.33×10^{-3}) in comparison to the other experiments (best relative $\mathcal{L}^2$-error 1.65×10^{-2}). Particularly, in comparison with the multiple embedding (red curve) and the proposed hierarchical training (blue curve), where the overall network structure to represent the solution is comparable, the hierarchical training process shows a better performance than the simultaneous training process.

Fig. 5 shows the point-wise errors of each level approximation in comparison with the standard learning method. Moreover, our method combined with Fourier feature embedding outperforms the performance of HiPINN using the standard MLP, as we can employ a neural network suitable for learning the target frequencies at each level. We address a question when it is appropriate to switch to the next level. Fig. 6 presents the six training procedures of the Fourier feature embedded neural networks ($\sigma = 1$, $\sigma = 5$ in level sequence). The experiment shows that transition to the next level after 25000 iterations provide comparable overall accuracy using the two-level representation.

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36024-4_42

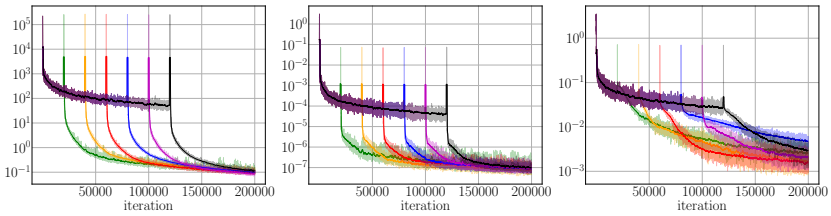

Fig. 6. Training procedures of Fourier feature embedded neural network (single embedding $\sigma_1 = 1$, $\sigma_2 = 5$ in sequence) in solving 2D Poisson equation, Eq. (10) with different second level initiations, $2n \times 10^4$, $n = 1, 2, 3, 4, 5, 6$. (left) interior losses, (middle) boundary losses, (right) relative $\mathcal{L}^2$-errors.

4.2 Steady-state advection-diffusion equation

The last test is to demonstrate the capability of the proposed method in handling the multiscale behavior, which could arise from the intertwined consequences of both differential operator and force term. We consider the steady-state advection-diffusion equation with mixed Dirichlet and Neumann boundary conditions,

$$\begin{aligned} \boldsymbol{w} \cdot \nabla u - \nu \Delta u &= f \quad \text{in} \quad \boldsymbol{x} \in [0,1]^2, \\ u(0, x_2) = g_1, u(1, x_2) &= g_2 \quad \text{for} \quad x_2 \in [0,1], \\ \frac{\partial u}{\partial n} u(x_1, 0) = \frac{\partial u}{\partial n} u(x_1, 1) &= 0 \quad \text{for} \quad x_1 \in [0,1]. \end{aligned} \tag{13}$$

We choose the diffusion coefficient $\nu = 0.01$, the force $f = \sin(4\pi x_2)$, Dirichlet boundary value, $g_1 = 0$ and $g_2 = 1$, and an incompressible velocity field $\boldsymbol{w}$,

$$\boldsymbol{w}(\boldsymbol{x}) = (-5\sin(6\pi x_1)\cos(6\pi x_2), 5\cos(6\pi x_1)\sin(6\pi x_2)) \tag{14}$$

We solve Eq. (13) using two levels, in which neural networks $v(\boldsymbol{x}; \boldsymbol{\theta}_1)$ and $v(\boldsymbol{x}; \boldsymbol{\theta}_2)$ are trained under the following loss functions

$$\begin{aligned} \mathcal{L}^{(1)}(\boldsymbol{\theta}_1) = &\frac{\lambda_\Omega}{N_r} \sum_{i=1}^{N_r} \left| \mathcal{R}(\boldsymbol{\theta}_1; \boldsymbol{x}_r^i) \right|^2 + \frac{\lambda_{\partial\Omega,1}}{N_{b,1}} \sum_{i=1}^{N_{b,1}} \left| \frac{\partial v}{\partial n}(\boldsymbol{x}_{b,1}^i; \boldsymbol{\theta}_1) \right|^2 \\ &+ \frac{\lambda_{\partial\Omega,2}}{N_{b,2}} \sum_{i=1}^{N_{b,2}} \left| v(\boldsymbol{x}_{b,2}^i; \boldsymbol{\theta}_1) - g(\boldsymbol{x}_{b,2}^i) \right|^2, \end{aligned} \tag{15}$$

$$\begin{aligned} \mathcal{L}^{(2)}(\boldsymbol{\theta}_2) = &\frac{\lambda_\Omega}{N_r} \sum_{i=1}^{N_r} \left| \mathcal{R}(\boldsymbol{\theta}_2; \boldsymbol{x}_r^i) \right|^2 + \frac{\lambda_{\partial\Omega,1}}{N_{b,1}} \sum_{i=1}^{N_{b,1}} \left| \frac{\partial v}{\partial n}(\boldsymbol{x}_{b,1}^i; \boldsymbol{\theta}_1^*) + \frac{\partial v}{\partial n}(\boldsymbol{x}_{b,1}^i; \boldsymbol{\theta}_2) \right|^2 \\ &+ \frac{\lambda_{\partial\Omega,2}}{N_{b,2}} \sum_{i=1}^{N_{b,2}} \left| v(\boldsymbol{x}_{b,2}^i; \boldsymbol{\theta}_1^*) + v(\boldsymbol{x}_{b,2}^i; \boldsymbol{\theta}_2) - g(\boldsymbol{x}_{b,2}^i) \right|^2, \end{aligned} \tag{16}$$

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36024-4_42

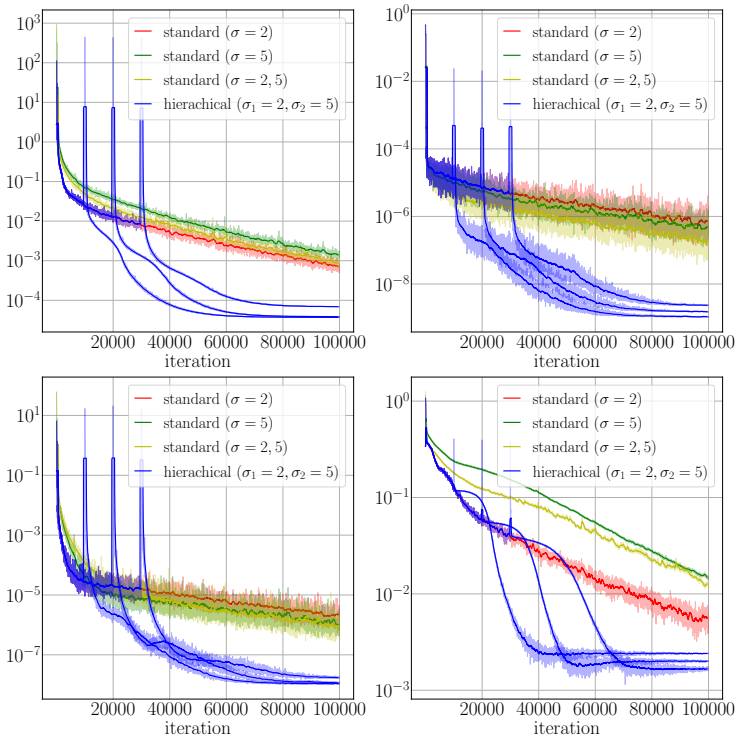

Fig. 7. Training procedures for steady-state advection-diffusion equation, Eq. (13), by standard learning and proposed hierarchical learning. (first row, left) interior losses, (first row, right) Dirichlet boundary losses, (second row, left) Neumann boundary losses, (second row, right) relative $\mathcal{L}^2$-errors. The standard learning corresponds to single Fourier feature embedding with $\sigma = 2$, $\sigma = 5$, separately, and multiple embeddings $\sigma = 2, 5$. The hierarchical learning runs with single embedding $\sigma = 2$ at the first level, and $\sigma = 5$ at the second level, in sequence. The second level training is initiated at $n \times 10^4$, $n = 1, 2, 3$, iterations. The detail approximations are also presented in Fig. 8.

where the residuals of PDE at each level are

$$\begin{aligned} \mathcal{R}(\boldsymbol{\theta}_1; \boldsymbol{x}_r^i) &= \boldsymbol{w} \cdot \nabla v(\boldsymbol{x}_r^i; \boldsymbol{\theta}_1) - \nu \Delta v(\boldsymbol{x}_r^i; \boldsymbol{\theta}_1) - f(\boldsymbol{x}_r^i) \\ \mathcal{R}(\boldsymbol{\theta}_2; \boldsymbol{x}_r^i) &= \boldsymbol{w} \cdot \nabla v_c - \nu \Delta v_c - f(\boldsymbol{x}_r^i) \qquad v_c = v(\boldsymbol{x}_r^i; \boldsymbol{\theta}_1^*) + v(\boldsymbol{x}_r^i; \boldsymbol{\theta}_2). \end{aligned} \tag{17}$$

Here, $\boldsymbol{x}_{b,1}$ and $\boldsymbol{x}_{b,2}$ are sampling points for the Neumann and the Dirichlet boundary conditions, respectively, g is read as g_1 or g_2 depending on the location of $\boldsymbol{x}_{b,2}^i$, and $\boldsymbol{\theta}_1^*$ in Eq. (16) and Eq. (17) is the updated $\boldsymbol{\theta}_1$ at the first level and is fixed at the second level. Moreover, we also apply the adaptive weight algorithm [16] by treating the weight λ_Ω on boundary loss separately into two parts, $\lambda_{\partial\Omega,1}$ on the Neumann boundary loss and $\lambda_{\partial\Omega,2}$ on the Dirichlet boundary loss.

We apply the hierarchical learning method with the Fourier feature embedded neural networks. The exact size neural networks are considered at both

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36024-4_42

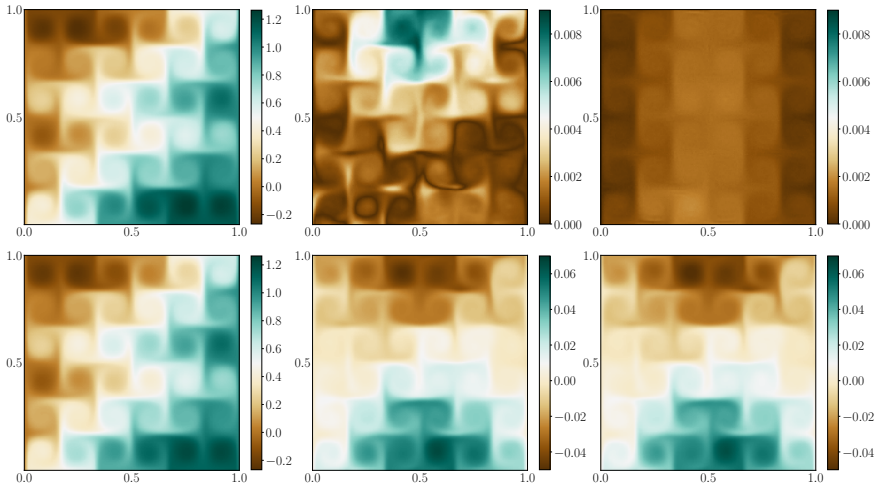

Fig. 8. first row: The numerical solutions of Eq. (13) by standard (single embedding $\sigma = 2$) and hierarchical (single embedding $\sigma_1 = 2$, $\sigma_2 = 5$ in sequence) learning. (left) the reference solution, (middle) the pointwise error corresponding to standard learning, (right) the pointwise error corresponding to proposed hierarchical learning. second row: the approximations at each level in the hierarchical learning. (left) the approximation $v(\cdot; \boldsymbol{\theta}_1^*)$ at first level, (middle) the approximation $v(\cdot; \boldsymbol{\theta}_2^*)$ at second level, (right) the target function for $v(\cdot; \boldsymbol{\theta}_2)$ at second level, which is equal to $(u_{\text{reference}} - v(\cdot; \boldsymbol{\theta}_1^*))$.

levels using different single embedding; $\sigma = 2$ at the first level and $\sigma = 5$ at the second level to learn low and high-frequency components. The rest of the network comprises the feature extractor with three hidden layers of dimension 200 followed by the last 200-dimensional dense layer. We train the neural networks over 1×10^5 iterations, where we switch to the second level at various instances (which are at $n \times 10^4$, $n = 1, 2, 3$, iterations).

We compare the hierarchical learning method with the standard learning approach using the same size neural network with different Fourier feature embeddings; single embedding using $\sigma = 2$ or $\sigma = 5$, and multiple embeddings using $\sigma = 2, 5$. Fig. 7 shows the training procedures in terms of three losses and relative $\mathcal{L}^2$-errors. Among the standard learning experiments, $\sigma = 2$ embedding is suitable for this example as it has the most accurate approximation with a $\mathcal{L}^2$ error 4.76×10^{-3}. In comparison with the hierarchical learning method, hierarchical learning has the lowest error 2.41×10^{-3} when the second level training is triggered after 1×10^4 iterations of the first level. We also note that hierarchical learning converges after 4×10^4 iterations, which is 2.5 times faster than the other method. Fig. 8 shows the numerical solutions from both standard learning and hierarchical learning for reference.

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36024-4_42

5 Discussions and conclusions

Several research efforts have been focused on the design of a network to efficiently represent a PDE solution that contains a wide range of scales, including hierarchical networks. However, the training process can suffer from slow convergence due to the interruptions of different scale components of the multiscale network. Thus, the trainability of the network becomes an issue even though the network can represent the multiscale solution if sufficiently trained. This study proposed a hierarchical learning method to solve PDEs using neural networks, which uses sequential training based on a hierarchy of networks. The rationale of the sequential training is to focus on the target characteristic scales of each network by training them separately rather than by training all networks simultaneously. Among several other methods to impose a hierarchy in the network, we tested two methods; 1) multi-layer perceptrons (MLPs) with various network complexities, and 2) Fourier feature embedded networks. The first approach has a computational efficiency in solving a low complexity network while capturing the low-frequency components of the solution. The second approach does not provide any computational gain as each network at different levels has the same complexity. Still, we can explicitly impose the range of scales of the solution through the Fourier feature embedded layers. The proposed hierarchical learning method has been tested through a suite of numerical tests including the advection-diffusion problem with a multiscale velocity field.

There are several issues to be addressed for the proposed hierarchical learning method. It is unclear to see the connection between the network complexity and its characteristic scales to represent a function. We have checked in our numerical experiments that changing the complexity of a network will change its corresponding scales. Still, we lack explicit and rigorous criteria to determine the characteristic scales.
Also, we used the same network complexity for each level for the Fourier feature embedding approach, assuming that the Fourier embedded layer will determine its characteristic scales.

In applying the hierarchical learning to time dependent problems, there are two approaches. One approach is to use a network to learn the spatiotemporal scales at the same time. The other approach is to march the problem where the spatial variations are learned through a network [10]. We are interested in designing hierarchical networks to resolve multiscale behaviors in the temporal domain, particularly to capture the long-time behavior of a dynamical system, such as the climatology of geophysical fluid systems. Lastly, we have tested the hierarchical learning method in the PINN framework in the current study. As the overarching idea of the proposed method is in the efficient representation of a multiscale function using hierarchical networks, we expect that the proposed method can apply to other network-based methods for solving PDEs, which we leave as future work.

Acknowledgements YL is supported in part by NSF DMS-1912999 and ONR MURI N00014-20-1-2595.

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36024-4_42

References

1. Arpit, D., Jastrzębski, S., Ballas, N., Krueger, D., Bengio, E., Kanwal, M.S., Maharaj, T., Fischer, A., Courville, A., Bengio, Y., et al.: A closer look at memorization in deep networks. In: International Conference on Machine Learning. pp. 233–242. PMLR (2017)
2. Briggs, W.L., Henson, V.E., McCormick, S.F.: A multigrid tutorial. SIAM (2000)
3. Cai, W., Xu, Z.Q.J.: Multi-scale deep neural networks for solving high dimensional pdes. arXiv preprint arXiv:1910.11710 (2019)
4. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the thirteenth international conference on artificial intelligence and statistics. pp. 249–256 (2010)
5. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
6. Lee, Y., Engquist, B.: Multiscale numerical methods for advection-diffusion in incompressible turbulent flow fields. J. Comput. Phys. **317**, 33–46 (2016)
7. Liu, Z., Cai, W., Xu, Z.Q.J.: Multi-scale deep neural network (mscalednn) for solving poisson-boltzmann equation in complex domains. arXiv preprint arXiv:2007.11207 (2020)
8. van der Meer, R., Oosterlee, C.W., Borovykh, A.: Optimally weighted loss functions for solving pdes with neural networks. J. Comput. Appl. Math **405**, 113887 (2022)
9. Rahaman, N., Baratin, A., Arpit, D., Draxler, F., Lin, M., Hamprecht, F., Bengio, Y., Courville, A.: On the spectral bias of neural networks. In: International Conference on Machine Learning. pp. 5301–5310. PMLR (2019)
10. Raissi, M., Perdikaris, P., Karniadakis, G.E.: Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. J. Comput. Phys. **378**, 686–707 (2019)
11. Sirignano, J., Spiliopoulos, K.: Dgm: A deep learning algorithm for solving partial differential equations. J. Comput. Phys. **375**, 1339–1364 (2018)
12. Tancik, M., Srinivasan, P., Mildenhall, B., Fridovich-Keil, S., Raghavan, N., Singhal, U., Ramamoorthi, R., Barron, J., Ng, R.: Fourier features let networks learn high frequency functions in low dimensional domains. Advances in Neural Information Processing Systems **33**, 7537–7547 (2020)
13. Wang, B., Zhang, W., Cai, W.: Multi-scale deep neural network (mscalednn) methods for oscillatory stokes flows in complex domains. arXiv preprint arXiv:2009.12729 (2020)
14. Wang, S., Teng, Y., Perdikaris, P.: Understanding and mitigating gradient flow pathologies in physics-informed neural networks. SIAM J. Sci. Comput. **43**(5), A3055–A3081 (2021)
15. Wang, S., Wang, H., Perdikaris, P.: On the eigenvector bias of fourier feature networks: From regression to solving multi-scale pdes with physics-informed neural networks. Comput. Methods Appl. Mech. Engrg. **384**, 113938 (2021)
16. Wang, S., Yu, X., Perdikaris, P.: When and why pinns fail to train: A neural tangent kernel perspective. J. Comput. Phys. **449**, 110768 (2022)
17. Xu, Z.Q.J., Zhang, Y., Luo, T., Xiao, Y., Ma, Z.: Frequency principle: Fourier analysis sheds light on deep neural networks. Commun. Comput. Phys. (2019)
18. Xu, Z.Q.J., Zhang, Y., Xiao, Y.: Training behavior of deep neural network in frequency domain. In: International Conference on Neural Information Processing. pp. 264–274. Springer (2019)

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36024-4_42