RESEARCH ARTICLE

*National Science Open*
3: 20240001, 2024
https://doi.org/10.1360/nso/20240001

Information Sciences

# Learning neural operators on Riemannian manifolds

Gengxiang Chen[1,#], Xu Liu[2,#,*], Qinglu Meng[1], Lu Chen[1], Changqing Liu[1] & Yingguang Li[1,*]

[1] *College of Mechanical and Electrical Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China;*
[2] *School of Mechanical and Power Engineering, Nanjing Tech University, Nanjing 211816, China*

#Equally contributed to this work.
*Corresponding authors (emails: liuxu.smpe@njtech.edu.cn (Xu Liu); liyingguang@nuaa.edu.cn (Yingguang Li))

Received 9 January 2024; Revised 5 March 2024; Accepted 10 April 2024; Published online 12 April 2024

**Abstract:** Learning mappings between functions (operators) defined on complex computational domains is a common theoretical challenge in machine learning. Existing operator learning methods mainly focus on regular computational domains, and have many components that rely on Euclidean structural data. However, many real-life operator learning problems involve complex computational domains such as surfaces and solids, which are non-Euclidean and widely referred to as Riemannian manifolds. Here, we report a new concept, neural operator on Riemannian manifolds (NORM), which generalises neural operator from Euclidean spaces to Riemannian manifolds, and can learn the operators defined on complex geometries while preserving the discretisation-independent model structure. NORM shifts the function-to-function mapping to finite-dimensional mapping in the Laplacian eigenfunctions' subspace of geometry, and holds universal approximation property even with only one fundamental block. The theoretical and experimental analyses prove the significant performance of NORM in operator learning and show its potential for many scientific discoveries and engineering applications.

**Keywords:** deep learning, neural operator, partial differential equations, Riemannian manifold

## INTRODUCTION

Many scientific discoveries and engineering research activities involve the exploration of the intrinsic connection and relationship between functions [1,2]. In mathematics, the mapping between two functions is called the operator [3]. Establishing the operator defined on complex computational domains has been a theoretical challenge [4]. One ubiquitous example of operator is the solution operator of partial differential equations (PDEs) [5], which provides the foundational descriptions of many nature laws. Solving PDEs under different parameters, initial and boundary conditions can be regarded as finding the solution operators [6,7]. A more practical example is that, for nuclear fusion, establishing the operator that links the input controlling coil voltage to the plasma distribution in the complex tokamak vessel could enable rapid and accurate forecasting of plasma field evolution, thus pointing to a promising direction towards sustainable fusion [8]. There are also requirements for establishing operators in a wide range of other complex field prediction scenarios, such as predicting the blood flow dynamics of the human body for the purpose of cardiovascular disease diagnosis and treatment [9], and predicting the pressure field of an aircraft for fuselage structure optimisation [10,11]. Physical experiments and numerical simulations are commonly used methods for finding the mapping between two functions (i.e., operators) [1]. Due to the complex process of the underlying operators,

especially when involving complex computational domains like tokamak vessels, human organs or aircraft structures, the high computational and experimental costs of these methods are prohibitive for real-world situations [12, 13].

Artificial intelligence (AI) techniques recently emerged as a promising paradigm shift for learning operators directly from data [1, 14, 15]. Classical deep learning methods, such as convolutional neural networks (CNNs) and deconvolution techniques [2], can learn the mapping between discretised picture-like uniform grid data to approximate the operator [16, 17]. Graph neural networks (GNNs) can represent the computational domain as a graph and then learn the properties of the nodes through message passing [18, 19]. However, since the network structure and the parameterisation of CNNs and GNNs heavily depend on the discretisation-resolution of the computational domain [20], the high-dimensional discretisation of the computational domains will bring significant computational burdens to model training, and lead to slow convergence or even divergence when learning general nonlinear operators [21]. Recently, neural operators (NOs), such as DeepONet [22] and fourier neural operator (FNO) [23], were proposed as a new deep learning approach that could directly learn mappings between functions on continuous domains with a discretisation-independent model structure (i.e., the parameterisation of the model is independent of the discretisation of the computational domain) [24]. Despite the significant success of NOs, they mainly focus on learning the mapping between functions defined on regular computational domains (data in the form of a picture-like uniform grid), and many components of these methods rely on Euclidean structural data, such as Fast Fourier Transform in FNO [23] and its variant Factorized-FNO [25], image convolution layer in U-shaped neural operator (UNO) [26], and Wavelet transform in wavelet neural operator (WNO) [27]. However, real-life applications are more complex and many are in irregular computational domains. Existing NOs often have to convert irregular data to the form as regular uniform grid by coordinate transformation [28, 29] or grid interpolation [20, 30]. However, coordinate transformation techniques are normally limited to converting simple two-dimensional (2D) irregular computational domains due to the poor intrinsic representation [20, 28], whilst grid interpolation often leads to high-dimensional discretisation and thus brings significant computational burdens to model training, especially for three-dimensional (3D) computational domains [17]. Therefore, existing NOs has limitations in solving operator learning problems of real-life applications with irregular computational domains, including complex surfaces and solids, which are non-Euclidean structural data, and widely referred to as Riemannian manifolds.

This research proposed a deep learning framework with a new concept called neural operator on Riemannian manifolds (NORM), as shown in Figure 1A. NORM could break the limitations of existing NOs and extend the applicability from Euclidean spaces to Riemannian manifolds. NORM can learn the mapping between functions defined on any Riemannian manifolds, including 2D and 3D computational domains, while maintaining a model structure independent of discretisation. Compared with learning operators directly in the Euclidean space, the fundamental blocks of NORM shift the function-to-function mapping to the finite-dimensional mapping in the Laplacian eigenfunctions' subspace of geometry (Figure 1C). Because Laplacian eigenfunctions have been proven to be the optimal basis for approximating functions on Riemannian manifolds [31], NORM can learn the global geometric information effectively and accurately without increasing the complexity of parameterisation. Besides, we have proved that NORM could hold the universal approximation property even with only one fundamental block. The effectiveness of the proposed framework was demonstrated through several different tasks in science and engineering, including learning solution operators

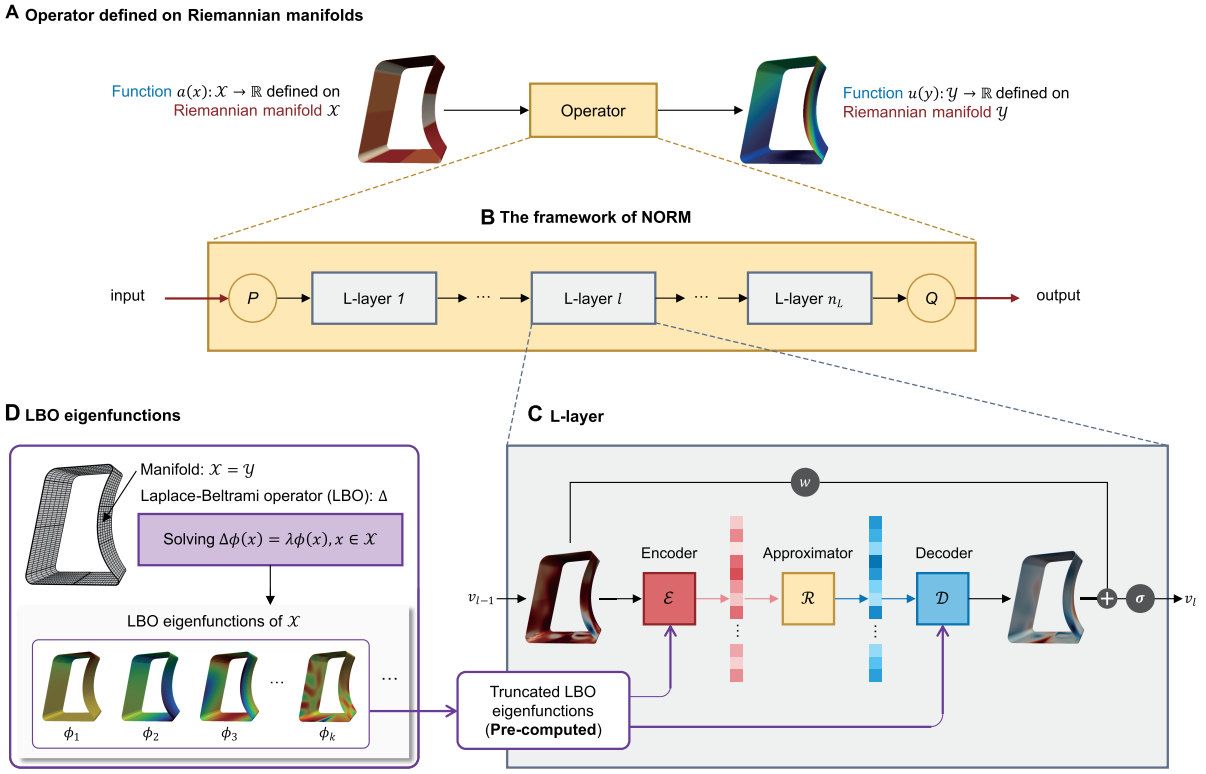


**Figure 1** The illustration of NORM. (A) Operators defined on Riemannian manifolds, where the input function and output function can be defined on the same or different Riemannian manifolds. The example for this illustration is the operator learning problem of the composite curing case, where the input temperature function and the output deformation function are both defined on the same manifold, the composite part. (B) The framework of NORM, consists of two feature mapping layers (P and Q) and multiple L-layers. (C) The structure of L-layer, consists of the encoder-approximator-decoder block, the linear transformation, and the non-linear activation function. (D) Laplace-Beltrami operator (LBO) eigenfunctions for the geometric domain (the composite part).

for classical PDEs, composite workpiece deformation prediction and blood flow dynamics prediction.

## NEURAL OPERATOR ON RIEMANNIAN MANIFOLDS

### Problem definition

Learning operators on Riemannian manifolds refers to learning a mapping between two functions defined on Riemannian manifolds, as shown in Figure 1A. Denote $\mathcal{G} : \mathcal{A}(\mathcal{X};\mathbb{R}) \to \mathcal{U}(\mathcal{Y};\mathbb{R})$ a continuous operator, namely the underlying mapping between the input and the output functions. The input $a \in \mathcal{A}(\mathcal{X};\mathbb{R})$ is a function $a(x) : \mathcal{X} \to \mathbb{R}$, $x \in \mathcal{X}$, the output $u \in \mathcal{U}(\mathcal{Y};\mathbb{R})$ is a function $u(y) : \mathcal{Y} \to \mathbb{R}$, $y \in \mathcal{Y}$. $\mathcal{X}$ and $\mathcal{Y}$ are Riemannian manifolds. Assuming that both $\mathcal{A}$ and $\mathcal{U}$ are $L^2$ spaces, then, the problem of learning operator on Riemannian manifolds is to learn a parameterised operator $\mathcal{G}_\theta$ to approximate $\mathcal{G}$, i.e., $\mathcal{G}_\theta \approx \mathcal{G}, \theta \in \mathbb{R}^p$.

Since the input function $a$ and the output function $u$ are both defined on Riemannian manifolds, the obvious solution is to transfer them into a new representation that can be processed with existing Euclidean learning models. Ideally, the solution should be feasible and consistent for any functions defined on Riemannian manifolds. At the same time, the new representation should be low-dimensional while maintaining the

information of the original functions. Therefore, we first propose a simple approximation block with an encoder-approximator-decoder structure to transfer the mapping between functions on Riemannian manifolds to a finite-dimensional mapping on Euclidean space.

The approximation block for learning operators on Riemannian manifolds can be defined as a mapping $\mathcal{N} : \mathcal{A}(\mathcal{X};\mathbb{R}) \to \mathcal{U}(\mathcal{Y};\mathbb{R})$ of the form $\mathcal{N} = \mathcal{D} \circ \mathcal{R} \circ \mathcal{E}$, where $\mathcal{E} : \mathcal{A}(\mathcal{X};\mathbb{R}) \to \mathbb{R}^{d_X}$ denotes the encoder that maps the function on manifold $\mathcal{X}$ to Euclidean space, $\mathcal{R} : \mathbb{R}^{d_X} \to \mathbb{R}^{d_Y}$ is an approximator, a learning model for Euclidean data, $\mathcal{D} : \mathbb{R}^{d_Y} \to \mathcal{U}(\mathcal{Y};\mathbb{R})$ is an inverted mapping to recover the prediction function on manifold $\mathcal{Y}$.

Encoder-approximator-decoder structures were widely applied in the machine learning field, such as the transformer [32] and autoencoder [33], and also in operator learning problems [34,35]. However, the encoder and decoder of existing research are primarily designed for dealing with Euclidean structural data. To learn operators on Riemannian manifolds, the primary challenge lies in how to design the encoder and decoder mapping to process functions on manifolds without increasing the model complexity. These two mappings would not only influence the feature extraction capability of the learning model, but also determine whether the model holds universal approximation property.

### Constructing mappings using Laplacian

The discretisation-independent target of the neural operator reminds us of the mesh-free spectral method in PDE solving [36]. Intuitively, the spectrum of manifolds could naturally describe the intrinsic information of operators on manifolds. The ideal choice of the spectrum for operator learning is the eigenfunctions of the Laplacian, which is a set of orthonormal basis [37], and has been proven to be the optimal basis for approximating functions defined on Riemannian manifolds [31,38]. Therefore, the encoder $\mathcal{E}$ and the decoder $\mathcal{D}$ could be constructed as the spectral decomposition and the spectral reconstruction on the corresponding Laplacian eigenfunctions.

The Laplacian occurs in a wide range of differential equations describing science and engineering problems, such as the heat transfer function, Poisson's equation, diffusion equation, and wave equation [38]. For the Euclidean space $\mathbb{R}^d$ and a twice-differentiable function $f$, the Laplacian $\Delta f$ is a second-order differential operator defined as the divergence $\nabla\cdot$ of the gradient $\nabla f$, that is $\Delta f = \nabla \cdot \nabla f = \nabla^2 f$. The eigenvalue problem for the Laplacian can be defined as $\Delta\phi_i = \lambda_i\phi_i$, where the $\lambda_i$ ($\lambda_1 \leqslant \lambda_2 \leqslant \ldots$) and $\phi_i(x)$ that satisfying this equation are defined as the eigenvalues and the corresponding eigenfunctions. Actually, the Fourier basis $e^{2\pi ikx}$ is also the eigenfunction of the Laplacian with the eigenvalue $\lambda = -(2\pi k)^2$ [39].

Since the divergence operator $\nabla\cdot$ and gradient operator $\nabla f$ can also be defined on manifolds with Riemannian metric $g$, the Laplacian $\Delta f$ can be naturally extended to the Riemannian manifold, which is also called the Laplace-Beltrami operator (LBO) [40]. Therefore, we could obtain the Laplacian spectrum of manifolds in a similar way as in Euclidean space, as shown in Figure 1D.

For Riemannian manifold $\mathcal{M}$, the LBO eigenfunctions $\phi_i(x)$ is a set of orthonormal bases for the Hilbert space $L^2(\mathcal{M})$. It can be proved that a finite number of leading LBO eigenfunctions can approximate functions on manifolds with any accuracy [31]. Therefore, for the approximation block $\mathcal{N} = \mathcal{D} \circ \mathcal{R} \circ \mathcal{E}$, the encoder $\mathcal{E}$ can be defined as the spectral decomposition on the LBO eigenfunctions $\phi_{\mathcal{X},i}$ of the input manifold $\mathcal{X}$:

$$\mathcal{E} : \mathcal{A} \to \mathbb{R}^{d_X}, \quad \mathcal{E}(a) := (\langle a, \phi_{\mathcal{X},1}\rangle, \ldots, \langle a, \phi_{\mathcal{X},d_X}\rangle), \quad \forall a \in \mathcal{A}. \tag{1}$$

And the decoder can be defined as the spectral reconstruction on the LBO eigenfunctions $\phi_{\mathcal{Y},i}$ of the output manifold $\mathcal{Y}$:

$$\mathcal{D}: \mathbb{R}^{d_{\mathcal{Y}}} \to \mathcal{U}, \quad \mathcal{D}(\beta) = \sum_{i=1}^{d_{\mathcal{Y}}} \beta_i \phi_{\mathcal{Y},i}, \quad \forall \beta \in \mathbb{R}^{d_{\mathcal{Y}}}. \tag{2}$$

With the defined encoder $\mathcal{X}$ and decoder $\mathcal{D}$, an approximation block $\mathcal{N} = \mathcal{D} \circ \mathcal{R} \circ \mathcal{E}$ could potentially learn the mappings between functions on manifolds with a simple parameterised Euclidean learning model $\mathcal{R}$. Since LBO can be defined on any Riemannian manifold, the block $\mathcal{N}$ can naturally deal with any complex geometric domain, which breaks the limitations that existing neural operators relying on Euclidean structured data. Meanwhile, the approximation block holds the discretisation-independent property, because $\mathcal{R}$ is parameterised on Euclidean spaces with size only related to the truncated eigenfunctions of the input and output manifolds.

Although Laplacian is defined mathematically on smooth domains, practical numerical computation typically requires discrete approximations of domains, such as meshes or point clouds. The LBOs of common geometric meshes have been strictly defined in the differential geometry field [41], including triangular mesh, quadrilateral mesh, and tetrahedral mesh. In Supplementary information S2.1, an example of discretised LBO for triangular mesh is provided.

## Framework of neural operator on Riemannian manifold

The approximation block $\mathcal{N} = \mathcal{D} \circ \mathcal{R} \circ \mathcal{E}$ can transfer the mapping between functions on Riemannian manifolds to a finite-dimensional Euclidean space learning problem. However, one basic block only approximates the target operator by a linear subspace, which is inefficient in extracting non-linear low-dimensional structures of data. Here, we propose a new deep learning framework, the NORM, that consists of multiple layers and in which the approximation block $\mathcal{N}$ constitutes one layer of the model, like the convolution layer in traditional CNN.

We start from a common situation, assuming the input and output functions are defined on the same manifold $\mathcal{M}$, i.e., $\mathcal{X} = \mathcal{Y} = \mathcal{M}$. The structure of NORM can be represented as the form shown in Figure 1B, consisting of two feature mapping layers $\mathcal{P}$, $\mathcal{Q}$ and $n_L$ hidden layers. The shallow network $\mathcal{P}$ lifts the input function $a$ to get $v_0 = \mathcal{P}(a)$, where $\mathcal{P}: L^2(\mathcal{M}; \mathbb{R}) \to L^2(\mathcal{M}; \mathbb{R}^{d_v})$, so as to expand the dimension of features to increase the representation ability, similar to the convolution channel expansion in CNN. Multiple hidden layers, defined as the Laplace layer, or L-layer (Figure 1C), would update the input function iteratively, such as $v_{l-1} \mapsto v_l = \mathcal{L}_l(v_{l-1})$ in the L-layer $l$. After that, the final shallow network $Q$ would project the high-dimensional features to the output dimension, namely $u = Q(v_{n_L})$, where $\mathcal{Q}: L^2(\mathcal{M}; \mathbb{R}^{d_v}) \to L^2(\mathcal{M}; \mathbb{R})$. The iterative structure can be represented as

$$\mathcal{G}_\theta(a) = \mathcal{Q} \circ \mathcal{L}_{n_L} \circ \mathcal{L}_{n_L-1} \circ \cdots \circ \mathcal{L}_1 \circ \mathcal{P}(a). \tag{3}$$

The iteration of the hidden layers is given as follows:

$$v_l = \mathcal{L}_l(v_{l-1})(x) := \sigma\left(W_l v_{l-1}(x) + b_l(x) + \mathcal{N}(v_{l-1})(x)\right), \quad \forall x \in \mathcal{M}, \tag{4}$$

where the linear transformations $W_l \in \mathbb{R}^{d_v \times d_v}$ and the bias $b_l \in \mathbb{R}^{d_v}$ are defined as pointwise mapping. $\sigma$ is the non-linear activation function like in the traditional neural network. Note that, the LBO eigenfunctions

required in the approximation block can be pre-computated before training the model, as shown in Figure 1D. The detailed implementation of the discredited version of the approximation block $\mathcal{N}(v)$ is provided in Supplementary information S1.1.

The above definition introduces the NORM structure where the input and output are defined on the same manifold. Nevertheless, the structure can be easily generalised to the settings where the input and output are defined on different manifolds and several different structures are introduced in Supplementary information S1.2.

Note that the parameterisation of NORM is independent of the discretisation of the input and output functions, because all operations are defined directly in the function spaces on manifolds rather than the Euclidean coordinate spaces. $\mathcal{P}$, $\mathcal{Q}$ are learnable neural networks between finite-dimensional Euclidean spaces and have the same point-wise parameterisation for all $x \in \mathcal{M}$. Therefore, NORM can learn the mappings between functions on any Riemannian manifolds while maintaining the discretisation-independent property.

## Universal approximation of NORM

Many recent studies have investigated the universal approximation properties of neural operators between functions on Euclidean spaces [24, 42]. This section will show the advantage of the proposed method that even one approximation block $\mathcal{N}$ of NORM holds the universal approximation ability in learning operators between functions defined on Riemannian manifolds.

Let $\mathcal{N} = \mathcal{D} \circ \mathcal{R} \circ \mathcal{E}$ be a neural operator for the continuous mapping $\mathcal{A}(\mathcal{X}; \mathbb{R}) \to \mathcal{U}(\mathcal{Y}; \mathbb{R})$, and $\mathcal{R} : \mathbb{R}^{d_\mathcal{X}} \to \mathbb{R}^{d_\mathcal{Y}}$ represents a neural network that has universal approximation property. The encoder is defined as $\mathcal{E} : \mathcal{A} \to \mathbb{R}^{d_\mathcal{X}}$ and $\mathcal{E}(a) := (\langle a, \phi_{\mathcal{X},1} \rangle, \ldots, \langle a, \phi_{\mathcal{X},d_\mathcal{X}} \rangle), \forall a \in \mathcal{A}$. The decoder is defined as $\mathcal{D} : \mathbb{R}^{d_\mathcal{Y}} \to \mathcal{U}$, and $\mathcal{D}(\beta) = \sum_{i=1}^{d_\mathcal{Y}} \beta_i \phi_{\mathcal{Y},i}, \forall \beta \in \mathbb{R}^{d_\mathcal{Y}}$. $\mathcal{X}$ and $\mathcal{Y}$ are Riemannian manifolds. $\mathcal{A}$ and $\mathcal{U}$ are $L^2$ spaces. $\phi_{\mathcal{X},i}$ and $\phi_{\mathcal{Y},i}$ are LBO eigenfunctions of manifolds $\mathcal{X}$ and $\mathcal{Y}$, respectively. It should be noted that $\mathcal{N} = \mathcal{D} \circ \mathcal{R} \circ \mathcal{E}$ is a basic block of the NORM, and also can be treated as the simplified version of NORM. Therefore, the universal approximation property of $\mathcal{N}$ could guarantee the universal approximation property of the more complex NORM framework. The universal approximation theorem of neural operators on Riemannian manifolds is as follows.

**Theorem. Universal approximation theorem for neural operators on Riemannian manifolds.** Let $\mathcal{G} : \mathcal{A}(\mathcal{X}; \mathbb{R}) \to \mathcal{U}(\mathcal{Y}; \mathbb{R})$ be a Lipschitz continuous operator, $K \in \mathcal{A}$ is compact set. Then for any $\epsilon > 0$, there exists a neural operator $\mathcal{N} = \mathcal{D} \circ \mathcal{R} \circ \mathcal{E}$, such that:

$$\sup_{a \in K} \|\mathcal{G}(a) - \mathcal{N}(a)\|_{L^2} \leqslant \epsilon. \tag{5}$$

*Proof.* It is challenging to directly prove the approximation error from $\mathcal{A} \to \mathcal{U}$. Therefore, we establish a low-dimensional projection subspace of $\mathcal{A}$ and $\mathcal{U}$ spanned by the corresponding LBO eigenfunctions. It can be first proved that $\mathcal{N}$ holds universal approximation property in learning operators between the projection subspaces. Since LBO eigenfunctions is a group of basis in $L^2$ space, the projection error can be proven to be $\epsilon-$approximation. Therefore, the final approximation error of $\mathcal{N}$ can be obtained by combining the approximation error on the subspace, the encoding error on the input, and the decoding error on the output. The detailed proof can be found in Supplementary information S3.

**Table 1** Performance comparison for the five case studies [a]

| Case | Metric | GNN | DeepONet | POD-DeepONet | FNO | WNO | NORM |
|---|---|---|---|---|---|---|---|
| 1. Darcy problem | MME | 0.140(0.002) | $0.045(4\times10^{-4})$ | $0.044(3\times10^{-4})$ | 0.094(0.002) | $0.077(2\times10^{-4})$ | $0.039(4\times10^{-4})$ |
| | $E_{L_2}(\%)$ | 6.732(0.053) | 1.358(0.013) | 1.296(0.023) | 3.826(0.077) | 1.090(0.030) | 1.046(0.020) |
| 2. Pipe turbulence | MME | 2.358(0.125) | 0.960(0.002) | 0.241(0.017) | 0.896(0.001) | 0.892(0.001) | 0.116(0.003) |
| | $E_{L_2}(\%)$ | 23.583(1.411) | 9.358(0.107) | 2.587(0.275) | 3.801(0.002) | 3.786(0.003) | 1.008(0.020) |
| 3. Heat transfer | MME | – | 3.038(0.156) | 1.304(0.045) | – | – | 1.599(0.096) |
| | $E_{L_2}(\%)$ | – | 0.072(0.002) | 0.057(0.001) | – | – | 0.027(0.002) |
| 4. Composite | MME | 0.882(0.029) | 0.157(0.002) | 0.077(0.003) | – | – | 0.051(0.002) |
| | $E_{L_2}(\%)$ | 20.908(0.050) | 1.880(0.034) | 1.437(0.060) | – | – | 0.999(0.027) |
| 5. Blood flow | MME | – | 0.899(0.010) | 0.488(0.002) | – | – | 0.093(0.003) |
| | $E_{L_2}(\%)$ | – | 60.613(0.961) | 33.256(0.126) | – | – | 4.822(0.061) |

a: The values A(B) represent the mean and standard deviation of five repeated runs, respectively.

## RESULTS

The proposed NORM was verified using three toy cases and two practical engineering cases with 2D or 3D complex geometric domains. The three toy cases of learning PDEs solution operators involved different problem settings and input/output structures: (1) the Darcy problem case aims to learn the mapping from the parameter function (the diffusion coefficient field) to the solution function (the pressure function field), where both functions are defined on the same 2D manifold; (2) the pipe turbulence case is a classical dynamics systems prediction setting, namely, predicting the future state field based on the current state field (velocity field in the pipe); and (3) the heat transfer case tries to learn the mapping from the boundary condition (temperature function on 2D manifold) to the temperature field of the part (temperature function on 3D manifold). The two engineering cases are composite workpiece deformation prediction and blood flow dynamics prediction: (4) the composite case aims to learn the mapping from the temperature field to the final deformation field of a 3D composite workpiece, where the deformation mechanism involves complex physicochemical processes other than only PDEs, and (5) for the blood flow dynamics case, the inputs are multiple time series functions, and the output is the spatiotemporal velocity field of the aortic (3D manifold).

We compared NORM with several popular neural operators, including DeepONet [22], POD-DeepONet [20], FNO [23], WNO [27] and also one classical Graph Neural Networks (GNN), named GraphSAGE [43]. For the 2D cases, the irregular geometric domains were interpolated to a regular domain for the implementation of FNO and WNO. For the 3D cases, we did not compare with FNO and WNO because of the prohibitive complexity of 3D spatial interpolation. Since the message-passing mechanism in graph learning methods typically focuses on problems with the same input and output graphs, we did not compare GNN for the heat transfer case and blood flow dynamics case. The details about data generation and baseline model configurations are described in the Supplementary information S4, S5 and S6.2. The quantitative comparison results of all methods are presented in Table 1. We considered two error metrics: $E_{L_2}$ is the mean relative $L_2$ error of all test samples, and mean maximum error (MME) refers to the mean value of all test samples in terms of the maximum error in the whole computational domain.

## Learning PDEs solution operators

PDEs provide the foundational descriptions of many nature phenomena and physics laws. Machine learning methods have been reported with many successes in PDEs prediction and discovery [1, 15]. The recent proposed PDE discovering method, Sparse Spatiotemporal System Discovery ($S^3d$), can even automatically discover the physical terms of PDEs without leveraging prior knowledge or assumptions, thus holding significant potential to facilitate scientific discovery [44]. With more PDEs being discovered and established, learning solution operators for these PDEs under different parameters, initial, and boundary conditions becomes a necessary direction for accelerating engineering applications [4]. In this section, we will focus on learning the solution operators for three typical PDEs: the Darcy problem, pipe turbulence, and heat transfer.

### *Darcy problem (Case 1)*

Darcy flow equation is a classical law for describing the flow of a fluid through a porous medium. This problem is also widely used for various neural operator verification [24]. Darcy's law can be mathematically described by the following equation:

$$-\nabla \cdot (a\nabla u) = f, \tag{6}$$

where $a$ is the diffusion coefficient field, $u$ is the pressure field and $f$ is the source term to be specified. As shown in Figure 2, the computational domain is a 2D geometric shape represented by a triangle mesh with 2290 nodes. The geometric domain has an irregular boundary with a thin rectangle notch inside, which can increase the complexity of the learning problem. The operator learning target in the Darcy flow problem is the mapping from the diffusion coefficient field $a(\mathbf{x})$ to the pressure field $u(\mathbf{x})$:

$$\mathcal{G} : a(\mathbf{x}) \mapsto u(\mathbf{x}), \quad \mathbf{x} \in \mathcal{M}. \tag{7}$$

The labelled data for training the neural operator model is the pair of $a(\mathbf{x})$ and $u(\mathbf{x})$. 1200 sets of input data $a(\mathbf{x})$ are randomly generated first. Then the corresponding $u(\mathbf{x})$ is solved by Matlab's SOLVEPDE toolbox. 1000 of them are used as the training dataset, and the rest 200 groups are defined as the test dataset.

Figure 2A and B show the input field and output field of one representative sample in the test dataset. Figure 2C–L report the comparative prediction results of different methods and the corresponding prediction errors, in which $\Delta_{\text{mean}}$ refers to the average absolute error over all nodes in the geometric domain, and $\Delta_{\text{max}}$ means the maximum absolute error on all nodes. The comparison shows that the output field and the NORM predicted result exhibit excellent agreement. Due to the influence of grid interpolation, FNO has the most significant error, especially in the boundary region. WNO can provide a more accurate prediction compared to FNO, with the average error even approaching the result of NORM. DeepONet and POD-DeepONet show significant errors on the right side of the rectangle. The quantitative results of the test dataset are listed in Table 1. NORM can achieve the lowest error compared with all other baseline methods.

### *Pipe turbulence (Case 2)*

Turbulence is a vital flow state of the fluid, which reflects the instability of the fluid system [45]. Here, we considered turbulent flows in a complex pipe, of which the underlying governing law is the 2D Navier-Stokes

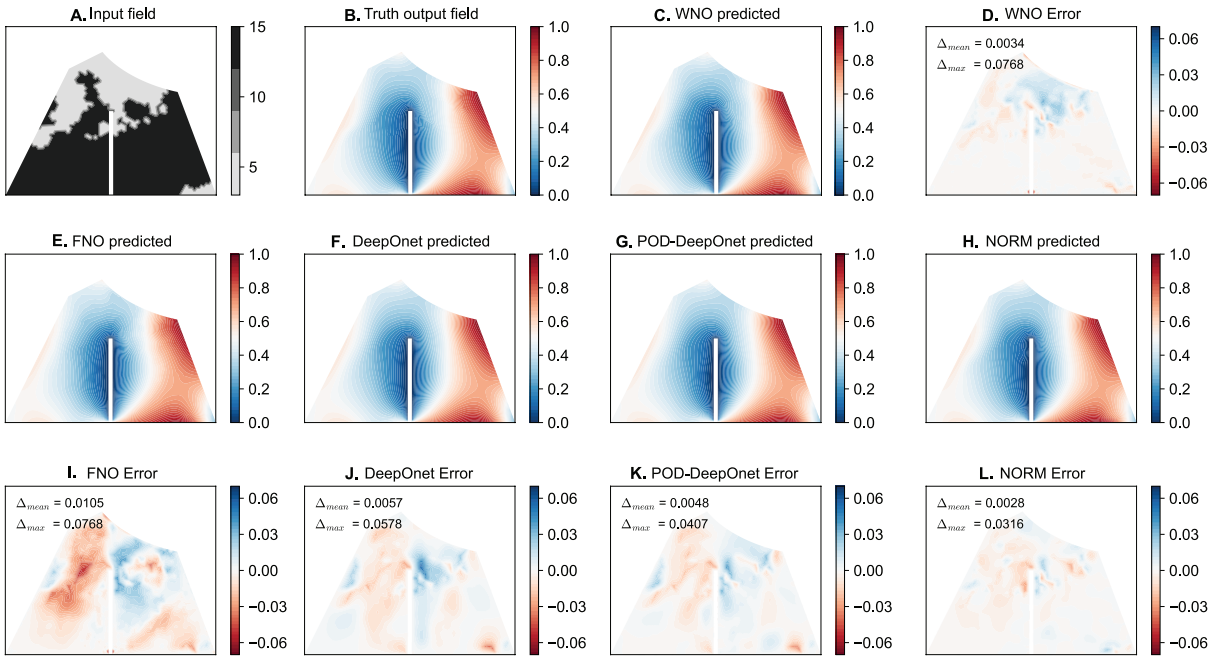


**Figure 2** Experimental results of the Darcy problem (Case 1). (A, B) The input and output fields for a representative sample. (C, E–H) The prediction results of different methods. (D, I–L) The prediction errors of different methods.

equation for a viscous incompressible fluid:

$$\begin{aligned} \frac{\partial \mathbf{v}}{\partial t} + (\mathbf{v} \cdot \nabla)\mathbf{v} &= -\nabla p + \mu \nabla^2 \mathbf{v}, \\ \nabla \cdot \mathbf{v} &= 0, \end{aligned} \tag{8}$$

where $\mathbf{v}$ is the velocity, $p$ is the pressure, and the fluid chosen is water. The geometric design of the irregular pipe is shown in Figure 3 where the left and right ends are inlet and outlet, respectively. For a given inlet velocity, we performed the transient simulation to predict the velocity distribution in the pipe. The velocity field data are represented by a triangular mesh with 2673 nodes. Details about data generation and simulation settings can be found in the Supplementary information S4.1.2. The operator learning problem of this case is defined as the mapping from the velocity field $\mathbf{v}(\mathbf{x}, t_1)$ to the velocity field $\mathbf{v}(\mathbf{x}, t_2)$, where $t_2 = t_1 + 0.1$ s:

$$\mathcal{G} : \mathbf{v}(\mathbf{x}, t_1) \mapsto \mathbf{v}(\mathbf{x}, t_2), \quad \mathbf{x} \in \mathcal{M}. \tag{9}$$

The prediction results and errors of baseline models are provided in Figure 3C–L. As shown in Figure 3L, NORM can give a consistent prediction compared with the ground truth. WNO and FNO achieve minor errors in smooth areas but large errors in sharp areas because of the grid interpolation, leading to small $\Delta_{\text{mean}}$ but large $\Delta_{\text{max}}$ (shown in Figure 3D and I). POD-DeepONet, like NORM, has a uniform distribution of errors, while the error value is slightly larger than NORM. DeepONet has the most significant prediction error compared to other methods in this task. The quantitative statistical results can be seen in Table 1.

### *Heat transfer (Case 3)*

Heat transfer describes the transfer of energy as a result of a temperature difference, which widely occurs in nature and engineering technology [46]. The heat equation can be written in the following form (assuming

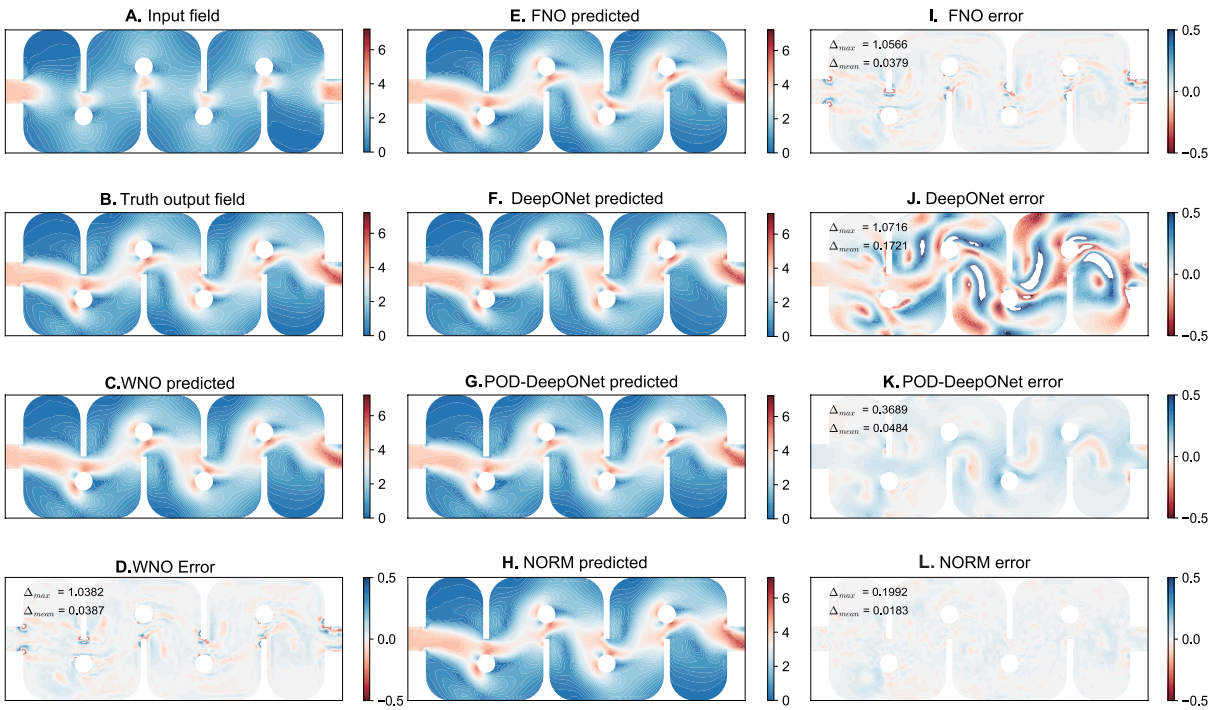


**Figure 3** Experimental results of the pipe turbulence (Case 2). (A, B) The input and output fields for a representative sample. (C, E–H) The prediction results of different methods. (D, I–L) The prediction errors of different methods.

no mass transfer or radiation):

$$\rho C \frac{\partial T}{\partial t} = \nabla \cdot K \nabla T + \dot{Q}, \tag{10}$$

where $T$ is the temperature as a function of time and space. $\rho$, $C$, and $K$ are the density, specific heat capacity, and thermal conductivity of the medium, respectively. And $\dot{Q}$ is the internal heat source.

The heat transfer case was designed on a 3D solid part, as shown in Figure 4C. The learning problem is defined as the mapping from the 2D boundary condition $T_{bc}(\mathbf{x})$ to the 3D temperature field $T_{t=3\mathrm{s}}(\mathbf{y})$ of the solid part after 3 s of heat transfer.

$$\mathcal{G} : T_{bc}(\mathbf{x}) \mapsto T_{t=3\ \mathrm{s}}(\mathbf{y}), \quad \mathbf{x} \in \mathcal{X}, \mathbf{y} \in \mathcal{Y}. \tag{11}$$

As shown in Figure 4A, the input geometric domain is represented by a triangular mesh with 186 nodes, and the output geometric domain is represented by a tetrahedral mesh with 7199 nodes. The labelled dataset was generated by the commercial simulation software Comsol. The training dataset consists of 100 labelled samples, and another 100 groups are defined as the test dataset. More details are given in the Supplementary information S4.1.3.

In this case, the input and output functions are defined on different manifolds, thus the different L-layers of NORM have to utilise different LBO eigenfunctions. The model structure of NORM is given in Figure S1B. The beginning L-layers of NORM employ the LBO eigenfunctions of the input manifold for both the encoder and decoder. One middle L-layer of NORM utilises the LBO eigenfunctions of the input manifold for the encoder while taking the LBO eigenfunctions of the output manifold for the decoder. The ending L-layers employ LBO eigenfunctions of the output manifold for both the encoder and decoder. FNO is not

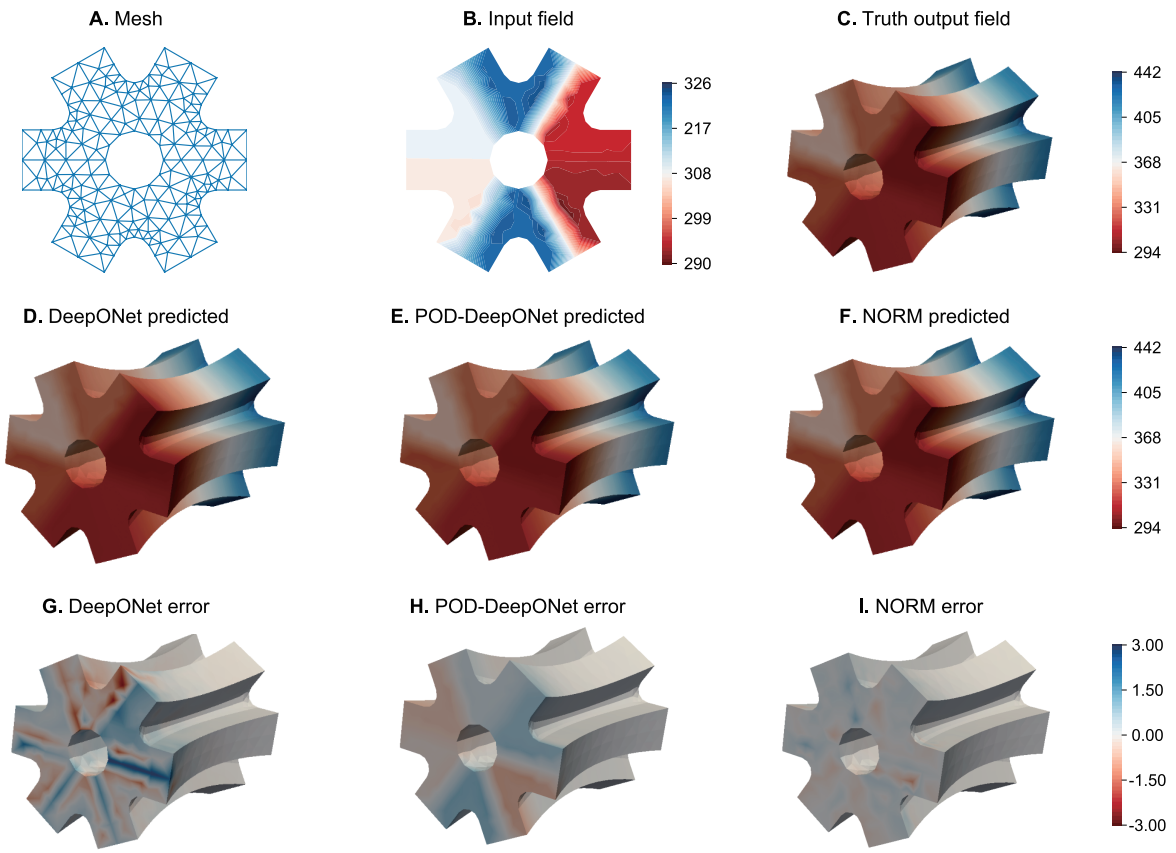


**Figure 4** Experimental results of the heat transfer case (Case 3). (A) The mesh for the input geometric domain. (B, C) The input and output fields for a representative sample. (D–F) The prediction results of different methods. (G–I) The prediction errors of different methods.

implemented for this case due to the prohibitive computational complexity of 3D spatial interpolation. The prediction results of different methods for one typical test data are shown in Figure 4. DeepONet has a large prediction error where the temperature gradient is large. POD-DeepONet has different errors on different temperature regions of the left end face, while the error of NORM is smaller and only appears in a few small areas. Moreover, the statistical results for all methods on the test dataset are shown in Table 1, where NORM shows the smallest relative $L_2$ error.

## Composite workpiece deformation prediction (Case 4)

This case study investigated the effectiveness of the proposed NORM on a complex 3D irregular geometry, specifically in predicting the curing deformation of a carbon fiber reinforced polymer (CFRP) composite part. CFRP composites are known for their lightweight and high-strength properties, thus becoming preferred materials for weight reduction and performance enhancement in modern aerospace industries [13]. The large size and high accuracy requirements of aerospace CFRP composite parts impose increased demands on deformation control during the curing process [47]. Regulating the curing temperature distribution of a part is an effective means of controlling curing deformation. Therefore, constructing the predictive model of the temperature-to-deformation field on the geometry can provide essential support for further curing process optimisation [48].

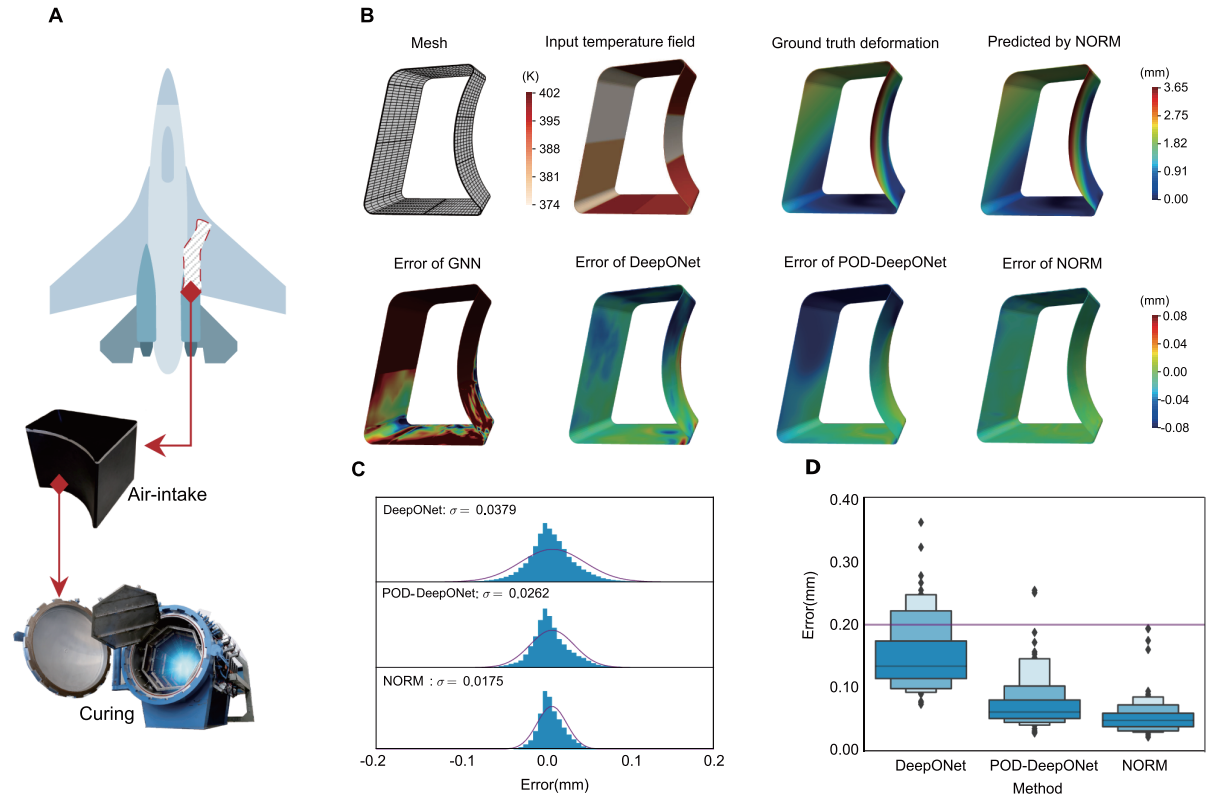


**Figure 5** Composite workpiece deformation prediction case (Case 4). (A) Illustration of the air-intake workpiece and the composite curing. (B) The input and output of the operator learning problem, the predicted deformation of NORM, and the prediction error of comparison methods. (C) The distribution of deformation prediction error over all nodes of all test samples. (D) The maximum prediction errors of all test cases for the three methods.

As shown in Figure 5A, the CFRP composite workpiece used for the case study is the air-intake structural part of a jet. This workpiece is a complex closed revolving structure formed by multiple curved surfaces, which would deform significantly after high-temperature curing. The learning problem of this case is defined as the mapping from the temperature field $a(x, y, z)$ to the deformation field $u(x, y, z)$ on the given composite part.

Figure 5B shows the prediction result of NORM and the prediction error of baseline methods of one test sample. It can be found that the error map of NORM is almost "green" for the whole part, which means that predicted deformation field is very close to the reference value. Table 1 shows that NORM outperforms all baseline methods in both $E_{L_2}$ and MME. Figure 5C shows the distribution of prediction error over all nodes of all test samples. It can be seen that the prediction errors of all nodes for all methods show Gaussian distributions with mean values approximating zero. The estimated standard deviations of different methods are marked in each figure. By comparison, NORM can achieve a lower prediction error uniformly and comprehensively for most nodes.

Composite manufacturing is a risk-sensitive problem, so it is not sufficient to consider only the relative $L_2$ error and average statistical results. According to the deformation prediction evaluation criteria provided by the engineers of the collaborating company, the maximum prediction error of the deformation field predicted by the data-driven model should be less than 0.2 mm. Figure 5D reports the maximum prediction errors of

all test cases. NORM is not only far outperforming the comparative methods but also has all test samples with a maximum prediction error of less than 0.2 mm.

### Blood flow dynamics prediction (Case 5)

Blood flow dynamics is the science of studying the characteristics and regularities of blood movement and its constituents in the organism, which is closely related to human health [49]. To explore the potential of NORM for aortic hemodynamic modelling (Figure 6A), we consider a similar scenario as described in reference [50] where the inputs $pv|_{[0,T]\times 6}$ are time-varying pressure and velocity at the inlet/outlets, and the output $\mathbf{v}|_{\mathcal{M}\times[0,T]\times 3}$ is the velocity field of blood flow consisting of velocity components in three directions [51], as shown in Figure 6B. The spatial domain is represented by a tetrahedral mesh with 1656 nodes, and the temporal domain is discrete with 121 temporal nodes. It is worth pointing out that the challenges of this case lie in two aspects: (1) time-space complexity, i.e., the output function defined on the complex geometric domain is time-varying; (2) unbalanced node values, i.e., the velocity of most nodes is close to zero due to the no-slip boundary condition.

Since the Fourier basis is also a group of the LBO eigenfunction, NORM can naturally deal with the temporal dimension of input and output functions using the Fourier basis, as discussed in Supplementary information S1.2. Hence NORM adopted the structure of Figure S1C. Statistics results of the NORM and two benchmarks (DeepONet and POD-DeepONet) are presented in Table 1. It is evident that NORM yields the smallest MME and relative $L_2$ error with minor variation. It stands to reason that at nodes $v \to 0$, even a slight prediction bias would lead to a significant relative $L_2$ error, but the proposed NORM achieves an impressive relative $L_2$ error 4.822%, compared with 89.26% of DeepONet and 37.42% of POD-DeepONet, which demonstrates the remarkable approximation capability of NORM. Figure 6C compares the visualisation of the velocity streamlines (snapshots at a representative time) against baseline methods. We observe that NORM achieves an excellent agreement with the corresponding ground truth, while POD-DeepONet and DeepONet only learn the general trend of velocity distribution but lose the predictive accuracy of the node value. Especially, DeepONet fails to capture the local details of streamlines at inlets and outlets. To further show the temporal trajectory prediction performance of NORM, the additional comparison visualisations of the blood velocity fields for the time steps $t = 0.3$ s, $t = 0.6$ s and $t = 0.9$ s can be found in Supplementary information S6.1. Additionally, we provided a video supplement showcasing the prediction performance across the entire time series.

Furthermore, Figure 6D provides the perspective to investigate the predictive accuracy of the node velocity evolution over time, which projects the 3D vector onto the $xy$-plane. NORM agrees well with ground truth regarding phase and amplitude, while POD-DeepONet shows a smaller overall amplitude, and DeepONet loses accuracy in both aspects. Finally, the comparison between ground truth and predictions for the magnitude of the velocity vector at 5000 spatiotemporal nodes randomly sampled from all test samples is plotted in Figure 6E. Compared to NORM ($R^2 = 0.998$), despite a quasi-linear relationship maintained by POD-DeepONet ($R^2 = 0.859$), a prediction bias amplifies as the velocity increases. We conjecture that it is the approximation bias introduced by using the linear superposition method to fit complex nonlinear problems. As for DeepONet ($R^2 = 0.567$), since its training mode is point-wise and the loss function used for training is the relative $L_2$ error, the updating of the model parameters is mainly driven by the nodes $v \to 0$. Then, the

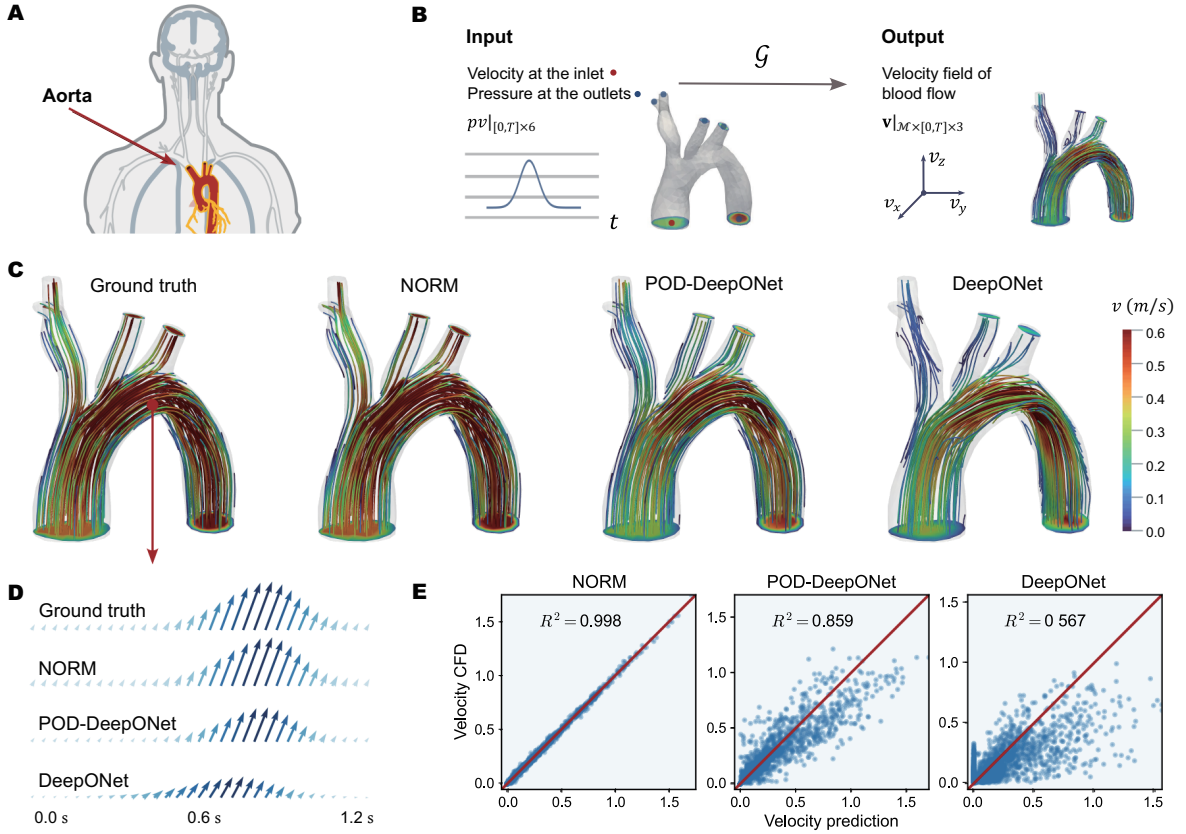


**Figure 6** Blood flow dynamics prediction case (Case 5). (A) Illustration of the human thoracic aorta, the largest human artery. (B) Illustration of the operator mapping $\mathcal{G}$. The inputs are the velocity at the inlet and the pressure at the outlets. The output is the velocity field of the blood flow. (C) Visualisation of the velocity streamlines (snapshots at a representative time) against baseline methods. (D) Comparison of node velocity evolution prediction over time. We project the 3D vector onto the *xy*-plane. (E) Comparison between ground truth and predictions for the magnitude of the velocity vector. We randomly sample 5000 spatiotemporal nodes from all test samples.

model outputs tend to be zero, resulting in a trade-off with the optimisation of other nodes. Therefore, the overall prediction of nodes in DeepONet appears more dispersed and does not show a linear relationship.

## Analysis

The encoder and the decoder of NORM are constructed by the spectral decomposition and the spectral reconstruction on the corresponding LBO eigenfunctions. This prompts a natural question: Could there be a more suitable basis than LBO eigenfunctions? From a model reduction point of view, the Proper Orthogonal Decomposition (POD) could also provide the projection basis to construct the encoder and the decoder. Consequently, NORM could be naturally extended to POD-NORM, wherein the POD modes of the training dataset replace the LBO eigenfunctions. Note that, the input data and the output data have different POD modes, so the structure of POD-NORM is similar to NORM with different input and output manifolds (Figure S1B in the Supplementary information). Therefore, NORM and POD-NORM were compared to demonstrate the advantages of LBO eigenfunctions. We focus on the performance comparison of the Darcy problem and the composite case, because the input fields of these two tasks are more complex, which brings more challenge to the representability of the spectrum. The data results reported in Figure 7 are the average based

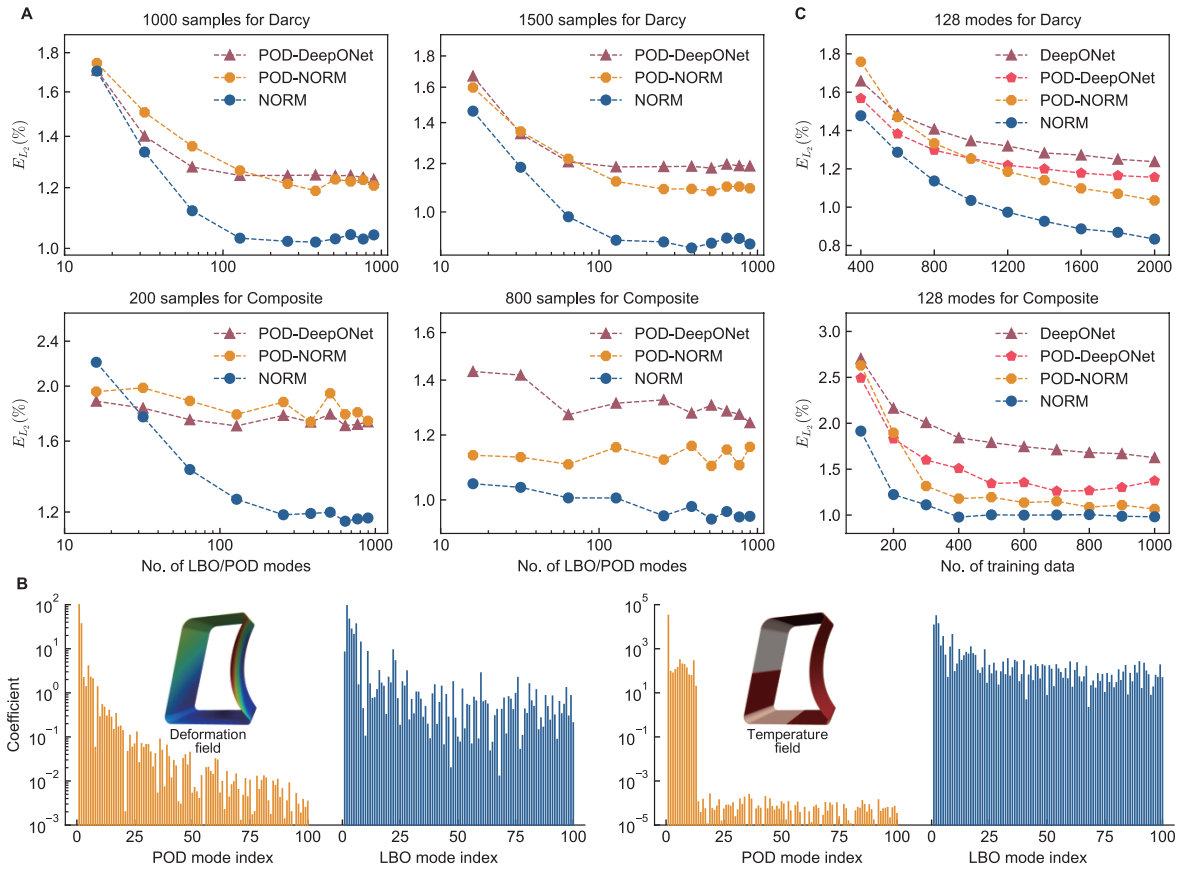


**Figure 7** Analysis results for different methods in the Darcy case and the composite case. (A) Comparison of POD-DeepONet, POD-NORM, and NORM for various numbers of modes in different sizes of the training dataset. (B) The coefficient analysis of the spectral decomposition of both the input temperature field and the output deformation field for the composite case using LBO and POD modes. (C) Comparison of DeepONet, POD-DeepONet, POD-NORM, and NORM for different sizes of training data while the number of LBO/POD modes is 128.

on five repeated runs.

We first compared the performance of NORM, POD-NORM, and POD-DeepONet across various numbers of modes from 16 to 896. Figure 7A shows the error tendency of the different methods with different mode numbers. Each case contains the results with two different sizes of the training dataset, $\{1000, 1500\}$ for the Darcy case and $\{200, 800\}$ for the composite case. For the Darcy case, the prediction errors of all three methods decrease rapidly as the number of modes increases, eventually converging to a stable performance level. Notably, POD-NORM and POD-DeepONet have similar performance, and NORM shows smaller errors under all number of modes. These findings indicate that the LBO eigenfunctions possess a more robust representation capability compared to POD modes. In Figure 7A, we can observe that, in the composite case, increasing the number of POD modes does not appear to reduce the prediction error of POD-DeepONet and POD-NORM significantly. In contrast, NORM continues to show a clear decreasing trend in error while maintaining its leading performance.

To further explain the performance difference between the two modes in the composite case, we conducted a comparative analysis of the spectral decomposition of both the input temperature field and the output deformation field using LBO and POD modes. As shown in Figure 7B, the top 100 POD decomposition

coefficients of the deformation field decreases rapidly from magnitudes of $10^2$ to $10^{-3}$, and the decomposition coefficients of the temperature field drop from $10^1$ to $10^{-4}$ suddenly. That indicates that the feature representation after the encoder $\mathcal{E}$ contains coefficients spanning a wide range, from $10^{-4}$ to $10^2$, which could bring challenges for the learning process. Besides, since the high-order POD coefficients of the deformation field are extremely small, any errors in these coefficients could lead to significant sensitivity in the reconstructed results generated by the decoder $\mathcal{D}$. By comparison, the LBO decomposition coefficients fluctuate in a relatively smaller range. This observation provides a potential explanation for why NORM consistently outperforms POD-NORM in most scenarios.

Another key distinction between these two modes lies in their underlying principles. POD modes are data-dependent while LBO eigenfunctions are geometry-dependent. For POD-DeepONet and POD-NORM, the POD modes are learnt from training data, thus the generalisability relies on the size of training data and low-data scenarios will lead to poor performance. By comparison, LBO eigenfunctions are only related to the geometric domain, but entirely independent of the training data. Therefore, the size of training samples will not influence the generalisability of LBO eigenfunctions. Figure 7C shows the error comparison for different operator learning methods with respect to different training dataset size. For the Darcy problem, the training dataset sizes vary from 400 to 2000, and the test dataset is an additional 200 groups labelled data. For the composite case, the training dataset sizes are set from 100 to 1000, and another 100 groups are defined as the test dataset. The number of modes is consistently set to 128 for POD-DeepONet, POD-NORM, and NORM. Notably, we observe that NORM exhibits a more rapid convergence rate as the training dataset increases, outperforming the other methods. In particular, for the Darcy problem, a NORM with 1200 samples can achieve a relative $L_2$ error of less than 1.00%, while DeepONet, POD-DeepONet, and POD-NORM with 2000 samples are 1.04%, 1.24% and 1.16% respectively. To sum up, integrating LBO eigenfunctions enables NORM with superior performance bound and enhances the convergence capability.

## DISCUSSION

In this work, we propose a deep learning framework with a new concept called the NORM, to learn mappings between functions defined on complex geometries, which are common challenges in science discovery and engineering applications. Unlike existing neural operator methods (such as FNO, UNO, WNO) that rely on regular geometric domains of Euclidean structure, NORM is able to learn mappings between input and output functions defined on any Riemannian manifolds via LBO subspace approximation. Furthermore, the optimality of LBO eigenfunctions allows NORM to capture the global feature of complex geometries with only a limited number of modes, rather than directly learning the operator in the high-dimensional coordinate space. The ability of LBO eigenfunctions to approximate functions on Riemannian manifolds also guarantees the universal approximation property of NORM.

NORM generalises the neural operator from Euclidean spaces to Riemannian manifolds, which has a wide range of potential applications, including PDEs solving, aerodynamics optimisation and other complex modelling scenarios. The case studies in parametric PDEs solving problems and engineering applications demonstrated that NORM can learn operators accurately and outperform the baseline methods. The discretisation-independent ability enables NORM with greater performance advantages compared with the

coordinate spaces based models (such as DeepONet [22]) when learning more complex operators (such as the blood flow dynamics case). The architecture of NORM draws inspiration from the iterative kernel integration structure employed in FNO [23]. Notably, since the Fourier basis is also a group of the LBO eigenfunction, NORM can be treated as a generalisation of FNO from the Euclidean space to Riemannian manifolds. In addition, NORM can deal with different input/output manifolds, including Euclidean space or complex geometries, and thus has broader application potential compared with GNN or FNO, which requires the input and output to be the same domains.

The integration of LBO eigenfunctions in NORM requires the definition of Riemannian metric, thus we introduce the Riemannian manifolds assumption for the geometric domains. In real-world engineering applications, the geometries either show Riemannian manifold properties or can be approximated to be Riemannian manifolds [52]. Besides, various Riemannian metrics have been defined for the commonly used mesh-represented geometries, thereby broadening the applicable scenarios of the Riemannian manifolds assumption [53]. Nevertheless, NORM can be applied to non-Riemannian data structures indirectly through proper pre-processing. For example, in the case of 3D point clouds, one feasible solution could be manually constructing the Riemannian metric from the point cloud and then calculating LBO eigenfunctions like described in reference [54]. Recent researchers have also started to develop Laplacian for non-manifold triangle meshes, which could be a potential solution for operator learning in non-manifold geometries [55].

Our method offers a new perspective for learning operators and solving PDEs on manifolds. Furthermore, the Laplacian-based approximation block in our method has strong extension potential to other neural operator structures or even physics-informed machine learning methods. For instance, the approximation block $\mathcal{N}$ could replace the branch net of DeepONet, making the new framework discretisation independent in both input and output functions. When solving PDEs with known equations, integrating the approximation block $\mathcal{N}$ into the physics-informed neural network could reduce the parameterisation complexity in coordinate spaces. In addition, the advantages of LBO eigenfunctions could be further discovered for more operator learning settings.

## Data availability

The source code and the datasets of all five case studies are available at https://github.com/gengxiangc/NORM.

## Funding

This work was supported by the National Science Fund for Distinguished Young Scholars (51925505), the General Program of National Natural Science Foundation of China (52275491), the Major Program of the National Natural Science Foundation of China (52090052), the Joint Funds of the National Natural Science Foundation of China (U21B2081), the National Key R&D Program of China (2022YFB3402600), and the New Cornerstone Science Foundation through the XPLORER PRIZE.

## Author contributions

G.C., X.L. and Y.L. conceptualised the problem. G.C., X.L. and Y.L. formulated the main ideas and the framework. G.C., Q.M., L.C. and C.L. developed the algorithm and performed the experiments and data processing. X.L., Y.L. and C.L. contributed to the theory and analysis. G.C., X.L. and Y.L. wrote the manuscript. Y.L. supervised the project.

## Conflict of interest

The authors declare no conflict of interest.

## Supplementary information

The supporting information is available online at https://doi.org/10.1360/nso/20240001. The supporting materials are published as submitted, without typesetting or editing. The responsibility for scientific accuracy and content remains entirely with the authors.

## References


1 Wang H, Fu T, Du Y, *et al*. Scientific discovery in the age of artificial intelligence. *Nature* 2023; **620**: 47–60.

2 Zhang R, Meng Q, Ma ZM. Deciphering and integrating invariants for neural operator learning with various physical mechanisms. *Natl Sci Rev* 2024; **11**: nwad336.

3 Li Z, Kovachki N, Azizzadenesheli K, *et al.* Neural operator: Graph kernel network for partial differential equations. arXiv: 2003.03485.

4 Li Z, Huang DZ, Liu B, *et al.* Fourier neural operator with learned deformations for pdes on general geometries. arXiv: 2207.05209.

5 Wang S, Wang H, Perdikaris P. Learning the solution operator of parametric partial differential equations with physics-informed deeponets. *Sci Adv* 2021; **7**: eabi8605.

6 Chen RT, Rubanova Y, Bettencourt J, *et al.* Neural ordinary differential equations. In: *Proceedings of the 32nd International Conference on Neural Information Processing Systems.* Red Hook: Curran Associates Inc., 2018.

7 Brunton SL, Kutz JN. Machine learning for partial differential equations. arXiv: 2303.17078.

8 Gopakumar V, Pamela S, Zanisi L, *et al.* Fourier neural operator for plasma modelling. arXiv: 2302.06542.

9 Corti M, Zingaro A, Dede'L, *et al*. Impact of atrial fibrillation on left atrium haemodynamics: A computational fluid dynamics study. *Comput Biol Med* 2022; **150**: 106143.

10 Sabater C, Stürmer P, Bekemeyer P. Fast predictions of aircraft aerodynamics using deep-learning techniques. *AIAA J* 2022; **60**: 5249–5261.

11 Taverniers S, Korneev S, Pietrzyk KM, *et al.* Accelerating part-scale simulation in liquid metal jet additive manufacturing via operator learning. arXiv: 2202.03665.

12 Azzizadenesheli K, Kovachki N, Li Z, *et al.* Neural operators for accelerating scientific simulations and design. arXiv: 2309.15325.

13 Ramezankhani M, Crawford B, Narayan A, *et al*. Making costly manufacturing smart with transfer learning under limited data: A case study on composites autoclave processing. *J Manuf Syst* 2021; **59**: 345–354.

14 Yuan Y, Liu J, Jin D, *et al*. DeceFL: A principled fully decentralized federated learning framework. *Natl Sci Open* 2023; **2**: 20220043.

15 Rao C, Ren P, Wang Q, *et al*. Encoding physics to learn reaction-diffusion processes. *Nat Mach Intell* 2023; **5**: 765–779.

16 Chen J, Viquerat J, Hachem E. U-net architectures for fast prediction of incompressible laminar flows. arXiv: 1910.13532.

17 Wu H, Hu T, Luo H, *et al.* Solving high-dimensional PDEs with latent spectral models. arXiv: 2301.12664.

18 Velickovic P, Cucurull G, Casanova A, *et al.* Graph attention networks. arXiv: 1710.10903.

19 Chen J, Hachem E, Viquerat J. Graph neural networks for laminar flow prediction around random two-dimensional shapes. *Phys Fluids* 2021; **33**: 123607.

20 Lu L, Meng X, Cai S, *et al*. A comprehensive and fair comparison of two neural operators (with practical extensions) based on FAIR data. *Comput Methods Appl Mech Eng* 2022; **393**: 114778.

21 You H, Yu Y, D'Elia M, *et al*. Nonlocal kernel network (NKN): A stable and resolution-independent deep neural network. *J Comput Phys* 2022; **469**: 111536.

22 Lu L, Jin P, Pang G, *et al*. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nat Mach Intell* 2021; **3**: 218–229.

23 Li Z, Kovachki N, Azizzadenesheli K, *et al.* Fourier neural operator for parametric partial differential equations. arXiv:

2010.08895.

24 Kovachki N, Li Z, Liu B, *et al.* Neural operator: Learning maps between function spaces. arXiv: 2108.08481.

25 Lehmann F, Gatti F, Bertin M, *et al*. 3D elastic wave propagation with a Factorized Fourier Neural Operator (F-FNO). *Comput Methods Appl Mech Eng* 2024; **420**: 116718.

26 Rahman MA, Ross ZE, Azizzadenesheli K. U-no: U-shaped neural operators. arXiv: 2204.11127.

27 Tripura T, Chakraborty S. Wavelet neural operator for solving parametric partial differential equations in computational mechanics problems. *Comput Methods Appl Mech Eng* 2023; **404**: 115783.

28 Li Z, Huang DZ, Liu B, *et al.* Fourier neural operator with learned deformations for pdes on general geometries. arXiv: 2207.05209.

29 Gao H, Sun L, Wang JX. PhyGeoNet: Physics-informed geometry-adaptive convolutional neural networks for solving parameterized steady-state PDEs on irregular domain. *J Comput Phys* 2021; **428**: 110079.

30 Seiler J, Jonscher M, Schoberl M, *et al*. Resampling images to a regular grid from a non-regular subset of pixel positions using frequency selective reconstruction. *IEEE Trans Image Process* 2015; **24**: 4540–4555.

31 Aflalo Y, Brezis H, Kimmel R. On the optimality of shape and data representation in the spectral domain. *SIAM J Imag Sci* 2015; **8**: 1141–1160.

32 Vaswani A, Shazeer N, Parmar N, *et al.* Attention is all you need. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems.* Red Hook: Curran Associates Inc., 2017.

33 Tschannen M, Bachem O, Lucic M. Recent advances in autoencoder-based representation learning. arXiv: 1812.05069.

34 Bhattacharya K, Hosseini B, Kovachki NB, *et al*. Model reduction and neural networks for parametric PDEs. *SMAI J Comput Math* 2021; **7**: 121–157.

35 Seidman JH, Kissas G, Perdikaris P, *et al.* Nomad: Nonlinear manifold decoders for operator learning. arXiv: 2206.03551.

36 Rai N, Mondal S. Spectral methods to solve nonlinear problems: A review. *Partial Differ Equ Appl Math* 2021; **4**: 100043.

37 Patanè G. Laplacian spectral basis functions. *Comput Aided Geometric Des* 2018; **65**: 31–47.

38 Aflalo Y, Kimmel R. Spectral multidimensional scaling. *Proc Natl Acad Sci USA* 2013; **110**: 18052–18057.

39 Terence Tao. Fourier transform. 2016. https://www.math.ucla.edu/~tao/preprints/fourier.pdf

40 Reuter M, Wolter FE, Peinecke N. Laplace-Beltrami spectra as "Shape-DNA" of surfaces and solids. *Comput-Aided Des* 2006; **38**: 342–366.

41 Alexa M, Herholz P, Kohlbrenner M, *et al*. Properties of laplace operators for tetrahedral meshes. *Comput Graphics Forum* 2020; **39**: 55–68.

42 Lanthaler S, Mishra S, Karniadakis GE. Error estimates for DeepONets: A deep learning framework in infinite dimensions. *Trans Math Its Appl* 2022; **6**: tnac001.

43 Hamilton W, Ying Z, Leskovec J. Inductive representation learning on large graphs. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems.* Red Hook: Curran Associates Inc., 2017.

44 Yuan Y, Li X, Li L, *et al*. Machine discovery of partial differential equations from spatiotemporal data: A sparse Bayesian learning framework. *Chaos-An Interdiscip J Nonlinear Sci* 2023; **33**: 113122.

45 Rouse H. Modern conceptions of the mechanics of fluid turbulence. *Trans Am Soc Civ Eng* 1937; **102**: 463–505.

46 Li Y, Li W, Han T, *et al*. Transforming heat transfer with thermal metamaterials and devices. *Nat Rev Mater* 2021; **6**: 488–507.

47 Shen Y, Lu Y, Liu S, *et al*. Self-resistance electric heating of shaped CFRP laminates: Temperature distribution optimization and validation. *Int J Adv Manuf Technol* 2022; **121**: 1755–1768.

48 Struzziero G, Teuwen JJE, Skordos AA. Numerical optimisation of thermoset composites manufacturing processes: A review. *Compos Part A-Appl Sci Manuf* 2019; **124**: 105499.

49 Secomb TW. Hemodynamics. *Compr Physiol* 2016; **6**: 975–1003.

50 Maul N, Zinn K, Wagner F, *et al.* Transient hemodynamics prediction using an efficient octree-based deep learning model. In: *Information Processing in Medical Imaging.* Cham: Springer, 2023; **13939**: 183–194.

51 Wen CY, Yang AS, Tseng LY, *et al*. Investigation of pulsatile flowfield in healthy thoracic aorta models. *Ann Biomed Eng* 2010; **38**: 391–402.

52 Masci J, Boscaini D, Bronstein M, *et al.* Geodesic convolutional neural networks on riemannian manifolds. In: *Proceedings of the 2015 IEEE International Conference on Computer Vision Workshop (ICCVW).* Santiago: IEEE, 2015.

53 Herzog R, Loayza-Romero E. A manifold of planar triangular meshes with complete riemannian metric. arXiv: 2012.05624.

54 Yan Q, Jiang SW, Harlim J. Spectral methods for solving elliptic PDEs on unknown manifolds. *J Comput Phys* 2023; **486**: 112132.

55 Sharp N, Crane K. A laplacian for nonmanifold triangle meshes. *Comput Graphics Forum* 2020; **39**: 69–80.