J. Nonlinear Var. Anal. 9 (2025), No. 1, pp. 45-64
Available online at http://jnva.biemdas.com
https://doi.org/10.23952/jnva.9.2025.1.04

PHYSICS-INFORMED FOURIER NEURAL OPERATORS: A MACHINE LEARNING METHOD FOR PARAMETRIC PARTIAL DIFFERENTIAL EQUATIONS

TAO ZHANG[1,*], HUI XIAO[1], DEBDULAL GHOSH[2]

[1]*School of Information and Mathematics, Yangtze University, Jingzhou 434023, China*
[2]*School of Applied Sciences (Mathematics), KIIT University,*
Campus-3 (Kathajori Campus), Bhubaneswar 751024, Odisha, India

Abstract. Current methods achieved reasonable success in solving short-term parametric partial differential equations (PDEs). However, solving long-term PDEs remains challenging, and existing techniques also suffer from low efficiency due to requiring finely-resolved datasets. In this paper, we propose a physics-informed Fourier neural operator (PIFNO) for parametric PDEs, which incorporates physical knowledge through regularization. The numerical PDE problem is reformulated into an unconstrained optimization task, which we solve by using an enhanced architecture that facilitates longer-term datasets. We compare PIFNO against standard FNO on three benchmark PDEs. Results demonstrate improved long-term performance with PIFNO. Moreover, PIFNO only needs coarse dataset resolution, which enhances computational efficiency.

Keywords. Discretization-invariance; Neural operators; Partial differential equations; Physic-informed operator.

1. INTRODUCTION

Nowadays, partial differential equations (PDEs) as a mathematical tool were widely used in scientific and engineering problems. The traditional methods commonly used to solve PDEs include the finite element method [1], finite difference method [2], and finite volume method [3], which rely on discretizing the space into very fine meshes. Despite significant progress in these traditional methods, there is still a need for improvement in terms of trade-off; coarse grids are fast but lack accuracy, while fine grids are high in accuracy but slow. Complex PDEs often require extremely fine grids. Thus it is challenging and time-consuming for traditional algorithms.

Due to the limitations of traditional algorithms, machine learning methods are recognized as a potential tool for PDEs. Dissanayake and Phan-Thien [4] first considered a numerical method. It can only be solved on extremely tiny datasets and linear problems. Lagaris, Likas, and Fotiadis [5] discussed a method for nonlinear PDEs. As the dimensionality increases, both memory and computation time requirements become intractable. Han, Jentzen, and E [6] proposed the deep BSDE method with a proper loss function to optimize parameters only for the specific PDEs with general high-dimensional parabolic. Consequently, Raissi, Perdikaris, and Karniadakis in [7] proposed a

*Corresponding author.
E-mail address: tzhang@yangtzeu.edu.cn (T. Zhang).
Received 12 June 2023; Accepted 10 March 2024; Published online 20 December 2024.

physics-informed neural network (PINN) that uses physics equations as an operational constraint, which makes the final training result approximately satisfy the laws of physics. Along the line of [7], E and Yu [8] presented the deep Ritz method for variational problems, which has made considerable progress. In recent years, the PINNs method gradually become a research hotspot at the cross-discipline of machine learning and computational mathematics, as well as yields relatively deeper variants, such as cPINNs [9] (conserved PINNs) and vPINNS [10] (variational PINNs), etc. However, most of the previous work using PINNs is restricted to solving only one PDE, which results from learning to map from vectors to vectors. In a word, this method has some problems in practical applications, with slow training or even non-convergence and low accuracy. The reason is that a new network needs to be retrained once the parameters of the equations change, for example, boundary conditions, the source terms, or other parameters in the PDEs.

Furthermore, in some cases, we have to obtain quickly available accurate solutions for the set of PDEs. One approach that makes such problems feasible is the neuron operator method, which is based on the fact that the set of PDEs can be learned by a map from the source function to the solution function [11]. Such a method can only be solved at a fixed point. Therefore, Lu, Jin, and Karniadakis [12] proposed the DeepOnet method, in which the source term is transmitted to the network at fixed grid points. However, the output can be predicted at any point by adding additional input points to the network. Nevertheless, this method is often inefficient or difficult to handle owing to the need for particularly large and precise sampling points. Kovachki et al. [13] proposed neural operator methods, which can be evaluated at any point in time and any space. They summarized and proved a universal approximation theorem of the neural operator. On the basis of [13], Li et al. [14] proposed Fourier neural operators. The difference lies in integral operators working directly in Fourier space by parameterizing. It is obvious that the neural operator method is a completely data-driven approach that requires a lot of fine datasets. However, when analyzing complex physical, biological, or engineering systems, the expense of data acquisition often poses a barrier, leading us to confront the challenge of drawing long-term conclusions with incomplete information. Therefore, these methods above are inefficient.

In this paper, we propose PIFNO for parametric PDEs. Over-parameterization is recognized as one of the fundamental components of architecture design in machine learning [15] crucial for performance [16] and optimization [17]. Therefore, the PIFNO architecture employs an optimization technique that joins the physical information as regularization. In the proposed approach, the model function is written as the sum of two terms: the first term satisfies the laws of physics (initial/boundary conditions and equation) that contain no adjustable parameters, and the second term involves a feed-forward neural operator to be trained, containing adjustable parameters. Compared with the above methods, we naturally encode an underlying physical law as prior information, which is fed into the training of the neural network as regularization, thus contributing to solutions in the long term that are consistent with our experimental results.

The rest of the paper is organized as follows. Some definitions and properties of operator learning and PINNs are reviewed in Section 2. In Section 3, we propose a Fourier operator learning method based on the integration of physical information. The numerical test examples of PDEs in the linear and non-linear cases are reported in Section 4. In Section 5, we provide a mathematical proof of the technical results presented in the previous sections. Section 6 concludes the paper.

2. PRELIMINARIES

2.1. **General parametric PDEs.** The generic form of a family of parametric PDEs can be written as follows:

$$\begin{aligned} (\mathbf{L}_a u)(x) &= f(x), \quad x \in D, \\ u(x) &= 0, \qquad\quad x \in \partial D, \end{aligned} \tag{2.1}$$

where the open set $\mathbf{A} = (D; \mathbb{R}^{d\mathbf{A}})$ and $\mathbf{U} = (D; \mathbb{R}^{d\mathbf{U}})$ denote input and output function spaces. The differential operator L_a depends on $a \in \mathbf{A}$, and the fixed function $f \in \mathbf{U}^*$ is determined by the structure of L_a on a bounded domain $D \subset \mathbb{R}^d$. It is assumed that

(i) the solution $u : D \to \mathbb{R}$ of problem (2.1) lies in the Banach space $\mathbf{U}$,
(ii) the operator L_a is a mapping from the parameter Banach space $\mathbf{A}$ to the space of linear operators (possibly unbounded) that map $\mathbf{U}$ to its dual $\mathbf{U}^*$, and
(iii) the problem (2.1) is well-posed.

Note that although the parametric PDEs in (2.1) are linear, but L_a's are not linear.

2.2. **Problem setting.** Our objective is to understand how to establish a mapping between two infinite-dimensional spaces by using only a finite collection of observations of input-output pairs derived from this mapping. We aim to define this problem with clarity in the following way.

Let the parameter space $\Theta \subseteq \mathbb{R}^p$ be finite-dimensional. Assume that we have a set of N data points, $\{a^{(i)}, u^{(i)}\}_{i=1}^N$, where the observations $a^{(i)} \sim \mu$ are i.i.d.. We aim to train a neural operator, denoted as $\mathbf{F}^\dagger$, which maps from the space $\mathbf{A}$ to the space $\mathbf{U}$. This operator is parameterized by θ. The map $a \mapsto u$ with $u^{(i)} = \mathbf{F}^\dagger(a^{(i)})$ is potentially with noise. The objective is to discover a finite-dimensional approximation, denoted as $\mathbf{F}_\theta : \mathbf{A} \to \mathbf{U}$, which is parameterized by $\theta \in \Theta$. This approximation is intended to satisfy the condition

$$\mathbf{F}_\theta \approx \mathbf{F}^\dagger, \text{ almost everywhere with respect to } \mu.$$

We frame this problem as an optimization problem with the objective function as $MSE_{\mathbf{F}}$, which represents the loss functional $\mathscr{C} : \mathbf{U} \times \mathbf{U} \to \mathbb{R}$. We adopt the squared-error loss within a suitable norm $||\cdot||_{\mathbf{U}}$ on $\mathbf{U}$ and approximate the expectation using the training data:

$$MSE_{\mathbf{F}} = \min_{\theta \in \mathbb{R}^p} \mathbb{E}_{a\sim\mu} \left\| \mathbf{F}^\dagger(a) - \mathbf{F}_\theta(a) \right\|_{\mathbf{U}}^2 \approx \min_{\theta \in \mathbb{R}^p} \frac{1}{N} \sum_{i=1}^N \left\| u^{(i)} - \mathbf{F}_\theta\left(a^{(i)}\right) \right\|_{\mathbf{U}}^2.$$

This formulation closely mirrors the classical finite-dimensional scenario [18].

3. ALGORITHM

In this section, we introduce the PIFNO, which comprises two components. The first satisfies the initial or boundary conditions and physics equations with no adjustable parameters. The second involves a Fourier neural operator containing tunable parameters. By incorporating physics priors into the Fourier operators, PIFNO integrates physical knowledge.

Definition 3.1. (*Fourier integral operator* κ). If $\mathbf{L}_a$ in equation (2.1) exhibits uniform ellipticity, the Green's function formula suggests that $u(x) = \int_D G_a(x,y) f(y) dy$. Given that G_a remains continuous for all points $x \neq y$, it is reasonable to represent the integral operator's action by using

$\kappa(\phi)$, where ϕ denotes the parameters. The following operator κ is called the Fourier integral operator:

$$(\kappa(\phi)v)(x) = \int_D \kappa_\phi(x,y,a(x),a(y))v(y)dy \quad \forall x \in D.$$

To simplify matters, we assume that $D = \mathbb{T}^d$ represents the unit torus and κ is periodic so that it admits a Fourier series expansion. Let $\mathscr{F} : D \to \mathbb{R}^{d_v}$ denote the Fourier transform, with $\mathscr{F}^{-1}$ as its inverse. By defining $\kappa(x,y) = \kappa(x-y)$ and employing the convolution theorem, we establish

$$(\kappa(\phi)v)(x) = \mathscr{F}^{-1}(\mathscr{F}(\kappa)\cdot\mathscr{F}(v))(x) \text{ for all } x \in D.$$

Subsequently, we introduce the Fourier integral operator

$$(\kappa(\phi)v)(x) = \mathscr{F}^{-1}(R_\phi \cdot \mathscr{F}(v))(x) \text{ for all } x \in D.$$

Here, R_ϕ signifies the Fourier transform of $\kappa : D \to \mathbb{C}^{d_v \times d_v}$, with $\phi \in \mathbb{R}^p$ serving as its parameterization.

Definition 3.2. (*Pre-training PFNO*). Pre-training to obtain initialized weights and the shared parameters between the initial or boundary conditions and Fourier neural operators can be acquired by minimization of the mean squared error loss

$$L = MSE_{\mathbf{F}} + \lambda MSE_{bc/initial},$$

where

$$MSE_{bc/initial} = \frac{1}{N_f}\sum_{i=1}^{N_f} |u(t_0^i, x_0^i) - u_0^i|^2 \text{ and } MSE_{\mathbf{F}} = \frac{1}{N_{\mathbf{F}}}\sum_{i=1}^{N_{\mathbf{F}}} |\mathbf{F}^\dagger(a_0^i) - \mathbf{F}_\theta(a_0^i)|.$$

In this work, $\{t_0^i, x_0^i, a_0^i, u_0^i\}_{i=1}^N$ represents the initial and boundary training datasets. The loss term $MSE_{bc/initial}$ corresponds to constraints imposed by the initial data and boundary data, while MSE_G imposes the structure of the Fourier neural operator.

Definition 3.3. (*Fine-tuning PFNO*). Frozen the network structure and inherited the network weight parameters to obtain a well-initialized model as described previously. This facilitates accelerated convergence. We then fine-tune the trained network by feeding a larger dataset and modifying the loss function through regularization adjustment:

$$L = MSE_{\mathbf{F}} + \lambda' MSE_f$$

where

$$MSE_f = \frac{1}{N_u}\sum_{i=1}^{N_u} |f(t_f^i, x_f^i)|^2 \text{ and } MSE_{\mathbf{F}} = \frac{1}{N_{\mathbf{F}}}\sum_{i=1}^{N_{\mathbf{F}}} |\mathbf{F}^\dagger(a^i) - \mathbf{F}_\theta(a^i)|.$$

Here, the set $\{t_f^i, x_f^i\}_{i=1}^N$ represents the collocation points for $f(t,x)$, with the loss function MSE_f enforcing the structural constraints imposed by equations.

We proceed next by delineating the PIFNO framework. We presuppose that the input functions denoted as $a \in \mathbf{A}$, are $\mathbb{R}^{d_a}$-valued and described within the bounded domain $D \subset \mathbb{R}^d$. Concurrently, the output functions, represented as $u \in \mathbf{U}$, are $\mathbb{R}^{d_u}$-valued and defined within the bounded domain

$D' \subset \mathbb{R}^{d'}$. The proposed architecture $\mathbf{F}_\theta : \mathbf{A} \to \mathbf{U}$ encompasses the following overarching structure:

Algorithm 1: Structure for physics-informed Fourier neural operator learning

Input: Provide the functions $\{a_j\}_{j=1}^N$

Output: Output the functions $\{u_j\}_{j=1}^N$

```
for i = 0, 1, 2, ..., do
    Compute MSE_F + λ MSE_{bc/initial} on boundary and initial points on (a_0^i, u_0^i);
    Update Fourier neural operator for pre-traning;
    Compute λ' MSE_f + MSE_F, fine-tune Fourier neural operator;
    for j = 1 to K do
        From the distribution, sample up a';
        Compute MSE_f + MSE_F on a';
        Update Fourier neural operator;
    end
end
```

Step 1. ***Initialized solution.*** Upon receiving an input function $a : D \to \mathbb{R}^{d_{\mathbf{A}}}$, we apply a point-wise operator P to a, yielding the computation of v_0:

$$v_0(x) = P(x, a(x), a_\varepsilon(x), \nabla a_\varepsilon(x)) + p.$$

Here, P is a matrix in $\mathbb{R}^{n \times 2(d+1)}$, parameterized by the function $P_\theta : \mathbb{R}^{d_{\mathbf{A}}} \to \mathbb{R}^{d_0}$, and p is a vector in $\mathbb{R}^n$. We augment the initialization $(x, a(x))$ with a Gaussian-smoothed version of the coefficients $a_\varepsilon(x)$, alongside their gradients $\nabla a_\varepsilon(x)$. This augmentation results in a $2(d+1)$-dimensional vector field at initialization. Typically, we opt for $d_{\mathbf{A}} \ll d_{v_0}$, making this a lifting operation executed by a fully local operator.

Step 2. ***Iterative update.*** We apply a sequence of nonlinear integral Fourier operators $(\kappa(\phi)v)(x) = \mathscr{F}^{-1}\left(R_t \cdot \mathscr{F} v_t\right)(x)$

$$\mathbf{F}_i : \{v_t : D_t \to \mathbb{R}^{d_{v_t}}\} \to \{v_{t+1} : D_{t+1} \to \mathbb{R}^{d_{v_{t+1}}}\}, \quad t \in \{0, 1, 2, ..., T\}$$

to update v_t:

$$v_{t+1}(x) = \sigma(W v_t(x) + \mathscr{F}^{-1}\left(R_t(k) \cdot \mathscr{F} v_t\right)(x)),$$

where σ is an activation function *GELU* [20], $R_t : \mathbb{Z}^d \to \mathbb{C}^{d_{t+1} \times d_t}$ is a linear transform which can work on lower Fourier modes and filter out higher modes, and $W \in \mathbb{R}^{n \times n}$ and ϕ entering kernel $\kappa_\phi : \mathbb{R}^{2(d+1)} \to \mathbb{R}^{n \times n}$, which are to be learned from data. Let $D_t \subset \mathbb{R}^d$ be a measurable set accompanied by a measure μ_i. In this context, we establish $D_0 = D$ and $D_T = D'$, with the stipulation that $D_t \subset \mathbb{R}^{d_t}$ represents a bounded domain.

Step 3. ***Solution function.*** Computing v_T for the T layer operator, we compute the last hidden representation $u : D' \to \mathbb{R}^{d_{\mathbf{U}}}$ by a fully local operator

$$u(x) = Q v_T(x) + q.$$

The Q is parameterized with a function $Q_\theta : \mathbb{R}^{d_{v_T}} \to \mathbb{R}^{d_{\mathbf{U}}}$, where $v_t(x) \in \mathbb{R}^n$ and $Q \in \mathbb{R}^{1\times n}$, $q \in \mathbb{R}$. Typically, $d_{v_T} \gg d_{\mathbf{U}}$.

Step 4. ***Initialized parameters.*** We initialize the neural network based on the initial conditions and boundary conditions. The loss function is as the following

$$L = MSE_{\mathbf{F}} + \lambda MSE_{bc/initial},$$

where $\lambda \geq 0$. Based on the properties of the PDEs, we select the boundary points (x at 0 or 1) and initial points $u(0)$ for training. This subset is more tractable to handle and effectively reduces training costs.

Step 5. ***Regularization.*** In order to impose physical constraints, the equations are appended to the loss function as a regularization term. The network architecture and parameters initialized in Steps 1–4 remain the same. Steps 1–3 and 5 are iterated to optimize further the parameters shared between the network and the equations by minimizing the loss function:

$$L = MSE_{\mathbf{F}} + \lambda' MSE_f,$$

where $\lambda' \geq 0$. Specifically, the term in MSE_f acts as a regularizer that penalizes solutions violating the physics equations and enforces the structure imposed by the equations at the collocation points.

The entire process is sketched in Figure 1.

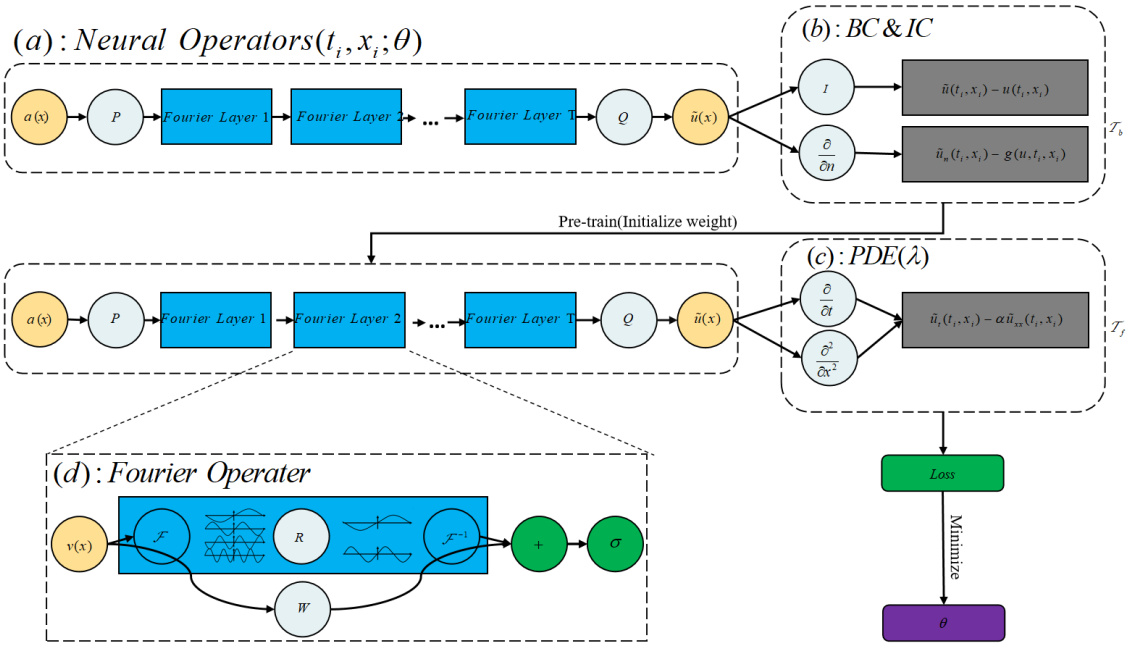

FIGURE 1. (a) The architecture of neural operators; (b) Informing physical boundary and initial conditions in the loss function as regularization of backpropagation; (c) Informing physical functions in the loss function as regularization of backpropagation; (d) Fourier operator.

Therefore, a key advantage of PIFNO is effective training with regular datasets, which can avoid the cost of data acquisition. In the next section, we demonstrate PIFNO on three benchmark operator learning tasks.

4. EXPRIMENTS

In this section, we present experiments on Burger's equation (1D), Darcy flow (2D), and Navier-Stokes equation (3D) to evaluate the performance of PIFNO. For each problem, we describe the setup, implementation details, and empirical results. In all experiments, we use the Adam optimizer [19] with a learning rate of 0.001 and the GeLu activation function [20], which can represent finite element method approximations [1] from an approximation theory perspective [21]. The overall PIFNO network architecture contains four layers of Fourier layers.

4.1. **Burgers' equation.**

Burgers' equation is a non-linear PDE with a Dirichlet boundary of the following form:

$$\begin{aligned} &\partial_t u(x,t) + \partial_x \left(u^2(x,t)/2\right) - \nu \partial_{xx} u(x,t) = 0, && x \in (0,1), t \in (0,1], \\ &u(x,0) = u_0(x), && x \in (0,1). \end{aligned}$$

Burgers' equation arises in various areas of applied mathematics, including traffic flow, gas dynamics, fluid dynamics, and nonlinear acoustics. Burgers' equation considers only convective effects and ignores viscous and source terms of the Navier-Stokes system. Here we study the initial value problem for Burgers' equation with periodic boundary conditions and viscosity coefficient $\nu \in \mathbb{R}_+$. Specifically, we assume the initial condition $u_0 \in L^2_{\text{per}}((0,1);\mathbb{R})$ and aim to learn the operator mapping from parameters to solutions.

Let us define $f(t,x)$ and $h(0,x_0)$ by

$$MSE_f = \partial_t u(x,t) + \partial_x \left(u^2(x,t)/2\right) - \nu \partial_{xx} u(x,t) \text{ and } MSE_{bc/initial} = u(0,x) - u_0(x).$$

The Dirichlet boundary and initial conditions are used to pre-train the model. The shared parameters between the neural operator $u(x,t)$ and $f(x,t)$ are optimized by minimizing the total loss function.

We solve the PDE on a spatial mesh with 8192 points and a time mesh with 2048 points. This high-resolution solution is used to generate datasets for training at lower resolutions. Specifically, the dataset consists of the evolution from a to u on a 2048×8192 grid, which can be downsampled to obtain lower precision training data. In this experiment, we only add a physical regularization term to the loss function to constrain the solution, laying the foundation for incorporating datasets with richer information and more complex PDE constraints in future work. The overall loss function can be written in the following form

$$L = \frac{1}{N_{\mathbf{F}}} \sum_{i=1}^{N_{\mathbf{F}}} |\mathbf{F}(t^i_{\mathbf{F}}, x^i_{\mathbf{F}}) - \mathbf{F}^i|^2 + \lambda' \frac{1}{N_f} \sum_{i=1}^{N_f} |f(t^i_f, x^i_f)|^2.$$

To train the PIFNO solution operator, we randomly sample 1000 initial conditions u_0 from the distribution μ, where μ follows a normal distribution $\mathcal{N}(0, 625(-\Delta + 25I)^{-2})$. We opt for the 1D Fourier neural operator as our foundational model, featuring four Fourier layers with 16 frequencies per channel and a width of 64. This entails a total of 1400 seconds of training on a single Nvidia GPU. PIFNO achieves an average relative L_2 error of 0.24% across 200 test instances,

showcasing an improvement over the 0.38% error observed with FNO under the same problem configuration.

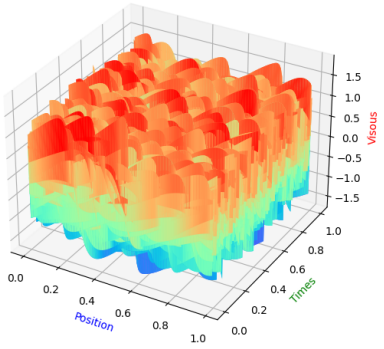

FIGURE 2. 100 solutions visulization

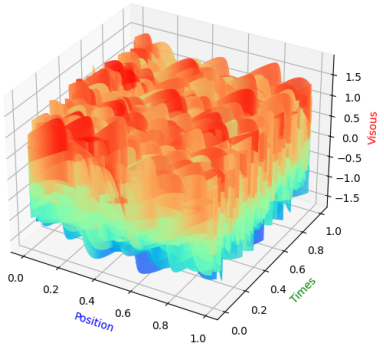

FIGURE 3. 1000 solutions visulization

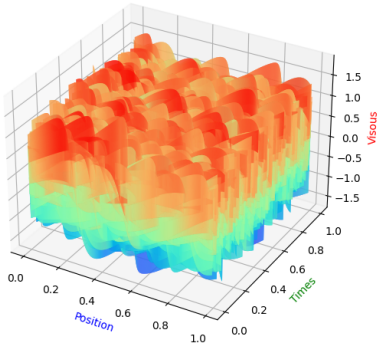

FIGURE 4. 2000 solutions visulization

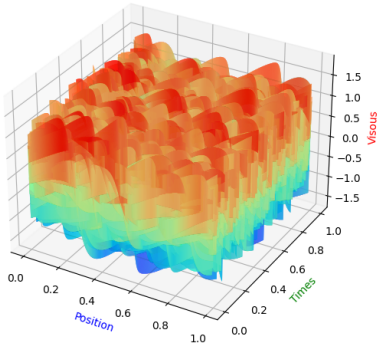

FIGURE 5. 8000 solutions visulization

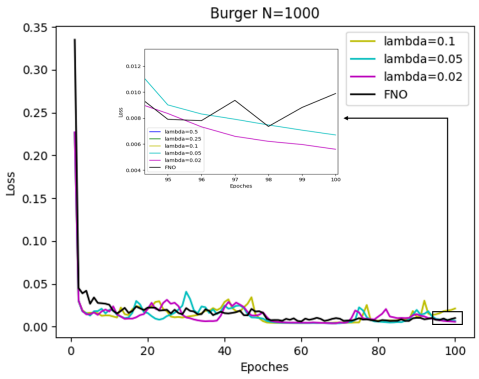

FIGURE 6. N=1000

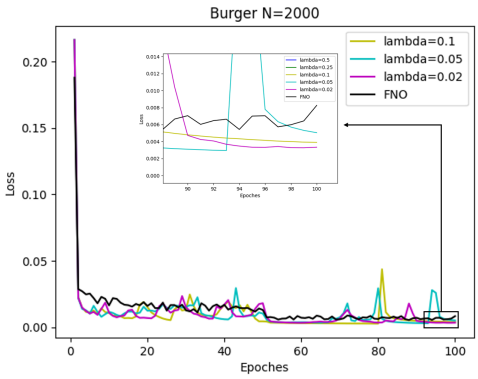

FIGURE 7. N=2000

The experiment result is as follows. Figures 2–5 show 3D visualizations of the PDEs solutions with an increasing number of data points N. We observe that the solutions become smoother as N increases. However, with the addition of physical constraints, the solutions at $N = 2000$ are not much different from $N = 8000$ even when scaled up. Figures 6–7 compare the loss for FNO and PIFNO at $N = 1000$ and $N = 2000$ data points, respectively. Except for the black FNO line, all

other lines denote PIFNO with different λ's. We found that λ between 0.02 and 0.1 gives the best performance for regular datasets. Thus, for subsequent experiments, λ is selected from this range.

4.2. **Darcy flow.** In this section, we showcase our method's ability to address boundary conditions by considering a two-dimensional steady-state Darcy flow equation, which is a linear elliptic PDE with a Dirichlet boundary condition of the following form:

$$\begin{aligned} -\nabla\cdot(a(x)\nabla u(x)) &= f(x), \quad && x\in(0,1)^2 \\ u(x) &= 0, && x\in\partial(0,1)^2. \end{aligned}$$

Here, a is the diffusion coefficient with $a\in\mathbf{A}\subseteq L^\infty(D;\mathbb{R}_+)$, $u\in\mathbf{U}\subseteq H_0^1(D;\mathbb{R})$ is the solution, and $f\in\mathbf{F}=H^{-1}(D;\mathbb{R})$ is the forcing function. The spatial domain is taken to be $D=(0,1)^2$. This PDE has wide-ranging applications, including simulation of groundwater flow, geothermal system modeling, oil reservoir permeability, and engineering seepage analysis.

Let us define $f(t,x)$ and the loss function to be given as follows

$$MSE_f = -\nabla\cdot(a(x)\nabla u(x)) - f(x)$$

$$\text{and } L=\frac{1}{N_{\mathbf{F}}}\sum_{i=1}^{N_{\mathbf{F}}}|\mathbf{F}(t_{\mathbf{F}}^i,x_{\mathbf{F}}^i)-\mathbf{F}^i|^2+\lambda'\frac{1}{N_f}\sum_{i=1}^{N_f}|f(t_f^i,x_f^i)|^2+\lambda\frac{1}{N_0}\sum_{i=1}^{N_0}|u(0,x_0^i)-u_0^i)|^2.$$

To prepare the dataset, we define $\mu\sim\mathcal{N}\left(0,(-\Delta+9I)^{-2}\right)$ as a probability measure incorporating zero Neumann boundary conditions on the Laplacian, where for all $x\geq 0$, $\psi(x)=12$ and for all $x<0,\psi(x)=3$. The forcing function is fixed at $f(x)=1$. The coefficient $a(x)$ is sampled from μ, and the solutions u's are obtained using the traditional finite element method on 241×241 and 421×421 grids in Matlab. Datasets of different resolutions can be generated by downsampling. We parameterize the 2D Fourier neural operator with four Fourier layers, each comprising 20 frequencies per channel and a width of 64.

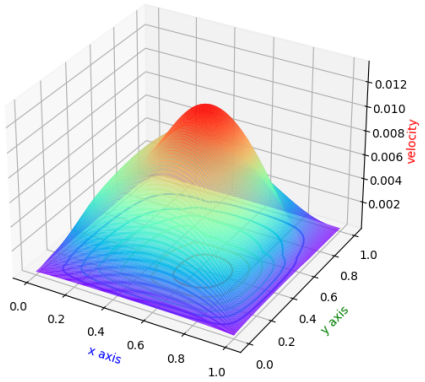

FIGURE 8. ($s=241$) visualization of one solution

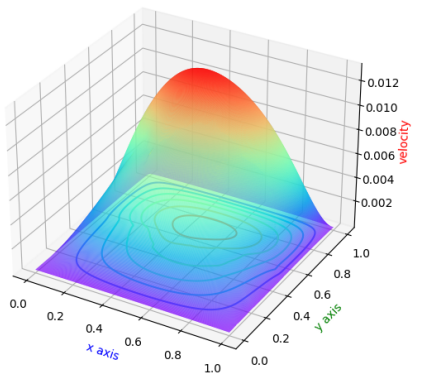

FIGURE 9. ($s=421$) visualization of one solution

Figures 8–9 show 3D visualizations of a solution to the equation. To clearly illustrate the solution, we have chosen to visualize a single equation rather than a series, unlike the multiple equations shown above. Figures 10–11 compare the experimental loss values for FNO and PIFNO on datasets with resolutions of 241×241 and 421×421, respectively. The black line indicates FNO and the purple line indicates PIFNO. Zooming in, we see that PIFNO achieves lower and

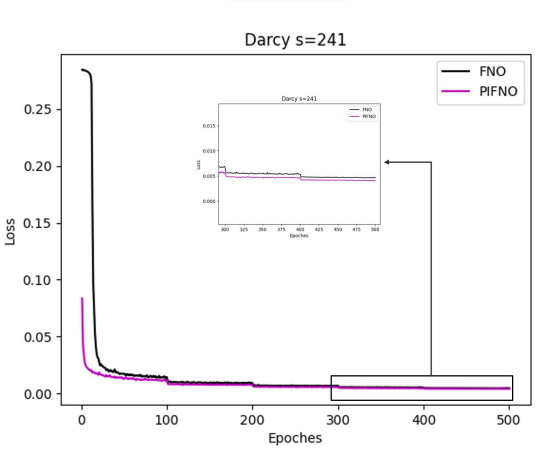

FIGURE 10. Loss values of FNO and PIFNO for $s = 241 \times 241$

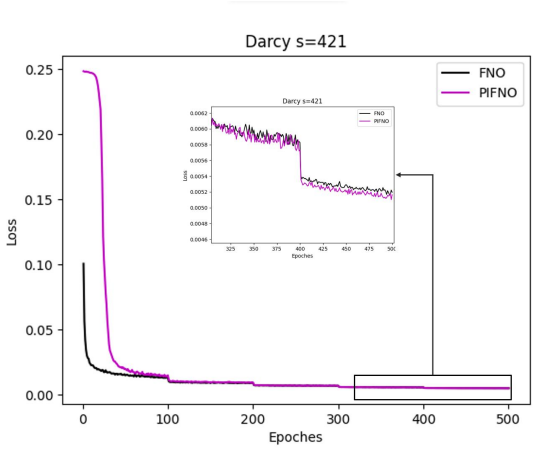

FIGURE 11. Loss values of FNO and PIFNO for $s = 421 \times 421$

more stable loss long term. The solution operator $\mathbf{F}_\theta$ maps a diffusion coefficient function a to the corresponding solution u.

4.3. **Navier-Stokes equation.** In this section, we assess the accuracy and efficiency of the proposed technique over time. We examine the following 2D Navier-Stokes equations for a viscous, incompressible fluid in vorticity form on the unit torus. The system is described as follows:

$$\begin{aligned}
&\partial_t w(x,t) + u(x,t)\cdot\nabla w(x,t) = \nu\Delta w(x,t) + f(x), && x\in(0,1)^2,\ t\in(0,T]\\
&\nabla\cdot u(x,t) = 0, && x\in(0,1)^2,\ t\in[0,T]\\
&w(x,0) = w_0(x), && x\in(0,1)^2.
\end{aligned}$$

The Navier-Stokes equation encapsulates fundamental laws governing viscous fluid flows and is of significant importance in fluid mechanics. Here, w denotes the vorticity field, $w_0(x)\in(0,1)^2$ is the initial vorticity, ν is the viscosity coefficient, and f is the forcing function.

To generate the dataset, initial vorticity conditions w_0 were randomly sampled from the distribution $\mu = \mathcal{N}(0, 7^{3/2}(-\Delta+49I)^{-2.5})$ with periodic boundary conditions. The forcing function f was fixed to $f(x) = 0.1(\sin(2\pi(x_1+x_2)) + \cos(\pi(x_1+x_2)))$. Data was generated on a 256×256 grid, which could then be downsampled for low-resolution training and high-resolution testing.

Let us define $f(t,x)$, $h(t,x)$ and loss function to be given as follows

$$\begin{aligned}
&MSE_f = \partial_t w(x,t) + u(x,t)\cdot\nabla w(x,t) - \nu\Delta w(x,t) + f(x),\\
&MSE_{bc/initial} = \nabla\cdot u(x,t) \ \text{ and}\\
&L = \frac{1}{N_{\mathbf{F}}}\sum_{i=1}^{N_{\mathbf{F}}}|\mathbf{F}(t^i_{\mathbf{F}},x^i_{\mathbf{F}}) - \mathbf{F}^i|^2 + \lambda\frac{1}{N_f}\sum_{i=1}^{N_f}|f(t^i_f,x^i_f)|^2 + \lambda'\frac{1}{N_0}\sum_{i=1}^{N_0}|u(0,x^i_0) - u^i_0)|^2.
\end{aligned}$$

Our aim is to train an operator capable of transforming the vorticity field from an initial time interval $[0,T_{in}]$ to a subsequent contrasting time interval $(T_{in},T]$. Notably, we trained PIFNO on low-resolution datasets and tested it on high-resolution. The details are presented in the following.

t=0.png

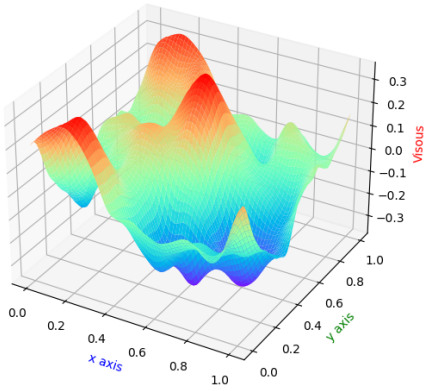

FIGURE 12. Solution for $T = 0$, $s = 64 \times 64$

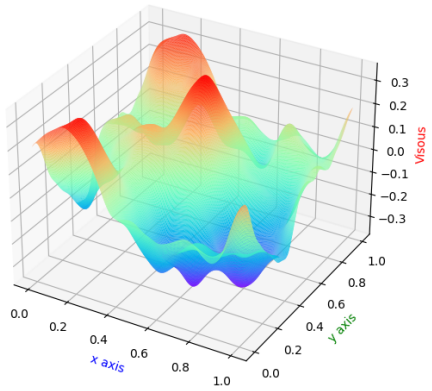

FIGURE 13. Solution for $T = 0$, $s = 256 \times 256$

t=10.png

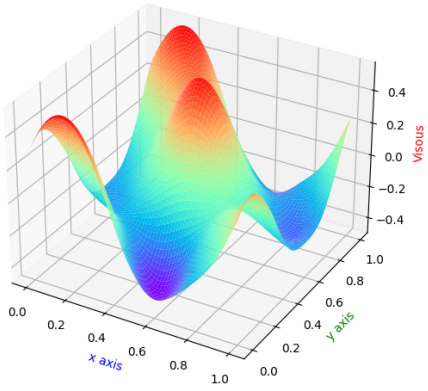

FIGURE 14. Solution for $T = 10$, $s = 64 \times 64$

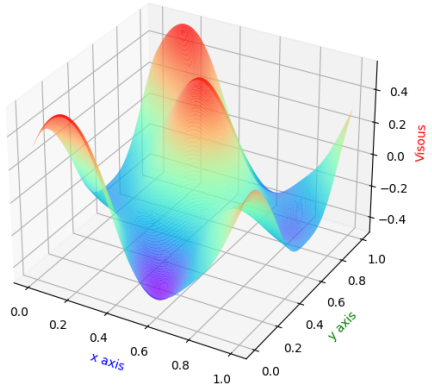

FIGURE 15. Solution for $T = 10$, $s = 256 \times 256$

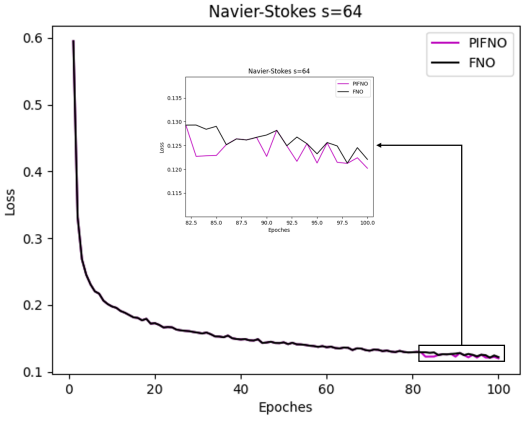

FIGURE 16. Loss values of FNO and PIFNO for $s = 64 \times 64$

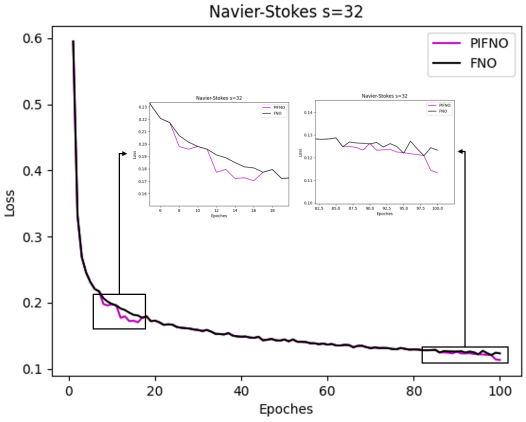

FIGURE 17. Loss values of FNO and PIFNO for $s = 32 \times 32$

Figures 12–13 show 3D visualizations of a solution to the equation at low and high resolutions, respectively, at the initial time. To clearly illustrate the solution, we have chosen to visualize a single equation rather than a series, unlike the multiple equations shown in Figures 2–5. The lower and higher resolutions were selected to demonstrate the behavior across scales. Figures 14–15 similarly show 3D visualizations at low and high resolution but at the final time rather than the initial. As before, a single equation is visualized for clarity rather than a series.

Figures 16–17 then compare the experimental loss values for FNO and PIFNO at dataset resolutions of 64×64 and 32×32, respectively. The black line indicates FNO, while PIFNO is in purple. At low resolution, PIFNO is observed to converge faster initially and achieve better long-term performance. FNO and PIFNO converge simultaneously at high resolution, but PIFNO retains superior long-term performance.

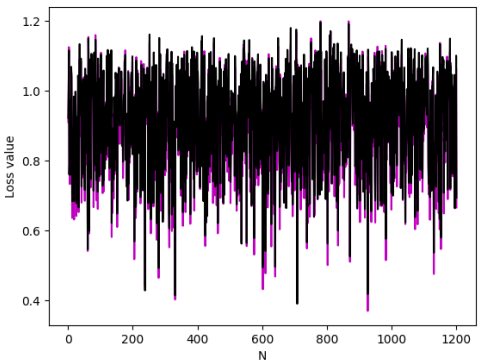

FIGURE 18. Loss value for $T = 20, s = 64, N = 1200$

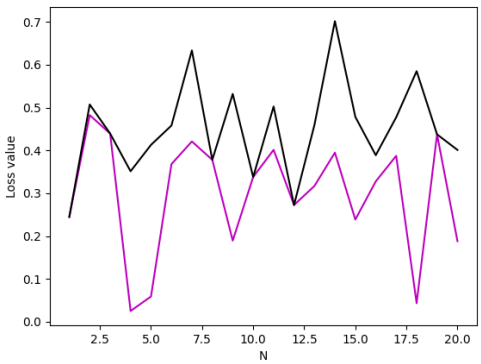

FIGURE 19. Loss value for $T = 50, s = 256, N = 20$

In Figure 18, we test both models on a dataset with a size 1200, a resolution of 256, and a time span of 20 seconds. The results demonstrate that PIFNO achieves lower or comparable loss versus FNO in most cases, highlighting its advantages over the baseline. We further challenge both models on a longer time series in Figure 19, using data of size 20, resolution 256, and time 50 seconds. PIFNO starts to significantly outperform FNO for long-time prediction tasks. Moreover, we train both models on low-resolution data but test on high-resolution counterparts. The superior performance of PIFNO thus indicates its ability to retain resolution invariance, a highly desirable property for robust generalization. In summary, our quantitative experiments with varied dataset complexities reveal PIFNO's capabilities for accurate long-term forecasting. The consistent improvements over FNO verify PIFNO's efficacy in extracting invariant physical features in a resolution-preserving manner. These results corroborate the design of PIFNO and highlight its potential for tackling real-world complex systems. Further rigorous benchmarking on large-scale problems will be conducted as future work to fully leverage the strengths of the proposed architecture.

5. CONVERGENCE ANALYSIS OF THE ALGORITHM

5.1. **Settings.** We consider a fixed spatial dimension $d \in \mathbb{N}$, where $D \subset \mathbb{R}^d$ denotes a domain in $\mathbb{R}^d$. Our focus lies in approximating operators $\mathbf{F}_\theta : \mathbf{A}\left(D; \mathbb{R}^{d_a}\right) \to \mathbf{U}\left(D; \mathbb{R}^{d_u}\right)$, mapping a to

$u := \mathbf{F}_\theta(a)$. Here, the input $a \in \mathbf{A}\left(D;\mathbb{R}^{d_a}\right)$, $d_a \in \mathbb{N}$, represents a function $a : D \to \mathbb{R}^{d_a}$ with d_a components, and the output $u \in \mathbf{U}\left(D;\mathbb{R}^{d_u}\right)$, $d_u \in \mathbb{N}$, signifies a function $u : D \to \mathbb{R}^{d_u}$ with d_u components. The spaces $\mathbf{A}\left(D;\mathbb{R}^{d_a}\right)$ and $\mathbf{U}\left(D;\mathbb{R}^{d_u}\right)$ are Banach spaces, or suitable subsets thereof. Typical examples of $\mathbf{A}$ and $\mathbf{U}$ encompass the space of continuous functions $C\left(D;\mathbb{R}^{d_u}\right)$, or Sobolev spaces $H^s\left(D;\mathbb{R}^{d_u}\right)$ with $s \geq 0$.

5.2. **Universal approximation.** We demonstrate that PIFNO possesses universality. Within the framework outlined earlier, one can discover a PIFNO capable of approximating a broad array of operators to the desired level of precision. To elucidate further, we present the following theorem.

Theorem 5.1. (Universal approximation). *Let $s, s' \geq 0$. Consider $\mathbf{F}_\theta : H^s\left(\mathbb{T}^d;\mathbb{R}^{d_a}\right) \to H^{s'}\left(\mathbb{T}^d;\mathbb{R}^{d_u}\right)$ as a continuous operator. Let $K \subset H^s\left(\mathbb{T}^d;\mathbb{R}^{d_a}\right)$ be a compact subset. For any $\varepsilon > 0$, there exists a PIFNO $\mathbf{F}^\dagger : H^s\left(\mathbb{T}^d;\mathbb{R}^{d_a}\right) \to H^{s'}\left(\mathbb{T}^d;\mathbb{R}^{d_u}\right)$, continuous as an operator $H^s \to H^{s'}$, satisfying*

$$\sup_{a\in K} \|\mathbf{F}_\theta(a) - \mathbf{F}^\dagger(a)\|_{H^{s'}} \leq \varepsilon.$$

Sketch of the proof: The comprehensive proof of this universal approximation theorem is furnished in Subsection 5.3, and we provide an outline here. For the sake of simplicity in notation, we set $d_a = d_u = 1$.

Lemma 5.1. *Suppose the universal approximation Theorem* 5.1 *holds for $s' = 0$. Then, it extends to arbitrary $s' \geq 0$.*

The primary aim is to establish Theorem 5.1 for the specific scenario of $s' = 0$. In other words, given a continuous operator $\mathbf{F}_\theta : H^s\left(\mathbb{T}^d\right) \to L^2\left(\mathbb{T}^d\right)$, where $K \subset H^s\left(\mathbb{T}^d\right)$ is compact and $\varepsilon > 0$, our goal is to construct a PIFNO $\mathbf{F}^\dagger : H^s\left(\mathbb{T}^d\right) \to L^2\left(\mathbb{T}^d\right)$ such that $\sup_{a\in K} \|\mathbf{F}_\theta(a) - \mathbf{F}^\dagger(a)\|_{H^{s'}} \leq \varepsilon$. To begin, we define the operator

$$\mathbf{F}_N : H^s\left(\mathbb{T}^d\right) \to L^2\left(\mathbb{T}^d\right), \quad \mathbf{F}_N(a) := P_N \mathbf{F}\left(P_N a\right),$$

where P_N is the operator. In essence, $\mathbf{F}_N$ loosely represents the Fourier projection of the continuous operator $\mathbf{F}_\theta$. Furthermore, we can demonstrate that for any given $\varepsilon > 0$, there exists $N \in \mathbb{N}$ such that

$$\|\mathbf{F}_\theta(a) - \mathbf{F}_N(a)\|_{L^2} \leq \varepsilon \quad \forall a \in K.$$

Thus, the crux of the proof lies in identifying a PIFNO capable of approximating the operator $\mathbf{F}_N$ with any desired level of accuracy.

5.3. **Proofs and technical details.** The proof of Lemma 5.1 depends on the subsequent technical lemma.

Lemma 5.2. *Give $s' \geq 0$ and $N \in \mathbb{N}$. Let $K \subset H^{s'}$ be compact. Suppose $\sigma \in C^m$, where $m > s'$ is an integer. Then, for any $\varepsilon > 0$, there exists a single-layer PIFNO $\mathscr{L} : H^{s'} \to H^{s'}$ such that*

$$\sup_{v\in K} \|P_N v - \mathscr{L}(v)\|_{H^{s'}} \leq \varepsilon.$$

Proof. Initially, we acknowledge that the Fourier projection $P_N : H^{s'} \to H^{s'}$ is a continuous operator. Consequently, the image $P_N K \subset H^{s'}$ is compact. Moreover, P_N maps into a finite-dimensional

subspace of $H^{s'}$. As a result of the norm equivalence on finite-dimensional spaces, there exists $C_0 = C_0(N,K) > 0$ such that

$$\sup_{v\in K} \|P_N v\|_{L^\infty} \le C_0 \quad \text{and} \quad \sup_{v\in K} \|P_N v\|_{H^m} \le C_0.$$

Let $x_0 \in \mathbb{R}$ be such that $\sigma'(x_0) \neq 0$. For $h > 0$, we define

$$\psi_h(x) := \frac{\sigma(x_0 + hx) - \sigma(x_0 - hx)}{2h\sigma'(x_0)}.$$

It is easily verified that $\psi_h \in C^m$, and there exists a constant $C_1 = C_1(\sigma, C_0) > 0$ such that

$$\|\psi_h\|_{C^m([-C_0,C_0])} \le C_1 \quad \forall h \in (0,1].$$

Moreover, by Taylor expansion, we obtain

$$|\psi_h(x) - x| \le Ch, \quad \forall x \in [-C_0, C_0] \ \forall h \in (0,1].$$

Applying the composition rule for Sobolev functions, we deduce that $\psi_h \circ P_N a \in H^m$ for $P_N a \in H^m$. Additionally, there exists a constant $C_2 = C_2(C_1, C_0) > 0$ such that

$$\|\psi_h(P_N v)\|_{H^m} \le C_2 \quad \forall v \in K. \tag{5.1}$$

We note that the mapping $v \mapsto \mathscr{L}_h(v) := \psi_h(P_N v)$ can be succinctly represented by a single-layer PIFNO. By equation (5.1), we have $\|\mathscr{L}_h(v)\|_{H^m} \le C_2$ for all $v \in K$. Applying the interpolation inequality between Sobolev spaces, we deduce that

$$\|\mathscr{L}_h(v) - P_N v\|_{H^{s'}} \le \|\mathscr{L}_h(v) - P_N v\|_{L^2}^{\theta} \|\mathscr{L}_h(v) - P_N v\|_{H^m}^{1-\theta},$$

independently of $h > 0$. Utilizing equations (5.4) and (5.1), we obtain

$$\|\mathscr{L}_h(v) - P_N v\|_{L^2} = \|\psi_h(P_N v) - P_N v\|_{L^2} \le Ch$$

where $C > 0$ is a constant independent of h and $v \in K$. Consequently, we conclude that

$$\|\mathscr{L}_h(v) - P_N v\|_{H^{s'}} \le Ch^{\theta} \to 0,$$

as $h \to 0$, where $C > 0$ is a constant independent of h. This concludes the proof. □

Proof of Lemma 5.1.

Proof. Let $\mathbf{F}_\theta : H^s \to H^{s'}$ represent a continuous operator, and consider $K \subset H^s$ as a compact set. Assuming the universal approximation capability of FNOs for operators $H^s \to L^2$, we aim to establish the existence of a PIFNO approximation $\mathbf{F} : H^s \to H^{s'}$ within an accuracy of $\varepsilon > 0$. Initially, we observe that due to the compactness of $\mathbf{F}(K) \subset H^{s'}$, there exists $N \in \mathbb{N}$ such that

$$\sup_{a\in K} \|\mathbf{F}_\theta(a) - P_N \mathbf{F}_\theta(a)\|_{H^{s'}} \le \varepsilon/2$$

Let us set $\delta > 0$ for now, with the specific value of δ to be determined later in this proof. According to the assumption on the universal approximation of operators $H^s \to L^2$, there exists a PIFNO $\widetilde{\mathbf{F}^\dagger} : H^s \to L^2$, continuously operating as $H^s \to L^2$, satisfying

$$\sup_{a\in K} \left\| P_N \mathbf{F}_\theta(a) - \tilde{\mathbf{F}}^\dagger(a) \right\|_{L^2} \le \delta.$$

One challenge in the current setup is that there is no assurance that $\tilde{\mathscr{N}}$ defines a mapping $H^s \to H^{s'}$, particularly when $s' > s$. To address this limitation, we incorporate an additional FNO layer $\widetilde{\mathscr{L}} : L^2 \to H^{s'}$. Leveraging Lemma 5.2, we obtain a single-layer FNO $v \mapsto \widetilde{\mathscr{L}}(v)$, which satisfies the equation $\widetilde{\mathscr{L}}(v) = \widetilde{\mathscr{L}}(P_N v)$ for all v, and establishes a continuous operator $H^{s'} \to H^{s'}$. This operator ensures that

$$\sup_{v \in K'} \left\| P_N v - \widetilde{\mathscr{L}}(v) \right\|_{H^{s'}} \leq \delta \tag{5.2}$$

where $K' := P_N \tilde{\mathbf{F}}^\dagger(K) \subset H^{s'}$ is a compact subset of $H^{s'}$.

Next, we introduce a PIFNO denoted by $\mathbf{F}^\dagger := \widetilde{\mathscr{L}} \circ \widetilde{\mathbf{F}^\dagger} : H^s \to H^{s'}$. $\mathbf{F}^\dagger$ represents a continuous operator $H^s \to H^{s'}$, structured as the composition

$$H^s \xrightarrow{\widetilde{\mathbf{F}^\dagger}} L^2 \xrightarrow{P_N} H^{s'} \xrightarrow{\widetilde{\mathscr{L}}} H^{s'},$$

consisting of continuous operators. For any $a \in K$, we establish the following bound

$$\begin{aligned} \left\| P_N \mathbf{F}_\theta(a) - \mathbf{F}^\dagger(a) \right\|_{H^{s'}} &\leq \left\| P_N \mathbf{F}_\theta(a) - P_N \widetilde{\mathbf{F}^\dagger}(a) \right\|_{H^{s'}} + \left\| P_N \tilde{\mathbf{F}}^\dagger(a) - \mathbf{F}^\dagger(a) \right\|_{H^{s'}} \\ &\leq C N^{s'} \left\| P_N \mathbf{F}_\theta(a) - \tilde{\mathbf{F}}^\dagger(a) \right\|_{L^2} + \left\| P_N \tilde{\mathbf{F}}^\dagger(a) - \widetilde{\mathscr{L}}\left(P_N \tilde{\mathbf{F}}^\dagger(a) \right) \right\|_{H^{s'}}. \end{aligned}$$

We employ the inequality $\|P_N v\|_{H^{s'}} \leq C N^{s'} \|P_N v\|_{L^2}$ with a constant $C = C\left(\mathbb{T}^d, s'\right) > 0$, independent of N. Additionally, note that $\mathbf{F}^\dagger(a) = \widetilde{\mathscr{L}}(\widetilde{\mathbf{F}^\dagger}(a)) = \widetilde{\mathscr{L}}\left(P_N \widetilde{\mathbf{F}^\dagger}(a) \right)$. So,

$$C N^{s'} \left\| P_N \mathbf{F}_\theta(a) - \tilde{\mathbf{F}}^\dagger(a) \right\|_{L^2} \leq C N^{s'} \delta.$$

The bound (5.2) implies that $\left\| P_N \widetilde{\mathbf{F}^\dagger}(a) - \widetilde{\mathscr{L}}\left(P_N \widetilde{\mathbf{F}^\dagger}(a) \right) \right\|_{H^{s'}} = \left\| P_N v - \widetilde{\mathscr{L}}(v) \right\|_{H^{s'}} \leq \delta$, with $v := P_N \mathbf{F}^\dagger(a) \in K'$. We thus obtain

$$\left\| P_N \mathbf{F}_\theta(a) - \mathbf{F}^\dagger(a) \right\|_{H^{s'}} \leq \left(C N^{s'} + 1 \right) \delta,$$

where $C = C\left(\mathbb{T}^d, s'\right) > 0$ is independent of δ. Since $\delta > 0$ was arbitrary, we can conclude that there exists a PIFNO $\mathbf{F}^\dagger : H^s \to H^{s'}$ such that

$$\sup_{a \in K} \|\mathbf{F}_\theta(a) - \mathbf{F}^\dagger(a)\|_{H^{s'}} \leq \sup_{a \in K} \|\mathbf{F}_\theta(a) - P_N \mathbf{F}_\theta(a)\|_{H^{s'}} + \sup_{a \in K} \left\| P_N \mathbf{F}_\theta(a) - \mathbf{F}^\dagger(a) \right\|_{H^{s'}} \leq \varepsilon.$$

Hence, the result follows. □

Due to Lemma 5.1, given a continuous operator $\mathbf{F}_\theta : H^s\left(\mathbb{T}^d\right) \to L^2\left(\mathbb{T}^d\right)$, where $K \subset H^s\left(\mathbb{T}^d\right)$ is compact and $\varepsilon > 0$, our goal is to create a PIFNO $\mathbf{F}^\dagger : H^s\left(\mathbb{T}^d\right) \to L^2\left(\mathbb{T}^d\right)$. This PIFNO should satisfy $\sup_{a \in K} \|\mathbf{F}_\theta(a) - \mathbf{F}^\dagger(a)\|_{L^2} \leq \varepsilon$.

Proof of Theorem 5.1.

Proof. In this proof, we simplify notations by considering $d_a = d_u = 1$, although the general case with $d_a, d_u > 1$ follows similarly. For any $N \in \mathbb{N}$, P_N represents the orthogonal Fourier projection operator. Initially, we observe that if $K \subset H^s\left(\mathbb{T}^d\right)$ is compact, basic arguments demonstrate that the set $\widetilde{K}$, defined as $\widetilde{K} := K \cup \bigcup_{N \in \mathbb{N}} P_N K$, remains compact. As $\mathbf{F}_\theta$ is continuous, its behavior on

$\tilde{K}$ is uniformly continuous. In other words, there exists a continuity function $\omega : [0,\infty) \to [0,\infty)$, where

$$\left\|\mathbf{F}_\theta(a) - \mathbf{F}_\theta\left(a'\right)\right\|_{L^2} \leq \omega\left(\left\|a - a'\right\|_{H^s}\right),$$

for all $a, a' \in \tilde{K}$. Note that the expression $\mathbf{F}_N$ is formulated as follows:

$$\begin{aligned}
& \left\|\mathbf{F}_\theta(a) - \mathbf{F}_N(a)\right\|_{L^2} \\
\leq\ & \left\|\mathbf{F}_\theta(a) - P_N\mathbf{F}_\theta(a)\right\|_{L^2} + \left\|P_N\mathbf{F}_\theta(a) - P_N\mathbf{F}_\theta\left(P_N a\right)\right\|_{L^2} \\
\leq\ & \left\|\mathbf{F}_\theta(a) - P_N\mathbf{F}_\theta(a)\right\|_{L^2} + \left\|\mathbf{F}_\theta(a) - \mathbf{F}_\theta\left(P_N a\right)\right\|_{L^2} \\
\leq\ & \sup_{v\in\mathscr{G}(\widetilde{K})} \left\|\left(1 - P_N\right)v\right\|_{L^2} + \omega\left(\sup_{a\in\widetilde{K}} \left\|\left(1 - P_N\right)a\right\|_{H^\sigma}\right).
\end{aligned}$$

Observe that $\widetilde{K}$ is compact, so is the image $\mathbf{F}_\theta(\widetilde{K})$, which leads to

$$\limsup_{N\to\infty} \sup_{u\in\mathbf{F}_\theta(\widetilde{K})} \left\|\left(1 - P_N\right)v\right\|_{L^2} = 0 = \limsup_{N\to\infty} \sup_{a\in\widetilde{K}} \left\|\left(1 - P_N\right)a\right\|_{H^s}.$$

In essence, there exists $N \in \mathbb{N}$ such that $\left\|\mathbf{F}_\theta - \mathbf{F}_N(a)\right\|_{L^2} \leq \varepsilon$ for all $a \in K \subset \tilde{K}$.

In the subsequent part of this proof, our aim is to devise a PIFNO approximation of $\mathbf{F}_N$. Specifically, we observe that $\mathbf{F}_N$ establishes a continuous operator $\mathbf{F}_N : L^2\left(\mathbb{T}^d\right) \to L^2\left(\mathbb{T}^d\right)$ through the mapping $a \mapsto P_N\mathscr{G}\left(P_N a\right)$. Moreover, the compact set K remains compact even when considered as a subset of $L^2\left(\mathbb{T}^d\right)$. Our goal is to demonstrate the existence of a PIFNO $\mathbf{F}^\dagger : L^2\left(\mathbb{T}^d\right) \to L^2\left(\mathbb{T}^d\right)$, satisfying $\sup_{a\in K}\left\|\mathbf{F}_N(a) - \mathbf{F}^\dagger(a)\right\|_{L^2} < \varepsilon$. Consequently, the restriction of $\mathbf{F}^\dagger$ to $\overset{\circ}{H^*}\left(\mathbb{T}^d\right) \subset L^2\left(\mathbb{T}^d\right)$ furnishes an approximation of $\mathbf{F}$, ensuring

$$\sup_{a\in K} \left\|\mathbf{F}_\theta(a) - \mathbf{F}^\dagger(a)\right\|_{L^2} < 2\varepsilon.$$

Since $\varepsilon > 0$ was chosen arbitrarily, the assertion naturally follows. As elucidated in the overview, our proof establishing the existence of an FNO approximating $\mathbf{F}_N$ relies on a decomposition scheme, which is defined through the Fourier conjugate operator $\widehat{G}_N$. Our objective is to demonstrate that each operator in the decomposition can be accurately approximated by FNOs. To accomplish this, let $\varepsilon > 0$ be arbitrary, and select $R_K, R_{\hat{K}}, R_{\hat{g}} > 0$, satisfying the conditions

$$\left.\begin{aligned}
& K \subset B_{R_{\mathscr{K}}}(0) := \left\{\|u\|_{L^2} \leq R_K\right\} \subset L^2\left(\mathbb{T}^d\right), \\
& \mathscr{F}_N \circ P_N\left(B_{R_K}(0)\right) \subset \left[-\tfrac{R_{\hat{K}}}{2}, \tfrac{R_{\hat{K}}}{2}\right]^{2K_N}, \\
& \widehat{\mathbf{F}}_N\left(\left[-R_{\hat{K}}, R_{\hat{K}}\right]^{2K_N}\right) \subset \left[-\tfrac{R_{\hat{g}}}{2}, \tfrac{R_{\hat{g}}}{2}\right]^{2K_N}.
\end{aligned}\right\} \tag{5.3}$$

Introducing $R_K, R_{\hat{K}}, R_{\hat{G}}$ serves the purpose of ensuring that each PIFNO in the composition properly maps its own domain into the subsequent PIFNO's domain. The PIFNO approximations for the individual steps in the composition $\mathbf{F}_N = \mathscr{F}_N^{-1} \circ \mathbf{F}_N \circ \left(\mathscr{F}_N \circ P_N\right)$ are outlined below.

5.3.1. *PIFNO approximation of $\mathscr{F}_N^{-1}$.* We commence our endeavor by devising a PIFNO approximation for the final step in the composition. To facilitate this, we interpret the mapping

$$\mathscr{F}_N^{-1} : [-R, R]^{2\mathscr{K}_N} \subset \mathbf{R}^{2\mathscr{K}_N} \to L^2\left(\mathbb{T}^d\right),$$

as follows:

$$\mathscr{F}_N^{-1} : \begin{cases} L^2\left(\mathbb{T}^d;[-R,R]^{2K_N}\right) \to L^2\left(\mathbb{T}^d\right), \\ \left\{\mathrm{Re}\left(\widehat{v}_k\right),\mathrm{Im}\left(\widehat{v}_k\right)\right\}_{|k|\le N} \mapsto v(x), \end{cases} \tag{5.4}$$

where the input $\left\{\mathrm{Re}\left(\widehat{v}_k\right),\mathrm{Im}\left(\widehat{v}_k\right)\right\}_{|k|\le N} \in [-R,R]^{2K_N}$ corresponds to a constant function in the space $L^2\left(\mathbb{T}^d;[-R,R]^{2K_N}\right)$. For non-constant input functions $v(x)$, we apply $\mathscr{F}_N^{-1}$ to the constant function $x \mapsto f_{\mathrm{T}^d} v(\xi)d\xi$ to define the mapping (5.4), allowing for general inputs. This mapping can be approximated with any desired accuracy by a PIFNO $\mathbf{F}_{\mathrm{IFT}}^\dagger : L^2\left(\mathbb{T}^d;\mathbf{R}^{2K_N}\right) \to L^2\left(\mathbb{T}^d\right)$, satisfying

$$\left\|\mathbf{F}_{\mathrm{IFT}}^\dagger(\widehat{v}) - \mathscr{F}_N^{-1}(\widehat{v})\right\|_{L^2} \le \varepsilon/3, \tag{5.5}$$

for any constant input functions $\widehat{v} \in L^2\left(\mathbb{T}^d;[-R,R]^{2K_N}\right)$.

5.3.2. *PIFNO approximation of* $\widehat{\mathbf{F}}_N$. We view the operator $\widehat{\mathbf{F}}_N$ as a continuous mapping

$$\hat{\mathbf{F}}_N : \left[-R_{\hat{K}},R_{\hat{K}}\right]^{2K_N} \subset \mathbf{R}^{2K_N} \to \mathbf{R}^{2K_N}.$$

As $\left[-R_{\hat{K}},R_{\hat{K}}\right]^{2K_N}$ is compact, there exists a finite-dimensional canonical neural network $\widehat{\mathbf{F}^\dagger} : \mathbf{R}^{2K_N} \to \mathbf{R}^{2K_N}$ such that

$$\sup_{\hat{v}\in\left[-R_{\hat{\mathscr{K}}},R_{\hat{K}}\right]^{\kappa_N}} \left\|\widehat{\mathbf{F}_N}(\hat{v}) - \widehat{\mathbf{F}^\dagger}(\hat{v})\right\|_{\ell^2} \le \varepsilon/3. \tag{5.6}$$

Moreover, by (5.3), we deduce

$$\widehat{G}_N\left(\left[-R_{\hat{K}},R_{\hat{K}}\right]^{2K_N}\right) \subset \left[-\frac{R_{\hat{G}}}{2},\frac{R_{\hat{G}}}{2}\right]^{2K_N}.$$

Thus, by selecting a neural network approximation $\mathbf{F}^\dagger$ with adequate precision, we ensure

$$\widehat{\mathbf{F}^\dagger}\left(\left[-R_{\hat{K}},R_{\hat{K}}\right]^{2\mathscr{K}_N}\right) \subset \left[-R_{\hat{g}},R_{\hat{g}}\right]^{2\kappa_N},$$

alongside (5.6). Notably, for $v \in L^2\left(\mathbb{T}^d;\mathbf{R}^{2K_N}\right)$, the corresponding mapping

$$\widehat{\mathbf{F}^\dagger} : L^2\left(\mathbb{T}^d;\mathbb{R}^{2\mathscr{K}_N}\right) \to L^2\left(\mathbb{T}^d;\mathbf{R}^{2\mathscr{K}_N}\right), \quad v(x) \mapsto \widehat{\mathbf{F}^\dagger}(v(x)),$$

is essentially an operator featuring solely local layers of the form

$$v_\ell(x) \mapsto \sigma\left(A_\ell v_\ell(x) + b_\ell\right), \quad \left(A_\ell \in \mathbb{R}^{d_v\times d_v}, b_\ell \in \mathbb{R}^{d_v}\right),$$

where $d_v := 2\left|\mathscr{K}_N\right|$, i.e., a PIFNO characterized by all $P_\ell \equiv 0$. In particular, denoting $\mathrm{Lip}(\widehat{N})$ the Lipschitz constant of the PIFNO constructed in the previous step, we can ensure that

$$\mathrm{Lip}(\mathbf{F}^\dagger)\left\|\mathscr{F}_N P_N v - \mathbf{F}_{\mathbf{FT}}^\dagger(v)\right\|_e \le \varepsilon/3, \quad \forall v \in B_{R_K}(0), \tag{5.7}$$

and furthermore, since by (5.3), we have

$$\mathscr{F}_N \circ P_N\left(B_{R_K}(0)\right) \subset \left[-\frac{R_{\widehat{K}}}{2},\frac{R_{\widehat{K}}}{2}\right]^{2K_N}.$$

We can, in addition, ensure that $\mathbf{F}_{\mathrm{FT}}^\dagger\left(B_{R_K}(0)\right) \subset \left[-R_{\hat{K}},R_{\hat{K}}\right]^{2K_N}$.

5.3.3. *Error estimate for resulting PIFNO approximation.* We now define a PIFNO $\mathbf{F}^\dagger(a) := \mathbf{F}^\dagger_{\mathrm{IFT}} \circ \mathbf{F}^\dagger \circ \mathbf{F}^\dagger_{\mathrm{FT}}(a)$, where the right-hand side terms have been constructed above. We note that

$$\begin{aligned}
\sup_K \left\| \mathbf{F}_N - \mathbf{F}^\dagger \right\|_{L^2} \leq & \sup_{B_{K_K}(0)} \left\| \mathscr{F}_N^{-1} \circ \widehat{\mathbf{F}}_N \circ \mathscr{F}_N \circ P_N - \mathbf{F}^\dagger_{\mathrm{IFT}} \circ \hat{\mathbf{F}}^\dagger \circ \mathbf{F}^\dagger_{\mathrm{FT}} \right\|_{L^2} \\
\leq & \sup_{B_{\kappa_K}(0)} \left\| \mathscr{F}_N^{-1} \circ \mathbf{F}_N \circ \mathscr{F}_N \circ P_N - \mathscr{F}_N^{-1} \circ \mathbf{F}^\dagger \circ \mathbf{F}^\dagger_{\mathrm{FT}} \right\|_{L^2} \\
& + \sup_{B_{K_K}(0)} \left\| \mathscr{F}_N^{-1} \circ \hat{\mathbf{F}}^\dagger \circ \mathbf{F}^\dagger_{\mathrm{FT}} - \mathbf{F}^\dagger_{\mathrm{IFT}} \circ \hat{\mathbf{F}}^\dagger \circ \mathbf{F}^\dagger_{\mathrm{FT}} \right\|_{L^2} \\
\leq & \sup_{B_{K_K}(0)} \left\| \widehat{\mathbf{F}}_N \circ \mathscr{F}_N \circ P_N - \widehat{\mathbf{F}^\dagger} \circ \mathbf{F}^\dagger_{\mathrm{FT}} \right\|_{L^2} + \sup_{\hat{N}\left(\mathscr{N}_{YT}\left(B_{K_K}(0)\right)\right)} \left\| \mathscr{F}_N^{-1} - \mathbf{F}^\dagger_{\mathrm{IFT}} \right\|_{L^2} \\
=: & \ (I) + (II).
\end{aligned}$$

For the second term (II), we note that

$$\widehat{\mathbf{F}^\dagger}\left(\mathbf{F}^\dagger_{\mathrm{FT}}\left(B_{R_K}(0)\right)\right) \subset \widehat{\mathbf{F}^\dagger}\left(\left[-R_{\hat{K}}, R_{\hat{K}}\right]^{2K_N}\right) \subset \left[-R_{\hat{g}}, R_{\hat{g}}\right]^{2K_N},$$

and hence, by (5.5), we can bound

$$(II) \leq \sup_{\left[-R_{\mathbf{F}}, R_g\right]^{2K_N}} \left\| \mathscr{F}_N^{-1} - \mathbf{F}^\dagger_{\mathrm{IFT}} \right\|_{L^2} \leq \varepsilon/3.$$

To estimate the first term (I), we note that

$$\begin{aligned}
(I) \ = & \sup_{B_{K_K}(0)} \left\| \widehat{\mathbf{F}}_N \circ \mathscr{F}_N \circ P_N - \widehat{\mathbf{F}^\dagger} \circ \mathbf{F}^\dagger_{\mathrm{FT}} \right\|_{L^2} \\
\leq & \sup_{B_{\pi_K}(0)} \left\| \widehat{\mathbf{F}}_N \circ \mathscr{F}_N \circ P_N - \hat{\mathbf{F}}^\dagger \circ \mathscr{F}_N \circ P_N \right\|_{L^2} + \sup_{B_{K_K}(0)} \left\| \hat{\mathbf{F}}^\dagger \circ \mathscr{F}_N \circ P_N - \widehat{\mathbf{F}^\dagger} \circ \mathbf{F}^\dagger_{\mathrm{FT}} \right\|_{L^2} \\
=: & \ (Ia) + (Ib).
\end{aligned}$$

To estimate (Ia), we note that $\mathscr{F}_N\left(P_N\left(B_{R_{\mathscr{K}}}(0)\right)\right) \subset \left[-R_{\hat{K}}, R_{\hat{K}}\right]^{2K_N}$, and hence

$$(Ia) \leq \sup_{\left[-R_{\widehat{\mathscr{K}}}, R_{\tilde{K}}\right]^{2\mathscr{K}_N}} \left\| \widehat{\mathbf{F}_N} - \widehat{\mathbf{F}^\dagger} \right\|_{L^2} \leq \varepsilon/3,$$

by (5.6). Finally, to estimate (Ib), we note that

$$(Ib) \leq \mathrm{Lip}(\widehat{\mathbf{F}^\dagger}) \sup_{B_{K_K}(0)} \left\| \mathscr{F}_N \circ P_N - \mathbf{F}^\dagger_{\mathrm{FT}} \right\|_{L^2} \leq \varepsilon/3,$$

by (5.7). Combining these estimates, we conclude that $\sup_{a \in K} \left\| \mathbf{F}_N(a) - \mathbf{F}^\dagger(a) \right\|_{L^2} \leq \varepsilon$, which shows that the continuous operator $\mathbf{F}_N$ can be approximated by a PIFNO $\mathbf{F}^\dagger$ to any desired accuracy $\varepsilon > 0$, and together with (2.9 concludes our proof of the universal approximation Theorem 5.1 for the special case $s' = 0$. The general case with $s' \geq 0$ now follows from Lemma 5.2. □

6. Conclusion

In this paper, we proposed a PIFNO, a new neural operator architecture with several advantages compared to prior works. Our approach is inspired by PINN and demonstrates superior long-term performance, especially when fine dataset resolution is limited. PIFNO is highly convenient to tune, achieves faster convergence, and exhibits minimal need for hyperparameter tuning. Notably, PIFNO retains performance even when trained on low-resolution datasets and tested on high-resolution ones, evidencing discretization invariance. However, performance still relies heavily on dataset quality, an aspect that we aim to improve in future work.

Acknowledgments
The research was supported by the National Science Foundation of China (62373066).

References

[1] M. J. Turner, R. W. Clough, H. C. Martin, L. J. Topp, Stiffness and deflection analysis of complex structures, J. Aeronaut. 23 (1956), 805-823.
[2] R. D. Richtmyer, E. H. Dill, Difference methods for initial-value problems, Phys. Today 12 (1959), 50-50.
[3] A. Jameson, W. Schmidt, E. Turkel, Numerical solution of the Euler equations by finite volume methods using Runge Kutta time stepping schemes, in: 14th Fluid and Plasma Dynamics Conference, p. 1259, (1981)
[4] M. Dissanayake, N. Phan-Thien, Neural-network-based approximations for solving partial differential equations, Commun. Numer. Meth. 10 (1994), 195-201.
[5] I. E. Lagaris, A. Likas, D. I. Fotiadis, Artificial neural networks for solving ordinary and partial differential equations, IEEE Trans. Neural Netw. 9 (1998), 987-1000.
[6] J. Han, A. Jentzen, W. E, Solving high-dimensional partial differential equations using deep learning, Proc. Natl. Acad. Sci. 115 (2018), 8505-8510.
[7] M. Raissi, P. Perdikaris, G. E. Karniadakis, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, J. Comput. Phys. 378 (2019), 686-707.
[8] W. E, B. Yu, The deep Ritz method: A deep learning-based numerical algorithm for solving variational problems, Commun. Math. Stat. 6 (2018), 1-12.
[9] A. D. Jagtap, E. Kharazmi, G. E. Karniadakis, Conservative physics-informed neural networks on discrete domains for conservation laws: Applications to forward and inverse problems, Comput. Methods Appl. Mech. 365 (2020), 113028.
[10] E. Kharazmi, Z. Zhang, G. E. Karniadakis, Variational physics-informed neural networks for solving partial differential equations, arXiv preprint (2019) arXiv:1912.00873.
[11] Y. Fan, L. Lin, L. Ying, L. Zepeda-Núnez, A multiscale neural network based on hierarchical matrices, Multiscale Model. Simul. 17 (2019), 1189-1213.
[12] L. Lu, P. Jin, G. E. Karniadakis, Deeponet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators, arXiv preprint (2019) arXiv:1910.03193.
[13] N. Kovachki, Z. Li, B. Liu, K. Azizzadenesheli, K. Bhattacharya, A. Stuart, A. Anandkumar, Neural operator: Learning maps between function spaces with applications to PDEs. J. Mach. Learn. Res. 24 (2023), 1–97.
[14] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, A. Anandkumar, Fourier neural operator for parametric partial differential equations, arXiv preprint (2020) arXiv:2010.08895.
[15] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778, 2016.
[16] M. Belkin, D. Hsu, S. Ma, S. Mandal, Reconciling modern machine-learning practice and the classical bias–variance trade-off, Proc. Natl. Acad. Sci. 116 (2019), 15849-15854.
[17] S. Du, J. Lee, H. Li, L. Wang, X. Zhai, Gradient descent finds global minima of deep neural networks, In: International Conference on Machine Learning, PMLR, pp. 1675-1685, 2016.

[18] V. Vapnik, The Nature of Statistical Learning Theory, Springer, 1999.
[19] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, arXiv preprint (2014) arXiv:1412.6980.
[20] D. Hendrycks, K. Gimpel, Gaussian error linear units (gelus), arXiv preprint (2016) arXiv:1606.08415.
[21] J. He, L. Li, J. Xu, C. Zheng, Relu deep neural networks and linear finite elements, arXiv preprint (2018) arXiv:1807.03973.