PHILOSOPHICAL TRANSACTIONS A

royalsocietypublishing.org/journal/rsta

Research

Cite this article: Zhang Z, Liu H, Liao W, Lin G. 2025 Coefficient-to-Basis Network: a fine-tunable operator learning framework for inverse problems with adaptive discretizations and theoretical guarantees. *Phil. Trans. R. Soc. A* **383**: 20240054.
https://doi.org/10.1098/rsta.2024.0054

Received: 19 February 2025
Accepted: 10 June 2025

One contribution of 14 to a theme issue 'Frontiers of applied inverse problems in science and engineering'.

Subject Areas:
artificial intelligence, computational mathematics, applied mathematics, mechanical engineering

Keywords:
operator learning, inverse problem, fine tuning, approximation theory, generalization error

Author for correspondence:
Guang Lin
e-mail: guanglin@purdue.edu

THE ROYAL SOCIETY PUBLISHING

Coefficient-to-Basis Network: a fine-tunable operator learning framework for inverse problems with adaptive discretizations and theoretical guarantees

Zecheng Zhang[1], Hao Liu[2], Wenjing Liao[3] and Guang Lin[4]

[1]Department of Applied and Computational Mathematics and Statistics, University of Notre Dame, Notre Dame, IN, USA
[2]Hong Kong Baptist University, Hong Kong, Hong Kong
[3]Department of Mathematics, Georgia Institute of Technology, Atlanta, GA, USA
[4]Department of Mathematics & School of Mechanical Engineering, Purdue University, West Lafayette, IN, USA

ZZ, 0000-0002-1412-8457; GL, 0000-0002-0976-1987

We propose a Coefficient-to-Basis Network (C2BNet), a novel framework for solving inverse problems within the operator learning paradigm. C2BNet efficiently adapts to different discretizations through fine-tuning, using a pre-trained model to significantly reduce computational cost while maintaining high accuracy. Unlike traditional approaches that require retraining from scratch for new discretizations, our method enables seamless adaptation without sacrificing predictive performance. Furthermore, we establish theoretical approximation and generalization error bounds for C2BNet by exploiting low-dimensional structures in the underlying datasets. Our analysis demonstrates that C2BNet adapts to low-dimensional structures without relying on explicit encoding mechanisms, highlighting its robustness and efficiency. To validate our theoretical findings, we conducted extensive numerical experiments that showcase the superior performance of C2BNet on several inverse problems. The results confirm that C2BNet effectively balances computational efficiency and accuracy, making it a promising tool to solve inverse problems in scientific

computing and engineering applications.

This article is part of the theme issue 'Frontiers of applied inverse problems in science and engineering'.

1. Introduction

Operator learning between infinite-dimensional function spaces is an important task that arises in many disciplines of science and engineering. In recent years, deep neural networks have been successfully applied to learn operators for solving numerical partial differential equations (PDEs) [1–3], image processing [4] and inverse problems [5,6].

Operator learning is challenging in general, since the input and output functions lie in infinite-dimensional spaces. To address this difficulty, many deep operator learning approaches are proposed in an encoder–decoder framework. This framework employs encoders to map the input and output functions to finite-dimensional vectors and then learn a map between these dimension-reduced vectors. Popular deep operator learning methods in this encoder–decoder framework include PCANet with principal component analysis [1,7,8] and Fourier neural operators (FNOs) based on fast Fourier transforms [9–13]). These methods utilize deterministic or data-driven linear encoders and decoders for dimension reduction, and neural networks are used to learn the map. For nonlinear dimension reduction, autoencoders have demonstrated success in extracting low-dimensional nonlinear structures in data [14–17], and autoencoders have been applied to operator learning in Seidman *et al*. [18]; Kontolati *et al*. [19]; Liu *et al*. [20]. Other widely used deep operator learning architectures include the Deep Operator Network (DeepONet) [3,21–29]; random feature models [30], among others.

Apart from the computational advances, theoretical works were established to understand the representation and generalization capabilities of deep operator learning methods. A theoretical foundation on the approximation property of PCANet was established in Bhattacharya *et al*. [1]. This was followed by a more comprehensive study in Lanthaler [31], which derived both upper and lower bounds for such approximations. Further contributions were made in [17], where a generalization error bound was established for the encoder–decoder-based neural networks, including PCANet as a specific instance. The universal approximation property of FNOs was analysed in Kovachki *et al*. [32]. DeepONets were proposed based on a universal approximation theory in Chen & Chen [33], and were further analysed in Lanthaler *et al*. [34]; Schwab *et al*. [35]. More recently, neural scaling laws governing DeepONets were investigated in [36] based on approximation and generalization theories. Lanthaler [37] explored lower bounds on the parameter complexity of operator learning, demonstrating that achieving a power scaling law for general Lipschitz operators is theoretically impossible, despite the empirical observations in [3,8]. However, these theoretical frameworks remain limited in fully explaining the empirical success of deep neural networks in operator learning, primarily due to the inherent challenges posed by the curse of dimensionality.

In science and engineering applications, many datasets exhibit low-dimensional structures. For example, even though the images in ImageNet [38] have an ambient dimension exceeding 150 000. Pope *et al*. [39] showed that the ImageNet dataset has an intrinsic dimension of approximately 40. Many high-dimensional datasets exhibit repetitive patterns and special structures including rotation and translation, which contribute to a low intrinsic dimensionality [40,41]. In PDE-related inverse problems, one often needs to infer certain unknown parameters in the PDE from given observations, such as inferring the permeability in porous media equations given the solutions [42]. In inverse problems, the unknown parameters often exhibit a low-dimensional structure. For example, the unknown parameters considered in Lu *et al*. [43] and Hasani & Ward [44] are generated by just a few Fourier bases.

By leveraging low-dimensional structures, existing deep learning theory has shown that the approximation and generalization errors of deep neural networks for function estimation converge at a fast rate depending on the intrinsic dimension of data or learning tasks [45–49], by contrast to a slower rate of convergence in high-dimensional spaces [43,50,51]. A generalization error bound of an autoencoder-based neural network (AENet) for operator learning was established in Liu *et al*. [20], where the convergence rate depends on the intrinsic dimension of the dataset. Recently, generalization error bounds were established in Dahal *et al*. [52]; Havrilla & Liao [53] for generative models and transformer neural networks when the input data lie on a low-dimensional manifold. Another approach to mitigate the curse of dimensionality considers operators with special structures, such as those arising in elliptic PDEs [54] or holomorphic operators [55,56].

In this paper, we address operator learning problems arising from inverse problems [11,57] to determine inverse Quantities of Interest (QoIs) associated with PDEs given observations on PDE solutions. In many practical scenarios, the inverse QoIs in PDEs exhibit a simpler structure compared to the corresponding PDE solutions. For instance, in the context of heat diffusion in non-uniform materials, the diffusivity parameter is inherently material dependent. If the domain can be partitioned into several subregions, each characterized nearly by a uniform material property, the diffusivity parameter can be represented by a piecewise constant function [58]. A similar framework is explored in Vaidya *et al*. [59] for the convection–diffusion of solutes in heterogeneous media. When the unknown parameter in PDEs is piecewise constant, it can be expressed by a linear combination of characteristic functions, each corresponding to a distinct subregion. These characteristic functions form a set of orthogonal bases. By contrast, the solutions to PDEs often exhibit more complex structures. Consequently, it is natural to assume that the PDE parameters possess low-dimensional linear structures, while the PDE solutions exhibit low-dimensional nonlinear structures.

Motivated by this observation, we propose a novel framework termed a Coefficient-to-Basis Network (C2BNet) to solve inverse problems. The proposed network architecture comprises two key components: (i) a coefficient network, which maps the input function (representing the PDE solution) to the coefficients corresponding to the basis functions of the output (PDE parameter), and (ii) a basis network (a linear layer), which is responsible for learning the basis functions in the output space. C2BNet is different from existing encoder–decoder based networks, such as PCANet [1] and AENet [20]. PCANet uses principle component analysis (PCA) for dimension reduction, which can only discover low-dimensional linear structures, while C2BNet can learn low-dimensional nonlinear structures of PDE solutions. AENet uses an autoencoder to explore low-dimensional nonlinear structures in data. The training of AENet contains two steps: in the first step, an autoencoder is trained to learn low-dimensional data structures in the input. In the second step, another network is trained to learn a mapping from the input latent variable given by the trained autoencoder to the output. In C2BNet, we do not need to explicitly train an autoencoder for dimension reduction. Both the coefficient network and basis network are trained together, which simplifies the training process.

A significant challenge in solving inverse problems by machine learning lies in adapting to new tasks or inferring QoIs in different regions of the domain based on new data. A common scenario arises when the new QoIs are discretized on a finer mesh, leading to a higher-dimensional output compared with the previous tasks. To address this challenge, we propose a fine-tuning approach for a pre-trained network that was initially trained on coarser discretizations. Our derivation demonstrates that only updating a linear layer of the pre-trained network is sufficient to adapt to the new task on finer discretizations. This approach significantly reduces the computational complexity associated with retraining the entire network on the new dataset while maintaining the accuracy of the prediction. By assuming that the input functions reside on a low-dimensional manifold, we establish approximation and generalization error bounds for our proposed C2BNet. In particular, these error bounds depend crucially on the intrinsic dimension of the input functions. Our contributions are summarized as follows:

— We introduce C2BNet, a novel framework for learning operators designed for PDE-based inverse problems. C2BNet can be efficiently fine-tuned to accommodate different discretizations using a pre-trained network, significantly reducing the computational cost of retraining without sacrificing accuracy.
— We establish approximation and generalization error bounds for C2BNet incorporating low-dimensional structures in datasets. Our results demonstrate that C2BNet adapts to low-dimensional data structures without an additional encoder–decoder mechanism.
— We validate the performance of C2BNet and the theoretical findings through comprehensive numerical experiments.

The remainder of this paper is structured as follows. In §2, we introduce some concepts and background definitions necessary for an understanding of the proposed framework. In §3 we present details of the proposed C2BNet. A theoretical analysis of C2BNet is provided in §4, with proofs detailed in appendix A. The efficacy of C2BNet is demonstrated through a series of numerical experiments in §5. Finally, we conclude the paper in §6.

Notation: In this paper, we use bold letters to denote vectors and normal letters to denote scalars. Calligraphic letters are used to denote sets. For a set Ω, we use $|\Omega|$ to denote its volume.

2. Preliminary

We first introduce some definitions about manifolds. We refer readers to Tu [60] and Lee [61] for more details.

Definition 1 (Chart)**.** Let $\mathcal{M}$ be a d-dimensional manifold embedded in $\mathbb{R}^D$. A chart of $\mathcal{M}$ is a pair (Q, ϕ) where $Q \subset \mathcal{M}$ is an open subset of $\mathcal{M}$, $\phi : Q \to \mathbb{R}^d$ is a homeomorphism.

The transformation ϕ in a chart defines a coordinate map on Q. An atlas of $\mathcal{M}$ is a collection of charts that covers $\mathcal{M}$:

Definition 2 (C^s Atlas)**.** Let $\{(Q_k, \phi_k)\}_{k\in\mathcal{K}}$ be a collection of charts of $\mathcal{M}$ with $\mathcal{K}$ denoting the set of indices. It is a C^s atlas of $\mathcal{M}$ if

(i) $\cup_{k\in\mathcal{K}} Q_k = \mathcal{M}$,
(ii) the mappings

$$\phi_j \circ \phi_k^{-1} : \phi_k\left(Q_j \cap Q_k\right) \to \phi_j\left(Q_j \cap Q_k\right) \text{ and } \phi_k \circ \phi_j^{-1} : \phi_j\left(Q_j \cap Q_k\right) \to \phi_k\left(Q_j \cap Q_k\right)$$

are C^s for any $j, k \in \mathcal{K}$.

A finite atlas is an atlas with a finite number of charts.

On a smooth manifold, we define C^s functions as follows.

Definition 3 (C^s functions on $\mathcal{M}$)**.** Let $\mathcal{M}$ be a smooth manifold and $f : \mathcal{M} \to \mathbb{R}$ be a function defined on $\mathcal{M}$. The function is called a C^s function on $\mathcal{M}$ if for any chart (Q, ϕ) of $\mathcal{M}$, the composition $f \circ \phi^{-1} : \phi(Q) \to \mathbb{R}$ is a C^s function.

To measure the complexity of a manifold, we define reach as follows.

Definition 4 (Reach [62,63])**.** Let $\mathcal{M}$ be a manifold embedded in $\mathbb{R}^D$. We define

$$G = \left\{\mathbf{x} \in \mathbb{R}^D : \exists \mathbf{y} \neq \mathbf{z} \in \mathcal{M} \text{ such tha } d(\mathbf{x}, \mathcal{M}) = \|\mathbf{x} - \mathbf{y}\|_2 = \|\mathbf{x} - \mathbf{z}\|_2\right\},$$

where $d(\mathbf{x}, \mathcal{M})$ is the distance from $\mathbf{x}$ to $\mathcal{M}$. The reach of $\mathcal{M}$ is defined as

$$\tau = \inf_{\mathbf{x}\in\mathcal{M}} \inf_{\mathbf{y}\in G} \|\mathbf{x}-\mathbf{y}\|_2.$$

The reach of a manifold gives a characterization of curvature. A hyper-plane has a reach $\tau = \infty$, and a hyper-sphere with radius r has a reach $\tau = r$.

In this paper, we consider feedforward neural networks in the form of

$$f_{\mathrm{NN}}(\mathbf{x}) = W_L \cdot \mathrm{ReLU}(W_{L-1} \cdots \mathrm{ReLU}(W_1\mathbf{x} + \boldsymbol{b}_1) \cdots + \boldsymbol{b}_{L-1}) + \boldsymbol{b}_L, \tag{2.1}$$

where W_l's are weight matrices, $\boldsymbol{b}_l$'s are bias and $\mathrm{ReLU}(a) = \max\{a, 0\}$ is the rectified linear unit and is applied elementwise to its argument. The function defined in equation (2.1) is a network with L layers. The number of layers refers to the number of weight matrices that are used in the network. The width of the l-th layer refers to the number of rows of W_l, and the width of the network is the largest number of rows among all weight matrices.

We consider the following network class:

$$\begin{aligned}&\mathcal{F}_{\mathrm{NN}}\left(d_1, d_2, L, p, K, \kappa, R\right) = \\ &\quad\Big\{\mathbf{f}_{\mathrm{NN}} = \left[f_1, \cdots, f_{d_2}\right]^\top \,|\, f_k : \Omega \to \mathbb{R} \text{ is in the form of (1) with } L \text{ layers, width bounded by } p \\ &\quad \|f_k\|_{L^\infty(\Omega)} \le R, \|W_l\|_{\infty,\infty} \le \kappa, \|\boldsymbol{b}_l\|_\infty \le \kappa, \sum_{l=1}^{L} \|W_l\|_0 + \|b_l\|_0 \le K, \forall l \Big\}\end{aligned} \tag{2.2}$$

In equation (2.2), $\|W_l\|_\infty = \max_{i,j} |(W_l)_{i,j}|$, $\|b_l\|_\infty = \max_i |(\boldsymbol{b}_l)_i|$ and $\|\cdot\|_0$ gives the number of non-zero elements of its argument.

3. Coefficient-to-Basis Network (C2BNet) for operator learning

Our objective is to learn an unknown operator

$$\Psi : \mathcal{X} \to \mathcal{Y}, \tag{3.1}$$

where $\mathcal{X} \subset L^2(\Omega_{\mathcal{X}})$ and $\mathcal{Y} \subset L^2(\Omega_{\mathcal{Y}})$ are input and output function sets with domain $\Omega_{\mathcal{X}} \subset \mathbb{R}^{s_1}$ and $\Omega_{\mathcal{Y}} \subset \mathbb{R}^{s_2}$ respectively. In addition, $\mathcal{X}$ and $\mathcal{Y}$ belong to separable Hilbert spaces with inner products $\langle u_1, u_2\rangle_{\mathcal{X}}$ and $\langle v_1, v_2\rangle_{\mathcal{Y}}$, respectively.

In PDE-based inverse problems, the input set $\mathcal{X}$ contains the PDE solutions and the output set $\mathcal{Y}$ contains the PDE parameters to be determined. Motivated by low-dimensional linear structures in the PDE parameters [58,59], we assume that the output functions in $\mathcal{Y}$ approximately lie in a low-dimensional subspace.

Assumption 1. Let $d_2 > 0$ be an integer. Suppose there exists a set of orthonormal functions $\{\omega_k\}_{k=1}^{d_2}$, i.e. $\langle \omega_j, \omega_k\rangle_{\mathcal{Y}} = 1$ for $j = k$, and is $\langle \omega_j, \omega_k\rangle_{\mathcal{Y}} = 0$ for $j \neq k$, so that any $v \in \mathcal{Y}$ satisfies

$$\left\| v - \sum_{k=1}^{d_2} \langle v, \omega_k\rangle_{\mathcal{Y}} \omega_k \right\|_{L^\infty(\Omega_{\mathcal{Y}})} \le \zeta,$$

where ζ denotes the approximation error which may depend on d_2.

We denote

$$\mathrm{Proj}(v) = \sum_{k=1}^{d_2} \alpha_k^v(v)\omega_k, \quad \text{with } \alpha_k^v(v) = \langle v, \omega_k\rangle_{\mathcal{Y}}. \tag{3.2}$$

Assumption 1 ensures the existence of a set of orthonormal bases whose span can approximate functions in $\mathcal{Y}$ with ζ error. This assumption is inspired by finite element methods and existing

works on operator learning. In finite element methods [64], the output function space is approximated by a finite element space, spanned by a set of basis functions. For the encoder–decoder-based operator learning approaches, such as PCANet and FNOs [1,10,17,31], linear encoders and decoders are used based on certain bases. In these works, the bases are either deterministic, such as Fourier bases and Legendre polynomials, or estimated by data-driven tools, such as PCA.

For any input $u \in \mathcal{X}$ and output $v \in \mathcal{Y}$, we denote their discretized counterparts by $\mathbf{u} = S_{\mathcal{X}}(u) \in \mathbb{R}^{D_1}$ and $\mathbf{v} = S_{\mathcal{Y}}(v) \in \mathbb{R}^{D_2}$ respectively, where $S_{\mathcal{X}}$ and $S_{\mathcal{Y}}$ denote the corresponding discretization operators. For a given discretization operator $S_{\mathcal{X}}$, we denote the induced inner product in $\mathbb{R}^{D_1}$ by $\langle \cdot, \cdot \rangle_{S_{\mathcal{X}}}$. One way to define such an inner produce is by using a quadrature rule:

$$\langle S_{\mathcal{X}}(u_1), S_{\mathcal{X}}(u_2) \rangle_{S_{\mathcal{X}}} = \sum_{k=1}^{D_1} \tau_k (\mathbf{u}_1)_k (\mathbf{u}_2)_k, \tag{3.3}$$

where $\tau_k > 0$ are the weights in the quadrature rule. When the functions in $\mathcal{X}$ are smooth and the discretization grid of $S_{\mathcal{X}}$ is sufficiently fine, we expect $\|S_{\mathcal{X}}(u)\|_{S_{\mathcal{X}}} \approx \|u\|_{\mathcal{X}}$ for any $u \in \mathcal{X}$. We make the following assumption on the discretization operator $S_{\mathcal{Y}}$:

Assumption 2. Assume that, for any $v \in \mathcal{Y}$,

$$0.5\|v\|_{\mathcal{Y}} \le \|S_{\mathcal{Y}}(v)\|_{S_{\mathcal{Y}}} \le 2\|v\|_{\mathcal{Y}}, \quad 0.5\|\mathrm{Proj}(v)\|_{\mathcal{Y}} \le \|S_{\mathcal{Y}} \circ \mathrm{Proj}(v)\|_{S_{\mathcal{Y}}} \le 2\|\mathrm{Proj}(v)\|_{\mathcal{Y}}.$$

Assumption 2 holds as long as functions in $\mathcal{Y}$ are uniformly regular and the discretization grid is sufficiently fine. For example, according to the Nyquist–Shannon sampling theorem [65], bandlimited functions can be completely determined from their discretized counterparts on a sufficiently fine grid. A specific example is provided in [20, Example 1] to demonstrate that assumption 2 can be satisfied if the functions in $\mathcal{Y}$ are band limited and the discretization grid is sufficiently fine. For another example, when functions in $\mathcal{Y}$ are bounded and second-order differentiable, assumption 2 can be satisfied by choosing $\langle \cdot, \cdot \rangle_{S_{\mathcal{Y}}}$ as the composite trapezoidal rule with sufficiently fine grids. We can find the weight τ_k's in equation (3.3) using a quadrature rule to approximate the continuous inner product by the discretized counterpart.

(a) Coefficient-to-Basis Network (C2BNet) architecture

Our network structure is designed based on the decomposition in equation (3.2), which consists of two components: the coefficients $\{\alpha_k^v(v)\}_{k=1}^{d_2}$ and the bases $\{\omega_k\}_{k=1}^{d_2}$. We design a network for each component. In equation (3.2), for each k, the coefficient $\alpha_k^v(v)$ is a functional of the output function v in $\mathcal{Y}$. Since $v = \Psi(u)$, we have

$$\alpha_k^v(v) = \alpha_k^v \circ \Psi(u), \tag{3.4}$$

which implies that each coefficient is a functional of the input function u as well. We construct a coefficient network $\mathbf{f}_{coef} : \mathbb{R}^{D_1} \to \mathbb{R}^{d_2}$ to learn the mapping in equation (3.4):

3.4:

$$\mathbf{u} \xrightarrow{\mathbf{f}_{coef}} [\alpha_1^v \circ \Psi(u), ..., \alpha_{d_2}^v \circ \Psi(u)]^\top,$$

where $\mathbf{u} = S_{\mathcal{X}}(u)$ denotes the discretized counterpart of u.

Given a discretization grid in $\Omega_{\mathcal{Y}}$ associated with the discretization operator $S_{\mathcal{Y}}$, we can represent the basis functions $\{\omega_k\}_{k=1}^{d_2}$ by a set of vectors $\{S_{\mathcal{Y}}(\omega_k)\}_{k=1}^{d_2}$. In C2BNet, we use a linear layer to learn this set: $\mathbf{f}_{basis} : \mathbb{R}^{d_2} \to \mathbb{R}^{D_2}$.

Our operator network Ψ_{NN} that approximates the target operator Ψ in equation (3.1) is constructed as

$$\Psi_{\mathrm{NN}} = \mathbf{f}_{basis} \circ \mathbf{f}_{coef}. \tag{3.5}$$

The architecture of Ψ_{NN} is illustrated in figure 1a.

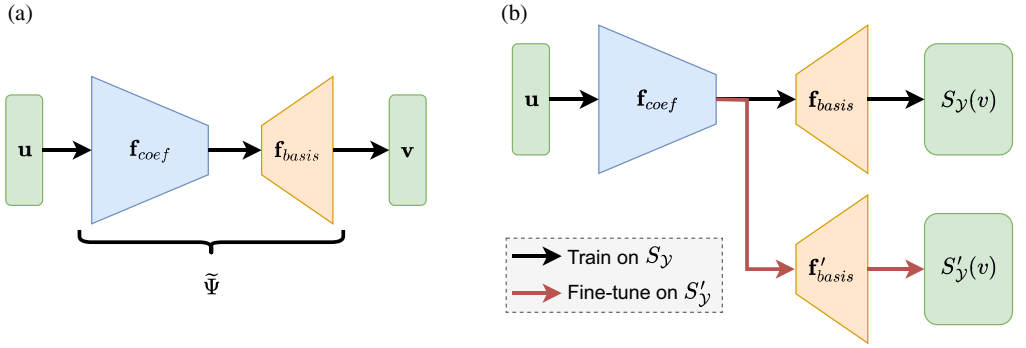

Figure 1. (a) An illustration of C2BNet in equation (3.5). (b) Fine-tuning from discretization $S_{\mathcal{Y}}$ to $S'_{\mathcal{Y}}$.

Although both C2BNet can DeepONet have two subnetworks, their structures are different. DeepONet approximates an operator through a linear combination of learnable basis functions (trunk) and the learnable coefficients (branch). However, C2BNet is a feedforward network. It is a composition of two networks instead of a product of two networks.

(b) Fine-tuning on a new discretization in $\Omega_{\mathcal{Y}}$

A key advantage C2BNet is adapting to different discretizations. Once the network is trained on a specific discretization associated with $S_{\mathcal{Y}}$, it can be efficiently fine-tuned to accommodate a new discretization associated with $S'_{\mathcal{Y}}$. As indicated by equation (3.4), the coefficients $\{\alpha_k^v(v)\}_{k=1}^{d_2}$ are determined solely by the input function u and are independent of the discretization $S_{\mathcal{Y}}$. In our framework, altering the discretization in $\Omega_{\mathcal{Y}}$ only requires updating the basis network $\mathbf{f}_{basis}$ to represent $\{S'_{\mathcal{Y}}(\omega_k)\}_{k=1}^{d_2}$ instead of $\{S_{\mathcal{Y}}(\omega_k)\}_{k=1}^{d_2}$. Consequently, if the coefficient network $\mathbf{f}_{coef}$ learns the coefficients well, only $\mathbf{f}_{basis}$ needs to be fine-tuned in the new dataset to learn the updated basis functions. This strategy eliminates the need to retrain the entire network, as only a single linear layer needs to be updated, which significantly reduces the computational cost. Our fine-tuning strategy is illustrated in figure 1b. In fact, since the fine-tuning of C2BNet only updates the last linear layer, the training loss is quadratic with respect to the weights of the last layer. We can solve it explicitly instead of training a network using stochastic gradient decent or any other algorithms. This property distinguishes C2BNet from other encoder–decoder based networks which need to retrain part or the whole network.

4. Approximation and generalization theory of C2BNet

In this section, approximation and generalization error bounds are established for the C2BNet in §3. C2BNet is designed to utilize low-dimensional linear structures in output functions. By further leveraging low-dimensional nonlinear structures in input functions, we prove approximation and generalization error bounds depending on the intrinsic dimension of the input functions. All proofs are deferred until appendix A.

Our theoretical analysis requires some assumptions. The first assumption below says that the functions in $\mathcal{X}$ and $\mathcal{Y}$ have a bounded domain and the function values are bounded.

Assumption 3. Suppose the followings hold for $\mathcal{X}$ and $\mathcal{Y}$: there exist $B_1, B_2, M_1, M_2 > 0$ such that

(i) For any $\mathbf{x} \in \Omega_{\mathcal{X}}, \mathbf{y} \in \Omega_{\mathcal{Y}}$, $\|\mathbf{x}\|_\infty \le B_1$, $\|\mathbf{y}\|_\infty \le B_2$.
(ii) For any $u \in \mathcal{X}, v \in \mathcal{Y}$, $\|u\|_{L^\infty(\Omega_{\mathcal{X}})} \le M_1$, $\|v\|_{L^\infty(\Omega_{\mathcal{Y}})} \le M_2$.

Assumption 3 implies that the domain and output of any function in $\mathcal{X}$ and $\mathcal{Y}$ are bounded. The next assumption leverages low-dimensional structures in input functions:

Assumption 4. Suppose the input function set $\mathcal{X}$ satisfies the followings:

(i) Suppose $S_{\mathcal{X}}$ discretizes functions in $\mathcal{X}$ without loss of information: there exists a map $D_{\mathcal{X}} : \mathbb{R}^{D_1} \to \mathcal{X}$ such that

$$D_{\mathcal{X}} \circ S_{\mathcal{X}}(u) = u$$

for any $u \in \mathcal{X}$.

(ii) Suppose that $\{\mathbf{u} = \mathcal{S}_{\mathcal{X}}(u) : u \in \mathcal{X}\}$ is located on a d_1-dimensional Riemannian manifold $\mathcal{M}$ isometrically embedded in $\mathbb{R}^{D_1}$, with a positive reach $\tau > 0$.

(iii) Suppose $D_{\mathcal{X}} : \mathcal{M} \to \mathcal{X}$ is Lipschitz. Let $\{(Q_k, \phi_k)\}_{k \in \mathcal{K}}$ be an atlas of $\mathcal{M}$. For any given chart (U_k, ϕ_k) in this atlas, there exists a constant C such that for any $\mathbf{z}_1, \mathbf{z}_2 \in \phi_k(U)$

$$\|D_{\mathcal{X}} \circ \phi_k^{-1}(\mathbf{z}_1) - D_{\mathcal{X}} \circ \phi_k^{-1}(\mathbf{z}_2)\|_{L^\infty(\Omega_{\mathcal{X}})} \leq C\|\mathbf{z}_1 - \mathbf{z}_2\|_2.$$

Assumption 4 assumes that $\mathcal{X}$ processes a low-dimensional nonlinear structure. Assumption 4(i) imposes a one-to-one correspondence between each function in $\mathcal{X}$ and their discretized counterpart. Assumption 4(i) holds when the functions in $\mathcal{X}$ are bandlimited and the discretization grid is sufficiently fine. Assumptions 4(ii)–(iii) assume that $\mathcal{X}$ exhibits a low-dimensional nonlinear structure.

Our last assumption is about the target operator Ψ:

Assumption 5. Suppose that the operator Ψ is Lipschitz with Lipschitz constant L_Ψ:

$$\|\Psi(u_1) - \Psi(u_2)\|_{\mathcal{Y}} \leq L_\Psi \|u_1 - u_2\|_{\mathcal{X}}.$$

Assumption 5 imposes a Lipschitz property of the target operator Ψ, which is common in the operator learning literature.

We next provide a simple example showing that assumptions 1–5 are satisfied.

Example 1. Consider the transport equation

$$\begin{cases} u_t = f(t)u_x \quad (x,t) \in [0, 2\pi] \times [0,1], \\ u(x,0) = \sin(x), \\ u(x,t) \text{ is periodic in space.} \end{cases}$$

The objective is to infer $f(t)$ from observed data sampled from $u(x,t)$. Suppose $f(x)$ is a linear function:

$$f(t) = a + bt,$$

for $a, b \in [0,1]$. Then $d_2 = 2, \omega_1(t) = 1, \omega_2(t) = 2\sqrt{3}(t - 1/2), \zeta = 0$, and assumption 1 is satisfied.

For any given a, b, the PDE solution is

$$u(x,t) = \sin\left(x + \left(at + \frac{b}{2}t^2\right)\right).$$

Assumption 3 is satisfied. The operator Ψ maps $u(x,t)$ to $f(t) = \arcsin(u(x,t) - x)$, assumption 5 is satisfied. We sample N_x equally spaced points along x and N_t equally spaced points along t. When $N_x \geq 2, N_t \geq 3$, the expression of $u(x,t)$ can be uniquely determined from the sampled points. Assumption 4(i) is satisfied by choosing $D_{\mathcal{X}}$ as this mapping. Since both $u(x,t)$ and $\mathbf{u}$ are parameterized by (a,b), they are located on a manifold $\mathcal{M}$ with $d_1 = 2$. Assumption 4(ii) is satisfied. Let ϕ

be the mapping from $\mathbf{u}$ to (a,b). Then $(\mathcal{M},\phi)$ is a chart (and also an atlas) of $\mathcal{M}$. For $(a,b)\in\phi(\mathcal{M})$, we have $D_{\mathcal{X}}\circ\phi^{-1}(a,b)=\sin\left(x+\left(at+\frac{b}{2}t^2\right)\right)$, which is Lipschtiz in a and b. Assumption 4(iii) is satisfied. By choosing $\langle\cdot,\cdot\rangle_{S_{\mathcal{X}}}$ as the composite trapezoidal rules, assumption 2 is satisfied when N_x, N_t is sufficiently large. In this example, all of assumptions 1–5 are satisfied.

(a) Approximation theory

Our first theoretical result is on the approximation power of C2BNet for Lipschitz operators under assumptions 2–5.

Theorem 1. Let $B_1, B_2, M_1, M_2, L_\Psi > 0$, and suppose assumptions 2–4 hold. For any $\varepsilon > 0$, there exists a network architecture $\mathcal{F}_{coef}=\mathcal{F}_{\mathrm{NN}}(D_1,d_2,L,p,K,\kappa,R)$ with

$$\begin{aligned} &L=O\left(\log\frac{1}{\varepsilon}\right),\ p=O(\varepsilon^{-d_1}),\ K=O\left(\varepsilon^{-d_1}\log\frac{1}{\varepsilon}+D_1\log\frac{1}{\varepsilon}+D_1\log D_1+D_2\right),\\ &\kappa=O(\varepsilon^{-1}),\ R=M_2|\Omega_{\mathcal{Y}}|, \end{aligned} \tag{4.1}$$

and a linear network $\mathcal{F}_{basis}=\mathcal{F}_{\mathrm{NN}}(d_2,D_2,1,D_2,d_2D_2,M_2,M_2)$ such that for any operator Ψ satisfying assumption 5, such an architecture gives rise to $\widetilde{\mathbf{f}}_{coef}\in\mathcal{F}_{coef}, \widetilde{\mathbf{f}}_{basis}\in\mathcal{F}_{basis}$ and $\widetilde{\Psi}=\widetilde{\mathbf{f}}_{basis}\circ\widetilde{\mathbf{f}}_{coef}$ with

$$\sup_{u\in\mathcal{X}}\|S_{\mathcal{Y}}\circ\mathrm{Proj}\circ\Psi(u)-\widetilde{\Psi}\circ S_{\mathcal{X}}(u)\|_\infty\le\varepsilon. \tag{4.2}$$

The constant hidden in O depends on $d_1, d_2, M_1, M_2, |\Omega_{\mathcal{X}}|, |\Omega_{\mathcal{Y}}|, L_\Psi, \tau$ and the surface area of $\mathcal{M}$.

Theorem 1 is proved in appendix A.1. Theorem 1 shows that if the network architecture is properly chosen, C2BNet can universally approximate Lipschitz operators after a linear projection with an arbitrary accuracy. Furthermore, the size of the network scales with ε with an exponent depending on the intrinsic dimension d_1 of the input functions , instead of the ambient dimension D_1.

Our C2BNet in theorem 1 is constructed to approximate Proj$\circ\Psi$. When the outputs of Ψ approximately lie in a low-dimensional subspace as assumed in assumption 1, we can combine theorem 1 and assumption 1 to guarantee an approximation error of Ψ.

Corollary 1. Let $B_1, B_2, M_1, M_2, L_\Psi, \zeta > 0$, and suppose that assumptions 1–4 hold. For any $\varepsilon > 0$, there exists a network architecture $\mathcal{F}_{coef}=\mathcal{F}_{\mathrm{NN}}(D_1,d_2,L,p,K,\kappa,R)$ with

$$\begin{aligned} &L=O\left(\log\frac{1}{\varepsilon}\right),\ p=O(\varepsilon^{-d_1}),\ K=O\left(\varepsilon^{-d_1}\log\frac{1}{\varepsilon}+D_1\log\frac{1}{\varepsilon}+D_1\log D_1+D_2\right),\\ &\kappa=O(\varepsilon^{-1}),\ R=M_2|\Omega_{\mathcal{Y}}| \end{aligned} \tag{4.3}$$

and a linear network $\mathcal{F}_{basis}=\mathcal{F}_{\mathrm{NN}}(d_2,D_2,1,D_2,d_2D_2,M_2,M_2)$ such that for any operator Ψ satisfying assumption 5, such an architecture gives rise to $\widetilde{\mathbf{f}}_{coef}\in\mathcal{F}_{coef}, \widetilde{\mathbf{f}}_{basis}\in\mathcal{F}_{basis}$ and $\widetilde{\Psi}=\widetilde{\mathbf{f}}_{basis}\circ\widetilde{\mathbf{f}}_{coef}$ with

$$\sup_{u\in\mathcal{X}}\|S_{\mathcal{Y}}\circ\Psi(u)-\widetilde{\Psi}\circ S_{\mathcal{X}}(u)\|_\infty\le\zeta+\varepsilon. \tag{4.4}$$

The constant hidden in O depends on $d_1, d_2, M_1, M_2, |\Omega_{\mathcal{X}}|, |\Omega_{\mathcal{Y}}|, L_\Psi, \tau$ and the surface area of $\mathcal{M}$.

Corollary 1 is proved in appendix A.2. Corollary 1 gives an upper bound of C2BNet for approximating the original Lipschitz operators. The bound depends on the approximation error by projection and the approximation error by networks.

(b) Generalization error

Suppose we are given input–output pairs $\mathcal{S} = \{u_i, \widehat{v}_i\}_{i=1}^n$ with $u_i \in \mathcal{X}$ sampled from a probability distribution ρ and

$$\widehat{v}_i = \Psi(u_i) + \xi_i,$$

where ξ_i's represent independent and identically distributed (i.i.d.) noise satisfying the following assumption:

Assumption 6. Suppose ξ_i's are i.i.d. copies of $\xi \in \mathcal{Y}$. For any $\mathbf{y} \in \Omega_{\mathcal{Y}}$, $\xi_i(\mathbf{y})$ follows a sub-Gaussian distribution with variance proxy σ^2.

Given a network architecture $\mathcal{F}_{coef}$ and $\mathcal{F}_{basis}$, we define the C2BNet class

$$\mathcal{F} = \{\mathbf{f}_{basis} \circ \mathbf{f}_{coef} : \mathbf{f}_{coef} \in \mathcal{F}_{coef}, \mathbf{f}_{basis} \in \mathcal{F}_{basis}\}. \tag{4.5}$$

Given the training data $\{\mathbf{u}_i, \widehat{\mathbf{v}}_i\}_{i=1}^n$ with $\mathbf{u}_i = S_{\mathcal{X}}(u_i), \widehat{\mathbf{v}}_i = S_{\mathcal{Y}}(\widehat{v}_i)$, we train C2BNet through the following empirical risk minimization:

$$\widehat{\Psi} = \underset{\Psi_{\mathrm{NN}} \in \mathcal{F}}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} \|\Psi_{\mathrm{NN}}(\mathbf{u}_i) - \widehat{\mathbf{v}}_i\|_{S_{\mathcal{Y}}}^2, \tag{4.6}$$

The generalization error of $\widehat{\Psi}$ satisfies the following upper bound.

Theorem 2. Let $B_1, B_2, M_1, M_2, L_\Psi, \zeta > 0$. Suppose assumptions 1–6 hold. Set the network architecture $\mathcal{F}_{coef} = \mathcal{F}_{\mathrm{NN}}(D_1, d_2, L, p, K, \kappa, R), \mathcal{F}_{basis} = \mathcal{F}_{\mathrm{NN}}(d_2, D_2, 1, D_2, d_2 D_2, M_2, M_2)$ with

$$\begin{aligned} &L = O(\log n),\ p = O(n^{\frac{2d_1}{2+d_1}}),\ K = O(n^{\frac{2d_1}{2+d_1}} \log n + D_1 \log n + D_1 \log D_1 + D_2), \\ &\kappa = O(\log n),\ R = M_2 |\Omega_{\mathcal{Y}}| \end{aligned} \tag{4.7}$$

and consider the C2BNet class $\mathcal{F}$ defined in equation (4.5). The minimizer of equation (4.6), denoted by $\widehat{\Psi}$, satisfies

$$\mathbb{E}_{\mathcal{S}} \mathbb{E}_{u \sim \rho} \left[\frac{1}{D_2} \|\widehat{\Psi} \circ S_{\mathcal{X}}(u) - S_{\mathcal{Y}} \circ \Psi(u)\|_2^2 \right] \leq C D_1 (\log D_1) n^{-\frac{2}{2+d_1}} \log^3 n + 8\zeta^2, \tag{4.8}$$

where C is a constant. Both C and the constants hidden in O depend on $d_1, d_2, M_1, M_2, |\Omega_{\mathcal{X}}|, |\Omega_{\mathcal{Y}}|$, L_Ψ, σ, τ and the surface area of $\mathcal{M}$.

Theorem 2 is proved in appendix A.3. The error bound in theorem 2 has two components: the first is the network estimation error, which converges to zero as n goes to infinity; the second represents the loss of information while output functions are projected to a d_2-dimensional subspace. If functions in $\mathcal{Y}$ lie on a d_2-dimensional subspace, then $\zeta = 0$ and the squared generalization error in theorem 2 converges at the rate $n^{-\frac{2}{2+d_1}}$. This rate of convergence depends on the intrinsic dimension d_1 instead of the ambient dimension D_1. By solving equation (4.6), C2BNet can adapt to low-dimensional structures in input functions and enjoys a fast convergence rate depending on the intrinsic dimension of the input functions, attenuating the curse of dimensionality. If $\mathcal{Y}$ is not exactly lying in a d_2-dimensional subspace, the projection error usually decays as d_2 increases. We can choose d_2 to be large enough so that ζ is small.

5. Numerical experiments

In this section, we demonstrate the effectiveness of C2BNet through three sets of numerical experiments. To show the effectiveness and robustness of the methods, we use the same network

structures in all examples. Specifically, the network has the size $d_{input} \to 100 \to 100 \to 100 \to d_{low} \to d_{output}$, where the number indicates the width of each layer (the number of neurons of each layer), and each layer is activated by the ReLU and with bias, except the last layer. The d_{input}, d_{output} and d_{low} are problem-dependent and are specified in each example. All codes and code running results will be published on Google Colab and Github after the paper is published; we can provide the link if requested by the readers.

(a) Radiative transfer equation

In this example, we examine the radiative transfer equation (RTE) [66–68] with a contrast scattering parameter $\sigma(x)$, the goal is to infer the inverse QoIs $\sigma(x)$ given the solution of the PDE. The RTE is a fundamental model in optical tomography and represents a classic inverse problem. The objective is to recover the scattering parameter based on given observations. The RTE is described by the following equation:

$$s \cdot \nabla I(x,s) = \frac{\sigma(x,\omega)}{\epsilon}\left(\int_{\mathcal{S}^{n-1}} I(x,s')\, ds' - I(x,s) \right), \quad \forall x \in D,\ s \in \mathcal{S}^{n-1}.$$

Here, s is a vector in the unit sphere $\mathcal{S}^{n-1}$, and n denotes the spatial dimension of the problem.

In our experiments, we set $n = 2$, making $\mathcal{S}^{n-1} = \mathcal{S}^1$, which corresponds to the unit circle. In addition, we set $\epsilon = 1$ and the domain $D = [0,1]^2$.

To close the model, Dirichlet boundary conditions are imposed. Specifically, for directions entering the domain ($s \cdot \mathbf{n} < 0$), the boundary condition is defined as $I(x,s) = I_{\text{in}}$. This applies to the boundary subset

$$\Gamma^- := \{(x,s) \in \partial D \times \mathcal{S}^{n-1} : s \cdot \mathbf{n} < 0\},$$

where $\mathbf{n}$ is the unit outward normal vector at $x \in \partial D$. The condition is expressed as

$$I(x,s) = I_{\text{in}}(x,s) \quad \text{for all } (x,s) \in \Gamma^-.$$

In our examples, the top, bottom and right-hand boundaries are assigned zero incoming boundary conditions, while the left-hand boundary is assumed to have a non-zero flow, injecting energy or particles into the domain.

The term 'contrast scattering' refers to the difference in scattering properties between specific channels and the surrounding background. Each scattering includes five channels, where the size of each is randomly determined by two free variables: length and width. Figure 2 illustrates four realizations of the scattering parameter σ, highlighting the variability in channel structures across different samples.

(i) Results for RTE

In this section, we present the numerical results. For input, the solution (observation) is uniformly discretized on an 11×11 spatial mesh, with four velocity directions uniformly distributed on the unit circle. The inverse QoIs correspond to the target scattering parameter, which is discretized on a 10×10 spatial mesh, $d_{low} = 50$ for this example. In figure 3, we present the error decay in the L_1 norm with respect to the number of training samples. From the left-hand panel of the figure, we observe that the relative error decreases to as low as 3% when 144 training samples are used, demonstrating the effectiveness of the neural network. Moreover, the right-hand panel shows the error decay on a log scale linearly, validating the power-law behaviour shown in theorem 2.

(b) Elliptic equation

In this example, we consider the elliptic equation,

$$-\nabla \cdot \kappa(x,y)\nabla u = f, x \in [0,1]^2,$$

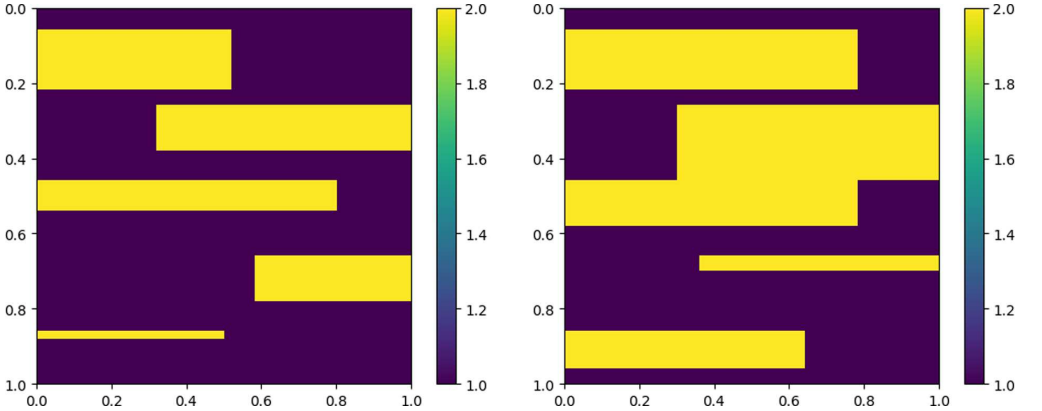

Figure 2. Demonstration of two scatter parameters for RTE. Each scattering field has five channels and each channels has two degrees of freedom width and height.

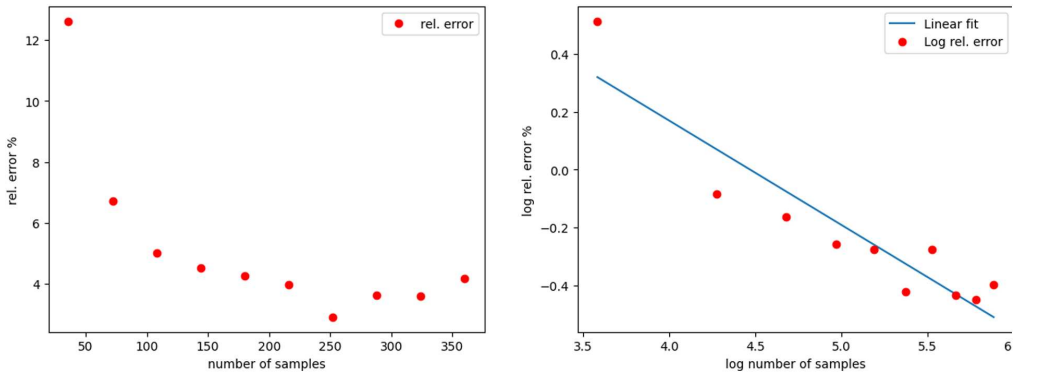

Figure 3. RTE results. Left: Relative error decay with respect to the number of training samples. Right: Error decay in log-scale. Note a linear line is fit to better demonstrate the linear decay in log.

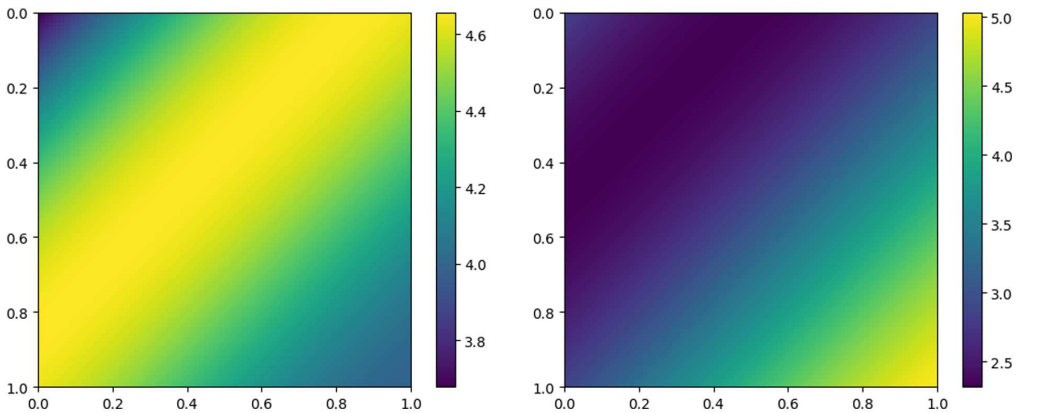

Figure 4. Demonstration of two permeability realizations for elliptic equation. Each permeability is determined by four degrees of freedom.

where $\kappa(x)$ is the permeability. The goal is to recover $\kappa(x, y)$ given the observations. We set $\kappa(x, y) = w_1 \sin(x + y) + w_2 \cos(x + y) + w_3 \sin(2(x + y)) + w_4 \cos(2(x + y))$+4.1, where $w_i \sim \text{Uniform}(-1, 1)$. We present two realizations in figure 4.

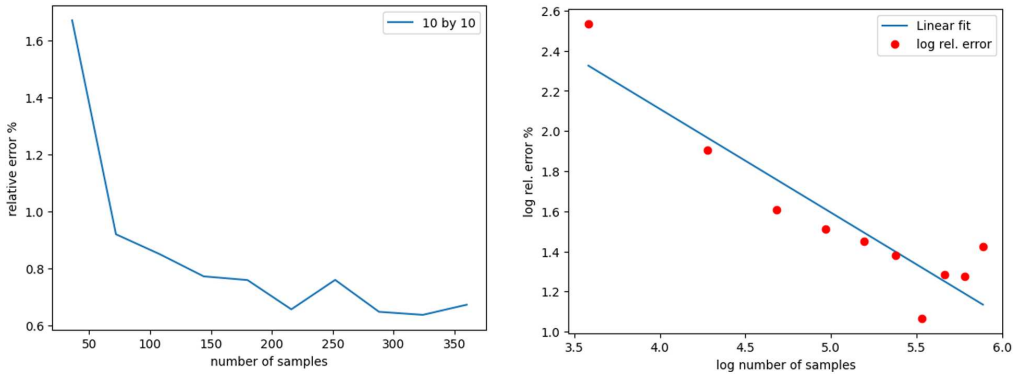

Figure 5. Elliptic example. Left: Error decay with respect to the number of training samples. Right: Error decay in logarithmic scale. We infer the inverse QoIs, which correspond to the permeability discretized on a 10×10 mesh. To better illustrate the convergence rate, we fit a linear line to the error in the logarithmic scale on the right-hand panel.

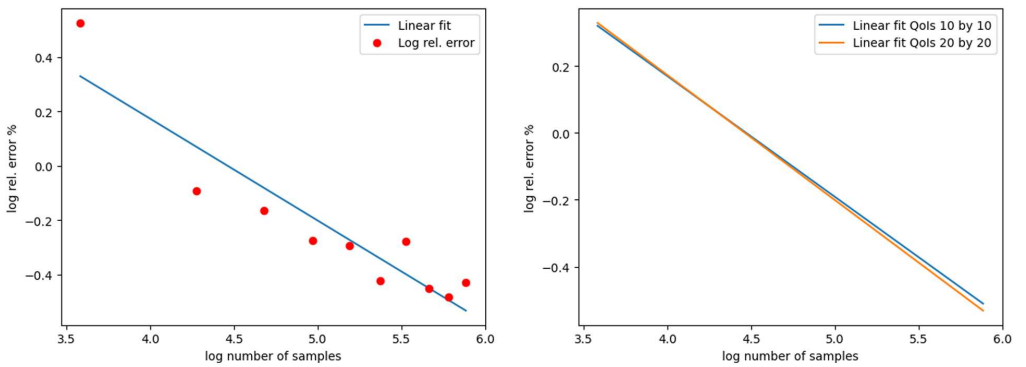

Figure 6. Elliptic example. Left: Error decay in logarithmic scale for the proposed method, which updates only the last linear layer for the downstream task with 20×20-dimensional inverse QoIs. Right: Convergence comparison between the pre-trained network for 10×10-dimensional inverse QoIs and the new network.

(i) Results for elliptic equation

In this section, we present the numerical results. The input observations (solutions of the equation) are uniformly sampled using a 10×10 mesh, while the inverse QoIs are also discretized on a 10×10 mesh. The prediction error is measured using the L_2 norm, with the results illustrated in figure 5. From the left-hand panel of figure 5, we observe that the relative L_2 error decreases to 0.7% with 360 training samples and remains below 1% even with only 108 training samples. This demonstrates the accuracy of the network's predictions. Furthermore, as shown in the right-hand panel of figure 5, errors exhibit a linear decay in the logarithmic scale, which validates the theoretical results presented in theorem 2.

(ii) Transfer to new QoIs (permeability on a finer mesh)

We test the performance of C2BNet by modifying the QoIs while keeping the encoding part of the trained network fixed. This approach significantly reduces the computational cost of inferring new QoIs by avoiding the need to train an entirely new network. Specifically, the new inverse QoIs correspond to solutions defined on a finer 20×20 grid, which differs from the 10×10 training mesh evaluated in §5b(i) To adapt the model, we retain all but the final layer and train only the last linear layer, which has a dimension of 12×400 ($d_{low} = 12$). The convergence of errors with respect to the number of training samples is presented in figure 6.

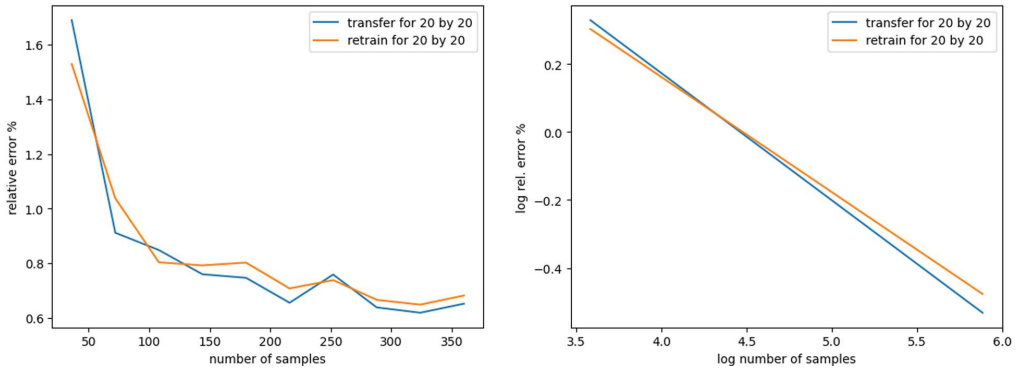

Figure 7. Time-dependent problem. Left: Error comparison between the proposed method, which transfers the trained weights except for the last layer and updates only the last layer, and the approach of retraining all layers for the downstream task with 20×20 inverse QoIs. Notably, with significantly less trainable parameters, the proposed method achieves a similar prediction error as the full retraining. Right: error decay in log scale and linear fit the log error.

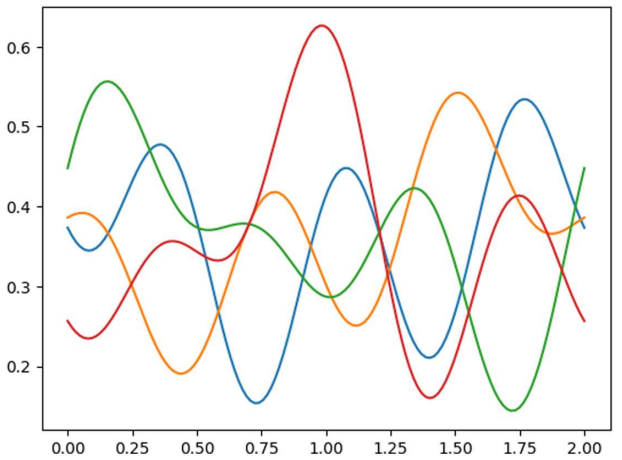

Figure 8. Demonstration of the four initial condition realizations for the time-dependent problems. Each realization is determined by six degrees of freedom.

For comparison, we also retrain all layers of the network for the new task, where the QoIs are defined on a 20×20 mesh. In particular, the full retraining process requires more cost than the proposed approach, which updates only the final layer. Specifically, complete retraining requires updating a total of 36 312 parameters, while the proposed method only updates the last layer, which consists of 4800 parameters. However, the proposed method achieves relative errors similar to full retraining, demonstrating its effectiveness and accuracy; see figure 7.

(c) Time-dependent diffusion equation

In this example, we consider the parabolic equation,

$$u_t - u_{xx} = f, x \in [0, 2], t \in [0, 0.01],$$

with Dirichlet boundary conditions. The goal is to find the initial condition given the solution at the terminal time. The initial condition used has the form: $u(x, 0) = \sum_{i=1}^{3} w_i \sin(i\pi x) + p_i \cos(i\pi x)$, where w_i, p_i are the weights uniformly sampled from $[-1, 1]$. We display four initial condition realizations in figure 8.

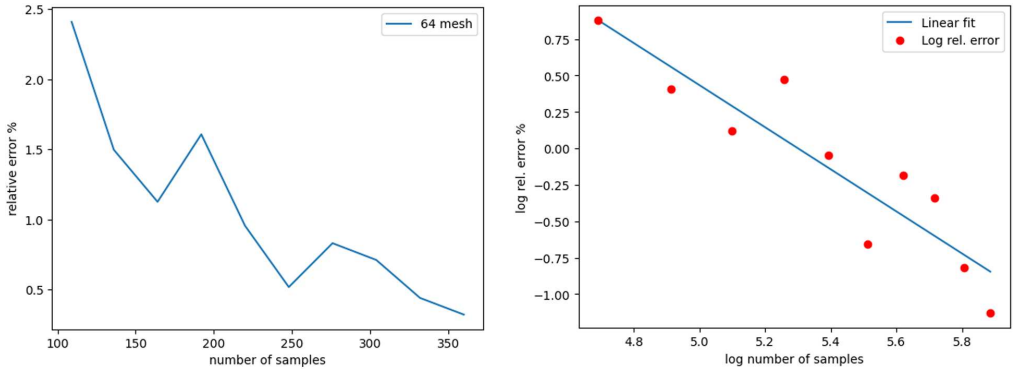

Figure 9. Left: Error decay with respect to the number of training samples. Right: Error decay in logarithmic scale. We infer the inverse QoIs, which correspond to initial conditions discretized on a 64-point mesh. To better illustrate the convergence rate, we fit a linear line to the error on the logarithmic scale on the right-hand panel.

(i) Results for the time-dependent diffusion equation

The network input (observations) are solutions at $t = 0.01$, and we take 64 points uniformly from $[0, 2]$, while the inverse QoIs are the initial conditions $u(x, 0)$ discretized by a 64-equal-distance mesh. The results are presented in figure 9. As illustrated in the left-hand panel, the network achieves prediction relative errors below 1% when approximately 225 training samples are used. Furthermore, the error decay is plotted on a logarithmic scale with respect to the number of training samples. The observed linear decay confirms the power-law behaviour established in the theorem.

(ii) Transfer to new QoIs (solutions on a finer mesh)

To demonstrate that the latent layer (the second-to-last layer) provides valuable information, as established in the theorem, we consider a new set of QoIs corresponding to initial conditions defined on a more dense mesh with 127 grid points. The proposed method updates only the final linear layer of the network, which has a size of 20×127 ($d_{low} = 20$), while keeping all preceding layers unchanged. This approach significantly reduces training costs compared with full retraining. The convergence results are presented in figure 10. From the left-hand panel of the figure, we observe a linear decay when applying a logarithmic scale to both the error and the number of training samples. The right-hand panel illustrates a close match in convergence rates between the pre-trained network, which predicts the original 64-point discretized initial condition, and the new network, where only a single linear layer is retrained for the updated QoIs.

For comparison, we evaluate the proposed method against the approach of retraining all layers for the new tasks. The total number of trainable parameters in the full retraining method is 312,60, approximately 12.31 times larger than the proposed method with only 2540 parameters, which only tunes the last linear layer. Despite significant savings in training cost, the accuracy of the proposed method remains unchanged. See figure 11 for the comparison.

6. Conclusion and discussion

In this work, we introduced C2BNet to solve inverse PDE problems within the operator learning paradigm. C2BNet consists of two components, a coefficient network and a linear basis network. C2BNet efficiently adapts to different discretizations through fine-tuning, leveraging a pre-trained model to significantly reduce computational costs while maintaining high accuracy. We also established approximation and generalization error bounds for learning Lipschitz operators using C2BNet. Our theories show that C2BNet is adaptive to low-dimensional data structures

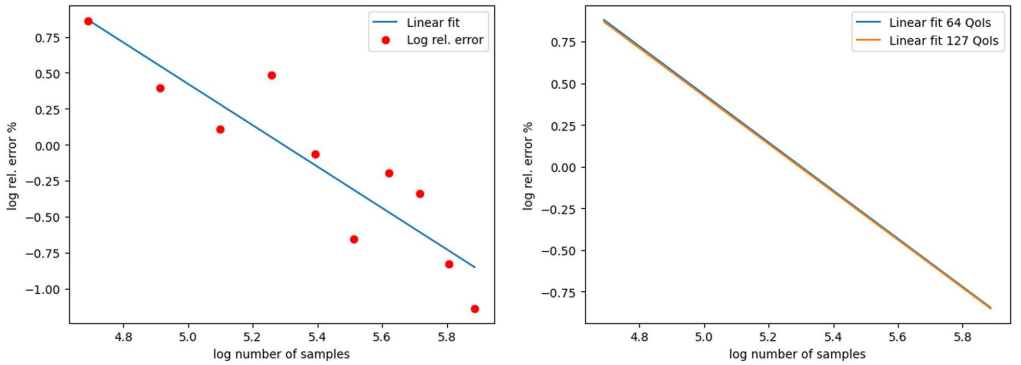

Figure 10. Left: Error decay in logarithmic scale for the proposed method, which updates only the last linear layer for the downstream task with 127-dimensional inverse QoIs. Right: Convergence comparison between the pre-trained network for 64-dimensional inverse QoIs and the new network.

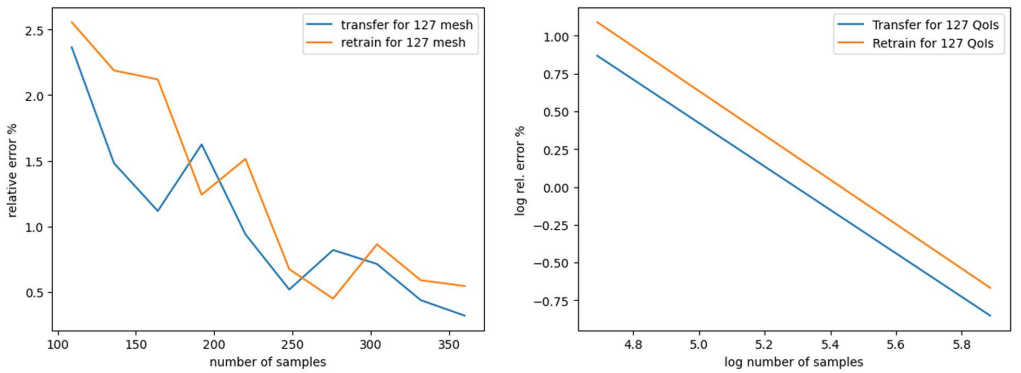

Figure 11. Time-dependent problem. Left: Error comparison between the proposed method, which transfers the trained weights except for the last layer and updates only the last layer, and the approach of retraining all layers for the downstream task with 127-dimensional inverse QoIs. Notably, the full retraining involves approximately 12 times more trainable parameters than the proposed method; however, the prediction errors behave similarly. Right: error decay in log scale and linear fit the log error.

and achieves a fast convergence rate depending on the intrinsic dimension of the dataset. The efficacy of C2BNet is demonstrated by systematic numerical experiments. The results confirm that C2BNet achieves a strong balance between computational efficiency and accuracy. These findings highlight the potential of C2BNet as a powerful and scalable tool to solve inverse problems in scientific computing and engineering applications.

The success of C2BNet relies on low-dimensional data structures. When the data exhibit complex, high-dimensional structures lacking such low-dimensional representations, larger values of d_1 and d_2 must be employed to mitigate information loss. Under these conditions, the inherent advantages of C2BNet may be diminished. Nevertheless, the framework remains applicable and can be effectively utilized for efficient fine-tuning tasks.

Data accessibility. The data for this manuscript are available at Purdue University Research Repository [69].

Declaration of AI use. We have not used AI-assisted technologies in creating this article.

Authors' contributions. Z.Z.: conceptualization, data curation, formal analysis, investigation, methodology, software, validation, visualization, writing—original draft, writing—review and editing; H.L.: conceptualization, data curation, formal analysis, investigation, methodology, software, validation, visualization, writing—original draft, writing—review and editing; W.L.: conceptualization, funding acquisition, project administration, supervision, writing—review and editing; G.L.: conceptualization, funding acquisition, project administration, resources, supervision, writing—review and editing.

All authors gave final approval for publication and agreed to be held accountable for the work performed therein.

Conflict of interest declaration. We declare we have no competing interests.

Funding. Hao Liu acknowledges the National Natural Science Foundation of China under the 12201530, and HKRGC ECS 22302123. Wenjing Liao acknowledges the National Science Foundation under the NSF DMS 2145167 and the U.S. Department of Energy under the DOE SC0024348. Guang Lin acknowledges the National Science Foundation under grants DMS-2053746, DMS-2134209, ECCS-2328241, CBET-2347401 and OAC-2311848. The U.S. Department of Energy also supports this work through the Office of Science Advanced Scientific Computing Research program (DESC0023161) and the Office of Fusion Energy Sciences (DE-SC0024583).

Appendix A. Proof of main results

A.1. Proof of theorem 1

Proof of theorem 1. We have

$$S_{\mathcal{Y}}\circ\mathrm{Proj}\circ\Psi(u)=S_{\mathcal{Y}}\left(\sum_{k=1}^{d_2}\alpha_k^v\circ\Psi(u)\omega_k\right)=\sum_{k=1}^{d_2}\alpha_k^v\circ\Psi\circ D_{\mathcal{X}}(\mathbf{u})S_{\mathcal{Y}}(\omega_k), \tag{A 1}$$

which is a linear combination of $S_{\mathcal{Y}}(\omega_k)$ with weight $\alpha_k^v\circ\Psi\circ D_{\mathcal{X}}(\mathbf{u})$. Here we denote $\mathbf{u}=S_{\mathcal{X}}(u)$.

For the coefficients, we use the following lemma to show that each $\alpha_k^v\circ\Psi\circ D_{\mathcal{X}}(\mathbf{u})$ is a Lipschitz function defined on $\mathcal{M}$ (see a proof in appendix B).

Lemma 1. Suppose assumptions 4(iii), 1 and 5 hold. For each k,

(i) $\sup_{v\in\mathcal{Y}}|\alpha_k^v(v)|\le M_2|\Omega_{\mathcal{Y}}|$,
(ii) $\alpha_k^v\circ\Psi\circ D_{\mathcal{X}}(\mathbf{u})$ is a Lipschitz function on $\mathcal{M}$ with Lipschitz constant $C_{\mathcal{M}}L_\Psi|\Omega_{\mathcal{X}}|$, where $C_{\mathcal{M}}$ is a constant depending on $\mathcal{M}$.

We denote $\alpha_k^{\mathbf{u}}(\mathbf{u})=\alpha_k^v\circ\Psi\circ D_{\mathcal{X}}(\mathbf{u})$. According to Chen *et al*. [45, Theorem 3.1], for any $\varepsilon_1>0$ and for each k, there exists a network architecture $\mathcal{F}_k=\mathcal{F}_{\mathrm{NN}}(D_1,1,L_k,p_k,K_k,\kappa_k,R_k)$ that gives rise to $\widetilde{\alpha}_k^{\mathbf{u}}\in\mathcal{F}_k$ such that

$$\|\widetilde{\alpha}_k^{\mathbf{u}}-\alpha_k^{\mathbf{u}}\|_{L^\infty(\mathcal{M})}<\varepsilon_1.$$

Such a network architecture has

$$L_k=O(\log\frac{1}{\varepsilon_1}),\ p=O(\varepsilon_1^{-d}),\ K_k=O(\varepsilon_1^{-d}\log\frac{1}{\varepsilon}+D_1\log\frac{1}{\varepsilon_1}+D_1\log D_1),$$
$$\kappa_k=O(\varepsilon_1^{-1}),\ R_k=M_2|\Omega_{\mathcal{Y}}|,$$

where the constant hidden in O depends on $d_1,M_1,M_2,|\Omega_{\mathcal{X}}|,|\Omega_{\mathcal{Y}}|,L_\Psi,\tau$ and the surface area of $\mathcal{M}$.

Denote

$$L'=\max_k L_k,\ p'=\max_k p_k,\ K'=\max_k K_k,\ \kappa'=\max_k \kappa_k,\ R'=\max_k R_k.$$

There exists a network architecture $\mathcal{F}_{coef}=\mathcal{F}_{\mathrm{NN}}(L',d_2p',d_2K',\kappa',R')$ giving rise to $\widetilde{\boldsymbol{\alpha}}=[\widetilde{\alpha}_1,...,\widetilde{\alpha}_{d_2}]^\top\in\mathcal{F}_{coef}$ such that

$$\sup_{\mathbf{u}\in\mathcal{M}}\|\widetilde{\boldsymbol{\alpha}}(\mathbf{u})-\boldsymbol{\alpha}^{\mathbf{u}}(\mathbf{u})\|_\infty<\varepsilon_1, \tag{A 2}$$

where we denote $\boldsymbol{\alpha}^{\mathbf{u}}=[\alpha_1^{\mathbf{u}},...,\alpha_{d_2}^{\mathbf{u}}]^\top$.

Now we consider the bases in equation (A 1). Since $\{\omega_k\}_{k=1}^{d_2}$ is independent of u, for any given discretization grids $S_{\mathcal{Y}}$, $\{S_{\mathcal{Y}}(\omega_k)\}_{k=1}^{d_2}$ is a set of fixed vectors, which can be realized by a matrix of dimension $D_2 \times d_2$. Specifically, let $\bar{W} = [S_{\mathcal{Y}}(\omega_1), ..., S_{\mathcal{Y}}(\omega_{d_2})] \in \mathbb{R}^{D_2 \times d_2}$. We construct $\widetilde{\Psi}$ as

$$\widetilde{\Psi} = \begin{bmatrix} \bar{W} & \mathbf{0} \\ \mathbf{0} & -\bar{W} \end{bmatrix} \mathrm{ReLU}\left(\begin{bmatrix} \widetilde{\boldsymbol{\alpha}} \\ -\widetilde{\boldsymbol{\alpha}} \end{bmatrix} \right).$$

We have

$$\begin{aligned}
&\sup_{u\in\mathcal{X}} \|S_{\mathcal{Y}} \circ \mathrm{Proj} \circ \Psi(u) - \widetilde{\Psi}(\mathbf{u})\|_\infty \\
=&\sup_{u\in\mathcal{X}} \left\| S_{\mathcal{Y}} \left(\sum_{k=1}^{d_2} \alpha_k^{\mathbf{u}}(\mathbf{u}) \omega_k \right) - \begin{bmatrix} \bar{W} & \mathbf{0} \\ \mathbf{0} & -\bar{W} \end{bmatrix} \mathrm{ReLU}\left(\begin{bmatrix} \widetilde{\boldsymbol{\alpha}}(\mathbf{u}) \\ -\widetilde{\boldsymbol{\alpha}}(\mathbf{u}) \end{bmatrix} \right) \right\|_\infty \\
=&\sup_{u\in\mathcal{X}} \| \sum_{k=1}^{d_2} \alpha_k^{\mathbf{u}}(\mathbf{u}) S_{\mathcal{Y}}(\omega_k) - \bar{W} \widetilde{\boldsymbol{\alpha}}^{\mathbf{u}}(\mathbf{u}) \|_\infty \\
=&\sup_{u\in\mathcal{X}} \| \bar{W} \boldsymbol{\alpha}^{\mathbf{u}}(\mathbf{u}) - \bar{W} \widetilde{\boldsymbol{\alpha}}^{\mathbf{u}}(\mathbf{u}) \|_\infty \\
=&\sup_{u\in\mathcal{X}} \| \bar{W} (\boldsymbol{\alpha}^{\mathbf{u}}(\mathbf{u}) - \widetilde{\boldsymbol{\alpha}}^{\mathbf{u}}(\mathbf{u})) \|_\infty \\
\leq& d_2 M_2 \varepsilon_1,
\end{aligned}$$

where the last inequality uses equation (A 2) and the fact that the absolute value of every element of $\bar{W}$ is bounded by M_2. Setting $\varepsilon_1 = \frac{\varepsilon}{d_2 M_2}$ finishes the proof. ■

A.2. Proof of corollary 1

Proof of corollary 1. Let $\widetilde{\Psi}$ be the network constructed in theorem 1. According to assumption 1, we have

$$\begin{aligned}
&\sup_{u\in\mathcal{X}} \|S_{\mathcal{Y}} \circ \Psi(u) - \widetilde{\Psi} \circ S_{\mathcal{X}}(u)\|_\infty \\
\leq& \sup_{u\in\mathcal{X}} \|S_{\mathcal{Y}} \circ \Psi(u) - S_{\mathcal{Y}} \circ \mathrm{Proj} \circ \Psi(u)\|_\infty + \sup_{u\in\mathcal{X}} \|S_{\mathcal{Y}} \circ \mathrm{Proj} \circ \Psi(u) - \widetilde{\Psi} \circ S_{\mathcal{X}}(u)\|_\infty \\
\leq& \zeta + \varepsilon.
\end{aligned}$$

■

A.3. Proof of theorem 2

Proof of theorem 2. We decompose the error as

$$\begin{aligned}
&\mathbb{E}_{\mathcal{S}} \mathbb{E}_{u\sim\rho} \left[\|\widehat{\Psi}(\mathbf{u}) - S_{\mathcal{Y}} \circ \Psi(u)\|_{S_{\mathcal{Y}}}^2 \right] \\
=& \underbrace{2\mathbb{E}_{\mathcal{S}} \left[\frac{1}{n} \sum_{i=1}^{n} \|\widehat{\Psi}(\mathbf{u}_i) - S_{\mathcal{Y}} \circ \Psi(u_i)\|_{S_{\mathcal{Y}}}^2 \right]}_{\mathrm{T}_1} \\
&+ \underbrace{\mathbb{E}_{u\sim\rho} \mathbb{E}_{\mathcal{S}} \left[\|\widehat{\Psi}(\mathbf{u}) - S_{\mathcal{Y}} \circ \Psi(u)\|_{S_{\mathcal{Y}}}^2 \right] - 2\mathbb{E}_{\mathcal{S}} \left[\frac{1}{n} \sum_{i=1}^{n} \|\widehat{\Psi}(\mathbf{u}_i) - S_{\mathcal{Y}} \circ \Psi(u_i)\|_{S_{\mathcal{Y}}}^2 \right]}_{\mathrm{T}_2}.
\end{aligned}$$

We next derive bounds for T_1 and T_2.

Bounding T_1. We bound T_1 as

$$\begin{aligned}
\mathrm{T}_1 =& 2\mathbb{E}_{\mathcal{S}}\left[\frac{1}{n}\sum_{i=1}^{n}\|\widehat{\Psi}(\mathbf{u}_i) - S_{\mathcal{Y}}\circ\Psi(u_i) - S_{\mathcal{Y}}(\xi_i) + S_{\mathcal{Y}}(\xi_i)\|_{S_{\mathcal{Y}}}^2\right]\\
=& 2\mathbb{E}_{\mathcal{S}}\left[\frac{1}{n}\sum_{i=1}^{n}\|\widehat{\Psi}(\mathbf{u}_i) - S_{\mathcal{Y}}\circ\Psi(u_i) - S_{\mathcal{Y}}(\xi_i)\|_{S_{\mathcal{Y}}}^2\right]\\
&+ 4\mathbb{E}_{\mathcal{S}}\left[\frac{1}{n}\sum_{i=1}^{n}\langle\widehat{\Psi}(\mathbf{u}_i) - S_{\mathcal{Y}}\circ\Psi(u_i) - S_{\mathcal{Y}}(\xi_i), S_{\mathcal{Y}}(\xi_i)\rangle_{S_{\mathcal{Y}}}\right] + 2\mathbb{E}_{\mathcal{S}}\left[\frac{1}{n}\sum_{i=1}^{n}\|S_{\mathcal{Y}}(\xi_i)\|_{S_{\mathcal{Y}}}^2\right]\\
=& 2\mathbb{E}_{\mathcal{S}}\left[\frac{1}{n}\sum_{i=1}^{n}\|\widehat{\Psi}(\mathbf{u}_i) - \widehat{\mathbf{v}}_i\|_{S_{\mathcal{Y}}}^2\right]\\
&+ 4\mathbb{E}_{\mathcal{S}}\left[\frac{1}{n}\sum_{i=1}^{n}\langle\widehat{\Psi}(\mathbf{u}_i) - S_{\mathcal{Y}}\circ\Psi(u_i), S_{\mathcal{Y}}(\xi_i)\rangle_{S_{\mathcal{Y}}}\right] - 2\mathbb{E}_{\mathcal{S}}\left[\frac{1}{n}\sum_{i=1}^{n}\|S_{\mathcal{Y}}(\xi_i)\|_{S_{\mathcal{Y}}}^2\right]\\
=& 2\mathbb{E}_{\mathcal{S}}\left[\inf_{\Psi_{\mathrm{NN}}\in\mathcal{F}}\left[\frac{1}{n}\sum_{i=1}^{n}\|\Psi_{\mathrm{NN}}(\mathbf{u}_i) - \widehat{\mathbf{v}}_i\|_{S_{\mathcal{Y}}}^2\right]\right]\\
&+ 4\mathbb{E}_{\mathcal{S}}\left[\frac{1}{n}\sum_{i=1}^{n}\langle\widehat{\Psi}(\mathbf{u}_i) - S_{\mathcal{Y}}\circ\Psi(u_i), S_{\mathcal{Y}}(\xi_i)\rangle_{S_{\mathcal{Y}}}\right] - 2\mathbb{E}_{\mathcal{S}}\left[\frac{1}{n}\sum_{i=1}^{n}\|S_{\mathcal{Y}}(\xi_i)\|_{S_{\mathcal{Y}}}^2\right]\\
\le& 2\inf_{\Psi_{\mathrm{NN}}\in\mathcal{F}}\left[\mathbb{E}_{\mathcal{S}}\left[\frac{1}{n}\sum_{i=1}^{n}\|\Psi_{\mathrm{NN}}(\mathbf{u}_i) - S_{\mathcal{Y}}\circ\Psi(u_i) - S_{\mathcal{Y}}(\xi_i)\|_{S_{\mathcal{Y}}}^2 - \|S_{\mathcal{Y}}(\xi_i)\|_{S_{\mathcal{Y}}}^2\right]\right]\\
&+ 4\mathbb{E}_{\mathcal{S}}\left[\frac{1}{n}\sum_{i=1}^{n}\langle\widehat{\Psi}(\mathbf{u}_i) - S_{\mathcal{Y}}\circ\Psi(u_i), S_{\mathcal{Y}}(\xi_i)\rangle_{S_{\mathcal{Y}}}\right]\\
=& 2\inf_{\Psi_{\mathrm{NN}}\in\mathcal{F}}\left[\mathbb{E}_{\mathcal{S}}\left[\frac{1}{n}\sum_{i=1}^{n}\|\Psi_{\mathrm{NN}}(\mathbf{u}_i) - S_{\mathcal{Y}}\circ\Psi(u_i)\|_{S_{\mathcal{Y}}}^2\right]\right] + 4\mathbb{E}_{\mathcal{S}}\left[\frac{1}{n}\sum_{i=1}^{n}\langle\widehat{\Psi}(\mathbf{u}_i) - S_{\mathcal{Y}}\circ\Psi(u_i), S_{\mathcal{Y}}(\xi_i)\rangle_{S_{\mathcal{Y}}}\right]\\
=& 2\inf_{\Psi_{\mathrm{NN}}\in\mathcal{F}}\left[\mathbb{E}_{u\sim\rho}\left[\|\Psi_{\mathrm{NN}}(\mathbf{u}) - S_{\mathcal{Y}}\circ\Psi(u)\|_{S_{\mathcal{Y}}}^2\right]\right] + 4\mathbb{E}_{\mathcal{S}}\left[\frac{1}{n}\sum_{i=1}^{n}\langle\widehat{\Psi}(\mathbf{u}_i) - S_{\mathcal{Y}}\circ\Psi(u_i), S_{\mathcal{Y}}(\xi_i)\rangle_{S_{\mathcal{Y}}}\right].
\end{aligned} \tag{A 3}$$

For the first term, we have

$$\begin{aligned}
&2\inf_{\Psi_{\mathrm{NN}}\in\mathcal{F}}\left[\mathbb{E}_{u\sim\rho}\left[\|\Psi_{\mathrm{NN}}(\mathbf{u}) - S_{\mathcal{Y}}\circ\Psi(u)\|_{S_{\mathcal{Y}}}^2\right]\right]\\
\le& 2\inf_{\Psi_{\mathrm{NN}}\in\mathcal{F}}\left[\mathbb{E}_{u\sim\rho}\left[\|2\Psi_{\mathrm{NN}}(\mathbf{u}) - S_{\mathcal{Y}}\circ\mathrm{Proj}\circ\Psi(u)\|_{S_{\mathcal{Y}}}^2 + 2\|S_{\mathcal{Y}}\circ\mathrm{Proj}\circ\Psi(u) - S_{\mathcal{Y}}\circ\Psi(u)\|_{S_{\mathcal{Y}}}^2\right]\right]\\
\le& 4\inf_{\Psi_{\mathrm{NN}}\in\mathcal{F}}\left[\mathbb{E}_{u\sim\rho}\left[\|2\Psi_{\mathrm{NN}}(\mathbf{u}) - S_{\mathcal{Y}}\circ\mathrm{Proj}\circ\Psi(u)\|_{S_{\mathcal{Y}}}^2\right]\right] + 4\zeta^2.
\end{aligned}$$

For any $\varepsilon_1 > 0$, theorem 1 shows that there exist network architectures $\mathcal{F}_{coef} = \mathcal{F}_{\mathrm{NN}}(D_1, d_2, L, p, K, \kappa, R)$ $\mathcal{F}_{basis} = \mathcal{F}_{\mathrm{NN}}(d_2, D_2, 1, D_2, d_2D_2, M_2, M_2)$ and $\mathcal{F}$ defined in equation (4.5) giving rise to a $\widetilde{\Psi}\in\mathcal{F}$ such that

$$\sup_{u\in\mathcal{X}}\|S_{\mathcal{Y}}\circ\mathrm{Proj}\circ\Psi(u) - \widetilde{\Psi}(\mathbf{u})\|_{\infty} \le \varepsilon_1. \tag{A 4}$$

The network architecture $\mathcal{F}_{coef}$ has

$$\begin{aligned}
&L = O(\log\frac{1}{\varepsilon_1}),\ p = O(\varepsilon_1^{-d_1}),\ K = O(\varepsilon_1^{-d_1}\log\frac{1}{\varepsilon_1} + D_1\log\frac{1}{\varepsilon_1} + D_1\log D_1 + D_2),\\
&\kappa = O(\varepsilon_1^{-1}),\ R = M_2|\Omega_{\mathcal{Y}}|.
\end{aligned} \tag{A 5}$$

The constant hidden in O depends on $d_1, d_2, M_1, M_2, |\Omega_{\mathcal{X}}|, |\Omega_{\mathcal{Y}}|, L_\Psi, \tau$ and the surface area of $\mathcal{M}$. Thus the first term in equation (A 3) is bounded by

$$2 \inf_{\Psi_{\mathrm{NN}} \in \mathcal{F}} \left[\mathbb{E}_{u\sim\rho} \left[\frac{1}{D_2} \|\Psi_{\mathrm{NN}}(\mathbf{u}) - S_{\mathcal{Y}} \circ \Psi(u)\|_2^2 \right] \right] \leq 4\varepsilon_1^2 + 4\zeta^2. \tag{A 6}$$

To bound the second term, we can use [20, Lemma 10]:

Lemma 2. Under the condition of theorem 2, for any $\delta > 0$, we have

$$\begin{aligned} &\mathbb{E}_{\mathcal{S}} \left[\frac{1}{n} \sum_{i=1}^{n} \langle \widehat{\Psi}(\mathbf{u}_i) - S_{\mathcal{Y}} \circ \Psi(u_i), S_{\mathcal{Y}}(\xi_i) \rangle_{S_{\mathcal{Y}}} \right] \\ &\leq 2\sigma \left(\sqrt{\mathbb{E}_{\mathcal{S}}[\|\widehat{\Psi} - S_{\mathcal{Y}} \circ \Psi\|_n^2]} + \delta \right) \sqrt{\frac{4 \log \mathcal{N}(\delta, \mathcal{F}, \|\cdot\|_{L^{\infty,\infty}}) + 6}{n}} + \delta\sigma, \end{aligned} \tag{A 7}$$

where

$$\|\widehat{\Psi} - S_{\mathcal{Y}} \circ \Psi\|_n^2 = \frac{1}{n} \sum_{i=1}^{n} \|\widehat{\Psi}(\mathbf{u}_i) - S_{\mathcal{Y}} \circ \Psi(u_i)\|_{S_{\mathcal{Y}}}^2.$$

Substituting equations (A 6) and (A 7) into equation (A 3) gives rise to

$$\begin{aligned} \mathrm{T}_1 =& 2\mathbb{E}_{\mathcal{S}} \left[\|\widehat{\Psi} - S_{\mathcal{Y}} \circ \Psi\|_n^2 \right] \\ \leq& 2\varepsilon_1^2 + 8\sigma \left(\sqrt{\mathbb{E}_{\mathcal{S}}[\|\widehat{\Psi} - S_{\mathcal{Y}} \circ \Psi\|_n^2]} + \delta \right) \sqrt{\frac{4 \log \mathcal{N}(\delta, \mathcal{F}, \|\cdot\|_{L^{\infty,\infty}}) + 6}{n}} + 4\delta\sigma. \end{aligned} \tag{A 8}$$

Denote

$$\begin{aligned} s &= \sqrt{\mathbb{E}_{\mathcal{S}}[\|\widehat{\Psi} - S_{\mathcal{Y}} \circ \Psi\|_n^2]} \\ a &= \varepsilon_1^2 + 4\sigma\delta \sqrt{\frac{4 \log \mathcal{N}(\delta, \mathcal{F}, \|\cdot\|_{L^{\infty,\infty}}) + 6}{n}} + 2\delta\sigma, \\ b &= 2\sigma \sqrt{\frac{4 \log \mathcal{N}(\delta, \mathcal{F}, \|\cdot\|_{L^{\infty,\infty}}) + 6}{n}}. \end{aligned}$$

Relation equation (A 8) can be rewritten as

$$s^2 \leq a + 2bs,$$

which implies

$$\begin{aligned} &(s-b)^2 \leq \sqrt{a} + b \\ \Rightarrow &|s-b| \leq \sqrt{a+b^2} \leq \sqrt{a} + b. \end{aligned}$$

When $s \geq b$, we have

$$\begin{aligned} &s - b \leq \sqrt{a} + b, \\ \Rightarrow &s \leq \sqrt{a} + 2b, \\ \Rightarrow &s^2 \leq (\sqrt{a} + 2b)^2 \leq 2a + 8b^2. \end{aligned}$$

When $s < b$, the relation $s^2 \leq 2a + 8b^2$ is also true. Substituting the expression of s, a, b into the relation, we get

$$\begin{aligned} \mathrm{T}_1 = 2s^2 \leq& 8\varepsilon_1^2 + 8\zeta^2 + 16\sigma\delta \sqrt{\frac{4 \log \mathcal{N}(\delta, \mathcal{F}, \|\cdot\|_{L^{\infty,\infty}}) + 6}{n}} + 8\delta\sigma \\ &+ 128\sigma^2 \frac{2 \log \mathcal{N}(\delta, \mathcal{F}, \|\cdot\|_{L^{\infty,\infty}}) + 3}{n}. \end{aligned}$$

Bounding T_2. To bound T_2, we use [20, Lemma 11]:

Lemma 3 (Lemma 11 of [20])**.** Under the condition of theorem 2, for any $\delta > 0$, we have

$$\mathrm{T}_2 \leq \frac{48M_2^2}{n} \log \mathcal{N}\left(\frac{\delta}{4M_2}, \mathcal{F}, \|\cdot\|_{L^{\infty,\infty}}\right) + 6\delta. \tag{A 9}$$

Putting T_1 **and** T_2 **together.** Combining the error bound of T_1 and T_2 gives rise to

$$\begin{aligned}
&\mathbb{E}_{\mathcal{S}}\mathbb{E}_{u\sim\rho}\left[\|\widehat{\Psi}(\mathbf{u}) - S_{\mathcal{Y}}\circ\Psi(u)\|_{S_{\mathcal{Y}}}^2\right] \\
\leq& 8\varepsilon_1^2 + 8\zeta^2 + 16\sigma\delta\sqrt{\frac{4\log\mathcal{N}(\delta,\mathcal{F},\|\cdot\|_{L^{\infty,\infty}})+6}{n}} + 8\delta\sigma \\
&+ 128\sigma^2\frac{2\log\mathcal{N}(\delta,\mathcal{F},\|\cdot\|_{L^{\infty,\infty}})+3}{n} + \frac{48M_2^2}{n}\log\mathcal{N}\left(\frac{\delta}{4M_2},\mathcal{F},\|\cdot\|_{L^{\infty,\infty}}\right) + 6\delta.
\end{aligned} \tag{A 10}$$

The covering number of a network class can be bounded by the following lemma:

Lemma 4 (Lemma 5.3 of [45])**.** Let $\mathcal{F}_{\mathrm{NN}}(D_1, D_2, L, p, K, \kappa, R)$ be a network architecture from $[-B_1, B_1]^{D_1}$ to $[-B_2, B_2]^{D_2}$ for some $B_1, B_2 > 0$. For any $\delta > 0$, we have

$$\mathcal{N}(\delta, \mathcal{F}_{\mathrm{NN}}, \|\cdot\|_{L^{\infty,\infty}}) \leq \left(\frac{2L^2(pB_1+2)\kappa^L p^{L+1}}{\delta}\right)^K.$$

Substituting the network architecture into equation (A 5) into lemma 4 gives rise to

$$\log\mathcal{N}(\delta,\mathcal{F},\|\cdot\|_{L^{\infty,\infty}}) \leq C_1 D_1(\log D_1)\varepsilon_1^{-d_1}\log^2\frac{1}{\varepsilon_1}\left(\log\frac{1}{\varepsilon_1} + \log\frac{1}{\delta}\right) \tag{A 11}$$

for some constant C_1 depending on $d_1, d_2, M_1, M_2, |\Omega_{\mathcal{X}}|, |\Omega_{\mathcal{Y}}|, L_\Psi, L_D, \tau$ and the surface area of $\mathcal{M}$. Setting $\delta = 1/n$ and substituting equation (A 11) into equation (A 10) gives rise to

$$\mathbb{E}_{\mathcal{S}}\mathbb{E}_{u\sim\rho}\left[\|\widehat{\Psi}(\mathbf{u}) - S_{\mathcal{Y}}\circ\Psi(u)|_{S_{\mathcal{Y}}}^2\right] \leq 8\varepsilon_1^2 + 8\zeta^2 + \frac{C_2 D_1(\log D_1)}{n}\varepsilon_1^{-d_1}\log^2\frac{1}{\varepsilon_1}\left(\log\frac{1}{\varepsilon_1} + \log n\right),$$

where C_2 is a constant depending on $d_1, d_2, M_1, M_2, |\Omega_{\mathcal{X}}|, |\Omega_{\mathcal{Y}}|, L_\Psi, \sigma, \tau$ and the surface area of $\mathcal{M}$. Setting $\varepsilon_1 = n^{-\frac{2}{2+d_1}}$ finishes the proof. The network architecture is specified in equation (4.7). ■

Appendix B. Proof of lemma 1

Proof of lemma 1. For (i), we have

$$|\alpha_k^v(v)| = \langle v, \omega_k\rangle_{\mathcal{Y}} \leq \|v\|_{\mathcal{Y}}\|\omega_k\|_{\mathcal{Y}} = \|v\|_{\mathcal{Y}} \leq |\Omega_{\mathcal{Y}}|\|v\|_{L^\infty(\Omega_{\mathcal{Y}})} \leq M_2|\Omega_{\mathcal{Y}}|.$$

To prove (ii), for simplicity, we denote $\alpha_k^{\mathbf{u}}(\mathbf{u}) = \alpha_k^v\circ\Psi\circ D_{\mathcal{X}}(\mathbf{u})$. Let (U, ϕ) be a chart of $\mathcal{M}$ such that ϕ is Lipscthiz with Lipschitz constant L_ϕ. We need to show that $\alpha_k^{\mathbf{u}}\circ\phi^{-1} : \phi(U) \to \mathbb{R}$ is a Lipschitz

function. For any $\mathbf{z}_1, \mathbf{z}_2 \in \phi(U)$, denote $\mathbf{u}_j = \phi^{-1}(\mathbf{z}_j)$, $u_j = D_{\mathcal{X}}(\mathbf{u}_j)$, $v_j = \Psi(u_j)$ for $j = 1, 2$. We have

$$\begin{aligned}
&|\alpha_k^{\mathbf{u}} \circ \phi^{-1}(\mathbf{z}_1) - \alpha_k^{\mathbf{u}} \circ \phi^{-1}(\mathbf{z}_2)| \\
=&|\alpha_k^{v}(v_1) - \alpha_k^{v}(v_2)| \\
=&|\langle v_1, \phi_k \rangle_{\mathcal{Y}} - \langle v_2, \phi_k \rangle_{\mathcal{Y}}| \\
=&|\langle v_1 - v_2, \phi_k \rangle_{\mathcal{Y}}| \\
\leq&\|v_1 - v_2\|_{\mathcal{Y}} \\
=&\|\Psi(u_1) - \Psi(u_2)\|_{\mathcal{Y}} \\
\leq&L_\Psi \|u_1 - u_2\|_{\mathcal{X}} \\
=&L_\Psi |\Omega_{\mathcal{X}}| \|u_1 - u_2\|_{L^\infty(\Omega_{\mathcal{X}})} \\
=&L_\Psi |\Omega_{\mathcal{X}}| |D_{\mathcal{X}} \circ \phi^{-1}(\mathbf{z}_1) - D_{\mathcal{X}} \circ \phi^{-1}(\mathbf{z}_2)|,
\end{aligned} \tag{B 1}$$

According to assumption 4(iii), $D_{\mathcal{X}} \circ \phi^{-1}$ is a Lipschitz function. For any given finite atlas of $\mathcal{M}$, there exists a constant $C_{\mathcal{M}}$ that is an upper bound of the Lipschitz constant of $D_{\mathcal{X}} \circ \phi^{-1}$ for all ϕ's in the atlas. Applying this property to equation (B 1) gives rise to

$$|\alpha_k^{v}(v_1) - \alpha_k^{v}(v_2)| \leq C L_\Psi |\Omega_{\mathcal{X}}| \|\mathbf{z}_1 - \mathbf{z}_2\|_2.$$

■

References

1. Bhattacharya K, Hosseini B, Kovachki NB, Stuart AM. 2021 Model reduction and neural networks for parametric PDEs. *SMAI J. Comput. Math.* **7**, 121–157. (doi:10.5802/smai-jcm.74)
2. Li Z, Azizzadenesheli K, Bhattacharya K, Anandkumar A. 2020b Fourier neural operator for parametric partial differential equations. *arXiv Preprint arXiv:2010.08895*
3. Lu L, Jin P, Pang G, Zhang Z, Karniadakis GE. 2021b Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nat. Mach. Intell.* **3**, 218–229. (doi:10.1038/s42256-021-00302-5)
4. Ronneberger O, Fischer P, Brox T. 2015 U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Lecture notes in computer science medical image computing and computer-assisted intervention – miccai 2015*, pp. 234–241. Cham, Switzerland: Springer International Publishing. (doi:10.1007/978-3-319-24574-4_28)
5. Fan Y, Ying L. 2019 Solving inverse wave scattering with deep learning. *arXiv Preprint arXiv:1911.13202*
6. Li H, Schwab J, Antholzer S, Haltmeier M. 2020a NETT: solving inverse problems with deep neural networks. *Inverse Probl.* **36**, 065005. (doi:10.1088/1361-6420/ab6d57)
7. Hesthaven JS, Ubbiali S. 2018 Non-intrusive reduced order modeling of nonlinear problems using neural networks. *J. Comput. Phys.* **363**, 55–78. (doi:10.1016/j.jcp.2018.02.037)
8. Hoop MV, Huang DZ, Stuart AM. 2022 The cost-accuracy trade-off in operator learning with neural networks. *arXiv Preprint arXiv:2203.13181*
9. Wen G, Li Z, Azizzadenesheli K, Anandkumar A, Benson SM. 2022 U-FNO—an enhanced fourier neural operator-based deep-learning model for multiphase flow. *Adv. Water Resour.* **163**, 104180. (doi:10.1016/j.advwatres.2022.104180)
10. Li Z, Azizzadenesheli B, Stuart A. 2021 Fourier neural operator for parametric partial differential equations. In *International conference on learning representations*.
11. Zhu M, Feng S, Lin Y, Lu L. 2023 Fourier-DeepONet: fourier-enhanced deep operator networks for full waveform inversion with improved accuracy, generalizability, and robustness. *Comput. Methods Appl. Mech. Eng.* **416**, 116300. (doi:10.1016/j.cma.2023.116300)
12. GuibasJ M, LiZ T, AnandkumarA C. 2021 Adaptive fourier neural operators: efficient token mixers for transformers https://arxiv. org/abs/2111.13587

13. Li Z, Liu B. 2022 Fourier neural operator with learned deformations for pdes on general geometries. *arXiv Preprint arXiv:2207.05209*
14. Bourlard H, Kamp Y. 1988 Auto-association by multilayer perceptrons and singular value decomposition. *Biol. Cybern*. **59**, 291–294. (doi:10.1007/BF00332918)
15. Hinton GE, Zemel R. 1993 Autoencoders, minimum description length and helmholtz free energy. In *Advances in neural information processing systems*, vol. 6.
16. Schonsheck M, Klock T, Lai R. 2022 Semi-supervised manifold learning with complexity decoupled chart autoencoders. *arXiv Preprint arXiv:2208.10570*
17. Liu H, Chen M, Liao W. 2024a Deep nonparametric estimation of operators between infinite dimensional spaces. *J. Mach. Learn*. 1–67.
18. Seidman J, Kissas G, Perdikaris P. 2022 NOMAD: nonlinear manifold decoders for operator learning **35**, 5601–5613. (doi:10.5270/esa-cx7leig)
19. Kontolati K, Karniadakis GE. 2023 Learning in latent spaces improves the predictive accuracy of deep neural operators. *arXiv Preprint arXiv:2304.07599*
20. Liu H, Dahal B, Lai R, Liao W. 2025 Generalization error guaranteed auto-encoder-based nonlinear model reduction for operator learning. *Appl. Comput. Harmon. Anal.* **74**, 101717. (doi:10.1016/j.acha.2024.101717)
21. Lin G, Moya C, Zhang Z. 2023 B-DeepONet: an enhanced Bayesian DeepONet for solving noisy parametric PDEs using accelerated replica exchange SGLD. *J. Comput. Phys*. **473**, 111713. (doi:10.1016/j.jcp.2022.111713)
22. Lin G, Zhang Z. 2021 Accelerated replica exchange stochastic gradient langevin diffusion enhanced bayesian DeepONet for solving noisy parametric pdes. *arXiv Preprint arXiv:2111.02484*
23. Zhang Z, Wing Tat L, Schaeffer H. 2023 BelNet: basis enhanced learning, a mesh-free neural operator. Proc. R. Soc. A **479**, 20230043. (doi:10.1098/rspa.2023.0043)
24. Goswami S, Yin M, Yu Y, Karniadakis GE. 2022 A physics-informed variational DeepONet for predicting crack path in quasi-brittle materials. *Comput. Methods Appl. Mech. Eng*. **391**, 114587. (doi:10.1016/j.cma.2022.114587)
25. Wang S, Wang H, Perdikaris P. 2021 Learning the solution operator of parametric partial differential equations with physics-informed DeepONets. *Sci. Adv*. **7**, eabi8605. (doi:10.1126/sciadv.abi8605)
26. Yu Y, Lu F, Jafarzadeh S. 2024 Nonlocal attention operator: materializing hidden knowledge towards interpretable physics discovery. In *Advances in neural information processing systems*, pp. 113797–113822, vol. 37.
27. Hao W, Wang J. 2025 Laplacian eigenfunction-based neural operator for learning nonlinear partial differential equations. *arXiv Preprint arXiv:2502.05571*
28. Zhang Z. 2024 MODNO: multi-operator learning with distributed neural operators. *Comput. Methods Appl. Mech.***431**, 117229. (doi:10.1016/j.cma.2024.117229)
29. Zhang Z, Moya C, Lu L, Lin G, Schaeffer H. 2024 D2no: efficient handling of heterogeneous input function spaces with distributed deep neural operators. *Comput. Methods Appl. Mech.***428**, 117084. (doi:10.1016/j.cma.2024.117084)
30. Nelsen NH, Stuart AM. 2024 Operator learning using random features: a tool for scientific computing. *SIAM Rev.* **66**, 535–571. (doi:10.1137/24M1648703)
31. Lanthaler S. 2023 Operator learning with PCA-net: upper and lower complexity bounds. *J. Mach. Learn*. 1–67.
32. Kovachki N, Mishra S. 2021 On universal approximation and error bounds for fourier neural operators. *J. Mach. Learn*. 1–76.
33. Chen T, Chen H. 1995 Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems. *IEEE Trans. Neural Netw*. **6**, 911–917. (doi:10.1109/72.392253)
34. Lanthaler S, Mishra S, Karniadakis GE. 2022 Error estimates for DeepONets: a deep learning framework in infinite dimensions. *Trans. Math. Its Appl*. **6**, c001. (doi:10.1093/imatrm/tnac001)
35. Schwab S, Zech J. 2023 Deep operator network approximation rates for lipschitz operators. *arXiv Preprint arXiv:2307.09835*
36. Liu H, Liao W. 2024b Neural scaling laws of deep ReLU and deep operator network: a theoretical study. *arXiv Preprint arXiv:2410.00357*
37. Lanthaler S. 2023 The parametric complexity of operator learning. *arXiv Preprint arXiv:2306.15924 539*

38. Russakovsky O *et al*. 2015 ImageNet large scale visual recognition challenge. *Int. J. Comput. Vis.* **115**, 211–252. (doi:10.1007/s11263-015-0816-y)
39. Pope P, Abdelkader A, Goldstein T. 2021 The intrinsic dimension of images and its impact on learning. In *International conference on learning representations*.
40. Tenenbaum JB, de Silva V, Langford JC. 2000 A global geometric framework for nonlinear dimensionality reduction. *Science* **290**, 2319–2323. (doi:10.1126/science.290.5500.2319)
41. Osher S, Shi Z, Zhu W. 2017 Low dimensional manifold model for image processing. *SIAM J. Imaging Sci*. **10**, 1669–1690. (doi:10.1137/16m1058686)
42. Efendiev Y, Hou T, Luo W. 2006 Preconditioning markov chain monte carlo simulations using coarse-scale models. *SIAM J. Sci. Comput*. **28**, 776–803. (doi:10.1137/050628568)
43. Lu J, Shen Z, Yang H, Zhang S. 2021a Deep Network Approximation for Smooth Functions. *SIAM J. Math. Anal*. **53**, 5465–5506. (doi:10.1137/20m134695x)
44. Hasani E. 2024 Generating synthetic data for neural operators. *arXiv Preprint arXiv:2401.02398*
45. Chen M, Jiang H, Liao W, Zhao T. 2022 Nonparametric regression on low-dimensional manifolds using deep ReLU networks: function approximation and statistical recovery. *Inf. Inference* **11**, 1203–1253. (doi:10.1093/imaiai/iaac001)
46. Nakada R. 2020 Adaptive approximation and generalization of deep neural network with intrinsic dimensionality. *J. Mach. Learn* 1–38.
47. Cloninger A, Klock T. 2021 A deep network construction that adapts to intrinsic dimensionality beyond the domain. *Neural Networks* **141**, 404–419. (doi:10.1016/j.neunet.2021.06.004)
48. Schmidt-Hieber J. 2020 Nonparametric regression using deep neural networks with ReLU activation function. *Ann. Statist*. **48**, 1875–1897. (doi:10.1214/19-AOS1875)
49. Liu H, Chen M, Zhao T, Liao W. 2021 Besov function approximation and binary classification on low-dimensional manifolds using convolutional residual networks. In *International conference on machine learning*, pp. 6770–6780.
50. Yarotsky D. 2017 Error bounds for approximations with deep ReLU networks. *Neural Networks* **94**, 103–114. (doi:10.1016/j.neunet.2017.07.002)
51. Suzuki T. 2019 Adaptivity of deep ReLU network for learning in besov and mixed smooth besov spaces: optimal rate and curse of dimensionality. In *International conference on learning representations*.
52. Dahal B, Chen M, Liao W. 2022 On deep generative models for approximation and estimation of distributions on manifolds. In *Advances in neural information processing systems*, pp. 10615–10628, vol. 35.
53. Havrilla A. 2024 Understanding scaling laws with statistical and approximation theory for transformer neural networks on intrinsically low-dimensional data. In *The thirty-eighth annual conference on neural information processing systems*.
54. Marcati C, Schwab C. 2023 Exponential convergence of deep operator networks for elliptic partial differential equations. *SIAM J. Numer. Anal*. **61**, 1513–1545. (doi:10.1137/21m1465718)
55. Opschoor JAA, Schwab Ch, Zech J. 2022 Exponential ReLU DNN expression of holomorphic maps in high dimension. *Constr. Approx*. **55**, 537–582. (doi:10.1007/s00365-021-09542-5)
56. Adcock B, Moraga S. 2024 Optimal deep learning of holomorphic operators between banach spaces. *arXiv Preprint arXiv:2406.13928*
57. Wu Y, Lin Y. 2020 InversionNet: an efficient and accurate data-driven full waveform inversion. *IEEE Trans. Comput. Imaging* **6**, 419–433. (doi:10.1109/tci.2019.2956866)
58. Liu Z, Emami-Meybodi H. 2021 Rate transient analysis of infinite-acting linear flow by use of piecewise constant diffusivity coefficients. *J. Pet. Sci. Eng*. **196**, 107783. (doi:10.1016/j.petrol.2020.107783)
59. Vaidya DS, Nitsche JM, Diamond SL, Kofke DA. 1996 Convection-diffusion of solutes in media with piecewise constant transport properties. *Chem. Eng. Sci*. **51**, 5299–5312. (doi:10.1016/s0009-2509(96)00356-9)
60. Tu LW. 2011 *An introduction to manifolds*. New York, NY: Springer.
61. Lee JM. 2006 *Riemannian manifolds: an introduction to curvature*. vol. 176. New York, NY: Springer Science & Business Media.
62. Federer H. 1959 Curvature measures. *Trans. Am. Math*. **93**, 418–491. (doi:10.1090/S0002-9947-1959-0110078-1)

63. Niyogi P, Smale S, Weinberger S. 2008 Finding the Homology of Submanifolds with High Confidence from Random Samples. *Discret. Comput. Geom.* **39**, 419–441. (doi:10.1007/s00454-008-9053-2)
64. Li Z, Tang T. 2017 *Numerical solution of differential equations: introduction to finite difference and finite element methods*. Cambridge, UK: Cambridge University Press.
65. Shannon CE. 1949 Communication in the presence of noise. *Proceedings of the IRE* **37**, 10–21. (doi:10.1109/JRPROC.1949.232969)
66. Lai RY, Li Q, Uhlmann G. 2019 Inverse problems for the stationary transport equation in the diffusion scaling. *SIAM J. Appl. Math.* **79**, 2340–2358. (doi:10.1137/18m1207582)
67. Newton K, Li Q, Stuart AM. 2020 Diffusive optical tomography in the Bayesian framework. *Multiscale Model. Simul.* **18**, 589–611. (doi:10.1137/19m1247346)
68. Li Q, Newton K. 2019 Diffusion equation-assisted Markov Chain Monte Carlo methods for the inverse radiative transfer equation. *Entropy* **21**, 291. (doi:10.3390/e21030291)
69. Lin G, Liu H, Liao W, Zhang Z. 2025 Data for: Coefficient-to-Basis Network: a fine-tunable operator learning framework for inverse problems with adaptive discretizations and theoretical guarantees. Purdue University Research Repository (doi:10.4231/WDY7-X547)