
SUPRA: Subspace Parameterized Attention for Neural Operator on General Domains

Zherui Yang¹ Zhengyang Xue¹ Ligang Liu¹

Abstract

Neural operators are efficient surrogate models for solving partial differential equations (PDEs), but their key components face challenges: (1) in order to improve accuracy, attention mechanisms suffer from computational inefficiency on large-scale meshes, and (2) spectral convolutions rely on the Fast Fourier Transform (FFT) on regular grids and assume a flat geometry, which causes accuracy degradation on irregular domains. To tackle these problems, we regard the matrix-vector operations in the standard attention mechanism on vectors in Euclidean space as bilinear forms and linear operators in vector spaces and generalize the attention mechanism to function spaces. This new attention mechanism is fully equivalent to the standard attention but impossible to compute due to the infinite dimensionality of function spaces. To address this, inspired by model reduction techniques, we propose Subspace Parameterized Attention (SUPRA) neural operator, which approximates the attention mechanism within a finite-dimensional subspace. To construct a subspace on irregular domains for SUPRA, we propose using the Laplacian eigenfunctions, which naturally adapt to domains' geometries and guarantee the optimal approximation for smooth functions. Experiments show that the SUPRA neural operator reduces error rates by up to 33% on various PDE datasets while maintaining state-of-the-art computational efficiency.

1. Introduction

Partial Differential Equations (PDEs) are critical in modeling physical and engineering systems, such as weather forecasting (Bonev et al., 2023), fluid dynamics (Horie & Mitsume, 2024), and structural analysis (Li et al., 2022b).

¹School of Mathematical Sciences, University of Science and Technology of China, Hefei, Anhui, China. Correspondence to: Ligang Liu <lgliu@ustc.edu.edu>.

Preliminary Work.

Solving PDEs has historically relied on numerical methods, such as finite element methods (Brenner & Scott, 2008), while they often become computationally expensive, especially for large-scale meshes and irregular domains (Smith et al., 1996). Recently, deep learning methods have shown great potential in accelerating PDE solving (Karniadakis et al., 2021). By learning mappings from inputs to solutions, neural operators (Kovachki et al., 2023), such as Fourier Neural Operators (FNOs) (Li et al., 2020), and DeepONet (Lu et al., 2019), offer significant speed-ups compared to traditional methods, while also being flexible and scalable.

Despite these advances, neural operators still face deficiencies in addressing the challenges of complex physical fields (e.g. Navier-Stokes equations with high Reynolds number) and irregular computational domains in multi-scale problems (e.g. NACA Airfoil). There are two main problems with existing approaches: (1) **The struggle between computational complexity and expressive power of attention.** As one of the most fundamental components of deep learning, the attention mechanism (Vaswani et al., 2017) has also been introduced in PDE-solving tasks. Fourier attention (Cao, 2021) treats sample points as tokens, resulting in a time complexity that is quadratic in the number of sample points, which is computationally expensive for large-scale meshes. Galerkin attention reduces the complexity to linear levels, forcing dot products to be applied only in the space spanned by the input functions (Wang & Wang, 2024) and limiting the expressive power of the attention mechanism. Although additional modules, such as FNOs (Rahman et al., 2024), can be used to improve accuracy, they also introduce additional overhead and constraints. (2) **Discontinuities in functions induced by cuts to irregular domains.** (see Figure 1). In order that FFT can be applied to conduct spectral convolution (Li et al., 2020), irregular physical domains are cut and mapped to a regular computational grid (Li et al., 2022b). However, the cut introduces discontinuities to smooth functions defined on the computational grid after they are mapped back to the original physical domain. The subsequent works extend the FFT to spheres (Bonev et al., 2023) or point clouds (Wang et al., 2024), but there still lacks a discussion on constructing suitable basis functions for more general domains and complex geometries.

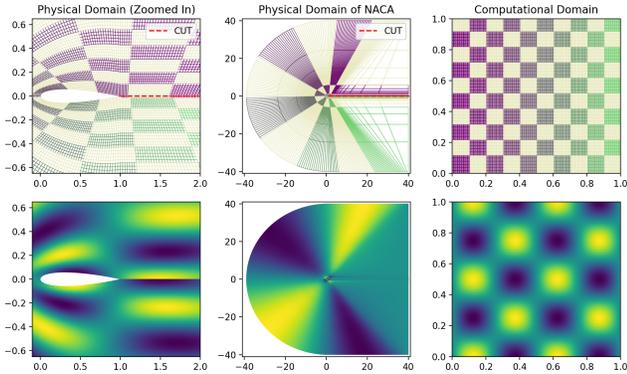


Figure 1. Discontinuities induced by domain cuts. The top row shows the mapping between a physical domain and a computation domain in the NACA Airfoil problem. To define the continuous map, the physical domain must be cut along the homology loop (red dashed line). Figures in the bottom row visualize a continuous function $f(x, y) = \sin(4\pi x) \cos(4\pi y)$ in different domains, where x, y are coordinates in the computational domains. After mapping back to the physical domain, the continuity of the function does not hold because of the cut.

To address the challenges above, we consider functions as fundamental elements in neural operators and PDE solving tasks, analogues to tokens in NLP tasks. Based on this idea, we derive the attention mechanism on function spaces by regarding the matrix-vector operations as bilinear forms and linear operators on vector spaces. This attention mechanism is fully equivalent to the standard defined on \mathbb{R}^N , aiming to capture complex relations between input functions. To tackle the inefficiency of computation in infinite-dimensional function spaces, we draw inspiration from model reduction techniques and propose Subspace Parameterized Attention (SUPRA), which parameterizes the attention mechanism within a finite-dimensional subspace. SUPRA transforms function-wise attention into standard attention operations on subspace coordinates, balancing its expressive power and computational efficiency.

To construct a suitable subspace for SUPRA on irregular domains, we leverage the subspace using the eigenfunctions of the Laplace operator. These eigenfunctions are obtained by solving the Laplacian eigenvalue problem defined by the geometry. They naturally encode the topology of the physical domain, ensure continuity across irregular meshes, and have excellent approximation properties for smooth functions. Leveraging their continuity and approximation properties, the SUPRA neural operator accurately captures the relations between functions while maintaining low computational costs on irregular domains.

In summary, our contributions are as follows:

1. We formulate the attention mechanism for infinite-dimensional function spaces and parameterize the at-

tention between functions within finite-dimensional subspace. To the best of our knowledge, our method is the first to directly extend the attention to function spaces while achieving a balance between attention’s expressive power and computational efficiency.

2. We leverage the Laplacian eigenfunctions to construct the subspace for irregular domains, which guarantees both the continuity of basis functions and optimal approximation property for smooth functions.
3. Through experiments and ablation studies on various PDE datasets, we demonstrate that our method achieves superior accuracy and computational efficiency compared to existing approaches.

2. Related Works

2.1. PDE Solvers

Solving PDEs is critical in many scientific research and industrial applications. Classical solvers, such as finite difference methods (Hesthaven, 2018), finite element methods (Brenner & Scott, 2008), and spectral methods (Bernardi & Maday, 1997), typically discretize the physical domain and solve the resulting linear systems. While well-established, these approaches are computationally expensive on high-dimensional problems or fine grids, driving interest in alternative solutions (Smith et al., 1996).

Physics-Informed Neural Networks (PINNs) Through defining a loss using PDEs’ residuals, PINNs use neural networks to represent the solution function directly, taking spatiotemporal coordinates as inputs and force the neural network to predict solution values under given initial and boundary conditions directly (Raissi et al., 2017; 2019). However, training a PINN typically addresses only one set of initial/boundary conditions and external forces and is not cheap even on modern hardware (Rathore et al., 2024).

Neural PDE Solvers Neural operators learn to approximate the mapping from inputs to solutions (Lu et al., 2019). Serving as surrogate models for numerical solvers. This approach significantly reduces the computational cost of solving PDEs approximately. As one of the pioneering works, FNO (Li et al., 2020) and its variants (Kossaifi et al., 2023; Wen et al., 2022) have achieved a significant leap in solving PDEs. FNOs use the Fast Fourier Transform (FFT) to perform efficient spectral convolution on regular grids. Although subsequent works, such as SFNO (Bonev et al., 2023), have extended the transformations to spherical domains using spherical harmonic transforms, studies on general regions remain limited. Geo-FNO (Li et al., 2022b) utilize mappings to transform irregular domains into regular rectangular regions, while the existence of continuous mappings is not guaranteed. GINO (Li et al., 2023b) propagates

information between irregular and regular domains through a GNO, but it does not fundamentally solve the problem of spectral convolution on irregular grids. Although Fourier transform can be extended to point clouds (Lingsch et al., 2024), the Fourier basis is also unsuitable for irregular domains and multi-scale problems.

2.2. Attention Mechanism in Neural PDE Solvers

As a breakthrough innovation in deep learning, the attention mechanism and Transformer model (Vaswani et al., 2017) have also been extensively applied to solving PDEs. In Fourier attention (Cao, 2021), point features are regarded as tokens, resulting in an attention weight matrix that scales quadratically with the number of points. Many works, such as Fact-Former (Li et al., 2023a), aim to reduce this computational complexity but are limited to regular domains. In contrast, others (Li et al., 2022a; Xiao et al., 2023; Hao et al., 2023) leverage linear transformers to address quadratic complexity by reordering computations but restricting attention to the subspace spanned by input functions. CoDA-NO (Rahman et al., 2024) use FNOs instead of linear combinations to generate attention inputs. Transolver (Wu et al., 2024) extracts global features by dividing the input physical field into slices to enable global information exchange. LNO (Wang & Wang, 2024) aggregate point data into global features to apply attention while failing to achieve high accuracy on irregular domains.

2.3. Model Reduction

Model reduction reduces computational complexity by projecting high-dimensional state spaces into low-dimensional subspaces, preserving key properties of the original system (Benner et al., 2017). Model reduction has many real-world applications. Proper Orthogonal Decomposition methods (POD) (Sirovich, 1987) and Principle Component Analysis (PCA) (Brunton et al., 2019) are standard model reduction techniques in fluid dynamics to transform from physical coordinates into a modal basis. In structural mechanics problems, model reduction also helps to accelerate the simulation (Sifakis & Barbic, 2012) or optimization problems (Choi et al., 2020; Benner et al., 2015).

3. Preliminaries

3.1. Problem Setup

As an instance, we consider a PDE with a boundary condition defined on domain $\Omega \subseteq \mathbb{R}^d$,

$$\begin{aligned} \mathcal{L}_a u &= 0, & x \in \Omega, \\ u &= 0, & x \in \partial\Omega, \end{aligned} \quad (1)$$

where \mathcal{L}_a is an operator that composed of the given function $a(x)$ and partial differentials of the unknown function $u(x)$.

The role of neural operators is to define mappings between functions. In the aforementioned problem, neural operators learn the mapping from the parameter to the solution, i.e. $a \mapsto u$. To enable computation on functions, both the input and output functions are represented by their values at M sample points x_i ($1 \leq i \leq M$).

Motivation of Generalizing Attention to Functions In the context of neural operators, functions are naturally treated as the fundamental "primitives". Generalizing the attention mechanism to function spaces losslessly is imperative, as it allows us to model complex relations between functions. In NLP tasks, attention is applied to tokens, represented as vectors in finite-dimensional space \mathbb{R}^N . To generalize the attention mechanism from vectors to functions, we first introduce two key concepts in this paper.

3.2. Linear Operators and Bilinear forms

Definition 3.1 (Linear Operator $b(\cdot)$). Given a vector space \mathbf{V} , a linear operator is a mapping $b(\cdot) : \mathbf{V} \rightarrow \mathbf{V}$ if it satisfies both additivity and homogeneity (Stein & Shakarchi, 2009)

$$\forall u, v \in \mathbf{V}, \alpha \in \mathbb{R}, \quad b(\alpha u + v) = \alpha b(u) + b(v). \quad (2)$$

Definition 3.2 (Bilinear Form $a(\cdot, \cdot)$). A bilinear form is a mapping $a(\cdot, \cdot) : \mathbf{V} \times \mathbf{V} \rightarrow \mathbb{R}$ if for all $u, v, w \in \mathbf{V}$ and $\alpha \in \mathbb{R}$

$$A(\alpha u + v, w) = \alpha A(u, w) + A(v, w), \quad (3)$$

and similarly for its second argument.

In the rest of this paper, we will focus linear operators and bilinear forms on $\mathbf{V} = \mathbb{R}^N$ or $\mathbf{V} = L^2(\Omega)$ since they are used to define the weights and outputs of attention mechanisms.

3.3. Standard Attention

Self-attention weights between two tokens $x_i, x_j \in \mathbb{R}^N$, $1 \leq i, j \leq C$ are defined with their query vectors $q_i, q_j \in \mathbb{R}^N$ and key vectors $k_i, k_j \in \mathbb{R}^N$ as follows (Vaswani et al., 2017):

$$w_{ij} = q_i^\top k_j, \quad 1 \leq i, j \leq C, \quad (4)$$

where q_i, k_i, v_i are linear transform of x_i :

$$q_i = W_Q x_i, \quad k_i = W_K x_i, \quad v_i = W_V x_i. \quad (5)$$

In Equation (5) $W_Q, W_K, W_V \in \mathbb{R}^{N \times N}$ are learnable matrices. The attention weights before softmax operation can be formulated as

$$w_{ij} = \frac{1}{\sqrt{N}} x_i^\top W_Q^\top W_K x_j. \quad (6)$$

Finally, the outputs of attention z_i are defined as

$$z_i = \sum_{j=1}^C \frac{\exp(w_{ij})}{\sum_{k=1}^C \exp(w_{ik})} v_j. \quad (7)$$

4. Our Method

4.1. Attention Mechanisms on Function Spaces

As shown in Equation (6), attention weights are essentially computed from a bilinear form $w_{ij} = a(x_i, x_j) = x_i^T W_Q^T W_K x_j$ in the vector space $\mathbf{V} = \mathbb{R}^N$, while in Equation (5), v_i are computed from a linear operator $v_i = b(x_i) = W_V x_i$. In this case, the attention operation's outputs z_i with input $x_i \in \mathbb{R}^N$ are

$$z_i = \sum_{j=1}^N \frac{\exp a(x_i, x_j)}{\sum_{k=1}^N \exp a(x_i, x_k)} b(x_j). \quad (8)$$

To define attention on any vector space, only a bilinear form $a(\cdot, \cdot)$ and a linear operator $b(\cdot)$ are needed. In the rest of the paper, we will focus mainly on the function space $\mathbf{V} = L^2(\Omega)$.

Attention on Function Spaces Let $u_i, u_j \in L^2(\Omega)$, $1 \leq i, j \leq C$ be functions defined on domain Ω , suppose $\mathcal{B} = \{e_k\}$ is a basis of $L^2(\Omega)$, u_i can be represented by the basis by $u_i = \sum_k \hat{u}_i^k e_k$. The coefficients \hat{u}_i^k represent the coordinates of the function u under the basis. By substituting u_i, u_j into the bilinear form $a(u_i, u_j)$ and the linear operator $b(u_i)$, and using their linear properties, the bilinear form and linear operator are represented by the basis \mathcal{B} as

$$\begin{aligned} a(u_i, u_j) &= a\left(\sum_k \hat{u}_i^k e_k, \sum_l \hat{u}_j^l e_l\right) = \sum_{k,l} a(e_k, e_l) \hat{u}_i^k \hat{u}_j^l, \\ b(u_i) &= \sum_k \hat{b}^k(u_i) e_k = \sum_{k,l} \hat{b}^k(e_l) \hat{u}_i^l e_k. \end{aligned} \quad (9)$$

In Equation (9), $a(e_k, e_l)$, $\hat{b}^k(e_l) \in \mathbb{R}$ are learnable coefficients, that do not depend on the input functions u_i , but only on the basis. Equation (8) and Equation (9) are direct generalizations of the standard attention. However, since the function space $L^2(\Omega)$ is infinite-dimensional, infinitely many coefficients are impossible to compute.

4.2. Efficient Attention on Function Spaces

Subspace Parameterized Attention Inspired by model reduction techniques, the bilinear form can be efficiently approximated by truncating the basis at the N -th term if the basis \mathcal{B} has a good approximation property:

$$\begin{aligned} a(u_i, u_j) &\approx \sum_{k,l=1}^N a(e_k, e_l) \hat{u}_i^k \hat{u}_j^l = \hat{u}_i^\top A \hat{u}_j, \\ b(u_i) &\approx \sum_{k,l=1}^N \hat{b}^k(e_l) \hat{u}_i^l e_k, \text{ and } \hat{b}(u_i) \approx B \hat{u}_i. \end{aligned} \quad (10)$$

In Equation (10), matrix $A = [a(e_k, e_l)]_{N \times N}$ parameterizes the bilinear form, $B = [\hat{b}^k(e_l)]_{N \times N}$ parameterizes the lin-

ear operator within the subspace, and $\hat{u}_i \in \mathbb{R}^N$ are vectors composed of the coefficients \hat{u}_i^k . Plugging the parameterizations into the Equation (8), the final output functions z_i , $1 \leq i \leq C$ of attention are

$$z_i = \sum_{l=1}^N \hat{z}_i^l e_l, \text{ where } \hat{z}_i = \sum_{k=1}^C \frac{\exp(w_{ik})}{\sum_{k=1}^C \exp(w_{ik})} B \hat{u}_k, \quad (11)$$

At this point, the attention mechanism on the function space has been defined (in Equation (8) and (9)) and parameterized within a subspace (in Equation (10) and (11)). Therefore, we refer to this attention mechanism as **Subspace Parameterized Attention (SUPRA)**. As proved in Appendix A, as N increases, SUPRA can approximate the attention in function spaces with arbitrary accuracy.

Subspace Projection and Reconstruction Besides operations on coordinates, projection from function u to subspace coordinates \hat{u} and reconstruction from \hat{u} to u are also required in SUPRA. Reconstruction is a trivial linear combination of the basis functions, while projection involves numerical integration. For common cases, such as regular grids $[0, 1]^2$ with resolution $H \times W$, the projection onto an orthonormal basis is simply a weighted sum,

$$\hat{u}^k = \int_{[0,1]^2} u(x) e_k(x) dx \approx \frac{1}{HW} \sum_{i,j=1}^{H,W} u(x_{ij}) e_k(x_{ij}), \quad (12)$$

where $x_{ij} = (i/H, j/W)$. This operation can also be conducted efficiently on GPUs.

4.3. Subspace Construction for General Domains

The subspace spanned by basis functions serves as the arena for the attention mechanism, influencing how functions engage and exchange information. On regular domains, any orthonormal basis with promising approximation properties can work in SUPRA, such as Chebyshev (Trefethen, 2013) and Fourier basis (Wright et al., 2015). For irregular domains, we provide a Laplacian-based method to construct proper basis functions (?).

Laplacian Eigensubspace The Laplacian eigensubspace is spanned by the smallest eigenfunctions of the Laplace operator, which are the Fourier basis on regular grids and spherical harmonics on spheres. *Therefore, the Laplacian eigensubspace can be interpreted as a natural extension of the Fourier basis from regular grids to general domains.*

The Laplacian eigensubspace guarantees optimal approximation for smooth functions that adapt to the geometry of any given domain. These eigenfunctions are naturally orthonormal and continuous (see Figure 3), ensuring efficient dimensionality reduction while reflecting the domain's topology (see also Appendix B).

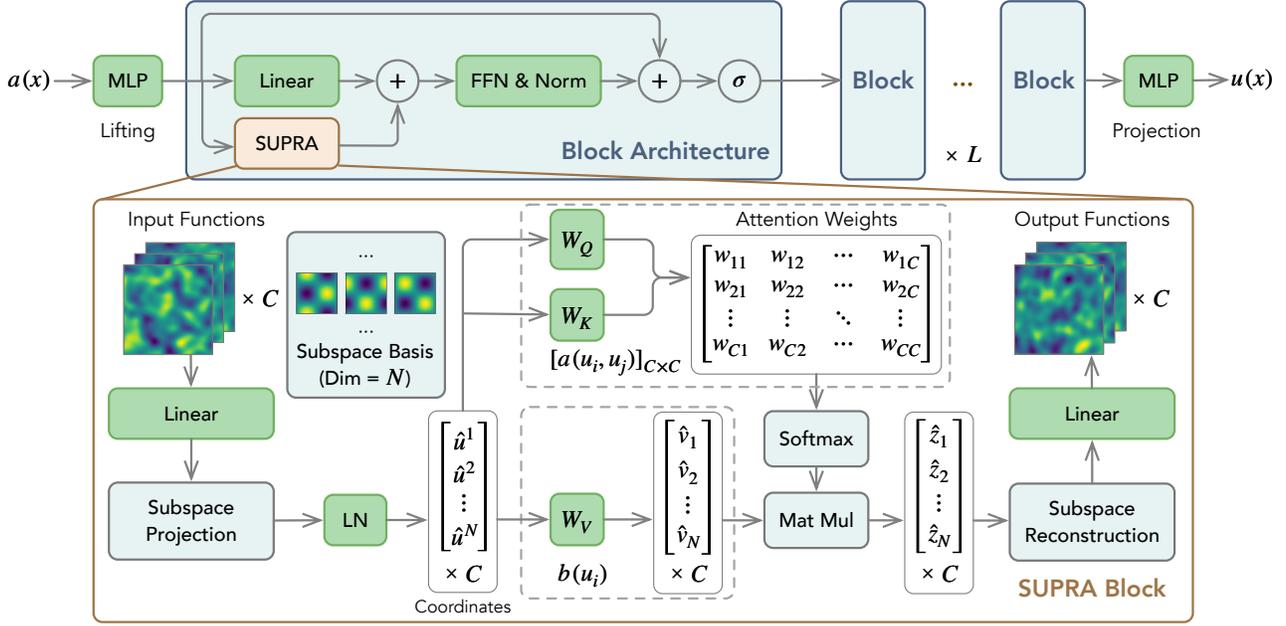


Figure 2. Overall design of SUPRA neural operator. We adopt the architecture proposed in (Kossaifi et al., 2023) while replacing spectral convolutions with SUPRA blocks. All trainable modules are colored green. LN stands for LayerNorm, W_V corresponds to the matrix B , and $W_Q^T W_K$ corresponds to the matrix A defined in Equation (10). Although LayerNorm (Ba et al., 2016) is a common choice, InstanceNorm (Ulyanov et al., 2016) can work better since each function is treated as a token in our framework.

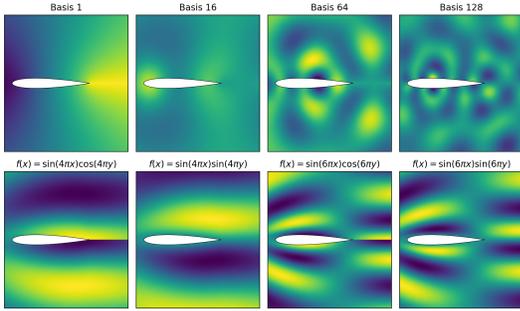


Figure 3. Comparison between Laplacian eigenfunctions and Fourier basis. The eigenfunctions guarantee continuity across the physical domain, while the Fourier basis defined on the computational mesh does not.

Basis Construction For irregular domains, the eigenvectors of the Laplace matrix resulting from FEM discretization are precomputed once by using classical methods. For regular grids, the basis is computed from a tensor product. See Section 5.2 and Appendix C for their implementation details. Typically, $N = 64$ to $N = 256$ is sufficient for SUPRA.

4.4. Further Discussions

Multiple heads in SUPRA To enhance SUPRA’s expressive power, we use multiple attention heads by decomposing $A = W_Q^T W_K$ and applying attention to subvectors

within each head. Focusing on the parameterized attention in Equation (10), it is clear that SUPRA can directly leverage highly optimized multi-head attention code, as shown in the SUPRA block in Figure 2.

Comparison to Standard Attention Comparing the matrix A to $W_Q^T W_K$ Equation (6) and the matrix B to W_V in Equation (5), it is clear that SUPRA transforms the attention between functions $u_i \in L^2(\Omega)$ into standard attention between the coordinate vectors $\hat{u}_i \in \mathbb{R}^N$ of these functions within the subspace.

Comparison to Galerkin Attention Given C input functions u_i , $1 \leq i \leq C$, attention weights in Galerkin attention are defined as the dot product between v_i, k_i , $w_{ij} = \langle v_i, k_j \rangle$, and the output is $z_i = \sum_{j=1}^N w_{ij} q_j$. Here, functions v_i, k_i, q_i are linear combinations of input functions u_i , forcing attention to be applied only within the subspace spanned by $\{u_i\}_{i=1}^C$. In contrast, SUPRA is based on bilinear forms and linear operators in a larger function space, allowing it to capture functions’ relations and apply transforms directly to each function.

Complexity Analysis SUPRA is efficient compared to all previous methods using the attention mechanism. For C functions sampled at M points in the domain, the complexity of different operations in SUPRA is: (1) projection to subspace coordinates: $O(CM)$, (2) attention between coordinates: $O(C^2N)$, and (3) reconstruction from

Test Case	Relative L^2 Error ($\times 10^{-2}$)					#Params ($\times 10^6$)		
	Galerkin	GNOT	Transolver	LNO	Ours	Transolver	LNO	Ours
Darcy	0.84	1.05	0.50	<u>0.49</u>	0.43	2.8	0.76	<u>1.7</u>
Navier Stokes	14.0	13.8	<u>7.83</u>	8.45	6.25	11.2	<u>5.0</u>	3.4
Plasticity	1.20	3.36	<u>0.08</u>	0.29	0.04	2.8	<u>1.4</u>	1.3
Airfoil	1.18	0.76	<u>0.43</u>	0.51	0.34	2.8	<u>1.4</u>	0.5
Pipe	0.98	0.47	<u>0.32</u>	0.26	<u>0.31</u>	3.0	<u>1.4</u>	1.1

Table 1. Performance Comparison. Relative L^2 error ($\times 10^{-2}$) is recorded, and a smaller value indicates better performance. The best recorded on each dataset is in bold, and the second best is underlined. The best configurations of our method are listed in Appendix C.2.

coordinates: $O(CM)$. The total complexity of SUPRA is $O(C^2N + CM)$. Correspondingly, Fourier attention considers points as tokens, requiring a complexity of $O(CM^2)$, while Galerkin attention requires a complexity of $O(C^2M)$. The complexity of SUPRA is similar to Galerkin attention, but achieves better expressive power.

5. Experiments

Test Cases We adopt five standard benchmark datasets provided by the community listed in Table 2. These test cases cover most of the input types in PDE problems, including changes to the parameters of the equation (Darcy), external inputs (Plasticity), time advance (Navier Stokes), and irregular domains (Pipe and Airfoil). See Appendix C for their background information and our experiment’s settings.

Test Case	Input	Output
Darcy	Porus Medium	Fluid Pressure
Navier Stokes	Previous Vorticity	Future Vorticity
Plasticity	External Force	Deformation
Airfoil	Structure	Mach Number
Pipe	Structure	Fluid Velocity

Table 2. List of our test cases and their inputs and outputs.

5.1. Accuracy Comparison

Baselines SUPRA neural operator is compared with the most recent SOTA methods: (1) Galerkin Transformer (Cao, 2021), (2) GNOT (Hao et al., 2023), (3) Transolver (Wu et al., 2024), and (4) LNO (Wang & Wang, 2024).

General Settings and Metrics We use the following relative L^2 error (i.e. relative root MSE) as our error metric:

$$\text{rel}_{L^2}(u, u^*) = \frac{\|u - u^*\|_2}{\|u^*\|_2}, \quad \|f\|_2 = \sqrt{\sum_i f(x_i)^2}, \quad (13)$$

where x_i are sample points, u, u^* corresponds to the model prediction and the ground truth. The final relative L^2 error is averaged over all test samples. The best performance of

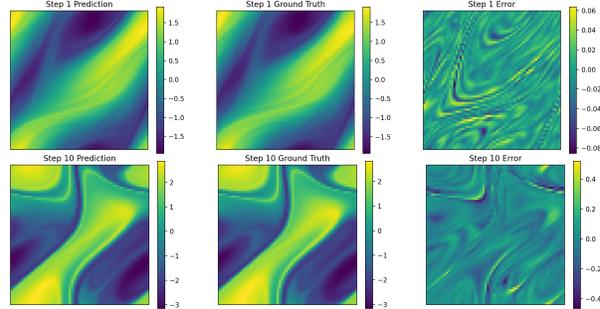


Figure 4. Comparisons between our prediction and ground truth at the first and last step in the Navier Stokes problem. Although the previous input is smooth, the output can be very sharp.

each method on different test cases is shown in Table 1.

As shown, the SUPRA neural operator achieves the best accuracy and lower computational cost in most test cases. Our model achieves performance close to the SOTA with lower computational cost in the Pipe case.

We visualize the prediction of the SUPRA neural operator for the Navier Stokes and Airfoil problems in Figure 4 and Figure 5. SUPRA neural operator accurately captures the vortex and shock wave, demonstrating its ability to capture complex dynamics even for sharp solutions.

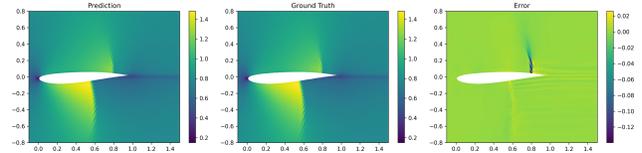


Figure 5. Comparison between our prediction and ground truth in Airfoil problem. Our method can capture the shock wave around the wing precisely.

5.2. Ablation Study

Number of Basis The performance of SUPRA with different number of basis functions is compared. As presented in Table 3, it is worth noticing that SUPRA will not degenerate

when the number of basis is small but also helps to filter out useless high-frequency modes.

Test Case	Basis	#Basis Functions		
		small	medium	large
NS	Fourier	6.86	6.32	6.50
Darcy	Fourier	0.433	0.436	0.466
Darcy	Chebyshev	0.432	0.449	0.454
Plasticity	Fourier	0.062	0.056	0.044
Airfoil	Laplacian	0.446	0.340	0.342

Table 3. Comparison between different numbers of basis functions: Relative L^2 error ($\times 10^{-2}$) is recorded. The numbers of basis functions range from 64 to 256: The small/medium/large corresponds to 64/100/144 for Darcy and Plasticity, 64/144/256 for Navier Stokes, and 64/128/192 for Airfoil.

Choice of Basis As an example to construct the basis, on 1D domain $[0, 1]$, the Fourier basis with n modes is:

$$\{\cos(2\pi ix), \sin(2\pi ix)\}_{i=1}^n. \quad (14)$$

We use a tensor product Fourier basis for the 2D domain $[0, 1]^2$. Assuming we have m, n modes in the x, y direction, the basis is constructed:

$$\{\cos(2\pi ix) \cos(2\pi jy), \cos(2\pi ix) \sin(2\pi jy), \sin(2\pi ix) \cos(2\pi jy), \sin(2\pi ix) \sin(2\pi jy)\}_{i,j=1}^{m,n}. \quad (15)$$

For a general d -dimensional structured mesh, the mesh is first mapped to the d -dimensional unit square $[0, 1]^d$, and the tensor product basis on the unit square is constructed accordingly. A d -dimensional basis with n modes per dimension results in $(2n)^d$ basis functions.

As mentioned in Section Section 4.3, the choice of basis is problem dependent. The performance of the Fourier basis and Chebyshev basis is compared on Darcy, Navier-Stokes, and Plasticity problems in Table 4.

Basis	Test Case		
	NS	Darcy	Plasticity
Fourier	6.32	0.466	0.044
Chebyshev	6.60	0.454	0.049

Table 4. Comparison between different orthonormal basis on regular grids. The table records the relative L^2 error ($\times 10^{-2}$). The number of basis is set to 144 in these experiments.

The performance of the subspace spanned by Laplacian eigenfunctions and the Fourier basis is compared on the Airfoil problem, with experiment results listed in the Table 5. We choose 128 Laplacian eigenfunctions and 6 modes for the Fourier basis (i.e. $(6 \times 2)^2 = 144$ basis functions in

total). Since the eigenfunctions are computed directly on the irregular physical domain, the Laplacian eigensubspace outperforms a Fourier basis defined on a regular computational grid.

Dataset	Laplacian	Fourier
Train	0.74	0.93
Test	3.40	4.58

Table 5. Comparison between Laplacian eigensubspace and Fourier basis. Relative L^2 Error ($\times 10^{-3}$) of Airfoil on train and test set is recorded with different subspace.

Normalization The performance of LayerNorm and InstanceNorm is compared in Table 6. It is clear that, the training process can fail without any normalization applied, while the performance of InstanceNorm is slightly better than LayerNorm.

Normalization	Test Case			
	NS	Airfoil	Darcy	Plasticity
Layer	6.92	0.340	0.433	0.046
Instance	6.50	0.350	0.463	0.044
None	↑	↑	0.479	↑

Table 6. Comparison between different normalizations. Relative L^2 error ($\times 10^{-2}$) is recorded in the table, and \uparrow indicates the optimization process blows up.

5.3. Model Scalability

Model Size The accuracy of the SUPRA neural operator is evaluated across different numbers of learnable parameters in Table 7. Increasing the number of hidden features increases the number of functions that participate in attention while increasing the number of hidden layers L enables the model to learn more complex features.

Model Size	Test Case		
	Navier Stokes	Airfoil	Darcy
$L = 4$	6.89	0.510	0.512
$L = 6$	6.25	0.340	0.471
$L = 8$	6.32	0.340	0.432
Small	11.1	0.484	0.488
Medium	7.99	0.340	0.432
Large	6.58	0.371	0.432

Table 7. Comparison between different number of hidden layers L and hidden features. Relative L^2 error ($\times 10^{-2}$) is recorded in the table. For hidden features, Small/Medium/Large correspond to 64/128/196 for Darcy and Navier Stokes and 32/64/96 for Airfoil.

Computational Cost The evaluation cost comparison of SUPRA and previous methods is listed in Table 8. A 2D

Hyper-parameters		Model	#Params	GPU Mem (MB)	Time (sec.)	
#Hiddens	Mesh Size				Forward	Backward
64	64×64	Transolver	712K	483	0.017	0.040
		LNO	334K	231	0.009	0.019
		Ours ($N = 128$)	635K	173	0.007	0.016
		Ours ($N = 256$)	2.2M	191	0.008	0.017
64	128×128	Transolver	712K	1822	0.063	0.148
		LNO	334K	741	0.029	0.058
		Ours ($N = 128$)	635K	635	0.013	0.034
		Ours ($N = 256$)	2.2M	658	0.016	0.042
256	64×64	Transolver	11M	1469	0.062	0.169
		LNO	5.0M	683	0.017	0.040
		Ours ($N = 128$)	2.1M	662	0.015	0.040
		Ours ($N = 256$)	3.7M	708	0.019	0.047
256	128×128	Transolver	11M	5647	0.236	0.677
		LNO	5.0M	2141	0.051	0.121
		Ours ($N = 128$)	2.1M	2430	0.053	0.133
		Ours ($N = 256$)	3.7M	2479	0.071	0.177

Table 8. Model efficiency. A single RTX-3060 is used for all experiments. All models and experiments use 8 hidden layers and a batch size of 4. N represents the number of subspace basis functions in the SUPRA block.

problem defined on a structured grid, with 1 function as input and 1 function as output, is considered in this experiment. By comparing Table 7 and 8, it can be concluded that SUPRA maintains high prediction accuracy even with a small number of parameters and computational costs.

6. Conclusion and Future Work

This paper introduces the SUPRA neural operator for solving PDEs on general domains. SUPRA defines attention between functions derived directly from the standard attention mechanism and parameterizes attention within a finite-dimensional subspace, achieving superior expressive power and computational efficiency. The SUPRA neural operator demonstrates stronger performance on irregular domains by leveraging the Laplacian eigensubspace defined directly on the physical domain. SUPRA achieves higher accuracy across various standard PDE benchmarks while maintaining low computational cost. In the future, manual subspace construction and choice can also be replaced by a machine learning model. Applying SUPRA to PDEs and 3D dynamics with a larger mesh is also challenging.

Impact Statement

This work aims to improve neural networks as surrogate models for PDE solvers by introducing the attention mechanism directly in function space. This approach offers superior efficiency and expressive power compared to current techniques. We are confident that our method can enhance performance in practical applications, including weather

prediction and topology optimization. In developing our approach, we have prioritized ethical considerations and ensured that our work carries no foreseeable ethical concerns.

References

- Ba, J., Kiros, J. R., and Hinton, G. E. Layer normalization. *ArXiv*, abs/1607.06450, 2016.
- Benner, P., Gugercin, S., and Willcox, K. A survey of projection-based model reduction methods for parametric dynamical systems. *SIAM Review*, 57(4):483–531, 2015. doi: 10.1137/130932715.
- Benner, P., Ohlberger, M., Cohen, A., and Willcox, K. *Model Reduction and Approximation*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2017. doi: 10.1137/1.9781611974829.
- Bernardi, C. and Maday, Y. Spectral methods. In *Techniques of Scientific Computing (Part 2)*, volume 5 of *Handbook of Numerical Analysis*, pp. 209–485. Elsevier, 1997.
- Bonev, B., Kurth, T., Hundt, C., Pathak, J., Baust, M., Kashinath, K., and Anandkumar, A. Spherical fourier neural operators: Learning stable dynamics on the sphere. In *International Conference on Machine Learning*, 2023.
- Brenner, S. and Scott, R. *The Mathematical Theory of Finite Element Method*, volume 15. 01 2008. ISBN 978-1-4757-4340-1. doi: 10.1007/978-1-4757-4338-8.

- Brunton, S. L., Noack, B. R., and Koumoutsakos, P. Machine learning for fluid mechanics. *ArXiv*, abs/1905.11075, 2019.
- Cao, S. Choose a transformer: Fourier or galerkin. In *Neural Information Processing Systems*, 2021.
- Choi, Y., Boncoraglio, G., Anderson, S., Amsallem, D., and Farhat, C. Gradient-based constrained optimization using a database of linear reduced-order models. *Journal of Computational Physics*, 423:109787, 2020. ISSN 0021-9991. doi: <https://doi.org/10.1016/j.jcp.2020.109787>.
- De Witt, T., Lessig, C., and Fiume, E. Fluid simulation using laplacian eigenfunctions. *ACM Trans. Graph.*, 31(1), February 2012. ISSN 0730-0301. doi: 10.1145/2077341.2077351.
- Evans, L. *Partial Differential Equations*. Graduate studies in mathematics. American Mathematical Society, 2010. ISBN 9780821849743.
- Gilbarg, D. and Trudinger, N. *Elliptic Partial Differential Equations of Second Order*. Grundlehren der mathematischen Wissenschaften. Springer Berlin Heidelberg, 2013. ISBN 9783642963797.
- Hao, Z., Wang, Z., Su, H., Ying, C., Dong, Y., Liu, S., Cheng, Z., Song, J., and Zhu, J. Gnot: A general neural operator transformer for operator learning. In *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pp. 12556–12569. PMLR, 2023.
- Hesthaven, J. S. *Numerical Methods for Conservation Laws*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2018. doi: 10.1137/1.9781611975109.
- Horie, M. and Mitsume, N. Graph neural PDE solvers with conservation and similarity-equivariance. In Salakhutdinov, R., Kolter, Z., Heller, K., Weller, A., Oliver, N., Scarlett, J., and Berkenkamp, F. (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 18785–18814. PMLR, 21–27 Jul 2024.
- Karniadakis, G. E., Kevrekidis, I. G., Lu, L., Perdikaris, P., Wang, S., and Yang, L. Physics-informed machine learning. *Nature Reviews Physics*, 3:422 – 440, 2021.
- Kossaifi, J., Kovachki, N. B., Azizzadenesheli, K., and Anandkumar, A. Multi-grid tensorized fourier neural operator for high-resolution pdes. *ArXiv*, abs/2310.00120, 2023.
- Kovachki, N., Li, Z., Liu, B., Azizzadenesheli, K., Bhattacharya, K., Stuart, A., and Anandkumar, A. Neural operator: Learning maps between function spaces with applications to pdes. *Journal of Machine Learning Research*, 24(89):1–97, 2023.
- Lévy, B. and Zhang, H. R. Spectral mesh processing. In *ACM SIGGRAPH 2010 Courses*, SIGGRAPH ’10, New York, NY, USA, 2010. Association for Computing Machinery. ISBN 9781450303958. doi: 10.1145/1837101.1837109.
- Li, Z., Meidani, K., and Farimani, A. B. Transformer for partial differential equations’ operator learning. *Trans. Mach. Learn. Res.*, 2023, 2022a.
- Li, Z., Shu, D., and Barati Farimani, A. Scalable transformer for pde surrogate modeling. In Oh, A., Naumann, T., Globerson, A., Saenko, K., Hardt, M., and Levine, S. (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 28010–28039. Curran Associates, Inc., 2023a.
- Li, Z.-Y., Kovachki, N. B., Azizzadenesheli, K., Liu, B., Bhattacharya, K., Stuart, A. M., and Anandkumar, A. Fourier neural operator for parametric partial differential equations. *ArXiv*, abs/2010.08895, 2020.
- Li, Z.-Y., Huang, D. Z., Liu, B., and Anandkumar, A. Fourier neural operator with learned deformations for pdes on general geometries. *J. Mach. Learn. Res.*, 24: 388:1–388:26, 2022b.
- Li, Z.-Y., Kovachki, N. B., Choy, C., Li, B., Kossaifi, J., Otta, S. P., Nabian, M. A., Stadler, M., Hundt, C., Azizzadenesheli, K., and Anandkumar, A. Geometry-informed neural operator for large-scale 3d pdes. *ArXiv*, abs/2309.00583, 2023b.
- Lingsch, L. E., Michelis, M. Y., De Bezenac, E., M. Perera, S., Katzschmann, R. K., and Mishra, S. Beyond regular grids: Fourier-based neural operators on arbitrary domains. In Salakhutdinov, R., Kolter, Z., Heller, K., Weller, A., Oliver, N., Scarlett, J., and Berkenkamp, F. (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 30610–30629. PMLR, 21–27 Jul 2024.
- Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2017.
- Lu, L., Jin, P., Pang, G., Zhang, Z., and Karniadakis, G. E. Learning nonlinear operators via deepnet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3:218 – 229, 2019.

- Rahman, M. A., George, R. J., Elleithy, M., Leibovici, D., Li, Z., Bonev, B., White, C., Berner, J., Yeh, R. A., Kossaifi, J., Azizzadenesheli, K., and Anandkumar, A. Pre-training codomain attention neural operators for solving multiphysics pdes. *Advances in Neural Information Processing Systems*, 37, 2024.
- Raissi, M., Perdikaris, P., and Karniadakis, G. E. Physics informed deep learning (part i): Data-driven solutions of nonlinear partial differential equations. *arXiv preprint arXiv:1711.10561*, 2017.
- Raissi, M., Perdikaris, P., and Karniadakis, G. E. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.
- Rathore, P., Lei, W., Frangella, Z., Lu, L., and Udell, M. Challenges in training PINNs: A loss landscape perspective. In Salakhutdinov, R., Kolter, Z., Heller, K., Weller, A., Oliver, N., Scarlett, J., and Berkenkamp, F. (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 42159–42191. PMLR, 21–27 Jul 2024.
- Sheng, C., Cao, D., and Shen, J. Efficient spectral methods for pdes with spectral fractional laplacian. *Journal of Scientific Computing*, 88, 07 2021. doi: 10.1007/s10915-021-01491-2.
- Sifakis, E. and Barbic, J. Fem simulation of 3d deformable solids: a practitioner’s guide to theory, discretization and model reduction. In *ACM SIGGRAPH 2012 Courses*, SIGGRAPH ’12, New York, NY, USA, 2012. Association for Computing Machinery. ISBN 9781450316781. doi: 10.1145/2343483.2343501.
- Sirovich, L. Turbulence and the dynamics of coherent structures. i. coherent structures. *Quarterly of Applied Mathematics*, 45:561–571, 1987.
- Smith, B. F., Bjørstad, P. E., and Gropp, W. D. *Domain decomposition: parallel multilevel methods for elliptic partial differential equations*. Cambridge University Press, USA, 1996. ISBN 052149589X.
- Smith, L. N. and Topin, N. Super-convergence: very fast training of neural networks using large learning rates. In *Defense + Commercial Sensing*, 2018.
- Stein, E. and Shakarchi, R. *Real Analysis: Measure Theory, Integration, and Hilbert Spaces*. Princeton University Press, 2009. ISBN 9781400835560.
- Trefethen, L. N. *Approximation theory and approximation practice*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2013. ISBN 978-1-611972-39-9.
- Ulyanov, D., Vedaldi, A., and Lempitsky, V. S. Instance normalization: The missing ingredient for fast stylization. *CoRR*, abs/1607.08022, 2016.
- Vaswani, A., Shazeer, N. M., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is all you need. In *Neural Information Processing Systems*, 2017.
- Wang, H., Jiabin, L., Dwivedi, A., Hara, K., and Wu, T. BENO: Boundary-embedded neural operators for elliptic PDEs. In *The Twelfth International Conference on Learning Representations*, 2024.
- Wang, T. and Wang, C. Latent neural operator for solving forward and inverse pde problems. In *Advances in Neural Information Processing Systems*, 2024.
- Wen, G., Li, Z., Azizzadenesheli, K., Anandkumar, A., and Benson, S. M. U-fno—an enhanced fourier neural operator-based deep-learning model for multiphase flow. *Advances in Water Resources*, 163:104180, 2022. ISSN 0309-1708. doi: <https://doi.org/10.1016/j.advwatres.2022.104180>.
- Wright, G. B., Javed, M., Montanelli, H., and Trefethen, L. N. Extension of Chebfun to periodic functions. *SIAM J. Sci. Comput.*, 37(5):C554–C573, 2015. ISSN 1064-8275,1095-7197. doi: 10.1137/141001007. URL <https://doi.org/10.1137/141001007>.
- Wu, H., Luo, H., Wang, H., Wang, J., and Long, M. Transolver: A fast transformer solver for PDEs on general geometries. In Salakhutdinov, R., Kolter, Z., Heller, K., Weller, A., Oliver, N., Scarlett, J., and Berkenkamp, F. (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 53681–53705. PMLR, 21–27 Jul 2024.
- Xiao, Z., Hao, Z., Lin, B., Deng, Z., and Su, H. Improved operator learning by orthogonal attention. *ArXiv*, abs/2310.12487, 2023.

A. Approximation property of SUPRA

In this section, we examine SUPRA will converge under the assumption of boundness of the linear operator $b(\cdot)$ and bilinear form $a(\cdot, \cdot)$. In a Banach space, we call a linear operator bounded if there is a constant C such that for all $u \in \mathbf{V}$,

$$\|b(u)\| \leq C \cdot \|u\|. \quad (16)$$

A bilinear form is bounded if there is a constant C such that for all $u, v \in \mathbf{V}$,

$$a(u, v) \leq C \cdot \|u\| \cdot \|v\|. \quad (17)$$

The boundness property of linear operators and bilinear forms are identical to continuity in Banach spaces: Suppose u_i and v_i converge to u and v in space \mathbf{V} , for linear operator $b(\cdot)$,

$$\lim_{i \rightarrow \infty} \|u_i - u\| = 0 \implies 0 \leq \lim_{i \rightarrow \infty} \|b(u_i) - b(u)\| = \lim_{i \rightarrow \infty} \|b(u_i - u)\| \leq C \lim_{i \rightarrow \infty} \|u_i - u\| = 0. \quad (18)$$

Therefore, $\lim_{i \rightarrow \infty} b(u_i) = b(u)$. Similarly, we have $\lim_{i, j \rightarrow \infty} a(u_i, v_j) = a(u, v)$ as well. Suppose we use N basis functions $\{e_i\}_{i=1}^N$ to approximate u, v , as N goes to infinity:

$$\lim_{N \rightarrow \infty} \sum_{k=1}^N \hat{u}^k e_k = u, \quad \lim_{N \rightarrow \infty} \sum_{k=1}^N \hat{v}^k e_k = v. \quad (19)$$

Substituting the summation into Equation (10), we have:

$$\begin{aligned} \lim_{N \rightarrow \infty} a \left(\sum_{k=1}^N \hat{u}^k e_k, \sum_{k=1}^N \hat{v}^k e_k \right) &= a(u, v), \\ \lim_{N \rightarrow \infty} b \left(\sum_{k=1}^N \hat{u}^k e_k \right) &= b(u). \end{aligned} \quad (20)$$

Therefore, as N increases, SUPRA's outputs will converge to the attention of the vector space.

Subspace Basis Here, we list some common basis. These basis share a property that, as the number of basis functions N increases, the approximations $u_N = \sum_{i=1}^N \hat{u}^i e_i$ will converge to u . These properties guarantee the existence of u_i, v_i in the proof above.

1. Fourier/Trigonometric representations have uniform properties across the interval of approximation, which is also the Laplacian eigenfunctions on regular grids such as $[0, 1]^d$,
2. Chebyshev polynomials are nonuniform, with greater resolution at the ends of $[-1, 1]$ than at the middle,
3. Spherical harmonics are the natural extension of the Fourier series to functions defined on the surface of a sphere, which are also the Laplacian eigenfunctions on that surface,

B. Eigensubspace of Laplace Operator

Benefits of the Eigensubspace Laplace operator (Laplace-Beltrami operator, or Laplacian) is defined as the divergence of gradient $\Delta f = \text{div}(\nabla f)$. The spectrum of the Laplacian consists of all eigenvalues λ and eigenfunctions f with:

$$-\Delta f = \lambda f. \quad (21)$$

This is also known as **Helmholtz Equation**. If the domain ω is bounded in \mathbb{R}^n , then eigenfunctions of the Laplacian are an orthonormal basis for Hilbert space $L^2(\Omega)$ (Gilbarg & Trudinger, 2013). **Every eigenfunction is infinitely differentiable, guaranteeing continuity across any irregular domain.** Besides, the theorem below describes the smallest eigenfunctions as the most smooth functions on Ω (Evans, 2010):

Theorem B.1. *The infimum is achieved if and only if φ is an eigenfunction of eigenvalue λ_k :*

$$\lambda_k = \inf \left\{ \frac{\|\nabla\varphi\|_{L^2}^2}{\|\varphi\|_{L^2}^2} : s.t. \varphi \in H_1(\Omega) \cap E_k \right\} \quad (22)$$

where $E_k := \{\varphi_1, \varphi_2, \dots, \varphi_{k-1}\}^\perp$.

Theory and application of the discrete version of Laplacian eigensubspace are also well-established in spectral mesh processing (Lévy & Zhang, 2010). This eigensubspace is used to perform effective and information-preserving model reduction while revealing global and intrinsic structures in geometric data, enabling efficient PDE solving (Sheng et al., 2021) and simulation (De Witt et al., 2012).

C. Experiments Setups

C.1. PDEs Descriptions and Data Preprocessing

Consider the boundary-value problem given by:

$$\begin{aligned} Lu &= f, & \text{in } \Omega, \\ u &= g, & \text{on } \partial\Omega. \end{aligned} \quad (23)$$

Our experiments cover most variables in the form: (1) coefficients in L , describing the property of the material at different positions, (2) external force f , (3) domain shape Ω , and (4) boundary shape $\partial\Omega$. Our experiments also consider time-dependent problems, which occur when previous states determine future states. We list our experiments in Table 9.

Test Case	Geometry	#Dim	Mesh Size	Dataset Split		Input Type
				Train	Test	
Darcy (Li et al., 2020)	Regular Grid	2	85×85	1000	200	Coefficients
Navier Stokes (Li et al., 2020)	Regular Grid	2+1	64×64	1000	200	Previous States
Plasticity (Li et al., 2022b)	Structured Mesh	2+1	101×31	900	80	External Force
Airfoil (Li et al., 2022b)	Structured Mesh*	2	221×51	1000	200	Boundary Shape
Pipe (Li et al., 2022b)	Structured Mesh	2	129×129	1000	200	Domain Shape

Table 9. Datasets and their split used in our experiments. Although the Airfoil case is structured, the mesh is cut to map from an annulus-like domain to a regular grid. To construct the Laplacian eigensubspace, we regard it as an irregular mesh.

Darcy The equation in Darcy describes the flow of fluid through a porous medium:

$$\begin{aligned} -\nabla \cdot (a(x)\nabla u(x)) &= f(x), & x \in (0, 1)^2, \\ u(x) &= 0, & x \in \partial(0, 1)^2, \end{aligned} \quad (24)$$

where $a(x)$ is the diffusion coefficient of a porous medium, which is the input of neural operators and is stored by a regular grid. The output is the solution u on the same grid. In this dataset, the original resolution is 421×421 . We perform a $5 \times$ downsample on the original one as previous works do. We further enhance the dataset by applying randomly flip horizontally or vertically to both input and output since the equation guarantees the operation.

Navier Stokes This equation models incompressible and viscous flow with periodic boundary conditions:

$$\begin{aligned} \partial_t w(x, t) + u(x, t) \cdot \nabla w(x, t) &= \nu \nabla^2 w(x, t) + f(x), & x \in (0, 1)^2, t \in (0, T], \\ \nabla \cdot u(x, t) &= 0, & x \in (0, 1)^2, t \in (0, T], \\ w(x, 0) &= w_0(x), & x \in (0, 1)^2, \end{aligned} \quad (25)$$

where u is the velocity and $w = \nabla \times u$ is the vorticity of the flow. We choose $\nu = 10^{-5}$ in this case. Each sample in the dataset has the vorticity field in 20 time steps. The neural operator takes 10 steps as input and predicts the next 1 step. It advances in time by dropping the first step in history and appending the prediction of the neural operator.

Plasticity In this task, neural operators need to predict future deformation of a given plasticity material under the different impacts from above. Since the initial state of the plasticity does not change, the neural operators take time t , boundary condition on the top as input, and directly predict the deformation field in the domain.

Pipe In this task, neural operators need to predict the x -component of the velocity field of the fluid in pipes with different shapes. The input is the structured mesh’s physical position under the pipe’s deformation, and the output is the x -component of the velocity field on the same structured mesh.

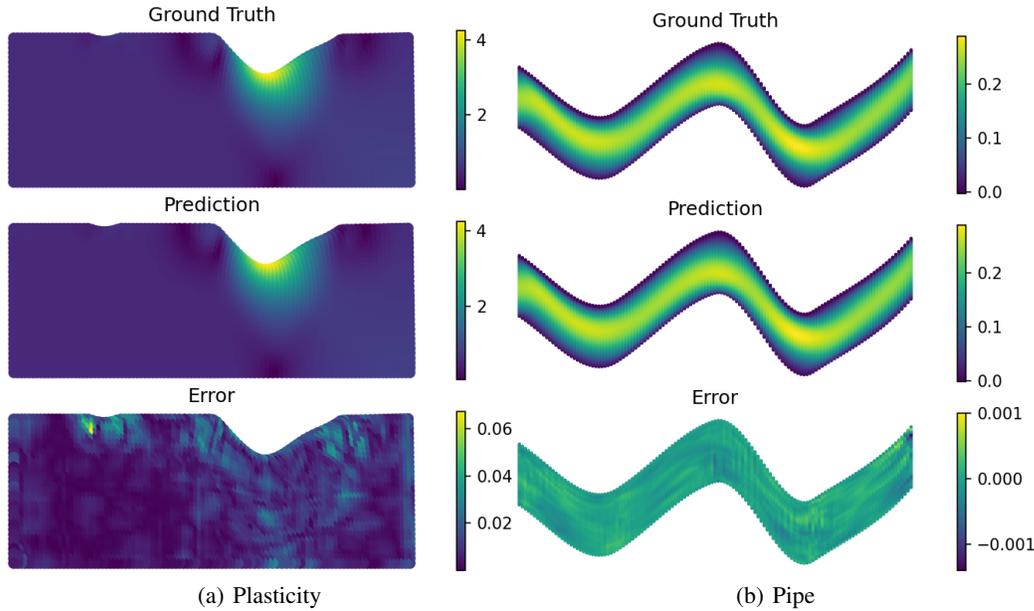


Figure 6. Comparison between ground truth in Plasticity and Pipe test set.

Airfoil Euler equation models the transonic flow over an airfoil. The input is mesh point locations, and the output is the Mach number on a structured mesh. We extract the average shape as the mean mesh to precompute the Laplacian eigensubspace. We compute the eigensubspace for only one mesh (i.e. the mean mesh), not for each sample.

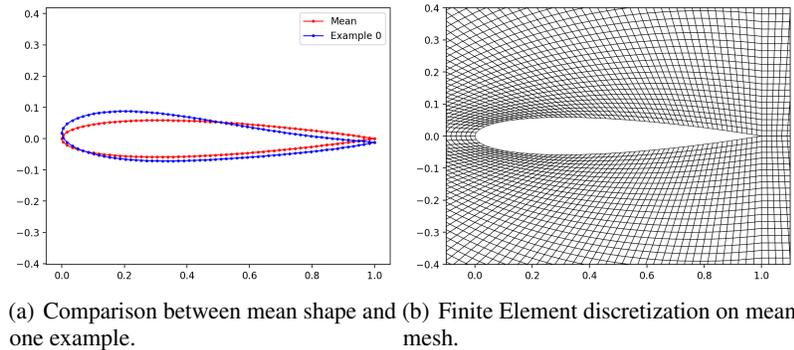


Figure 7. Laplacian eigensubspace construction in Airfoil.

We use input/output normalizers in each case, which is a common strategy to guarantee the stability of training progress. We use AdamW optimizer(Loshchilov & Hutter, 2017) and a OneCycle scheduler(Smith & Topin, 2018) for all the test cases.

C.2. Configurations for optimal records

We list the configuration of optimal records in Table 10. In the table, H_1 loss corresponds to the Sobolev seminorm in the function space, which measures the regularity of the function. In our experiments on Darcy, $|u|_{H_1} = \|\nabla u\|_{L_2}$. Besides, all the losses in the table are relative. Different time steps in a data sample are considered multiple samples in training.

Test Case	Loss Fn	Basis	#Basis	Normalization	#Layers	#Hiddens
Darcy	$L_2 + 0.1H_1$	Chebyshev	100	Layer	8	128
Navier Stokes	L_2	Fourier	144	Instance	8	256
Plasticity	L_2	Fourier	144	Instance	8	128
Airfoil	L_2	Eigensubspace	64	Layer	6	64
Pipe	L_2	Fourier	144	Layer	8	96

Table 10. The optimal configuration of SUPRA neural operator in Table 1.

The following L_2 norm on domain Ω is used to compute our error metric:

$$\|u\|_2^2 = \int_{\Omega} (u(x))^2 dx \approx \frac{1}{M} \sum_{i \in \text{mesh}} u(x_i)^2, \quad (26)$$

where M is the number of sample points. We assume a structured or regular grid is $[0, 1]^2$. For the relative L^2 error used in accuracy comparison, coefficient M is canceled:

$$\text{Relative } L_2 \text{ Error}(u, u^*) = \frac{\|u - u^*\|_2}{\|u^*\|_2} = \frac{\sqrt{\sum_{i \in \text{mesh}} (u(x_i) - u^*(x_i))^2}}{\sqrt{\sum_{i \in \text{mesh}} u^*(x_i)^2}}. \quad (27)$$