SirenFNO: Efficient and Full Frequency Learning of Fourier Neural Operators

Pengqing Shi, **Jie Yin**, **Stephen Tierney** and **Junbin Gao**
The University of Sydney, Australia
{pengqing.shi, jie.yin, stephen.tierney, junbin.gao}@sydney.edu.au

Abstract

Fourier neural operators (FNOs) are effective and efficient surrogates for approximating solutions of PDEs and generalize across discretizations. However, owing to the reliance on frequency truncation to maintain learning efficiency of FNOs, empirical studies suggest that FNOs exhibit spectral bias toward low-frequency information, which may hinder the learning capability especially for certain PDEs with strong high-frequency oscillations. To address this limitation, we propose SirenFNO, a novel framework that leverages sinusoidal representation networks (SIRENs) to learn implicit neural representations and performs mode-wise kernel parameterization. Our SIREN parameterization learns a full-grid spectrum with a constant and discretization-independent parameter count, thereby eliminating the need for frequency truncation. We further extend SirenFNO with functional tensor decompositions to enhance parameter and learning efficiency. Empirical results show that our SirenFNO consistently outperforms FNO with approximately 4 to 15 times parameter reductions with preserved discretization invariance, and our functional decomposition variants obtain performance improvements with a maximum of 73 times fewer parameters across multiple PDE benchmarks.

1 Introduction

Operator learning aims to approximate mappings between infinite-dimensional function spaces, with applications across scientific computing and physical simulation. Recently, neural operators (NOs) [Lu *et al.*, 2021; Li *et al.*, 2021; Kovachki *et al.*, 2023] have been used to efficiently solve families of partial differential equations (PDEs), including weather forecasting [Pathak *et al.*, 2022], engineering design [Liu *et al.*, 2023], forward/inverse problems [Raissi and Karniadakis, 2018; Raissi *et al.*, 2019].

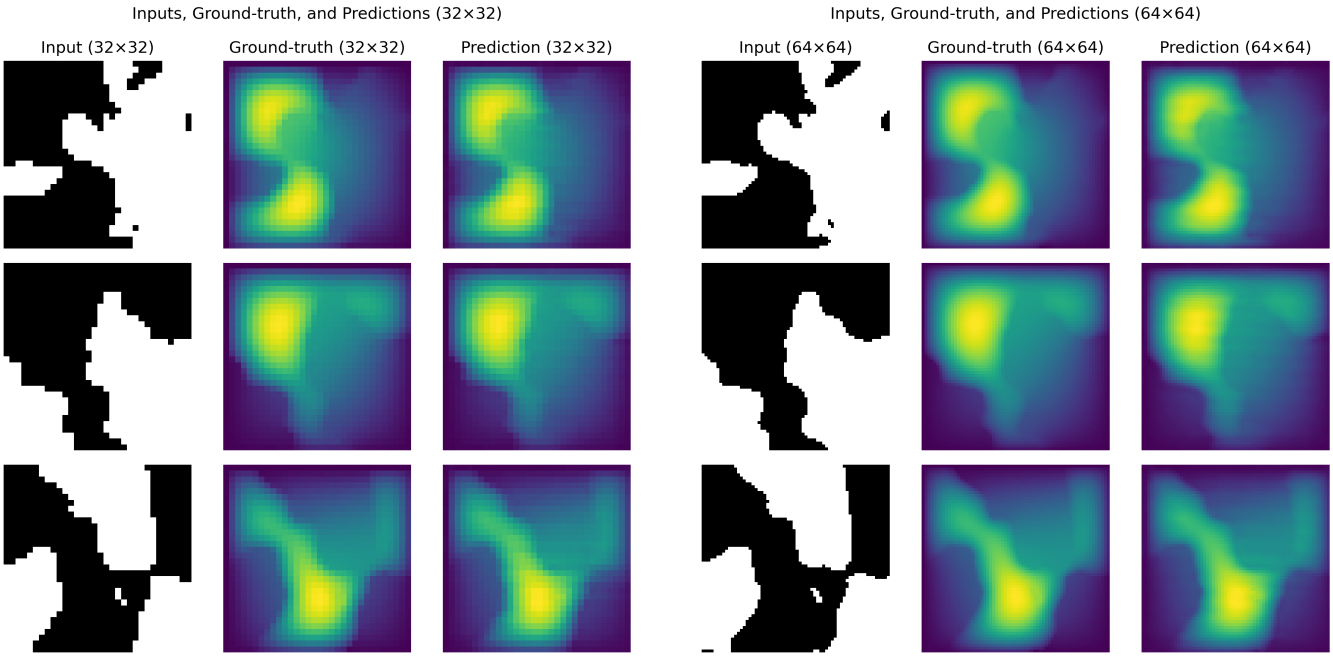
The Fourier Neural Operator (FNO) [Li *et al.*, 2021] is a class of NOs that has proven highly effective and efficient for learning PDE solution operators. Specifically, FNO computes the kernel integral in the frequency domain by applying the Fast Fourier Transform (FFT), performs a learned convolution there, and then maps back to the spatial domain via the inverse FFT (IFFT). In practice, instead of learning the convolution weights for every frequency exhaustively, FNO performs frequency truncation with a chosen threshold to retain only low-frequency modes and discard high-frequency components. This yields certain computational savings, fewer parameters, and near–discretization invariance. However, the use of frequency truncation may fail to preserve the equivalence between the continuous and discrete representations of neural operators [Gao *et al.*, 2025], and hinder the model's ability to capture high-frequency details in certain PDEs with strong high-frequency oscillations [Xiao *et al.*, 2024].

To address the aforementioned critical limitations that frequency truncation imposes on the FNO, we propose to parameterize the Fourier kernel via a hypernetwork. This hypernetwork performs mode-wise kernel weight generation, dynamically producing kernel parameters for each frequency mode without any explicit truncation, thereby enabling learning across arbitrary discretization resolutions. More importantly, as the spectral kernel learns a separate weight for each Fourier mode, the number of learnable kernel parameters in the plain FNO is determined by the modes retained after truncation, and without truncation it scales with the grid resolution. In contrast, in our approach, the number of parameters solely depends on the hypernetwork architecture and remains fixed across different resolutions.

Specifically, we use SIREN [Sitzmann *et al.*, 2020] as the hypernetwork to generate the Fourier kernel coefficients. Unlike conventional alternatives such as MLPs, which exhibit spectral bias toward low-frequency modes [Rahaman *et al.*, 2019; Qin *et al.*, 2024; Zhi-Qin John Xu *et al.*, 2020], SIREN's sinusoidal activations efficiently capture both high- and low-frequency information. Therefore, SIREN provides a smooth and continuous parameterization that is agnostic to any specific discretization and less susceptible to spectral bias, making it a well-suited hypernetwork for the FNO integral kernel. Furthermore, SIREN efficiently generates Fourier kernel coefficients with a relatively small number of learnable parameters, and we further improve its efficiency by incorporating functional tensor decompositions [Vemuri *et al.*, 2024; Vemuri *et al.*, 2025].

The main contributions of this work are as follows:

- We propose the Siren-FNO framework, which utilizes



(a) Trained and tested on 32×32

(b) Trained on 32×32 and tested on 64×64

Figure 1: Zero-Shot Super-Resolution on the Darcy Flow for SirenFNO trained on 32×32 resolution.

SIREN as a hypernetwork to generate convolution kernel weights for all frequency modes, avoiding spectral truncation, while maintaining computational and parameter efficiency.

- We extend the Siren-FNO framework with functional tensor decompositions of the Fourier integral kernel, specifically Canonical Polyadic (CP), Tensor-Train (TT), and Tucker decompositions.
- We demonstrate the effectiveness and resolution-agnostic behavior of the proposed methods across multiple PDE benchmarks.

2 Related Works

Neural Operators. Neural operators learn mappings between continuous function spaces and are often used for solving PDEs. For practical implementation, these continuous functions are generally represented on discretized grids. DeepONet [Lu *et al.*, 2021] employs two sub-networks: a branch network that encodes input function values at fixed sensor points and a trunk network learns the mapping between query locations to a set of basis functions. The outputs from these two networks are then combined to approximate operators, with guarantees from the universal approximation theorem. Laplace Neural Operator (LNO) [Cao *et al.*, 2024] uses the Laplace transform to decompose the input space and introduces Laplace layers to better learn non-periodic signals in the Laplace domain. Moreover, attention mechanisms have also been explored for operator learning [Cao, 2021]. Operator Transformer (OFormer) [Cao, 2021] is a transformer-based neural operator framework that employs an input encoder to embed function samples and a query encoder to project arbitrary output locations. General Neural Operator Transformer (GNOT) [Hao *et al.*, 2023] introduces a heterogeneous attention operator to process diverse input function types. Our work focuses on the Fourier neural operators [Li *et al.*, 2021].

Fourier Neural Operators. The FNO [Li *et al.*, 2021] defines its convolution kernel in the Fourier space and uses Fast Fourier Transform (FFT) to learn the operator efficiently. Several works seek to further enhance the efficiency of the vanilla FNO. Factorized FNO (F-FNO) [Tran *et al.*, 2023] deepens the FNO architecture to capture long-range dependencies and factorizes layers to learn each feature dimension efficiently with fewer parameters. Adaptive FNO (AFNO) [Guibas *et al.*, 2021] utilizes MLP layers with learnable weights adaptively shared across frequency modes to reduce computational complexity. Beyond efficiency, motivated by the low-frequency bias of FNOs, a number of studies aim to advance its high-frequency learning ability. U-FNO [Wen *et al.*, 2022] incorporates a U-Net pathway to each Fourier layer to enhance high-frequency learning. Amortized FNO (AM-FNO) [Xiao *et al.*, 2024] introduces a learnable amortized parameterization of the Fourier-kernel integral that captures high-frequency information without explicit truncation.

Dynamic Hypernetworks. Hypernetworks (hypernets) are neural networks that generate learnable weights of a primary neural network. Hypernets can be categorized by their weight-generation strategy as either static or dynamic [Chauhan *et al.*, 2024]. A hypernet is static if its conditioning inputs and outputs have fixed dimensions for a predetermined target architecture. In contrast, dynamic hypernets are used

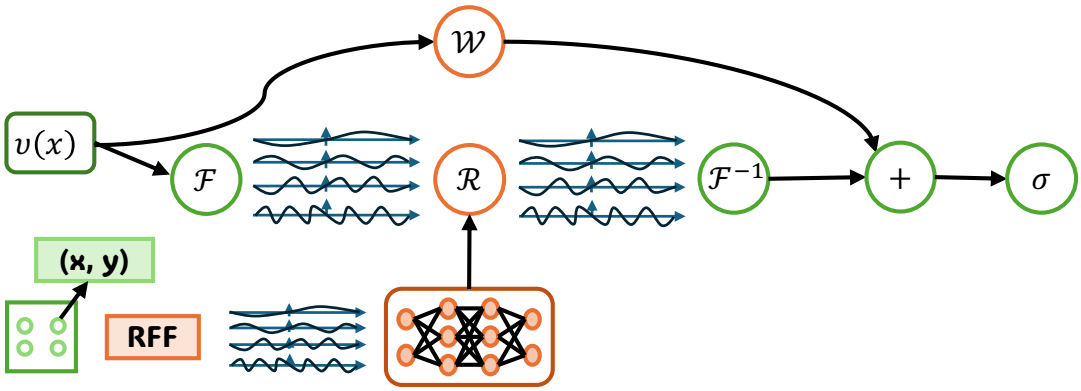


Figure 2: Illustration of SIREN Parameterization on a 2-D PDE discretized at resolution of 2×2

to generate model weights for a primary network with a dynamic architecture, which means that the structure of the primary neural network could change or be unknown during the training and inference stages. Dynamic hypernets have been leveraged for shape reconstruction [Littwin and Wolf, 2019], automatic network pruning [Li *et al.*, 2020], or efficient architecture search [Peng *et al.*, 2020]. In this work, we adopt a dynamic hypernetwork to enable parameterization generalization across discretizations.

Tensor Decomposition for Neural Operators. For operator learning, tensor decomposition is often used to separate PDE dimensions or variables for improving parameter efficiency. TDMD-DeepONet [Chen *et al.*, 2025] integrates tensor-train and dynamic mode decompositions into DeepONet for solving time-dependent PDEs. TensorGRaD [Loeschcke *et al.*, 2025] directly performs Tucker decomposition on the gradient tensor. F-FNO [Tran *et al.*, 2023] factorizes the Fourier transforms with separable spectral layers along the problem dimensions. D-FNO [Li and Ye, 2025] decomposes latent features into a sum of rank-1 tensor products, and substitutes the 3-D FFT by multiple 1-D FFTs, improving efficiency on 3-D PDE tasks. MG-TFNO [Kossaifi *et al.*, 2023] employs tensor decomposition in a multi-grid setting to separate FNOS Fourier convolution weights for greater parameter efficiency.

3 Preliminaries

This section describes the setting of the Fourier neural operator and its frequency truncation in detail.

3.1 Fourier Neural Operators

For an input function space $\mathcal{A} = \mathcal{A}(D; \mathbb{R}^{d_a})$ and a target function space $\mathcal{U} = \mathcal{U}(D; \mathbb{R}^{d_u})$ on a bounded domain $D \subset \mathbb{R}^d$, operator learning aims to learn a neural operator $\mathcal{G}_\theta$ with parameters $\theta \in \mathbb{R}^p$ to approximate the ground-truth mapping $\mathcal{G} : \mathcal{A} \to \mathcal{U}$ from a set of observed function samples $\{a_i, u_i\}_{i=1}^N$. For a loss function L, we learn the neural operator by solving the problem,

$$\min_\theta \frac{1}{N} \sum_{i=1}^{N} L\big(\mathcal{G}_\theta(a_i), u_i\big). \tag{1}$$

Neural operator (NO) takes the following composition form,

$$\mathcal{G}_\theta := \mathcal{P} \circ \mathcal{L}^L \circ \mathcal{L}^{L-1} \circ \cdots \circ \mathcal{L}^1 \circ \mathcal{Q}, \tag{2}$$

where $\mathcal{Q} : \mathcal{A}(D, \mathbb{R}^{d_a}) \to \mathcal{U}(D, \mathbb{R}^{d_v})$ is the lifting operator and $\mathcal{P} : \mathcal{U}(D, \mathbb{R}^{d_v}) \to \mathcal{U}(D, \mathbb{R}^{d_u})$ is the projection operator, and each NO layer $\mathcal{L}^l$ for $l = 1, 2, \ldots, L$ is defined as,

$$v^{(l+1)}(x) = \mathcal{L}^{(l)}(v^{(l)}(\cdot))(x) := \sigma\big(W^{(l)} v^{(l)}(x) + \int_D \kappa(x, x', a(x), a(x'); \phi) v^{(l)}(x') dx'\big), \forall x \in D \tag{3}$$

where σ is nonlinear activation, $W \in \mathbb{R}^{d_v \times d_v}$ is a learnable weight matrix, and κ is the integration kernel.

Fourier Neural Operator (FNO) [Li *et al.*, 2021] proposes a Fourier integral operator by employing convolution operation in the Fourier domain, satisfying translation invariance that $\kappa_\phi(x, x', a(x), a(x')) = \kappa_\phi(x - x')$. Applying the convolution theorem, the Fourier integral kernel is then defined as $\int_D \kappa_\phi(x - x') dx' = \mathcal{F}^{-1}(\mathcal{F}(\kappa_\phi) \cdot \mathcal{F}(v^l))(x)$, $\forall x \in D$ Then, FNO parameterizes the periodic kernel function κ in the Fourier domain by $\mathcal{R}_\phi(k) \in \mathbb{C}^{d_v \times d_v}$ such that $\mathcal{R}_\phi(k) = \mathcal{F}(\kappa)(k)$ for $k \in \mathbb{Z}^d$.

$$v^{(l+1)}(x) = \sigma\Big(W^{(l)} v^{(l)}(x) + \mathcal{F}^{-1}\big(\mathcal{R}^l_\phi(k) \cdot \mathcal{F}(v^{(l)})(k)\big)(x) + b^{(l)}\Big), \forall x \in D \tag{4}$$

3.2 Frequency Truncation

Low-frequency components generally have larger magnitudes than high-frequency components in many PDE problems [Boffetta and Ecke, 2012; George *et al.*, 2024; Jing *et al.*, 2025]. To prioritize the learning of low-frequency modes and improve parameter efficiency, FNO truncates the Fourier spectrum by discarding high-frequency components with a predetermined number of retained frequency modes $k_{\max} = |\{k \in \mathbb{Z}^d : |k_j| \le k_{\max,j} \text{ for } j = 1, \ldots, d\}|$. In this case, we have learnable kernel parameters $R_\phi \in \mathbb{C}^{k_{\max} \times d_v \times d_v}$, and the size of complex-valued tensor R_ϕ can be greatly reduced owing to frequency truncation. Then, FNO performs zero-padding on discarded modes and applies an IFFT to recover the solution field in the physical domain.

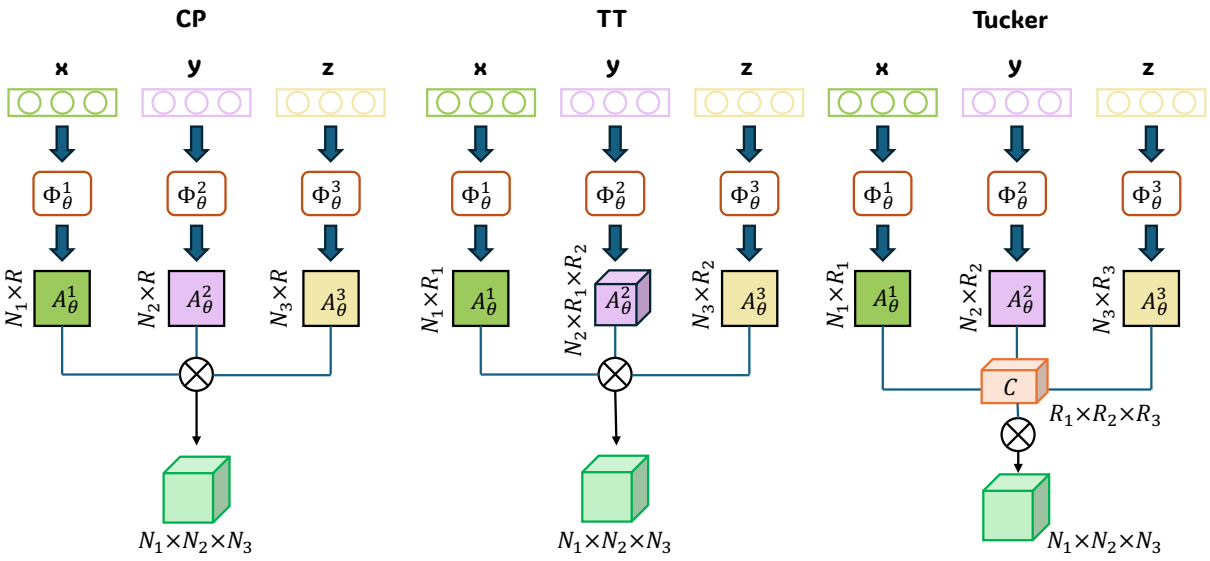


Figure 3: SIREN parameterization with CP, TT, and Tucker functional tensor decompositions on a 3D grid.

By parameterizing only the retained low-frequency components, FNO achieves superior efficiency and captures dominant components well. However, FNO exhibits Fourier parameterization bias or spectral bias due to its inability to capture non-dominant frequencies effectively [Qin *et al.*, 2024]. Moreover, the predetermined $k_{\max}$ imposes inductive bias and may degrade model performance [George *et al.*, 2024].

4 The Proposed Method

In this work, we propose SirenFNO, a novel framework that uses SIREN [Sitzmann *et al.*, 2020] to generate kernel coefficients mode by mode for all discrete Fourier components, eliminating the need for truncation. A key advantage of SirenFNO is that the number of learnable parameters is fixed by the SIREN architecture and does not depend on the grid resolution. Thereby, our proposed SirenFNO efficiently represents both lower- and higher-frequency details across varying discretizations.

4.1 Parameterization of Fourier Kernel via SIREN

SIREN [Sitzmann *et al.*, 2020] employs sinusoidal activations to to learn continuous implicit representations. In our parameterization, each discrete Fourier mode is indexed by a (normalized) frequency coordinate $\xi_k \in [-1,1]^d$ for $k \in \mathbb{Z}^d$. A SIREN network $\Phi_\theta : [-1,1]^d \to \mathbb{C}^{d_v \times d_v}$ maps ξ_k to the corresponding complex Fourier kernel coefficient matrix. For a batch of modes K at a given resolution, stacking inputs yields outputs in $\mathbb{C}^{|K|\times d_v \times d_v}$ the set size $|K|$ is adaptively determined by the discretization. Because all frequency coordinates are normalized to $[-1,1]$, the network is insensitive to changes in grid resolution. Specifically, for each k, the coefficients are produced by a SIREN stack,

$$\Phi_\theta(\xi_k) = (\mathcal{O} \circ \phi^{(L)} \circ \cdots \circ \phi^{(1)} \circ \mathcal{E}_B)(\xi_k), \tag{5}$$

where $\mathcal{E}_B : \mathbb{R}^d \to \mathbb{R}^{2m}$ is a learnable random Fourier feature (RFF) embedding of the input frequency coordinates, defined as

$$\mathcal{E}_B(\xi_k) = \left[\cos(\pi B^\top \xi_k),\ \sin(\pi B^\top \xi_k)\right] \in \mathbb{R}^{2m}, \tag{6}$$

where m is the number of RFF frequencies, and B is a learnable parameter initialized as $B^{(0)} = \sigma G$, with $G_{ij} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0,1)$ and $\sigma > 0$.

Then, we have $\phi^{(1)} : \mathbb{R}^{2m} \to \mathbb{R}^h$, and $\phi^{(l)} : \mathbb{R}^h \to \mathbb{R}^h$ for $l = 2, \dots L$. Each SIREN layer $\phi^{(l)}$ in the above (5) takes the following form,

$$\xi^{(l)} \leftarrow \phi^{(l)}(\xi_k^{(l-1)}) = \sin(w \cdot W^{(l)} \xi_k^{(l-1)} + b^{(l)}), \tag{7}$$

where a factor w is used in each SIREN layer to accelerate convergence and span multiple periods of the sine basis, enabling our parameterization method to efficiently represent high-frequency information. Finally, $\mathcal{O} : \mathbb{R}^h \to \mathbb{C}^{d_v \times d_v}$ is an MLP layer.

Now, we replace the discreted kernel coefficients $\mathcal{R}_\phi(k) \in \mathbb{C}^{d_v \times d_v}$ in (4) with coefficients $\Phi_\theta(\xi_k) \in \mathbb{C}^{d_v \times d_v}$ generated by the above SIREN parameterization, our SirenFNO layer is formulated as,

$$\begin{aligned} v^{(l+1)}(x) = \sigma\Big(&W^{(l)} v^{(l)}(x) \\ &+ \mathcal{F}^{-1}\big(\Phi_\theta^{(l)}(\xi_k) \cdot \mathcal{F}(v^{(l)})(k)\big)(x) + b^{(l)}\Big),\ \forall x \in D. \end{aligned} \tag{8}$$

In contrast to FNO, which directly maintains and optimizes a large discretization-dependent tensor of Fourier kernel coefficients, our SIREN parameterization learns a continuous mapping from frequency coordinates to kernel coefficients. This eliminates the need to store all coefficients explicitly, and they are generated on demand for each frequency mode instead. As such, SirenFNO can efficiently learn the Fourier integral operator over all frequency modes without frequency truncation, and the number of learnable parameters is independent of the discretization resolution, determined solely by the SIREN architecture. Figure 2 illustrates the process of parameter generation by our kernel parameterization.

4.2 Functional Tensor Decomposition for SirenFNO

For SirenFNO without truncation, we construct input coordinate points for SIREN block $\Phi_\theta : [-1,1]^d \to \mathbb{C}^{d_v \times d_v}$ on a full d-dimensional domain, and the SIREN block is shared across all resolution grids with N points per axis. Therefore, for a d-dimensional PDE discretized on a $K = N^d$ grids , the construction of input coordinates for SIREN takes up $O(N^d \times d)$ memory, which requires excessive memory usage, especially for finer grids and high-dimensional PDE applications. To alleviate this, we further extend SirenFNO by leveraging functional tensor decompositions.

In this work, we employ three tensor decomposition schemes, *Canonic-Polyadic (CP)*, *Tensor-Train (TT)*, and *Tucker*, within SirenFNO. In the functional tensor variants, the kernel is parameterized by factorizing the SIREN trunk into d branch sub-networks, one per geometric dimension. Instead of sampling a full N^d grid, each branch processes N coordinates for its dimension, yielding a factor vector/matrix; the full tensor of Fourier-kernel coefficients is then reconstructed via tensor (outer) products. This reduces the sampling requirement from N^d points to $N \times d$ and the memory footprint from exponential to linear in d, i.e. $\mathcal{O}(Nd)$.

CP-SirenFNO. With CP rank $R \ll \min_{i\in[d]} N_i$, the kernel tensor $\Phi_\theta \in \mathbb{C}^{N_1\times\cdots\times N_d}$ is parameterized as a sum of R rank-1 outer products. SirenFNO realizes this by using d SIREN branches to produce factor matrices $A^{(i)} \in \mathbb{R}^{N_i\times R}$ for $i = 1,\dots,d$. Let $a_r^{(i)} := A^{(i)}(:,r) \in \mathbb{R}^{N_i}$ denote the r-th column. The reconstruction is

$$\Phi_\theta = \sum_{r=1}^{R} a_r^{(1)} \otimes a_r^{(2)} \circ \cdots \circ a_r^{(d)}, \tag{9}$$

where $\otimes$ denotes the outer (tensor) product. This reduces storage from $\prod_{i=1}^d N_i$ to $\mathcal{O}\big(R\sum_{i=1}^d N_i\big)$ while enabling on-demand coefficient generation.

TT-SirenFNO. With TT ranks $R_0 = R_d = 1$ and $R_1,\dots,R_{d-1}$, SirenFNO uses d SIREN branches to produce cores $G^{(i)} \in \mathbb{C}^{R_{i-1}\times N_i\times R_i}$ for $i = 1,\dots,d$. Let $g^{(i)}_{r_{i-1},r_i} \in \mathbb{C}^{N_i}$ denote the mode-2 fiber of $G^{(i)}$ with TT indices (r_{i-1}, r_i) fixed. The kernel tensor $\Phi_\theta \in \mathbb{C}^{N_1\times\cdots\times N_d}$ is reconstructed as

$$\Phi_\theta = \sum_{r_1=1}^{R_1} \cdots \sum_{r_{d-1}=1}^{R_{d-1}} g^{(1)}_{1,r_1} \otimes g^{(2)}_{r_1,r_2} \otimes \cdots \otimes g^{(d)}_{r_{d-1},1}. \tag{10}$$

This representation reduces storage from $\prod_{i=1}^d N_i$ to $\mathcal{O}\big(\sum_{i=1}^d N_i R_{i-1} R_i\big)$.

Tucker-SirenFNO. Given Tucker ranks $R_1,\dots,R_d$, SirenFNO employs d SIREN branches to produce factor matrices $A^{(i)} \in \mathbb{C}^{N_i\times R_i}$ for $i = 1,\dots,d$, together with a learnable core $\mathcal{C} \in \mathbb{C}^{R_1\times\cdots\times R_d}$. The kernel tensor $\Phi_\theta \in \mathbb{C}^{N_1\times\cdots\times N_d}$ is reconstructed via the Tucker product

$$\Phi_\theta = \mathcal{C} \times_1 A^{(1)} \times_2 A^{(2)} \cdots \times_d A^{(d)}, \tag{11}$$

where $\times_i$ means mode-i tensor-matrix product.

For SirenFNO with functional tensor decompositions, these factor matrices $\{A^1,\dots,A^d\}$ are generated and parameterized using a number of d small SIREN networks $\{\Phi^1_{\theta_1},\dots,\Phi^d_{\theta_d}\}$ accordingly. The reconstructed tensor is then used as Fourier kernel coefficients in the FNO. Figure 3 illustrates an example of SIREN parameterization with functional tensor decompositions.

4.3 Model Architecture

Our SirenFNO follows the standard FNO architecture,

$$\mathcal{G}_\theta := \mathcal{P} \circ \mathcal{L}^L \circ \mathcal{L}^{L-1} \circ \cdots \circ \mathcal{L}^1 \circ \mathcal{Q}, \tag{12}$$

where $\mathcal{Q}$ is the lifting layer to transform input data α to hidden state z, and $\mathcal{P}$ is the projection layer to project hidden state z to output prediction u. Furthermore, intermediary layers $\mathcal{L}^l$ for $l = 1,\dots,L$ are our SirenFNO layers introduced in (8). Now, we use $\mathcal{K}_\Phi(v^{(l)}(z)) := \mathcal{F}^{-1}\big(\Phi^{(l)}_\theta(\xi_k)\cdot \mathcal{F}(v^{(l)})(k)\big)(z)$ to denote our SirenFNO parameterization kernel, we have

$$v^{(l+1)}(z) = \sigma\Big(W^{(l)}v^{(l)}(z) + \mathcal{K}_\Phi(v^{(l)}(z)) + b^{(l)}\Big), \quad \forall z \in D. \tag{13}$$

In this work, we leverage the same model architecture as AM-FNO [Xiao *et al.*, 2024; Tran *et al.*, 2023], using residual connections and replacing linear mapping $W^l v^l(z)$ such that

$$\begin{aligned} v^{(l+1)}(z) = \sigma\Big(&v^{(l)}(z) + \\ &W_2^{(l)}\sigma(W_1^{(l)}\mathcal{K}_\Phi(v^{(l)}(z)) + b_1^{(l)}) + b_2^{(l)}\Big), \quad \forall z \in D. \end{aligned} \tag{14}$$

It is worth noting that each SirenFNO operator layer includes additional nonlinearities after the residual connection, and the last operator layer omits the activation after the residual connection.

5 Experiments

This section describes our experimental setups and results. All experiments are conducted on the same machine equipped with an NVIDIA GeForce RTX 5090. Our Code is available at https://github.com/pengqingshi/SirenFNO.

5.1 Benchmarks and Baselines

We compare our proposed models against FNO [Li *et al.*, 2021] and its several well-known variants, including UFNO in [Wen *et al.*, 2022], Tensorized FNO with CP decomposition (TFNO-CP) [Kossaifi *et al.*, 2023], and AM-FNO with MLPs parameterization [Xiao *et al.*, 2024], and UNO in [Rahman *et al.*, 2023].

Name	Dataset	Resolution	No. train	No. test
Darcy	2D Darcy flow	128×128	1000	200
NS	2D incompressible Navier Stokes	128×128	1000	200
Burgers	1D Burgers' equation	$1024, T=20$	1000	200
1DCFD	1D computational fluid dynamics	$1024, T=20$	1800	200
2DCFD	2D computational fluid dynamics	$128\times128, T=10$	1800	200
ReacDiff	1D Reaction-Diffusion	$1024, T=20$	1000	200

Table 1: Description of benchmark datasets.

Dataset	**Darcy**		**NS**	**Burgers**	**1DCFD**	**2DCFD**	**ReacDiff**
Resolution	32×32	128×128	128×128	1024	1024	128×128	1024
FNO	$7.30e$-2	$6.26e$-2	$5.61e$-2	$1.12e$-2	$8.27e$-3	$1.64e$-2	$1.18e$-4
UFNO	$7.07e$-2	$6.00e$-2	$5.37e$-2	$1.70e$-2	$8.29e$-3	$1.21e$-2	$1.98e$-4
TFNO-CP	$5.20e$-2	$4.81e$-2	$5.17e$-2	$1.05e$-2	$6.31e$-3	$8.25e$-3	$1.14e$-4
U-NO	$6.57e$-2	$7.68e$-2	$4.96e$-2	$7.53e$-3	$7.62e$-3	$1.62e$-2	$7.79e$-5
AM-FNO (MLP)	$3.72e$-2	$5.10e$-2	$4.02e$-2	$8.92e$-3	$7.89e$-3	$1.21e$-2	$1.78e$-3
SirenFNO†	$6.32e$-2	$5.82e$-2	$5.52e$-2	$8.21e$-3	$7.41e$-3	$1.15e$-2	$7.40e$-5
CP-SirenFNO†	$4.99e$-2	$3.96e$-2	$5.04e$-2	$8.12e$-3	$5.91e$-3	$7.28e$-3	$8.98e$-5
TT-SirenFNO†	$5.42e$-2	$5.36e$-2	$5.87e$-2	$7.94e$-3	$5.58e$-3	$8.75e$-3	$8.51e$-5
Tucker-SirenFNO†	$4.45e$-2	$4.51e$-2	$4.19e$-2	$7.18e$-3	$7.31e$-3	$7.30e$-3	$7.81e$-5
SirenFNO	**3.51e-2**	**2.36e-2**	$3.51e$-2	**5.52e-3**	$6.77e$-3	**6.83e-3**	**6.31e-5**
CP-SirenFNO	$4.04e$-2	$3.03e$-2	$3.78e$-2	$8.48e$-3	**4.75e-3**	$7.40e$-3	$7.58e$-5
TT-SirenFNO	$4.18e$-2	$3.33e$-2	**3.24e-2**	$7.52e$-3	$6.32e$-3	$7.41e$-3	$8.92e$-5
Tucker-SirenFNO	$3.98e$-2	$3.18e$-2	$3.86e$-2	$8.11e$-3	$5.29e$-3	$6.90e$-3	$8.71e$-5

Table 2: ℓ_2 relative test errors. All models are train and test on the same corresponding resolution. Models highlighted with † are using the identical architecture with FNO for ablation study. SirenFNO and its variants in boldface but without † are our proposed methods. Note that our proposed methods learn full-frequency information without truncation.

Table 1 lists the benchmark datasets used in our experiments. Darcy and Navier-Stokes datasets are obtained from the NeuralOperator library [Kossaifi *et al.*, 2024; Kovachki *et al.*, 2023]; and 1D Burgers[1], 1D CFD[2], 2D CFD[3], and reaction-diffusion[4] are sourced from PDEBench [Takamoto *et al.*, 2024].

5.2 Implementation Details

The ℓ_2 relative error is utilized for training and evaluation. Given data pairs $\{a_i, u_i\}_{i=1}^N$ and a model $\mathcal{G}_\theta : \mathcal{A} \to \mathcal{U}$, the ℓ_2 relative error is computed as,

$$L(\mathcal{G}_\theta(a), u) = \frac{1}{N} \sum_{i=1}^{N} \frac{\|\mathcal{G}_\theta(a_i) - u_i\|^2}{\|u_i\|^2}. \tag{15}$$

For each experiment, we conduct 500 training epochs using AdamW optimizer with a batch size of 32, a learning rate of $1e$-3 with the cosine annealing schedule and a weight decay of $1e$-4. We set 4 layers with 32 hidden channels for FNO and its variants by default, including our SirenFNO. The hidden channels for lifting and projection layers are set to 64 by default. For FNO frequency truncation, we retain modes of $m_x = m_y = 32$ for Navier-Stokes and 2D CFD, and $m_x = m_y = 16$ for the Darcy flow. We learn full frequency for FNO on Burgers, 1D CFD, and reaction-diffusion. Please note that our proposed SirenFNO and its variant forms do not perform frequency truncation. Additionally, We evaluate time-dependent PDEs in an one-step autoregressive scheme, where we perform 10 rollout steps with previous 10 states for Burgers, reaction-diffusion, and 1D CFD; and we conduct 5 rollout steps with previous 5 states for 2D CFD.

5.3 Main Results

Table 2 reports the ℓ_2 relative test set error for each PDE benchmark, with all models trained and tested at identical discretization resolutions. Table 3 lists the number of learnable parameters in thousands for every neural operator in each experiment. For each problem, we boldface the lowest ℓ_2 relative test error and the smallest parameter count.

As shown in Table 2, our SirenFNO and its functional decomposition variants consistently surpass the vanilla FNO across all PDE benchmarks. Moreover, as indicated in Table 3, they achieve remarkably greater parameter efficiency without relying on frequency truncation.

For the 2D Darcy flow problem, our SirenFNO reduces the ℓ_2 relative test error from 0.0730 to 0.0351 (by 52%) and from 0.0626 to 0.0236 (by 62%) for discretizations of 32×32 and 128×128 respectively, with an approximately $4\times$ reduction in model parameters, compared to FNO. Moreover, CP, TT, and Tucker variants maintain comparable performance while further reducing the number of parameters. For solving the 2D Navier-Stokes equation, our SirenFNO obtains a ℓ_2 relative error of 0.0351, compared to that of 0.0561 for FNO; while using about 8 times fewer parameters. Notably, our TT-SirenFNO achieves the best overall performance on the Navier-Stokes benchmark, with a further reduction in parameters by 21 times compared to FNO. On the 1D Burgers benchmark, SirenFNO yields the lowest error, which is 51% lower than that of FNO, with about 14 times fewer parameters. CP-SirenFNO also outperforms FNO with roughly 60 times fewer parameters, and the TT and Tucker variants similarly improve over FNO with about 50 and 57 times fewer parameters. For 1D CFD, SirenFNO reduces the

[1] `1D_Burgers_Sols_Nu0.001.hdf5`

[2] `1D_CFD_Rand_Eta0.01_Zeta0.01_periodic_Train.hdf5`

[3] `2D_CFD_Rand_M0.1_Eta0.01_Zeta0.01_periodic_128_Train.hdf5`

[4] `ReacDiff_Nu0.5_Rho1.0.hdf5`

Dataset	**Darcy**		**NS**	**Burgers**	**1DCFD**	**2DCFD**	**ReacDiff**
Resolution	32×32	128×128	128×128	1024	1024	128×128	1024
FNO	1192.8	1192.8	4469.6	4216.2	4216.2	4469.9	4216.2
UFNO	5876.2	5876.2	990.8	4258.5	1892.2	991.0	1892.2
TFNO-CP	72.9	72.9	237.5	226.4	226.4	237.8	226.4
U-NO	1792.7	4250.3	8726.8	1726.1	1726.1	7602.3	5658.2
AM-FNO (MLP)	385.5	385.5	4443.1	207.6	823.1	385.6	823.1
SirenFNO	308.9	308.9	579.5	308.9	304.8	304.8	304.8
CP-SirenFNO	**63.9**	**63.9**	**138.9**	**70.1**	**57.7**	**64.0**	**57.7**
TT-SirenFNO	92.5	109.2	211.6	84.5	72.0	92.7	72.0
Tucker-SirenFNO	162.6	162.6	596.4	74.2	61.8	96.8	61.8

Table 3: Number of parameters used for each experiment, numbers are reported in thousands approximately. Note that our proposed methods learn full-frequency information without truncation.

ℓ_2 error by 18% lower) while using 14 times fewer parameters than FNO, and CP-SirenFNO is the best-performing model with only 57.7 thousands parameters. Similarly, for 2D CFD and reaction-diffusion, SirenFNO performs the best among all, while using significantly lower parameters than standard FNO. The improvement gains can be attributed to the ability of SIREN parameterization to capture high- and low-frequency information, enabling full-frequency learning irrespective of discretizations.

5.4 Ablation Study

As shown in Table 2, our proposed methods adopt the enhanced architecture used in AM-FNO (MLP). SirenFNO consistently outperforms all baselines while using significantly fewer parameters to learn full-frequency information, demonstrating both the effectiveness and efficiency of our model.

For the ablation study, we further isolate the effect of architectural changes by keeping the operator architecture identical to the standard FNO implementation. According to Table 2, under a strictly controlled setting, SirenFNO† results indicate that our SIREN parameterization yields consistent performance gains across all PDE benchmarks. Performance gains can be purely attributed to our kernel parameterization, suggesting that our methods mitigate the spectral bias of FNO and enhance its high-frequency learning capability while achieving superior parameter efficiency. Furthermore, our SirenFNO delivers further improvements, indicating a complementary and synergic effect by employing the improved operator architecture.

5.5 Zero-Shot Super-Resolution

A key advantage of FNO is resolution invariance, which enables a model trained on a coarse grid to generalize well to a finer grid without retraining. To preserve resolution-invariant generalization for our SirenFNO, we adopt frequency truncation with modes $m_x = m_y = 12$. Table 4 reports the zero-shot super-resolution performance on 2D Darcy flow, and Figure 1 visualizes the predictions. All models are trained on a discretization of 32×32 and then evaluated on 32×32 and 64×64 grids. Our SirenFNO consistently outperforms FNO and maintains strong performance across resolutions.

	32×32	64×64
FNO	0.0730	0.0803
SirenFNO	0.0451	0.0599
CP-SirenFNO	0.0502	0.0606
TT-SirenFNO	0.0459	0.0591
Tucker-SirenFNO	**0.0455**	**0.0576**

Table 4: ℓ_2 relative test error for zero-shot super-resolution performance on the 2D Darcy flow. Models are trained on 32×32 resolution.

6 Conclusion

In this work, we propose SirenFNO to perform full-frequency learning and mitigate spectral bias without truncating Fourier modes, we also consider functional tensor decompositions with CP, TT, and Tucker variants to further improve the parameter and learning efficiency. As suggested by our experiments across various PDE benchmarks, our SirenFNO and its variants surpass FNO in terms of ℓ_2 relative test error, using approximately 4 to 73 times fewer parameters without truncation. Overall, our proposed SirenFNO provides a more accurate and efficient alternative to standard FNO architectures for operator learning on PDE benchmarks.

References

[Boffetta and Ecke, 2012] Guido Boffetta and Robert E Ecke. Two-dimensional turbulence. *Annual review of fluid mechanics*, 44(1):427–451, 2012.

[Cao *et al.*, 2024] Qianying Cao, Somdatta Goswami, and George Em Karniadakis. Laplace neural operator for solving differential equations. *Nature machine intelligence*, 6(6):631–640, 2024.

[Cao, 2021] Shuhao Cao. Choose a transformer: Fourier or galerkin. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021.

[Chauhan *et al.*, 2024] Vinod Kumar Chauhan, Jiandong Zhou, Ping Lu, Soheila Molaei, and David A. Clifton. A brief review of hypernetworks in deep learning. *The Artificial intelligence review*, 57(9):250–, 2024.

[Chen *et al.*, 2025] Yuanhong Chen, Yifan Lin, Xiang Sun, Chunxin Yuan, and Zhen Gao. Tensor decomposition-based neural operator with dynamic mode decomposition for parameterized time-dependent problems. *Journal of Computational Physics*, 533:113996, 2025.

[Gao *et al.*, 2025] Wenhan Gao, Ruichen Xu, Yuefan Deng, and Yi Liu. Discretization-invariance? on the discretization mismatch errors in neural operators. In *The Thirteenth International Conference on Learning Representations*, 2025.

[George *et al.*, 2024] Robert Joseph George, Jiawei Zhao, Jean Kossaifi, Zongyi Li, and Anima Anandkumar. Incremental spatial and spectral learning of neural operators for solving large-scale PDEs. *Transactions on Machine Learning Research*, 2024.

[Guibas *et al.*, 2021] John Guibas, Morteza Mardani, Zongyi Li, Andrew Tao, Anima Anandkumar, and Bryan Catanzaro. Efficient token mixing for transformers via adaptive fourier neural operators. In *International Conference on Learning Representations*, 2021.

[Hao *et al.*, 2023] Zhongkai Hao, Zhengyi Wang, Hang Su, Chengyang Ying, Yinpeng Dong, Songming Liu, Ze Cheng, Jian Song, and Jun Zhu. Gnot: a general neural operator transformer for operator learning. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023.

[Jing *et al.*, 2025] Fengrui Jing, Chuchu Zhai, Peizhi Zhao, Xue Li, Peifu Han, Hongzhen Ding, Yunlong Dong, Long Hao, Xu Tian, and Tao Song. Multi-scale low-frequency enhanced spectral neural operator for reducing low-frequency error in partial differential equations solving. *Engineering Applications of Artificial Intelligence*, 154:110912, 2025.

[Kossaifi *et al.*, 2023] Jean Kossaifi, Nikola Kovachki, Kamyar Azizzadenesheli, and Anima Anandkumar. Multi-grid tensorized fourier neural operator for high-resolution pdes. *arXiv.org*, 2023.

[Kossaifi *et al.*, 2024] Jean Kossaifi, Nikola Kovachki, Zongyi Li, David Pitt, Miguel Liu-Schiaffini, Robert Joseph George, Boris Bonev, Kamyar Azizzadenesheli, Julius Berner, and Anima Anandkumar. A library for learning neural operators, 2024.

[Kovachki *et al.*, 2023] Nikola Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Learning maps between function spaces with applications to pdes. *Journal of Machine Learning Research*, 24(89):1–97, 2023.

[Li and Ye, 2025] Kangjie Li and Wenjing Ye. D-fno: A decomposed fourier neural operator for large-scale parametric partial differential equations. *Computer Methods in Applied Mechanics and Engineering*, 436:117732, 2025.

[Li *et al.*, 2020] Yawei Li, Shuhang Gu, Kai Zhang, Luc Van Gool, and Radu Timofte. Dhp: Differentiable meta pruning via hypernetworks. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 608–624, Cham, 2020. Springer International Publishing.

[Li *et al.*, 2021] Zongyi Li, Nikola Borislavov Kovachki, Kamyar Azizzadenesheli, Burigede liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. In *International Conference on Learning Representations*, 2021.

[Littwin and Wolf, 2019] Gidi Littwin and Lior Wolf. Deep meta functionals for shape representation. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1824–1833, 2019.

[Liu *et al.*, 2023] Ziyue Liu, Yixing Li, Jing Hu, Xinling Yu, Shinyu Shiau, Xin Ai, Zhiyu Zeng, and Zheng Zhang. Deepoheat: Operator learning-based ultra-fast thermal simulation in 3d-ic design. In *2023 60th ACM/IEEE Design Automation Conference (DAC)*, pages 1–6, 2023.

[Loeschcke *et al.*, 2025] Sebastian Loeschcke, David Pitt, Robert Joseph George, Jiawei Zhao, Cheng Luo, Yuandong Tian, Jean Kossaifi, and Anima Anandkumar. Tensorgrad: Tensor gradient robust decomposition for memory-efficient neural operator training, 2025.

[Lu *et al.*, 2021] Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3(3):218–229, 2021.

[Pathak *et al.*, 2022] Jaideep Pathak, Shashank Subramanian, Peter Harrington, Sanjeev Raja, Ashesh Chattopadhyay, Morteza Mardani, Thorsten Kurth, David Hall, Zongyi Li, Kamyar Azizzadenesheli, Pedram Hassanzadeh, Karthik Kashinath, and Animashree Anandkumar. Fourcastnet: A global data-driven high-resolution weather model using adaptive fourier neural operators. *arXiv preprint arXiv:2202.11214*, 2022.

[Peng *et al.*, 2020] Houwen Peng, Hao Du, Hongyuan Yu, Qi Li, Jing Liao, and Jianlong Fu. Cream of the crop: Distilling prioritized paths for one-shot neural architecture search. In *34th Conference on Neural Information Processing Systems, NeurIPS 2020*, volume 2020-, 2020.

[Qin *et al.*, 2024] Shaoxiang Qin, Fuyuan Lyu, Wenhui Peng, Dingyang Geng, Ju Wang, Xing Tang, Sylvie Leroyer, Naiping Gao, Xue Liu, and Liangzhu Leon Wang. Toward a better understanding of fourier neural operators from a spectral perspective, 2024.

[Rahaman *et al.*, 2019] Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred Hamprecht, Yoshua Bengio, and Aaron Courville. On the spectral bias of neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5301–5310. PMLR, 09–15 Jun 2019.

[Rahman *et al.*, 2023] Md Ashiqur Rahman, Zachary E Ross, and Kamyar Azizzadenesheli. U-NO: U-shaped

neural operators. *Transactions on Machine Learning Research*, 2023.

[Raissi and Karniadakis, 2018] Maziar Raissi and George Em Karniadakis. Hidden physics models: Machine learning of nonlinear partial differential equations. *Journal of Computational Physics*, 357:125–141, 2018.

[Raissi *et al.*, 2019] Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.

[Sitzmann *et al.*, 2020] Vincent Sitzmann, Julien N.P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In *NeurIPS*, 2020.

[Takamoto *et al.*, 2024] Makoto Takamoto, Timothy Praditia, Raphael Leiteritz, Dan MacKinlay, Francesco Alesiani, Dirk Pflüger, and Mathias Niepert. Pdebench: An extensive benchmark for scientific machine learning, 2024.

[Tran *et al.*, 2023] Alasdair Tran, Alexander Mathews, Lexing Xie, and Cheng Soon Ong. Factorized fourier neural operators. In *The Eleventh International Conference on Learning Representations*, 2023.

[Vemuri *et al.*, 2024] Sai Karthikeya Vemuri, Tim Büchner, Julia Niebling, and Joachim Denzler. Functional tensor decompositions for physics-informed neural networks, 2024.

[Vemuri *et al.*, 2025] Sai Karthikeya Vemuri, Tim Büchner, and Joachim Denzler. F-inr: Functional tensor decomposition for implicit neural representations, 2025.

[Wen *et al.*, 2022] Gege Wen, Zongyi Li, Kamyar Azizzadenesheli, Anima Anandkumar, and Sally M Benson. U-fno–an enhanced fourier neural operator-based deep-learning model for multiphase flow. *Advances in Water Resources*, page 104180, 2022.

[Xiao *et al.*, 2024] Zipeng Xiao, Siqi Kou, Zhongkai Hao, Bokai Lin, and Zhijie Deng. Amortized fourier neural operators. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[Zhi-Qin John Xu *et al.*, 2020] Zhi-Qin John Xu Zhi-Qin John Xu, Yaoyu Zhang Yaoyu Zhang, Tao Luo Tao Luo, Yanyang Xiao Yanyang Xiao, and Zheng Ma Zheng Ma. Frequency principle: Fourier analysis sheds light on deep neural networks. *Communications in computational physics*, 28(5):1746–1767, 2020.