SPATIOTEMPORAL-AWARE MAMBA NEURAL OPERATOR FOR TIME-DEPENDENT PDE PREDICTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Neural operators (NOs) have emerged as efficient surrogates for predicting PDE-governed spatiotemporal dynamical systems. However, accurately modeling both temporal evolution and spatial interactions remains challenging, particularly for turbulent and long-horizon dynamics. We introduce MCANO, an autoregressive Mamba-based neural operator with dedicated temporal and spatial processing. Specifically, the temporal-awareness module employs a lightweight pointwise polynomial-basis lifting and embedding strategy to explicitly characterize temporal evolution patterns at each spatial location, while the spatial-awareness module augments conventional directional scans with a learned offset-based scan to provide an additional feature-conditioned spatial traversal. Building upon MCANO, we further develop MCANOP, a scalable multiphysics pretraining and fine-tuning framework that combines random sub-trajectory sampling, instance-wise normalization, and task-specific adaptation for heterogeneous PDE systems. Across regular-grid, irregular-grid, short-trajectory, and long-trajectory benchmarks, MCANO achieves the best performance on six of nine datasets and ranks second on the remaining three. After task-specific fine-tuning, MCANOP-17M substantially outperforms pretrained models of comparable size across all in-domain and out-of-domain datasets. Our findings highlight the potential of MCANO and MCANOP as parameter-efficient paradigms for accurate PDE prediction, matching or surpassing the performance of significantly larger models.

1 INTRODUCTION

Time-dependent partial differential equations (PDEs) underpin critical applications, including weather forecasting, fluid dynamics, and magnetohydrodynamics (Pathak et al., 2022; Augenstein et al., 2023; Jiang et al., 2024). Neural operators (NOs) alleviate the costly repeated simulations of traditional numerical solvers by learning mappings from input function spaces to solution spaces, enabling rapid prediction once trained. Pioneering architectures such as DeepONet (Lu et al., 2021) and Fourier neural operators (FNO) (Li et al., 2020) established the operator learning paradigm, inspiring spectral, convolutional, attention-based, and latent-space variants (Tran et al., 2021; Raonic et al., 2023; Li et al., 2022; Wang & Wang, 2024).

A key challenge in NO design is to capture complex spatiotemporal dependencies with appropriate inductive biases and manageable model capacity. MLPs lack explicit spatial structure awareness, while CNNs provide strong locality but struggle with long-range interactions. Transformers enable global modeling through self-attention, yet often require substantial model capacity and incur quadratic complexity with sequence length. Recently, state-space models (SSMs), particularly Mamba (Dao & Gu, 2024b), have emerged as an alternative for global dependency modeling through selective state propagation. Their compact sequence modeling provides a promising backbone for time-dependent PDEs (Zheng et al., 2024; Tiwari et al., 2025; Soares et al., 2025).

While existing Mamba-based NOs (Zheng et al., 2024; Tiwari et al., 2025) show promise, their application to time-dependent PDEs still leaves temporal and spatial representations tightly coupled. On the temporal side, historical states are commonly stacked as input channels and subsequently mixed by spatial operator blocks, providing limited explicit structure for describing temporal evolution. On the spatial side, Mamba-based models typically serialize two-dimensional features using predefined scan orders. Such serialization is independent of the current input field, so spatial locations

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

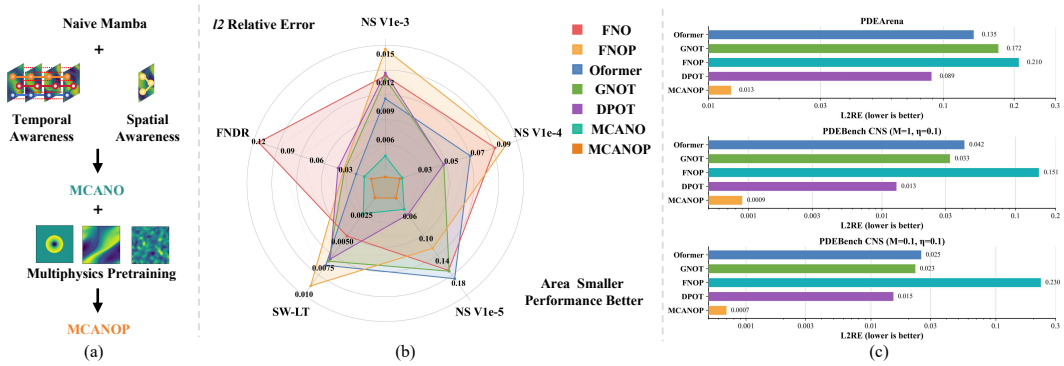

Figure 1: Schematic illustration (a) and performance comparison (b) of our proposed MCANO and MCANOP. (c) Accuracy comparison of MCANOP after fine-tuning on out-of-domain datasets.

that are adjacent in the scan sequence are not necessarily strongly correlated in the learned feature space. This limitation becomes more apparent for heterogeneous PDE fields containing localized interfaces, vortical structures, or rapidly varying regions. We therefore investigate explicit temporal feature lifting together with an additional feature-conditioned spatial traversal, while retaining conventional geometric scans as stable propagation paths.

In this work, we propose **M**amba-Based **C**ontent-**A**ware **N**eural **O**perator (**MCANO**), an autoregressive neural operator for spatiotemporal PDE prediction. MCANO separates temporal feature construction from subsequent spatial state propagation. First, the **T**emporal **A**wareness **M**odule (**TAM**) applies pointwise polynomial-basis lifting to the observed history to enrich temporal representations, which are then embedded into a compact feature space. Second, **S**patial **A**wareness **Mamba** (**SA-Mamba**) retains four conventional directional scans and introduces an additional learned offset-based scan conditioned on the current feature map. This feature-dependent sampling and reordering allow the traversal pattern to better conform to the spatial organization of the underlying field.

We further extend MCANO into **MCANOP**, a multiphysics pretraining and fine-tuning framework. Multi-PDE pretraining is challenging because different datasets exhibit substantially different field statistics, trajectory lengths, and physical variables. MCANOP addresses these differences through random sub-trajectory sampling and reversible instance normalization during pretraining. For downstream adaptation, the pretrained SA-Mamba processor is transferred while task-specific temporal lifting and projection layers are reinitialized, allowing the same pretrained processor to accommodate different physical variables. As shown in Figure 1, this compact pretraining framework yields substantial gains after task-specific fine-tuning without requiring a substantially larger backbone.

The main contributions are summarized as follows:

- **Mamba-Based Spatiotemporal-Aware NO Structure.** We propose MCANO, a Mamba-based NO that incorporates the TAM and SA-Mamba to enhance the modeling of temporal dynamics and spatial structures, achieving a favorable accuracy-parameter trade-off.
- **MCANO-Based Pretrained Framework.** We develop MCANOP, a scalable multiphysics pretraining and fine-tuning framework built upon MCANO, which learns shared cross-physics representations and improves prediction accuracy across diverse PDEs.
- **Empirical validation.** We evaluate the prediction accuracy of MCANO across nine PDE datasets with varying geometries and trajectory lengths. The fine-tuned MCANOP further surpasses state-of-the-art pretrained models on both in-domain and out-of-domain datasets.

2 Related Works

Neural operators for PDEs. NOs learn mappings between infinite-dimensional function spaces to solve PDEs efficiently. FNO (Li et al., 2020) parameterizes integral operators in the Fourier domain. Geo-FNO (Li et al., 2023) extends operator learning to irregular geometries via learned deformation.

OFormer (Li et al., 2022) introduces Transformer-based (Vaswani et al., 2017) attention to capture long-range dependencies. Latent Neural Operator (LNO) (Wang & Wang, 2024) introduces physics cross-attention to lift geometric inputs into a compact latent representation. CALM-PDE (Hagnberger et al., 2025) proposes a continuous and adaptive convolutional encoder-decoder tailored for time-dependent PDEs. While these methods have advanced operator learning, physical-domain processing and attention-heavy architectures may require substantial model capacity for capturing complex spatiotemporal dependencies.

State-space modeling for PDEs. State-space models, particularly Mamba (Dao & Gu, 2024b;a), provide an efficient alternative for modeling long sequences of spatiotemporal PDEs. MambaNO (Zheng et al., 2024) combines global Mamba integration with an alias-free local architecture, while LaMO (Tiwari et al., 2025) and PDE-FM (Soares et al., 2025) introduce latent-space Mamba modeling for time-dependent PDEs. Despite these advances, most existing Mamba-based NOs either treat time as an extra channel, missing explicit temporal dynamics, or apply fixed scanning paths that ignore the spatially varying structure of physical fields. Our proposed MCANO directly addresses these issues with the TAM and SA-Mamba modules, enabling explicit temporal-spatial decoupling and content-adaptive spatial state propagation within a compact model.

Pretraining for Time-dependent PDEs. Large-scale pretraining has recently been investigated to improve predictive capability across diverse PDE systems. MPP (McCabe et al., 2024) trains autoregressive models on multiple physics, and DPOT (Hao et al., 2024) incorporates denoising objectives to improve prediction stability. Unisolver (Zhou et al., 2024) uses symbolic PDE encoding for unified multi-physics prediction. These approaches primarily focus on scaling Transformer-based architectures or developing unified representations across heterogeneous PDE systems, with relatively limited emphasis on explicit spatiotemporal inductive biases. PDE-FM explores a Mamba backbone for PDE foundation models but does not explicitly design for temporal-spatial dynamics. Our MCANOP combines a compact spatiotemporal-aware Mamba backbone with multi-physics pretraining, demonstrating strong in-domain pretrained-only performance and effective downstream adaptation to datasets excluded from pretraining.

3 Proposed Method

We propose the autoregressive MCANO for spatiotemporal dynamical systems, which achieves accurate and stable long-term predictions. It consists of a lightweight TAM that explicitly enriches temporal features at each spatial location, and stacks of SA-Mamba integral layers that perform feature-conditioned spatial state propagation through fixed and dynamic scan paths. The architecture is further extended to a pretraining and fine-tuning framework (MCANOP) for multi-PDE learning. In the following, we first formulate the learning problem, then detail each component.

3.1 Problem Definition

Consider a system whose state $\mathbf{u}(t, \mathbf{x}) \in \mathbb{R}^C$, with C physical channels, evolves according to a time-dependent PDE

$$\partial_t \mathbf{u}(t, \mathbf{x}) = \mathcal{F}\big(t, \mathbf{x}, \mathbf{u}, \nabla \mathbf{u}, \nabla^2 \mathbf{u}, \boldsymbol{\lambda}\big), \tag{1}$$

subject to the initial condition $\mathbf{u}(0, \mathbf{x}) = \mathbf{u}^0(\mathbf{x})$ and boundary conditions $\mathcal{B}\mathbf{u} = 0$ on $[0, T] \times \partial\Omega$. Here $\mathcal{F}$ is a nonlinear parameterized operator, and $\boldsymbol{\lambda}$ denotes physical parameters. The continuous problem defines a solution mapping from the initial-boundary data to the full spatiotemporal field.

To make learning tractable, we work with finite-dimensional observations. On a regular $H \times W$ mesh, a snapshot at time t is represented as $\mathbf{u}^t \in \mathbb{R}^{H \times W \times C}$, and on unstructured meshes, the data consist of sets $\{\mathbf{u}(t, \mathbf{x}_i)\}_{i=1}^N$. In this work we focus on two-dimensional regular grids, but the proposed method can be straightforwardly extended to three dimensions and unstructured point sets.

Given a set of observed snapshots, the prediction task amounts to learning a discrete evolution operator that maps a recent history to the next state. This data-driven formulation applies regardless of whether the underlying PDE is known analytically. Owing to spatial discretization, a single snapshot may not fully resolve the latent dynamics relevant to future evolution. We therefore condition the prediction on a short history of T_{in} consecutive frames. Formally, given $\mathbf{U}^n = (\mathbf{u}^{n-T_{\text{in}}+1}, \ldots, \mathbf{u}^n)$,

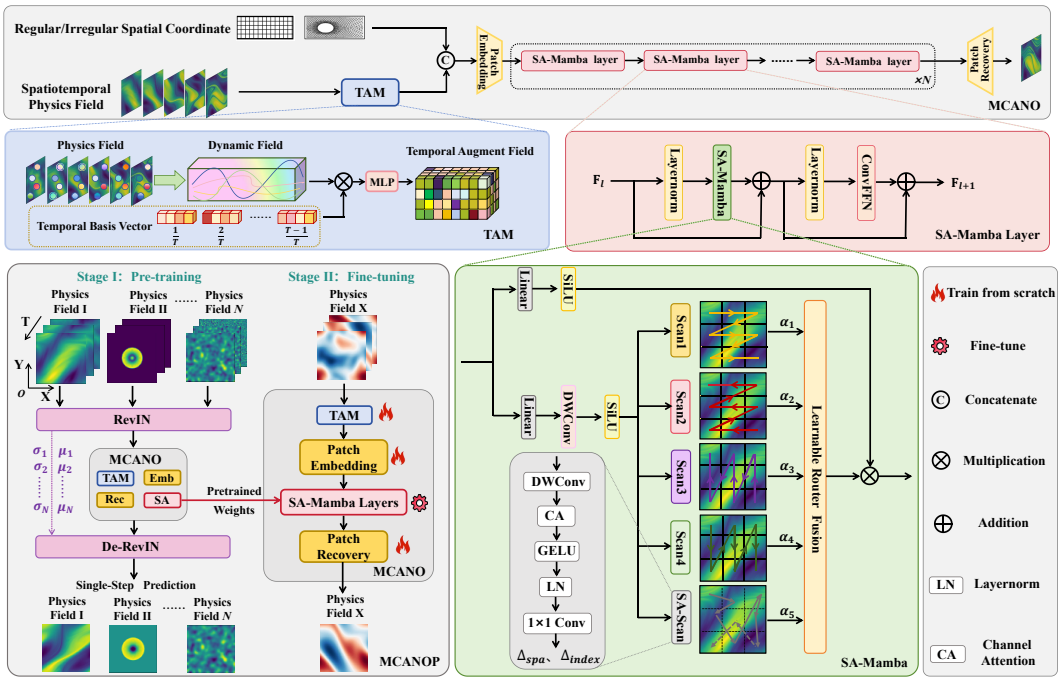

Figure 2: Illustration of the proposed MCANO and MCANOP framework. The main components and their respective data flows are annotated directly within the figure.

the goal is to learn an operator $\mathcal{G}_\theta$ such that

$$\hat{\mathbf{u}}^{n+1} = \mathcal{G}_\theta(\mathbf{U}^n). \tag{2}$$

By recursively applying $\mathcal{G}_\theta$, we obtain autoregressive forecasts of any desired length.

3.2 Overall Architecture

The overall design of MCANO follows an encoder–processor–decoder structure, depicted in Figure 2. An input trajectory of T_{in} frames $\mathbf{u}^t \in \mathbb{R}^{H\times W\times C}$, with $t = 0, 1, \ldots, T_{in}-1$, is first processed by the TAM. At each spatial location, TAM independently transforms the local time series of length T_{in} into a compact feature vector and yields a temporally enhanced feature map $\mathbf{H}_T \in \mathbb{R}^{H\times W\times d_T}$.

To inject spatial coordinate information, we concatenate $\mathbf{H}_T$ with normalized coordinate embeddings and map the result into a set of latent patches via a learnable patch embedding layer $\mathcal{P}_{\text{emb}}$, producing the initial token grid $\mathbf{F}_0$. The patch tokens are then fed into a stack of (L) SA-Mamba layers that propagate features along fixed and dynamic scan paths:

$$\mathbf{F}_\ell = \text{SA-Mamba}_{\ell-1}(\mathbf{F}_{\ell-1}), \quad \ell = 1, \ldots, L. \tag{3}$$

Finally, the refined representation $\mathbf{F}_L$ is decoded back to the physical space by a patch recovery module $\mathcal{P}_{\text{rec}}$, giving the prediction of the next-time snapshot $\hat{\mathbf{u}}^{T_{\text{in}}}$. All modules are trained end-to-end to minimize the prediction error over a training set of dynamical trajectories.

3.3 Temporal Awareness Module

We introduce the Temporal Awareness Module (TAM) to decouple temporal evolution from spatial mixing. TAM operates independently at each spatial location before tokenization, enriching temporal representations for subsequent spatial modeling.

TAM consists of two stages. First, a polynomial basis lifting operation constructs a structured temporal feature representation by combining the observed history with a set of predefined basis functions evaluated on normalized time coordinates. Subsequently, a pointwise temporal embedding module maps the lifted representation to a compact feature vector through a shared multilayer perceptron (MLP), which is then passed to the subsequent spatial processing stage.

Polynomial Basis Lifting. For a spatial location $\mathbf{x}$, let $\mathbf{u}_\mathbf{x} \in \mathbb{R}^{T_{\text{in}} \times C}$ be the input history at that point. We first normalize the observation times to $\tau_i = (t_i - t_1)/(t_{T_{\text{in}}} - t_1) \in [0, 1]$. A set of polynomial basis functions $\{\phi_k(\tau) = \tau^k\}_{k=0}^{K}$ is evaluated at each τ_i. The lifted representation $\mathbf{z}_\mathbf{x}$ is formed by multiplying the observed values with the basis functions and flattening

$$\mathbf{z}_\mathbf{x} = \text{vec}\left(\left[\mathbf{u}(\mathbf{x}, t_i)\,\phi_k(\tau_i)\right]_{i=1,\ldots,T_{\text{in}};\, k=0,\ldots,K}\right), \tag{4}$$

where the multiplication is scalar–channel multiplication and vec concatenates all entries into a single vector. The dimension of $\mathbf{z}_\mathbf{x}$ is $T_{\text{in}}(K+1) \times C$.

Pointwise Temporal Embedding. The high-dimensional lifted vector is then mapped to a compact temporal feature through a shared multilayer perceptron:

$$\mathbf{h}_\mathbf{x} = \text{MLP}_{\theta_T}(\mathbf{z}_\mathbf{x}), \qquad \mathbf{h}_\mathbf{x} \in \mathbb{R}^{d_T}. \tag{5}$$

The same MLP is applied independently at every spatial location, making TAM fully parallelizable across the grid. The resulting embedding vectors are rearranged into a spatial feature map $\mathbf{H}_T \in \mathbb{R}^{H \times W \times d_T}$ and passed to the subsequent patch embedding stage.

TAM can be viewed as a feature reparameterization rather than an explicit polynomial regression. It exposes correlations between the observed history and elementary temporal functions, since the basis values are fixed on a given temporal grid. This operation requires neither recurrent states nor attention and processes the entire input window in parallel, incurring trivial computational cost.

Moreover, TAM is inserted before the operator backbone, leaving the integral kernel architecture unchanged. This ensures compatibility with recent NO designs while explicitly enhancing temporal awareness. A theoretical polynomial approximation analysis is also provided in Appendix F.

3.4 Spatial Awareness Mamba-based Integral Layers

Building on geometric scans used in conventional 2D Mamba blocks, we introduce SA-Mamba with two components: SA-SCAN, a feature-conditioned offset scanning path that adaptively reorganizes token traversal according to the feature map, and a learnable multi-scan fusion that integrates different traversal paths. Each SA-Mamba layer follows a LayerNorm–SA-Mamba Module–LayerNorm–ConvFFN structure, reminiscent of a Transformer block (Dosovitskiy et al., 2021).

SA-SCAN Path. Given $\mathbf{F}_\ell \in \mathbb{R}^{H \times W \times C}$ with $N = HW$ spatial tokens, SA-SCAN predicts for each anchor a spatial offset $\hat{\boldsymbol{\delta}}_i^{\text{sp}} \in \mathbb{R}^2$ and a scalar path offset $\hat{\delta}_i^{\text{path}}$:

$$\left\{\hat{\boldsymbol{\delta}}_i^{\text{sp}}, \hat{\delta}_i^{\text{path}}\right\} = f_{\text{offset}}(\mathbf{F}_\ell)_i. \tag{6}$$

After bounded transformation, the spatial offset deforms the sampling location and bilinear interpolation retrieves the corresponding feature,

$$\tilde{\mathbf{x}}_i = \mathcal{B}\big(\mathbf{F}_\ell, \mathbf{p}_i^0 + \boldsymbol{\delta}_i^{\text{sp}}\big) + \mathbf{b}_i^{\text{rpe}}. \tag{7}$$

The path offset defines a feature-dependent priority score

$$q_i = r_i + \delta_i^{\text{path}}, \qquad \boldsymbol{\pi} = \text{argsort}(q_1, \ldots, q_N), \tag{8}$$

where r_i denotes the fixed monotonic reference score of the raster order, and $\boldsymbol{\pi}$ reorders the sampled features into $\mathcal{S}_\Delta = [\tilde{\mathbf{x}}_{\pi_1}, \ldots, \tilde{\mathbf{x}}_{\pi_{N_{\text{tok}}}}] \in \mathbb{R}^{N_{\text{tok}} \times C}$. Thus, spatial offsets control where features are sampled, while path offsets determine their traversal order.

Multi-Scan Integration. The dynamic sequence $\mathcal{S}_\Delta$ is combined with four conventional directional scans, including horizontal, vertical, and their reversed counterparts. The five sequences are processed in parallel by the SSM layer:

$$\mathbf{h}_q = \exp(\mathbf{A}\Delta_q)\mathbf{h}_{q-1} + \mathbf{B}_q\mathbf{x}_q, \qquad \mathbf{y}_q = \mathbf{C}_q\mathbf{h}_q + \mathbf{D}\mathbf{x}_q, \tag{9}$$

where q indexes positions along the serialized scan, Δ_q is the input-dependent selective SSM step size, and $\mathbf{B}_q$ and $\mathbf{C}_q$ are conditioned on the input at scan position q.

Before fusion, all outputs are restored to the canonical raster order using the corresponding inverse transpose and reversal operations for geometric scans and the inverse permutation $\boldsymbol{\pi}^{-1}$ for SA-SCAN. Denoting the aligned outputs as $\{\bar{\mathcal{Y}}_k\}_{k=1}^{5}$, we compute

$$\mathbf{Y}_{\text{fused}} = \sum_{k=1}^{5} \alpha_k \bar{\mathcal{Y}}_k, \qquad \alpha_k = \frac{\exp(\lambda_k)}{\sum_{j=1}^{5} \exp(\lambda_j)}, \tag{10}$$

where λ_k is a learnable scalar.

Finally, a Convolutional Feed-Forward Network (ConvFFN) (Zhou et al., 2023) replaces the standard pointwise MLP to reinforce local spatial inductive biases, concluding one SA-Mamba layer.

3.5 Pretraining and Fine-tuning for MCANO

We extend MCANO to a pretraining and fine-tuning framework, termed **MCANOP**, which learns shared representations from multiple time-dependent PDE datasets for downstream adaptation. Below we describe the key techniques that enable multi-physics pretraining and subsequent fine-tuning.

Random Sub-trajectory Sampling. Standard NO training often processes complete PDE trajectories, which incurs high memory cost and may cause overfitting to the temporal patterns of individual realizations. To mitigate this and encourage the model to capture the underlying PDE dynamics, we randomly sample short segments from the full trajectories during pretraining. Each segment consists of T_{in} input frames and T_{out} target frames, matching the model's input-output configuration. Teacher forcing is employed, where ground-truth frames are used as the autoregressive input at each step, and the model predicts the subsequent T_{out} frames.

RevIN-based Multi-PDE Pretraining. Pretraining across diverse PDE datasets introduces substantial distribution shifts, as solution magnitudes and statistics can differ by orders of magnitude. Standard dataset-level z-score normalization (Cabello-Solorzano et al., 2023) fails to align these disparate data distributions, often leading to training instability. We adopt Reversible Instance Normalization (RevIN) (Kim et al., 2021). For a given input sample $\mathbf{U}_s \in \mathbb{R}^{T_{\text{in}} \times H \times W \times C}$, let μ_s and σ_s be its mean and standard deviation. RevIN normalizes the sample as

$$\tilde{\mathbf{U}}_s = \gamma \odot \frac{\mathbf{U}_s - \mu_s}{\sigma_s} + \beta, \tag{11}$$

where $\gamma, \beta \in \mathbb{R}^C$ are learnable affine parameters shared across samples. The normalized field $\tilde{\mathbf{U}}_s$ is then processed by MCANO to predict future states, and the prediction is restored to the original physical scale by reversing equation 11:

$$\hat{\mathbf{U}}_s^{\text{out}} = \sigma_s \odot \frac{\text{MCANO}(\tilde{\mathbf{U}}_s) - \beta}{\gamma} + \mu_s. \tag{12}$$

By normalizing each sample independently, RevIN effectively attenuates cross-dataset distribution discrepancies, stabilizing pretraining and facilitating representation learning.

Fine-tuning Strategy. The physical variables and channel semantics may differ substantially between pretraining and downstream PDEs. We therefore transfer the pretrained SA-Mamba processor rather than enforcing a shared input-output interface across all tasks. Specifically, the SA-Mamba layers are initialized from the multiphysics pretrained checkpoint, whereas TAM, patch embedding, and patch recovery layers are reinitialized for the target dataset. During downstream training, the transferred processor and the newly initialized task-specific layers are jointly optimized. This separation allows MCANOP to reuse the spatiotemporal representations learned during pretraining while adapting the temporal and variable-specific interfaces to each downstream system.

4 Experiments

We evaluate MCANO on nine benchmark datasets of time-dependent PDEs against representative models from both NOs and spatiotemporal forecasting. Ablation studies and visualizations validate the proposed TAM and SA-Mamba components, and multi-PDE pretraining and fine-tuning experiments with MCANOP demonstrate cross-equation generalization.

Table 1: Benchmark results on time-dependent PDE datasets. The L2RE is reported for all datasets, where lower values indicate better performance. For each dataset, the best-performing model is highlighted in **bold**, and the second is underlined.

Method				Dataset				
Name	**Published**	**Params**	**Train Method**	**NS V1e-3**	**NS V1e-4**	**NS V1e-5**	**SW-ST**	**GSDR**
FNO2D	ICLR21	0.41M	Autoregressive	0.0128	0.0834	0.1556	0.00142	0.1564
FNO3D	ICLR21	6.56M	Autoregressive	0.0086	0.0820	0.1893	0.00207	0.1567
OFormer	TMLR23	1.85M	Autoregressive	0.0104	0.0645	0.1705	0.00047	0.2877
LNO-S	NeurIPS24	0.81M	Teacher Forcing	0.0137	0.0781	0.1331	0.00104	0.2889
LNO-L	NeurIPS24	5.34M	Teacher Forcing	**0.0031**	0.0131	0.0924	**0.00011**	0.2910
LaMO-S	ICML25	1.63M	Teacher Forcing	0.0174	0.0466	0.0998	0.00035	0.1649
LaMO-L	ICML25	3.13M	Teacher Forcing	0.0091	0.0261	0.0658	0.00035	0.1531
CALM-PDE	NeurIPS25	2.28M	Autoregressive	0.0218	0.0709	0.1284	0.00077	0.2611
SimVP	CVPR22	1.85M	Autoregressive	0.0396	0.1699	0.2414	0.00118	0.2822
TAU	CVPR23	4.16M	Autoregressive	0.0314	0.1204	0.2381	0.00610	0.2842
PredFormer	TMLR26	5.28M	Autoregressive	0.0281	0.0981	0.1671	0.00277	0.2807
MCANO (Ours)	-	3.80M	Teacher Forcing	0.0044	**0.0126**	**0.0459**	0.00024	**0.1463**

4.1 Experiment Settings

The datasets span six time-dependent PDE families with distinct physical characteristics, as detailed in Appendix A. In particular, the Navier-Stokes equation (Li et al., 2020) provides two-dimensional viscous incompressible fluid flow in vorticity form. We adopt three viscosity coefficients $\nu = 10^{-3}, 10^{-4}, 10^{-5}$, corresponding to increasingly turbulent regimes. The Shallow-Water equation (Takamoto et al., 2022) captures mass and momentum conservation in two-dimensional free-surface flows, with short-trajectory (ST) and long-trajectory (LT) variants of $T = 26$ and $T = 101$ time steps, respectively. The Gray-Scott diffusion-reaction (GSDR) (Ohana et al., 2024) and FitzHugh-Nagumo diffusion-reaction (FNDR) (Takamoto et al., 2022) models describe two-component reaction-diffusion systems driven by diffusive coupling and nonlinear reaction dynamics. The Airfoil and Cylinder datasets (Pfaff et al., 2020) simulate compressible and incompressible fluid dynamics, respectively, around airfoil and cylinder geometries.

We select classical NOs, as detailed in Appendix B, including the CNN-based FNO2D, FNO3D, Geo-FNO, and CALM-PDE(Li et al., 2020; 2023; Hagnberger et al., 2025), the Transformer-based Oformer and LNO (Li et al., 2022; Wang & Wang, 2024), as well as the Mamba-based LaMO (Tiwari et al., 2025). Three spatiotemporal forecasting models, SimVP (Gao et al., 2022), TAU (Tan et al., 2023), and PredFormer (Tang et al., 2026), are also included for comparison.

All models are trained for 500 epochs on NVIDIA H100 GPUs. We adopt the original training implementations, and adjust them only when they result in significantly worse performance. For all experiments, the input length is fixed to $T_{\text{in}} = 10$, so that the model receives the first ten time steps of each trajectory and predicts the subsequent step. Each model is evaluated under two training strategies: *teacher forcing*, where ground-truth previous states are supplied during training, and *autoregressive rollout*, which uses the model's own predictions. We report the strategy that yields the best validation performance for each model. During evaluation, the model forecasts the entire trajectory in an autoregressive manner. The l^2 relative error (L2RE) (Li et al., 2020) serves as the training loss and the evaluation metric.

4.2 Main Results

As summarized in Table 1, on regular-grid PDEs, MCANO demonstrates superior or competitive performance with only 3.80M parameters. MCANO achieves the lowest error on NS $\nu = 10^{-4}$ and NS $\nu = 10^{-5}$, highlighting the effectiveness of the architecture for modeling turbulent dynamics. On the SW-ST dataset, MCANO attains the second-best performance. The overall low errors indicate that this task is relatively easier. For GSDR, MCANO achieves the best result, with FNO remaining competitive. Notably, larger parameter counts do not guarantee better performance: MCANO (3.80M) outperforms FNO3D (6.56M) and PredFormer (5.28M) on all NS and SW-ST datasets, underscoring the importance of architectural design over sheer parameter scaling.

Table 2: Experimental results comparing pre-trained models in terms of pretrained-only (*-P) and task-specific fine-tuning (*-F) accuracy on in-domain and out-of-domain datasets. Notably, the out-of-domain datasets are excluded from MCANOP-17M pre-training but are included in the pre-training data of the compared models.

	In-domain Datasets					**Out-of-domain Datasets**		
Model	NS V1e-3	NS V1e-4	NS V1e-5	SW-LT	FNDR	PDEArena	CNS M=1	CNS M=0.1
FNOP-7M-P	0.0156	0.0922	0.116	0.00912	0.0321	0.210	0.130	0.153
DPOT-7M-P	0.0095	0.0606	0.0976	0.00560	0.0321	0.125	0.0285	0.0176
MPP-7M-P	–	–	–	0.0066	0.0168	–	0.0442	0.0312
DPOT-30M-P	0.0131	0.0442	0.0553	0.00657	0.0379	0.0991	0.0245	0.0153
MPP-30M-P	–	–	–	0.0024	0.0112	–	0.0319	0.0213
Unisolver-33M-P	0.0061	0.0336	0.0417	0.0045	0.0439	0.0687	0.0206	0.0130
MCANOP-17M-P	**0.0032**	**0.0226**	0.0460	**0.0012**	0.0169	–	–	–
DPOT-7M-F	0.00580	0.0367	0.0520	0.00241	0.0148	0.0910	0.0153	0.0156
DPOT-30M-F	0.00437	0.0237	0.0322	0.00235	0.0129	0.0867	0.0148	0.0139
MCANOP-17M-F	**0.0022**	**0.0108**	**0.0258**	**0.00078**	**0.00358**	**0.0125**	**0.0017**	**0.0013**

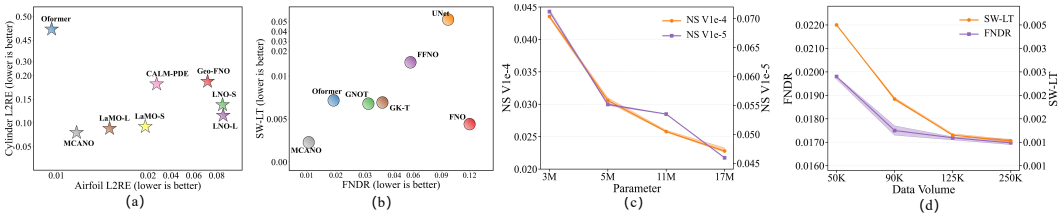

Figure 3: (a) Performance comparison on the airfoil and cylinder datasets. (b) Performance comparison on the SW-LT and FNDR datasets. (c) Scaling law experiments on model parameters. (d) Scaling law experiments on data volume.

To further evaluate robustness, we extend MCANO to two challenging settings: irregular spatial domains (Airfoil and Cylinder flows, Figure 3(a)) and long-trajectory prediction PDEs (SW-LT and FNDR, Figure 3(b)), each containing 101 time steps. Specifically, we adopt slice projection to handle irregular geometries and chunk-wise gradient backpropagation to enable stable learning over extended horizons (see Appendix C). MCANO remains competitive in the irregular-domain experiments and obtains the lowest L2RE on both SW-LT and FNDR, indicating that the architecture remains effective when either spatial geometry or rollout length becomes more challenging.

4.3 Experimental Results for MCANOP

Following prior multiphysics pretraining studies (Hao et al., 2024; McCabe et al., 2024; Zhou et al., 2024), we pretrain MCANOP on three Navier–Stokes datasets with different viscosities together with SW-LT and FNDR. All datasets are converted to 64×64 resolution with two physical channels. For downstream adaptation, we further evaluate on compressible Navier–Stokes datasets from PDEBench (Takamoto et al., 2022) and the PDEArena dataset (Gupta & Brandstetter, 2022). Additional implementation details are provided in Appendix E.

As shown in Table 2, despite modest computational resources that constrain the per-dataset data volume and limit the model to 17M parameters, MCANOP already matches or surpasses other pretrained models of comparable size. Without downstream fine-tuning, MCANOP-17M obtains the lowest L2RE among the compared pretrained models on NS $\nu = 10^{-3}$, NS $\nu = 10^{-4}$, and SW-LT, while remaining competitive on NS $\nu = 10^{-5}$ and FNDR.

Figure 1(c) and Table 2 show that task-specific fine-tuning produces substantially larger gains for MCANOP-17M, which substantially outperforms DPOT-30M-F on both in-domain and out-of-domain datasets. These results indicate that the pretrained processor provides a strong initialization that can be effectively specialized to individual PDE systems through downstream adaptation.

Table 3: Comparison of temporal representation designs and temporal embedding method on the NS $\nu = 10^{-5}$ dataset.

Architecture	L2RE↓
Attention	0.0629
Conv+MLP	0.0630
Mixer	0.0623
Gated MLP	0.0613
Naive MLP	0.0573
TAM	**0.0459**
Embedding	**L2RE↓**
Fourier	0.0599
Learnable	0.0631
Polynomial	**0.0459**

Table 4: Effects of Single-Dataset Training from Scratch, Pretraining, and Task-specific Fine-tuning for MCANOP-17M.

Training Scheme	NS $\nu = 10^{-5}$	CFD $M = 0.1, \eta = 0.1$
From scratch	0.0602	0.003563
Pre-training only	0.0460	–
Fine-tuning	**0.0258**	**0.000700**

Table 5: Ablation of the TAM and SA-Mamba components across multiple PDE datasets.

Settings	NS V1e-3	NS V1e-4	NS V1e-5	SW-ST	GSDR
MCANO	**0.0044**	**0.0126**	**0.0459**	**0.00024**	**0.1463**
w/o TAM	0.0059	0.0155	0.0643	0.00027	0.1497
w/o SA-Mamba	0.0054	0.0172	0.0540	0.00027	0.1525
Baseline	0.0085	0.0253	0.0661	0.00037	0.1530

MCANOP exhibits clear performance improvements as model capacity or data volume increases. We pretrain four variants with different parameter counts. Figure 3(c) shows that larger models consistently yield better results. Similarly, scaling the training data leads to significant gains (Figure 3(d)). These trends demonstrate that MCANOP can effectively leverage additional data and parameters to enhance predictive accuracy.

4.4 Ablation Results

To isolate the contribution of each proposed component, we conduct comprehensive ablation experiments. The full MCANO model is compared against three variants: (1) *w/o TAM*, which removes the TAM; (2) *w/o SA-Mamba*, which replaces the learned dynamic scan with a fixed four-direction scanning baseline; and (3) *Baseline*, which strips both TAM and the SA-SCAN path, yielding the simplest configuration. As shown in Table 5, the complete MCANO attains the lowest error on five datasets, confirming the complementary benefits of TAM and the SA-Mamba layer.

We further investigate both the architectural design and basis functions of TAM, comparing different temporal embedding structures as well as polynomial, Fourier, and learnable bases. As reported in Table 3, the simple design based on basis lifting followed by pointwise MLP embedding achieves the best performance, while more complex temporal architectures tend to degrade temporal feature modeling. Among the tested basis functions, the polynomial basis consistently performs better than the Fourier and learnable alternatives.

Table 4 shows that pre-training provides a stronger initialization than single-dataset training from scratch, while task-specific fine-tuning yields further gains. On NS $\nu = 10^{-5}$, the L2RE decreases from 0.0602 to 0.0460 after pre-training and further to 0.0258 after fine-tuning. On the CFD dataset with $M = 0.1$ and $\eta = 0.1$, fine-tuning reduces the error from 0.003563 to 0.000700, further demonstrating the transferability of the pretrained MCANOP-17M processor.

5 Conclusion

We introduced MCANO, an autoregressive Mamba neural operator that separates pointwise temporal feature construction from subsequent spatial state propagation. Across nine benchmarks spanning different geometries and rollout lengths, MCANO achieves the best performance on six datasets while maintaining a compact parameter budget.

We further developed MCANOP for multiphysics pretraining and task-specific adaptation. Fine-tuned MCANOP-17M achieves the best results among the compared fine-tuned pretrained models on all benchmarks, demonstrating that the pretrained processor can be effectively specialized to individual PDE systems. Future work includes scaling MCANO to larger models and extending the framework to three-dimensional problems and steady-state PDEs.

AI USE STATEMENT

In this work, we used generative AI tools to assist with manuscript drafting, rewriting, and language polishing. All AI-assisted suggestions were critically reviewed by the authors. Mathematical formulations, implementation details, experimental results, and citations were manually checked against the authors' code, experiments, and source literature. Generative AI was not treated as an authoritative source and did not independently determine the scientific conclusions. We take full responsibility for the final content of this work, including all text, claims, and artifacts produced with the assistance of generative AI.

REPRODUCIBILITY STATEMENT

We provide the mathematical formulation and architectural details of MCANO and MCANOP in the main text, including TAM, SA-SCAN, multi-scan integration, multi-PDE pretraining, and task-specific fine-tuning. The experimental section specifies the datasets, baseline models, training protocols, and evaluation metrics used in our study. Additional dataset descriptions, baseline configurations, implementation details for irregular-domain and long-trajectory experiments, pretraining settings, hyperparameters, and supplementary analyses are provided in the appendix. We will also provide a compact, anonymized, and fully functional code package as supplementary material.

REFERENCES

Yannick Augenstein, Taavi Repan, and Carsten Rockstuhl. Neural operator-based surrogate solver for free-form electromagnetic inverse design. *ACS Photonics*, 10(5):1547–1557, 2023.

Kelsy Cabello-Solorzano, Isabela Ortigosa de Araujo, Marco Peña, Luís Correia, and Antonio J. Tallón-Ballesteros. The impact of data normalization on the accuracy of machine learning algorithms: A comparative analysis. In *International conference on soft computing models in industrial and environmental applications*, pp. 344–353. Springer, 2023.

Tri Dao and Albert Gu. Transformers are ssms: Generalized models and efficient algorithms through structured state space duality. *arXiv preprint arXiv:2405.21060*, 2024a.

Tri Dao and Albert Gu. Transformers are SSMs: Generalized models and efficient algorithms through structured state space duality. In *International Conference on Machine Learning (ICML)*, 2024b.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=YicbFdNTTy.

Zhangyang Gao, Cheng Tan, Lirong Wu, and Stan Z Li. Simvp: Simpler yet better video prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3170–3180, 2022.

Jayesh K Gupta and Johannes Brandstetter. Towards multi-spatiotemporal-scale generalized pde modeling. *arXiv preprint arXiv:2209.15616*, 2022.

Jan Hagnberger, Daniel Musekamp, and Mathias Niepert. Calm-pde: Continuous and adaptive convolutions for latent space modeling of time-dependent pdes. *arXiv preprint arXiv:2505.12944*, 2025.

Zhongkai Hao, Chang Su, Songming Liu, Julius Berner, Chengyang Ying, Hang Su, Anima Anandkumar, Jian Song, and Jun Zhu. Dpot: Auto-regressive denoising operator transformer for large-scale pde pre-training. *arXiv preprint arXiv:2403.03542*, 2024.

Zhongyi Jiang, Min Zhu, and Lu Lu. Fourier-mionet: Fourier-enhanced multiple-input neural operators for multiphase modeling of geological carbon sequestration. *Reliability Engineering & System Safety*, 251:110392, 2024.

Taesung Kim, Jinhee Kim, Yunwon Tae, Cheonbok Park, Jang-Ho Choi, and Jaegul Choo. Reversible instance normalization for accurate time-series forecasting against distribution shift. In *International Conference on Learning Representations*, 2021.

Zijie Li, Kazem Meidani, and Amir Barati Farimani. Transformer for partial differential equations' operator learning. *arXiv preprint arXiv:2205.13671*, 2022.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. *arXiv preprint arXiv:2010.08895*, 2020.

Zongyi Li, Daniel Zhengyu Huang, Burigede Liu, and Anima Anandkumar. Fourier neural operator with learned deformations for pdes on general geometries. *Journal of Machine Learning Research*, 24(388):1–26, 2023.

Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. *Nature machine intelligence*, 3(3):218–229, 2021.

Michael McCabe, Bruno Régaldo-Saint Blancard, Liam Parker, Ruben Ohana, Miles Cranmer, Alberto Bietti, Michael Eickenberg, Siavash Golkar, Geraud Krawezik, Francois Lanusse, et al. Multiple physics pretraining for spatiotemporal surrogate models. *Advances in Neural Information Processing Systems*, 37:119301–119335, 2024.

Ruben Ohana, Michael McCabe, Lucas Meyer, Rudy Morel, Fruzsina Agocs, Miguel Beneitez, Marsha Berger, Blakesly Burkhart, Stuart Dalziel, Drummond Fielding, et al. The well: a large-scale collection of diverse physics simulations for machine learning. *Advances in Neural Information Processing Systems*, 37:44989–45037, 2024.

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

Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter Battaglia. Learning mesh-based simulation with graph networks. In *International Conference on Learning Representations*, 2020.

Bogdan Raonic, Roberto Molinaro, Tobias Rohner, Siddhartha Mishra, and Emmanuel de Bezenac. Convolutional neural operators. In *ICLR 2023 Workshop on Physics for Machine Learning*, 2023.

Eduardo Soares, Emilio Vital Brazil, Victor Shirasuna, Breno WSR de Carvalho, and Cristiano Malossi. Towards a foundation model for partial differential equations across physics domains. *arXiv preprint arXiv:2511.21861*, 2025.

Makoto Takamoto, Timothy Praditia, Raphael Leiteritz, Daniel MacKinlay, Francesco Alesiani, Dirk Pflüger, and Mathias Niepert. Pdebench: An extensive benchmark for scientific machine learning. *Advances in neural information processing systems*, 35:1596–1611, 2022.

Cheng Tan, Zhangyang Gao, Lirong Wu, Yongjie Xu, Jun Xia, Siyuan Li, and Stan Z Li. Temporal attention unit: Towards efficient spatiotemporal predictive learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18770–18782, 2023.

Yujin Tang, Lu Qi, Fei Xie, Xiangtai Li, Chao Ma, and Ming-Hsuan Yang. Video prediction transformers without recurrence or convolution. In *Transactions on Machine Learning Research*, 2026.

Karn Tiwari, Niladri Dutta, NM Krishnan, et al. Latent mamba operator for partial differential equations. *arXiv preprint arXiv:2505.19105*, 2025.

Alasdair Tran, Alexander Mathews, Lexing Xie, and Cheng Soon Ong. Factorized fourier neural operators. *arXiv preprint arXiv:2111.13802*, 2021.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Tian Wang and Chuang Wang. Latent neural operator for solving forward and inverse pde problems. *Advances in Neural Information Processing Systems*, 37:33085–33107, 2024.

Haixu Wu, Huakun Luo, Haowen Wang, Jianmin Wang, and Mingsheng Long. Transolver: A fast transformer solver for pdes on general geometries. *arXiv preprint arXiv:2402.02366*, 2024.

Jianwei Zheng, Wei Li, Ni Xu, Junwei Zhu, Xiaoxu Lin, and Xiaoqin Zhang. Alias-free mamba neural operator. *Advances in Neural Information Processing Systems*, 37:52962–52995, 2024.

Hang Zhou, Yuezhou Ma, Haixu Wu, Haowen Wang, and Mingsheng Long. Unisolver: Pde-conditional transformers are universal pde solvers. *arXiv preprint arXiv:2405.17527*, 2024.

Yupeng Zhou, Zhen Li, Chun-Le Guo, Song Bai, Ming-Ming Cheng, and Qibin Hou. Srformer: Permuted self-attention for single image super-resolution. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 12780–12791, 2023.

A A. DATASETS

The **Navier-Stokes (NS)** equation in (Li et al., 2020) models two-dimensional viscous incompressible fluid flow in vorticity form, considering varied viscosity coefficients that correspond to different Reynolds numbers. This setup introduces datasets spanning laminar, transitional, and turbulent regimes, thereby enabling the systematic evaluation of neural operators' ability to learn complex dynamical transitions and handle long-term chaotic evolution under varying physical constraints. Its mathematical formulation is as follows:

$$\begin{aligned} &\partial_t w(x,t) + u(x,t)\cdot\nabla w(x,t) \\ &\qquad = \nu\Delta w(x,t) + f(x), \quad x\in(0,1)^2,\ t\in(0,T], \end{aligned} \tag{13a}$$

$$\nabla\cdot u(x,t) = 0, \quad x\in(0,1)^2,\ t\in[0,T], \tag{13b}$$

$$w(x,0) = w_0(x), \quad x\in(0,1)^2, \tag{13c}$$

where $u \in C([0,T]; H^r_{\text{per}}((0,1)^2;\mathbb{R}^2))$ for any $r > 0$ is the velocity field, $w = \nabla\times u$ is the vorticity, $w_0 \in L^2_{\text{per}}((0,1)^2;\mathbb{R})$ is the initial vorticity, $\nu\in\mathbb{R}_+$ is the viscosity coefficient, and $f \in L^2_{\text{per}}((0,1)^2;\mathbb{R})$ is the forcing function. For $\nu =$ 1e-3, 1e-4, 1e-5, the datasets contain 4000/1000, 9000/1000, and 1000/200 samples for training/testing splits, respectively. The problem setup is given by the following equation:

$$\begin{aligned} &\text{NS } \nu = \text{1e-3}: \quad \mathcal{G}: u(\cdot,t)|_{t\in[0,10]} \mapsto u(\cdot,t)|_{t\in(10,50]}, \\ &\text{NS } \nu = \text{1e-4}: \quad \mathcal{G}: u(\cdot,t)|_{t\in[0,10]} \mapsto u(\cdot,t)|_{t\in(10,30]}, \\ &\text{NS } \nu = \text{1e-5}: \quad \mathcal{G}: u(\cdot,t)|_{t\in[0,10]} \mapsto u(\cdot,t)|_{t\in(10,20]}. \end{aligned} \tag{14}$$

The **Shallow-Water (SW)** equation in (Takamoto et al., 2022) describes two-dimensional free-surface flows governed by mass and momentum conservation. The system of hyperbolic PDEs models water depth and horizontal velocities, incorporating bathymetry effects and gravitational forcing. It preserves conservation laws even across hydraulic shocks, making it a suitable benchmark for evaluating neural operators in simulating complex, discontinuous flows such as tsunami propagation and flood dynamics:

$$\partial_t h + \nabla\cdot(h\boldsymbol{u}) = 0, \tag{15}$$

$$\partial_t(hu) + \nabla\cdot\left(\frac{1}{2}hu^2 + \frac{1}{2}g_r h^2\right) = -g_r h\nabla b. \tag{16}$$

where $h(x,t)$ is the water depth and $[-1,1]^2\times[0,5]$ is the domain. There are 900 training samples and 100 testing samples. The problem setup is given by the following equation:

$$\begin{aligned} &\text{SW-ST}: \quad \mathcal{G}: u(\cdot,t)|_{t\in[0,10]} \mapsto u(\cdot,t)|_{t\in(10,26]}, \\ &\text{SW-LT}: \quad \mathcal{G}: u(\cdot,t)|_{t\in[0,10]} \mapsto u(\cdot,t)|_{t\in(10,101]}. \end{aligned} \tag{17}$$

The **Gray-Scott Diffusion-Reaction (GSDR)** equation introduced by (Ohana et al., 2024) describes the spontaneous formation of ordered chemical patterns through coupled reaction and diffusion, characterized by an initial rapid change followed by convergence to stable or dynamic steady states. It is used to assess neural operators in simulating complex spatiotemporal self-organization and long-term pattern evolution. Its mathematical formulation is as follows:

$$\begin{aligned} \frac{\partial A}{\partial t} &= \delta_A\Delta A - AB^2 + f(1-A), \\ \frac{\partial B}{\partial t} &= \delta_B\Delta B + AB^2 - (f+k)B. \end{aligned} \tag{18}$$

where A and B are a set of coupled reaction–diffusion equations describing two chemical species, and the two parameters f and k control the "feed" and "kill" rates in the reaction, respectively. There are 1080 training samples and 120 testing samples. The problem setup is given by the following equation:

$$\begin{aligned} \text{GSDR}: \quad &\mathcal{G}: \{A(\cdot,\cdot,t), B(\cdot,\cdot,t)\}|_{t\in[0,10]} \mapsto \\ &\{A(\cdot,\cdot,t), B(\cdot,\cdot,t)\}|_{t\in(10,21]}. \end{aligned} \tag{19}$$

The **Fitzhugh-Nagumo Diffusion-Reaction (FNDR)** equation extends the classical diffusion-reaction system to a two-dimensional spatial domain, modeling the nonlinear coupling between an activator $u(t, x, y)$ and an inhibitor $v(t, x, y)$. This system is widely used to describe biological pattern formation and excitable media. Its mathematical formulation is given by:

$$\begin{aligned} \frac{\partial u}{\partial t} &= D_u \Delta u + R_u(u, v), \\ \frac{\partial v}{\partial t} &= D_v \Delta v + R_v(u, v), \end{aligned} \tag{20}$$

where $\Delta = \partial_{xx} + \partial_{yy}$ is the 2D Laplacian operator, D_u and D_v are the diffusion coefficients for the activator and inhibitor, respectively. The reaction terms follow the Fitzhugh-Nagumo kinetics:

$$R_u(u, v) = u - u^3 - k - v, \qquad R_v(u, v) = u - v, \tag{21}$$

with $k = 5 \times 10^{-3}$, $D_u = 1 \times 10^{-3}$, and $D_v = 5 \times 10^{-3}$. Initial conditions are drawn from a standard normal distribution $u(0, x, y) \sim \mathcal{N}(0, 1.0)$, and no-flow Neumann boundary conditions are imposed: $\partial_n u = \partial_n v = 0$ on the domain boundary. The dataset contains 1,000 samples, with 900 samples used for training and 100 for testing, each discretized on a 64×64 spatial grid with 101 time steps, including both u and v fields. Spatial discretization employs the finite volume method, and time integration uses the fourth-order Runge-Kutta method. The problem setup is defined as:

$$\begin{aligned} \text{FNDR}: \quad & \mathcal{G} : \{u(\cdot, \cdot, t), v(\cdot, \cdot, t)\}|_{t\in[0,10]} \mapsto \\ & \{u(\cdot, \cdot, t), v(\cdot, \cdot, t)\}|_{t\in(10,101]}. \end{aligned} \tag{22}$$

The **Airfoil** and the **Cylinder** from (Pfaff et al., 2020) simulate compressible and incompressible fluid dynamics, respectively. The Airfoil dataset models the aerodynamic fields around an airfoil cross-section, with the network predicting changes in momentum, density, and pressure. The Cylinder dataset represents flow around a cylinder, where the model predicts momentum change and pressure directly from node-type-augmented Eulerian mesh data. There are 1000 training samples and 200 testing samples for Airfoil and Cylinder. The problem setup is given by the following equation:

$$\text{Airfoil}: \quad \mathcal{G} : u(\cdot, t)|_{t\in[0,10]} \mapsto u(\cdot, t)|_{t\in(10,40]}. \tag{23}$$

$$\text{Cylinder}: \quad \mathcal{G} : u(\cdot, t)|_{t\in[0,10]} \mapsto u(\cdot, t)|_{t\in(10,35]}. \tag{24}$$

B B. MODELS

Fourier Neural Operator (FNO). FNO (Li et al., 2020) is a representative neural operator that learns mappings between function spaces by parameterizing integral operators in the Fourier domain. Given an input function discretized on a uniform grid, FNO alternates between pointwise linear transformations in physical space and global convolutions in Fourier space. The latter are implemented by truncating high-frequency modes and learning complex-valued weights on retained Fourier coefficients, enabling efficient modeling of long-range spatial interactions with linear complexity in the number of grid points. FNO assumes structured and regular spatial discretizations, which limits its applicability to irregular geometries or scattered observations without additional preprocessing or coordinate transformations. The training hyperparameters for FNO2D and FNO3D are detailed in Table 6.

Geo-Fourier Neural Operator (Geo-FNO). Geo-FNO (Li et al., 2023) extends FNO to irregular geometries by introducing a learned deformation mapping from the physical domain to a uniform latent grid, where standard Fourier convolutions can be applied. Specifically, Geo-FNO learns a geometry-dependent coordinate transformation that aligns irregular meshes or non-Euclidean domains to a canonical latent space. The operator learning is then performed in this latent domain using Fourier layers, and the output is mapped back to the original geometry. This design preserves the efficiency of Fourier-based operators while enabling generalization to complex domains, at the cost of an additional geometry encoding and deformation learning stage. The training hyperparameters for Geo-FNO are detailed in Table 6.

The **Operator Transformer (OFormer)** OFormer (Li et al., 2022) is an attention-based neural operator architecture designed for learning solution operators of partial differential equations. OFormer

Table 6: Hyperparameters for FNO2D, FNO3D and Geo-FNO used in the experiments.

2***Parameter**	**ST-PDE**				
	NS V1e-3	**NS V1e-4**	**NS V1e-5**	**SW**	**RD**
Width	20				
Modes	8				
Layers	4				
Learning Rate	1×10^{-3}				
Optimizer	Adam				
Scheduler	StepLR				
Batch Size	8				
Epochs	500				
Parameters	414517				414846

Table 7: Hyperparameters for Oformer used in the experiments.

2***Parameter**	**ST-PDE**				
	NS V1e-3	**NS V1e-4**	**NS V1e-5**	**SW**	**RD**
Encoder Dim	96				
Decoder Dim	192				
Encoder Layers	5				
Decoder Layers	1				
Learning Rate	5×10^{-4}				
Optimizer	AdamW				
Scheduler	OneCycleLR				
Batch Size	16	8		4	8
Epochs	500				
Parameters	1851745				1852898

leverages self-attention and cross-attention mechanisms to achieve flexibility with respect to input and output discretizations. The model comprises three core components: (i) a self-attention-based input encoder that processes function samples on arbitrary point sets, (ii) a query encoder that encodes arbitrary output locations via random Fourier features, and (iii) a cross-attention module that transfers information from input locations to query locations, enabling evaluation at arbitrary query points independent of the input grid. The model adopts softmax-free attention variants, including Fourier-type and Galerkin-type attention, which can be interpreted as learnable integral operators and computed efficiently via matrix associativity. OFormer demonstrates competitive performance on standard PDE benchmarks, particularly in data-sufficient regimes, and exhibits strong generalization to irregular geometries and varying discretizations without retraining. The training hyperparameters for OFormer are detailed in Table 7.

Latent Neural Operator (LNO). Latent Neural Operator (LNO) (Wang & Wang, 2024) proposes to perform operator learning entirely in a learned low-dimensional latent space to improve computational and memory efficiency. LNO introduces a Physics-Cross-Attention mechanism that lifts input functions defined on arbitrary geometries into a compact latent representation by attending to a set of learnable latent tokens. The operator is then applied in the latent space using attention-based transformations, after which the output is decoded back to the physical domain. By decoupling the computational cost from the spatial resolution of the input, LNO significantly reduces memory usage and enables scalable operator learning on large or irregular domains. The training hyperparameters for LNO are detailed in Table 8.

Latent Mamba Operator (LaMO). Latent Mamba Operator (LaMO) (Tiwari et al., 2025) further advances latent-space operator learning by integrating state-space models (SSMs) with kernel integral formulations. Instead of relying on attention mechanisms, LaMO employs Mamba-style selective state-space layers to model long-range dependencies within the latent space. This design offers linear-time complexity with respect to sequence length and improved numerical stability compared to attention-based latent operators. By combining SSM-based temporal modeling with latent integral operators, LaMO achieves strong approximation accuracy while maintaining favor-

Table 8: Hyperparameters for LNO used in the experiments.

2*Parameter	ST-PDE				
	NS V1e-3	NS V1e-4	NS V1e-5	SW	RD
Modes	128(LNO-S) 256(LNO-L)				
Dim	128(LNO-S) 256(LNO-L)				
Heads	8				
Layers	3				
Blocks	4(LNO-S) 8(LNO-L)				
Learning Rate	1×10^{-3}				
Optimizer	AdamW				
Scheduler	OneCycleLR				
Batch Size	8				
Epochs	500				
Parameters(LNO-S)	812801				814210
Parameters(LNO-L)	5339649				5342466

Table 9: Hyperparameters for LaMO used in the experiments.

2*Parameter	ST-PDE				
	NS V1e-3	NS V1e-4	NS V1e-5	SW	RD
Dim	128(LaMO-S) 128(LaMO-L)				
Heads	8				
Patch_sizes	2				
Blocks	4(LaMO-S) 8(LaMO-L)				
Learning Rate	5×10^{-4}				
Optimizer	AdamW				
Scheduler	OneCycleLR				
Batch Size	8				
Epochs	500				
Parameters(LaMO-S)	1634306				1639938
Parameters(LaMO-L)	3130114				3135746

able computational scaling, particularly for long-horizon and high-resolution PDE problems. The training hyperparameters for LaMO are detailed in Table 9.

CALM-PDE. CALM-PDE (Hagnberger et al., 2025) introduces a continuous and adaptive latent-space encoder–decoder architecture specifically designed for time-dependent PDEs with irregular spatial sampling. The model constrains convolutional kernels to local ε-neighborhoods in the physical domain, ensuring locality and reducing unnecessary global interactions. In addition, CALM-PDE learns adaptive query locations that dynamically adjust receptive fields based on the input geometry and temporal context. This continuous formulation allows CALM-PDE to efficiently process scattered observations and evolving spatial structures, offering memory and runtime advantages over transformer-based latent models while preserving strong expressiveness for spatiotemporal dynamics. The training hyperparameters for CALM-PDE are detailed in Table 10.

Instead of jointly modeling space and time with a single monolithic operator, SimVP, TAU and PredFormer explicitly separate spatial and temporal processing, often employing multi-input multi-output designs. Such decoupling has been shown to effectively mitigate error accumulation in autoregressive rollouts while significantly reducing computational overhead.

SimVP. SimVP (Gao et al., 2022) is a purely convolutional encoder–decoder framework for video prediction that decouples spatial representation learning from temporal dynamics modeling. The spatial encoder extracts per-frame features using convolutional layers, while the temporal module operates on the encoded feature sequences using lightweight convolutional blocks. This separation simplifies optimization and enables efficient spatiotemporal modeling without attention or recurrence. Due to its simplicity and strong empirical performance, SimVP serves as a competitive baseline for spatiotemporal forecasting tasks, including PDE trajectory prediction on regular grids. The training hyperparameters for SimVP are detailed in Table 11.

Table 10: Hyperparameters for CALM-PDE used in the experiments.

2***Parameter**	**ST-PDE**			
	NS V1e-3	**NS V1e-4**	**NS V1e-5**	**SW RD**
Encoder Dim	$128 \rightarrow 256$			
Decoder Dim	$256 \rightarrow 128$			
Encoder Layers	3			
Decoder Layers	3			
Decoder Layers	3			
Learning Rate	1×10^{-3}			
Optimizer	AdamW			
Scheduler	CosineAnnealingLR			
Batch Size	8			
Epochs	500			
Parameters	2240751			|2252422

Table 11: Hyperparameters for SimVP used in the experiments.

2***Parameter**	**ST-PDE**			
	NS V1e-3	**NS V1e-4**	**NS V1e-5**	**SW RD**
Spatial Encoder Blocks	4			
Temporal Encoder Blocks	4			
Spatial Encoder Dim	16			
Temporal Encoder Dim	128			
Groups	8			
Learning Rate	1×10^{-3}			
Optimizer	AdamW			
Scheduler	OneCycleLR			
Batch Size	8			
Epochs	500			
Parameters	1847713			|1847874

Temporal Attention Unit (TAU). Temporal Attention Unit (TAU) (Tan et al., 2023) introduces a spatiotemporal attention mechanism specifically tailored for video and temporal sequence prediction. TAU combines temporal attention with spatial feature aggregation to explicitly model dependencies across frames while preserving spatial locality. The attention formulation is designed to be more efficient than full spatiotemporal self-attention by focusing on temporally relevant features, enabling improved long-term forecasting accuracy. TAU demonstrates strong performance in capturing temporal correlations in dynamic systems, making it applicable to time-dependent PDE modeling where temporal dependencies play a critical role. The training hyperparameters for TAU are detailed in Table 12.

Table 12: Hyperparameters for TAU used in the experiments.

2***Parameter**	**ST-PDE**			
	NS V1e-3	**NS V1e-4**	**NS V1e-5**	**SW RD**
Spatial Encoder Blocks	4			
Temporal Encoder Blocks	4			
Spatial Encoder Dim	16			
Temporal Encoder Dim	256			
Learning Rate	1×10^{-3}			
Optimizer	AdamW			
Scheduler	OneCycleLR			
Batch Size	8		|	4
Epochs	500			
Parameters	4155329			|4155490

Table 13: Hyperparameters for PredFormer used in the experiments.

2*Parameter	ST-PDE			
	NS V1e-3	NS V1e-4	NS V1e-5	SW RD
Dim	256			
Heads	8			
Dim_head	32			
Blocks	8			
Learning Rate	1×10^{-3}			
Optimizer	AdamW			
Scheduler	OneCycleLR			
Batch Size	8			
Epochs	500			
Parameters	5276432			5284640

Table 14: Hyperparameters for GEO-FNO used in the experiments for ST-PDEs with irregular geometries.

2*Parameter	ST-PDE	
	Airfoil	Cylinder
Width	20	
Modes	8	
Layers	8	
Learning Rate	1×10^{-3}	
Optimizer	Adam	
Scheduler	StepLR	
Batch Size	8	
Epochs	500	
Parameters	517584	517255

PredFormer. PredFormer (Tang et al., 2026) adapts transformer architectures for spatiotemporal prediction by factorizing attention operations across spatial and temporal dimensions. Instead of applying full spatiotemporal attention, PredFormer sequentially models spatial attention within frames and temporal attention across frames, significantly reducing computational complexity. Additionally, gating mechanisms are introduced to regulate information flow and enhance long-range dependency modeling without explicit recurrence. This factorized design allows PredFormer to scale to longer sequences and higher resolutions, making it suitable for complex spatiotemporal forecasting tasks. The training hyperparameters for PredFormer are detailed in Table 13.

Table 15: Hyperparameters for OFormer used in the experiments for ST-PDEs with irregular geometries.

2*Parameter	ST-PDE	
	Airfoil	Cylinder
Encoder Dim	96	
Decoder Dim	192	
Encoder Layers	5	
Decoder Layers	1	
Learning Rate	5×10^{-4}	
Optimizer	AdamW	
Scheduler	OneCycleLR	
Batch Size	8	
Epochs	500	
Parameters	1854820	1853667

Table 16: Hyperparameters for LNO used in the experiments for ST-PDEs with irregular geometries.

2***Parameter**	**ST-PDE**	
	Airfoil	**Cylinder**
Modes	128(LNO-S)	256(LNO-L)
Dim	128(LNO-S)	256(LNO-L)
Heads	8	
Layers	3	
Blocks	4(LNO-S)	8(LNO-L)
Learning Rate	1×10^{-3}	
Optimizer	AdamW	
Scheduler	OneCycleLR	
Batch Size	8	
Epochs	500	
Parameters(LNO-S)	817028	815619
Parameters(LNO-L)	5348100	5345283

Table 17: Hyperparameters for LaMO used in the experiments for ST-PDEs with irregular geometries.

2***Parameter**	**ST-PDE**	
	Airfoil	**Cylinder**
Dim	128(LNO-S)	128(LNO-L)
Heads	8	
Patch_sizes	2	
Blocks	4(LNO-S)	8(LNO-L)
Learning Rate	5×10^{-4}	
Optimizer	AdamW	
Scheduler	OneCycleLR	
Batch Size	8	
Epochs	500	
Parameters(LaMO-S)	1021020	1018331
Parameters(LaMO-L)	1963660	1960971

Table 18: Hyperparameters for CALM-PDE used in the experiments for ST-PDEs with irregular geometries.

2***Parameter**	**ST-PDE**	
	Airfoil	**Cylinder**
Encoder Dim	$128 \rightarrow 256$	
Decoder Dim	$256 \rightarrow 128$	
Encoder Layers	3	
Decoder Layers	3	
Decoder Layers	3	
Learning Rate	1×10^{-3}	
Optimizer	Adam	
Scheduler	StepLR	
Batch Size	8	
Epochs	500	
Parameters	2277132	2310845

For Airfoil, Cylinder, and other irregular geometry datasets, FNO2D, FNO3D, SimVP, TAU, and PredFormer are not directly applicable, whereas Geo-FNO, OFormer, CALM-PDE, LNO, and LaMO can be directly employed. The training hyperparameters for these models are detailed in Tables 14, 15, 18, 16, and 17, respectively.

C C. Slice Projection and Chunk-wise Gradient Back-propagation

To further evaluate the robustness and extensibility of the proposed MCANO framework, we extend it to challenging unstructured spatial domains, specifically airfoil and cylinder equations, and long-trajectory spatiotemporal PDE prediction tasks such as SW-LT and FNDR.

Unlike regular-grid settings, where spatial operators can naturally exploit structured neighborhoods, irregular computational meshes introduce nonuniform node distributions and topology inconsistencies, making conventional convolutional or Fourier-based operators less effective. To address this issue, we adopt a geometry-aware slice projection mechanism inspired by latent token aggregation strategiesWu et al. (2024); Tiwari et al. (2025).

Given an irregular mesh containing N spatial nodes with coordinates, $\mathbf{x}_i \in \mathbb{R}^{d_s}, \quad i = 1, \ldots, N$, and corresponding physical features $\mathbf{f}_i \in \mathbb{R}^{d_f}$, the input representation is first constructed as

$$\mathbf{z}_i^{(0)} = \phi_{\text{pre}}\left([\mathbf{x}_i, \mathbf{f}_i]\right), \tag{25}$$

where $\phi_{\text{pre}}(\cdot)$ denotes the preprocessing MLP and $[\cdot, \cdot]$ represents channel-wise concatenation. The resulting hidden embeddings are then projected into multiple attention heads:

$$\mathbf{Q} = W_q\mathbf{Z}, \quad \mathbf{V} = W_v\mathbf{Z}, \quad \mathbf{Z} \in \mathbb{R}^{N\times C}. \tag{26}$$

Instead of relying on fixed-grid neighborhood relations, MCANO dynamically projects irregular node features into a compact latent slice space. Specifically, for each head, the model predicts soft slice assignment weights:

$$\mathbf{S}_{i,g} = \frac{\exp\left(\frac{\psi(\mathbf{q}_i)_g}{\tau}\right)}{\sum_{g'=1}^{G}\exp\left(\frac{\psi(\mathbf{q}_i)_{g'}}{\tau}\right)}, \tag{27}$$

where G denotes the number of latent slices, τ is a learnable temperature parameter, and $\psi(\cdot)$ is a linear projection. The slice tokens are then obtained through weighted aggregation:

$$\mathbf{T}_g = \frac{\sum_{i=1}^{N}\mathbf{S}_{i,g}\mathbf{v}_i}{\sum_{i=1}^{N}\mathbf{S}_{i,g} + \epsilon}. \tag{28}$$

This operation effectively compresses irregularly distributed node features into a small set of geometry-aware latent tokens $\mathbf{T} \in \mathbb{R}^{G\times d_h}, \quad G \ll N$. The aggregated slice tokens are subsequently used to update the original node representations through cross-token interactions inside each block.

For long-trajectory PDEs, directly processing the entire temporal sequence requires constructing a complete dynamic computation graph across all T time steps. During backpropagation, all intermediate activations must be retained. Consequently, GPU memory consumption grows approximately linearly with trajectory length, which becomes prohibitive for long-term PDE evolution.

To overcome this limitation, we introduce a chunk-wise training strategy. The temporal sequence is partitioned into K smaller chunks:

$$\{\mathbf{u}_1, \ldots, \mathbf{u}_T\} = \bigcup_{k=1}^{K}\mathcal{C}_k, \tag{29}$$

where each chunk contains a fixed number of time steps:

$$|\mathcal{C}_k| = L_c. \tag{30}$$

Within each chunk, all forward passes share a single computation graph:

$$\mathbf{h}_{t+1} = \mathcal{F}_\theta(\mathbf{h}_t), \quad t \in \mathcal{C}_k, \tag{31}$$

and the loss is accumulated over the entire chunk:

$$\mathcal{L}_k = \sum_{t \in \mathcal{C}_k} \ell(\hat{\mathbf{u}}_t, \mathbf{u}_t). \tag{32}$$

Instead of performing backpropagation at every individual time step, gradients are computed only once per chunk $\nabla_\theta \mathcal{L}_k$. In our experiments, the chunk size is empirically set to 10, which provides a favorable balance between temporal dependency modeling and GPU memory consumption.

D D. Different Temporal Encoders

Temporal Conv MLP (TCMLP). Given a local temporal sequence $\mathbf{u_x} \in \mathbb{R}^{T_{\text{in}}}$ at spatial location $\mathbf{x}$, Temporal Conv MLP (TCMLP) operates independently across spatial positions and shares parameters globally. Specifically, the input sequence is first reshaped to a single-channel temporal signal and processed by a one-dimensional convolution layer:

$$\mathbf{z_x} = \mathrm{Conv1D}(\mathbf{u_x}), \tag{33}$$

where the convolution kernel has size 3 with padding 1, enabling local temporal interaction among neighboring time steps. This convolution serves as a shallow temporal smoothing and local mixing operator, capturing short-term temporal correlations without introducing recurrent states or attention mechanisms. The convolved sequence is then passed through a position-wise multilayer perceptron (MLP):

$$\mathbf{h_x} = \mathrm{MLP}(\mathbf{z_x}), \tag{34}$$

where the MLP consists of two fully connected layers with a hidden expansion ratio of 4 and a GELU nonlinearity. The MLP enhances representational capacity by enabling nonlinear interactions across the temporal dimension while preserving the original temporal resolution. Finally, a residual connection is applied:

$$\mathrm{TCMLP}(\mathbf{u_x}) = \mathbf{u_x} + \mathbf{h_x}, \tag{35}$$

which stabilizes training and facilitates identity mapping when temporal correction is unnecessary. From a computational perspective, TCMLP has a complexity of $\mathcal{O}(T_{\text{in}})$ per spatial location. By combining local temporal convolution with a lightweight MLP, TCMLP provides an effective and scalable temporal encoder suitable for large-scale spatiotemporal neural operator architectures.

Temporal Attention (TA). Given an input spatiotemporal field $\mathbf{u} \in \mathbb{R}^{B \times H \times W \times T}$, temporal attention (TA) encoder first applies global spatial pooling to obtain a compact temporal representation:

$$\bar{\mathbf{u}} = \frac{1}{HW} \sum_{i=1}^{H} \sum_{j=1}^{W} \mathbf{u}_{i,j} \in \mathbb{R}^{B \times T}. \tag{36}$$

This operation summarizes the global temporal evolution of the field while avoiding the quadratic complexity associated with attention over all spatial locations. The pooled temporal sequence is then projected into a latent embedding space via a linear embedding:

$$\mathbf{z} = \mathrm{Linear}_{\text{in}}(\bar{\mathbf{u}}) \in \mathbb{R}^{B \times T \times D}, \tag{37}$$

where D denotes the temporal embedding dimension. The resulting sequence is processed by a stack of Temporal Attention Blocks, each consisting of a multi-head self-attention layer followed by a position-wise feed-forward network with residual connections and layer normalization:

$$\mathbf{z} \leftarrow \mathbf{z} + \mathrm{MHA}(\mathrm{LN}(\mathbf{z})), \quad \mathbf{z} \leftarrow \mathbf{z} + \mathrm{MLP}(\mathrm{LN}(\mathbf{z})). \tag{38}$$

Unlike standard spatiotemporal transformers that apply attention over all space–time tokens, attention here is restricted to the temporal axis, yielding a computational complexity of

$$\mathcal{O}(B \cdot T^2 \cdot D), \tag{39}$$

which is independent of the spatial resolution $H \times W$.This design significantly reduces memory and runtime costs while preserving the ability to model long-range temporal dependencies. After temporal attention, the latent sequence is projected back to a scalar temporal modulation signal:

$$\tilde{\mathbf{u}} = \mathrm{Linear}_{\text{out}}(\mathbf{z}) \in \mathbb{R}^{B \times T}. \tag{40}$$

Table 19: Model Hyperparameter Configuration for MCANO.

	Model Hyperparameters						
Model Name	Dim	Depth	Patch Size	Heads	SSM Expansion Factor	MLP Ratio	SSM d_state
MCANO	128	4	2	8	2.0	4	64

Table 20: Training Hyperparameter Configuration for MCANO.

	Training Hyperparameters				
Model Name	Epochs	Learning Rate	Scheduler	max_lr	pct_start
MCANO	500	5×10^{-4}	OneCycleLR	5×10^{-4}	0.3

Overall, TA serves as a global temporal context encoder, capturing long-range temporal correlations at the sequence level, while delegating fine-grained spatial modeling to the neural operator. This decoupled design aligns well with the structure of spatiotemporal PDEs, where temporal evolution often exhibits global coherence across the spatial domain.

Temporal Mixer (TMixer). To efficiently model short-range temporal dependencies while maintaining minimal computational overhead, we adopt a Temporal Mixer (TMixer) that operates independently at each spatial location. Given a local temporal signal, $\mathbf{u_x} = [u(\mathbf{x}, t_1), \ldots, u(\mathbf{x}, t_T)] \in \mathbb{R}^T$, the mixer first constructs a shifted temporal sequence: $\mathbf{u_x^{shift}} = [u(\mathbf{x}, t_1), u(\mathbf{x}, t_1), \ldots, u(\mathbf{x}, t_{T-1})]$, which explicitly encodes first-order temporal adjacency. The original and shifted sequences are then linearly interpolated via a fixed mixing coefficient $\alpha \in [0, 1]$:

$$\tilde{\mathbf{u}}_\mathbf{x} = \alpha \mathbf{u}_\mathbf{x} + (1 - \alpha)\mathbf{u}_\mathbf{x}^{\text{shift}},$$

providing an explicit inductive bias toward local temporal continuity, analogous to a first-order temporal differencing or smoothing operator. The mixed sequence is subsequently processed by a lightweight position-wise MLP:

$$\mathbf{y}_\mathbf{x} = \text{MLP}(\tilde{\mathbf{u}}_\mathbf{x}) = \mathbf{W}_2\sigma(\mathbf{W}_1\tilde{\mathbf{u}}_\mathbf{x}), \tag{41}$$

where $\sigma(\cdot)$ denotes the GELU nonlinearity. A residual connection is applied to stabilize training:

$$\mathbf{h}_\mathbf{x} = \mathbf{y}_\mathbf{x} + \tilde{\mathbf{u}}_x. \tag{42}$$

Importantly, TimeMixMLP is applied independently at each spatial location and shares parameters across the spatial domain. Its computational complexity scales linearly with the temporal length T and spatial resolution HW:

$$\mathcal{O}(B \cdot T \cdot D), \tag{43}$$

making it significantly more efficient than attention-based temporal modeling.

From a modeling perspective, the TMixer can be interpreted as a learned local temporal update operator, combining explicit short-range temporal coupling with nonlinear channel mixing. This structure aligns well with the inductive biases of time-dependent PDEs, where local temporal evolution is often governed by low-order dynamics. As a result, TMixer offers a favorable trade-off between expressiveness and computational efficiency, serving as a strong lightweight alternative to temporal attention or state-space models in spatiotemporal neural operators.

Temporal Gated MLP (GMLP). Given a temporal signal at a fixed spatial location, $\mathbf{x} \in \mathbb{R}^T$, the input is first projected into a doubled feature space:

$$[\mathbf{u}, \mathbf{v}] = \mathbf{W}_1\mathbf{x} + \mathbf{b}_1, \quad \mathbf{u}, \mathbf{v} \in \mathbb{R}^T. \tag{44}$$

The vector $\mathbf{u}$ is passed through a learned gating projection:

$$\mathbf{g} = \mathbf{W}_g\mathbf{u} + \mathbf{b}_g, \tag{45}$$

which modulates the transformed signal $\mathbf{v}$ after nonlinearity:

$$\mathbf{y} = \mathbf{x} + \text{GELU}(\mathbf{v}) \odot \mathbf{g}. \tag{46}$$

This structure resembles a simplified gated linear unit (GLU), but is applied along the temporal dimension rather than across feature channels. The gating mechanism enables the model to adaptively control the contribution of different temporal components, allowing it to suppress noisy or less informative time steps while amplifying salient temporal patterns. Unlike attention-based approaches, the Temporal Gated MLP (GMLP) introduces no explicit pairwise interactions between time steps, instead relying on learned global mixing and gating over the entire temporal window.

Table 21: Model Hyperparameter Configuration for MCANOP during Pretraining and Finetuning.

	Model Hyperparameters						
Model Name	Dim	Depth	Patch Size	Heads	SSM Expansion Factor	MLP Ratio	SSM d_state
MCANO	256	12	2	8	2.0	4	64

Table 22: Training Hyperparameter Configuration for MCANOP during Pretraining and Finetuning.

	Training Hyperparameters				
Phase	Epochs	Learning Rate	Scheduler	max_lr	pct_start
Pretraining	500	5×10^{-4}	OneCycle	5×10^{-4}	0.3
Finetuning	500	1×10^{-3}	OneCycle	1×10^{-3}	0.2

E E. Training details for MCANO and MCANOP

For MCANO, we adopt the model hyperparameter settings in Table 19, resulting in MCANO having 3.8M parameters, and adopt the training hyperparameter configuration in Table 20.

For the pretrained version of MCANO, namely MCANOP, we scale up the model parameters to enhance the model's representational capacity. Specifically, we increase the embedding dimension from 128 to 256 and the number of Transformer layers from 4 to 12, while keeping the patch size, number of attention heads, SSM expansion factor, MLP ratio, and SSM d_state unchanged. This parameter expansion enables the model to capture more complex temporal patterns and dependencies during pretraining on large-scale time-dependent multi-PDEs, thereby providing a stronger foundation for downstream fine-tuning tasks. The model hyperparameter settings for this pretrained model are shown in Table 21, and the training hyperparameters for training this model are shown in Table 22. For fine-tuning on downstream tasks, we adjust the training hyperparameters to better adapt to specific datasets. Specifically, we increase the learning rate to accommodate the randomly initialized projection layers and the absence of RevIN adaptation, enabling faster convergence. Additionally, we reduce the scheduler's pct_start to adopt a more aggressive warmup strategy, allowing the randomly initialized projection layers to enter the learning state more rapidly.

F F. Theoretical Analysis of Temporal Polynomial Lifting

In this section, we provide a theoretical justification for the polynomial lifting strategy used in TAM.

Consider a fixed spatial location $\mathbf{x} \in \Omega$. The temporal trajectory is

$$u_{\mathbf{x}}(t) = u(\mathbf{x}, t), \qquad t \in [0, 1]. \tag{47}$$

For a fixed spatial location $\mathbf{x} \in \Omega$, we assume that the temporal trajectory

$$u_{\mathbf{x}}(t) = u(\mathbf{x}, t), \qquad t \in [0, 1],$$

belongs to the Sobolev space $H^s([0, 1])$ with $s > \frac{1}{2}$:

$$u_{\mathbf{x}} \in H^s([0, 1]), \qquad s > \frac{1}{2}. \tag{48}$$

By Sobolev embedding,

$$H^s([0, 1]) \hookrightarrow C([0, 1]), \tag{49}$$

and therefore every temporal trajectory is continuous.

Let

$$t_i = \frac{i - 1}{T_{\text{in}} - 1}, \qquad i = 1, \ldots, T_{\text{in}}, \tag{50}$$

denote the normalized observation times.

The polynomial lifting used in TAM is

$$\Psi_K(\mathbf{u}) = \text{vec}\left([u(t_i)t_i^k]_{i=1,\ldots,T_{\text{in}};\, k=0,\ldots,K}\right), \tag{51}$$

where

$$\mathbf{u} = (u(t_1), \ldots, u(t_{T_{\text{in}}})).$$

The lifted representation belongs to $\mathbb{R}^{T_{\text{in}}(K+1)}$.

Unlike polynomial regression, TAM does not explicitly estimate polynomial coefficients. Instead, Ψ_K acts as a structured feature reparameterization that exposes correlations between the temporal signal and polynomial basis functions.

F.1 Approximation Property

The following result shows that smooth temporal trajectories admit accurate polynomial approximations.

Let $u \in H^s([0,1])$ with $s > \frac{1}{2}$. Then there exists a polynomial $P_K \in \Pi_K$ of degree at most K such that

$$\|u - P_K\|_{L^\infty([0,1])} \leq CK^{-s+\frac{1}{2}}, \tag{52}$$

where C depends only on u and s.

The result follows from classical Jackson-type approximation inequalities for Sobolev functions together with the embedding $H^s([0,1]) \hookrightarrow C([0,1])$ for $s > \frac{1}{2}$.

Proposition F.1 shows that the approximation error decreases polynomially with K. Therefore, sufficiently smooth temporal trajectories can be approximated with increasing accuracy in the finite-dimensional polynomial space Π_K as K grows.

F.2 Relation to Polynomial Approximation

Proposition F.1 shows that sufficiently smooth temporal trajectories can be accurately represented within a finite-dimensional polynomial space. The polynomial lifting adopted in TAM is motivated by this observation.

For the sampled temporal trajectory

$$\mathbf{u} = (u(t_1), \ldots, u(t_{T_{\text{in}}})), \tag{53}$$

TAM constructs the lifted feature

$$\Psi_K(\mathbf{u}) = \text{vec}\left([u(t_i)t_i^k]_{i=1,\ldots,T_{\text{in}};\, k=0,\ldots,K}\right). \tag{54}$$

The following proposition characterizes a basic property of the lifting operator. The lifting operator Ψ_K preserves all original temporal observations. In particular, the $k = 0$ component satisfies

$$u(t_i)t_i^0 = u(t_i), \qquad i = 1, \ldots, T_{\text{in}}. \tag{55}$$

Therefore, the original temporal trajectory is contained in the lifted representation.

Since $t_i^0 = 1$ for every sampling point, the entries corresponding to the zeroth-order basis are exactly

$$(u(t_1), \ldots, u(t_{T_{\text{in}}})).$$

Hence the original temporal observations appear explicitly as a subset of the lifted features.

Combining Proposition F.1 and Proposition F.2, TAM can be interpreted as a structured feature lifting mechanism inspired by polynomial approximation theory. The factors $\{t^k\}_{k=0}^K$ explicitly expose low-order temporal modes that characterize smooth temporal evolution, while retaining the complete original temporal history. Consequently, the subsequent MLP operates on a feature space that contains both the raw observations and their polynomially modulated counterparts.

Importantly, TAM does not perform explicit polynomial regression nor estimate polynomial coefficients. Instead, it enriches the temporal representation by exposing correlations between the observed history and polynomial basis functions before spatial operator learning. This provides an inductive bias toward smooth temporal dynamics while leaving the neural operator backbone unchanged.

G G. SUPPLEMENTARY EXPERIMENTAL RESULTS

Due to page limitations, additional experimental results are provided in the appendix to complement the main text and strengthen the credibility of our experiments.

The Necessity of MCANOP. Models pre-trained on large-scale datasets are capable of capturing generic feature representations, which is crucial for mitigating the data scarcity issue in downstream tasks. Moreover, the well-initialized parameters provided by pre-training can accelerate model convergence and effectively prevent overfitting when training on limited data.

To demonstrate this, we conduct experiments on the NS dataset, using MCANOP-17M as the baseline model, and evaluate three training schemes: training from scratch, pre-training, and fine-tuning after pre-training. According to the experimental results in Table 23, based on MCANOP-17M, training from scratch achieves the worst accuracy, while fine-tuning after pre-training achieves the best. Notably, the accuracy of training from scratch is even inferior to that of MCANO-3.8M, indicating that limited data causes overfitting on an overly large model. However, through pre-training, the model acquires the fundamental motion laws in the CFD domain, and when fine-tuned, it demonstrates accuracy far superior to that of task-specific models. Therefore, we adopt the pre-trained weights of MCANOP as the starting point for our fine-tuning experiments.

Table 23: Experimental results on the NS $\nu = 1\text{e-}5$ dataset under three training schemes.

Training Scheme	**L2RE**
Training from scratch	0.0602
Pre-training only	0.0460
Fine-tuning after pre-training	**0.0258**

Comparison with Larger Transformer-based PDE foundation models. Table 24 and Table 25 compare MCANOP with representative Transformer-based PDE foundation models under zero-shot and fine-tuning settings, respectively. Despite having only 17M parameters, MCANOP achieves competitive or superior performance compared with substantially larger Transformer-based models, demonstrating the effectiveness of the proposed spatiotemporal modeling design and pretraining strategy.

For the NS $\nu = 1\text{e-}3$ dataset, MCANOP-17M achieves a zero-shot L2RE of 0.0032, outperforming DPOT-7M (0.0095), DPOT-30M (0.0131), and DPOT-500M (0.00528), while approaching the performance of the much larger DPOT-1.03B (0.00229). This indicates that MCANOP with only 17M parameters can achieve comparable accuracy to a billion-scale Transformer-based model. After fine-tuning, MCANOP further improves the L2RE to 0.0022, surpassing all Transformer-based baselines except DPOT-500M-500 (0.00323) and achieving better accuracy than models with up to 500M parameters.

For the more challenging NS $\nu = 1\text{e-}4$ and NS $\nu = 1\text{e-}5$ datasets, where the dynamics exhibit stronger nonlinear evolution and more complex flow structures, MCANOP maintains strong scalability. In the zero-shot setting, MCANOP-17M achieves 0.0226 and 0.0459 L2RE on NS $\nu = 1\text{e-}4$ and NS $\nu = 1\text{e-}5$, respectively, outperforming DPOT-7M and DPOT-30M while approaching DPOT-122M and DPOT-500M. After fine-tuning, MCANOP-17M reaches 0.0108 on NS $\nu = 1\text{e-}4$, which is comparable to DPOT-500M-500 (0.0104), despite using less than 4% of its parameters. For NS $\nu = 1\text{e-}5$, MCANOP achieves 0.0258, remaining competitive with substantially larger Transformer-based models.

The strong performance on NS datasets can be attributed to the synergy between the spatiotemporal-aware architecture and the multi-PDE pretraining strategy. Unlike Transformer-based PDE models that implicitly mix temporal evolution with spatial representation learning through token interactions, MCANOP employs the Temporal-Awareness Module (TAM) to explicitly extract local temporal evolution patterns before spatial tokenization. This design provides a more suitable inductive bias for fluid dynamics, where the future state is strongly determined by the temporal evolution history at each spatial location. Meanwhile, the Spatial-Awareness Mamba (SA-Mamba) layers adap-

Table 24: Zero-Shot Comparison with Pre-Trained Models in Terms of L2RE.

Dataset	NS V1e-3	NS V1e-4	NS V1e-5	SW-LT	FNDR
DPOT-7M	0.0095	0.0606	0.0976	0.00560	0.0321
MPP-7M	–	–	–	0.0066	0.0168
DPOT-30M	0.0131	0.0442	0.0553	0.00657	0.0379
MPP-30M	–	–	–	0.0024	0.0112
Unisolver-33M	0.0061	0.0336	0.0417	0.0045	0.0439
DPOT-122M	0.00474	0.0285	0.0409	0.00290	0.0292
MPP-400M	–	–	–	0.00220	**0.0098**
DPOT-500M	0.00528	0.0274	0.0550	0.00233	0.0232
DPOT-1.03B	**0.00229**	**0.0131**	**0.0174**	0.00199	0.0191
MCANOP-17M	0.0032	0.0226	0.0460	**0.0012**	0.0169

Table 25: Fine-Tuned Accuracy Comparison with Pre-Trained Models in Terms of L2RE.

Dataset	NS V1e-3	NS V1e-4	NS V1e-5	SW-LT	FNDR
DPOT-7M-200	0.00729	0.0431	0.0511	0.00280	0.0194
DPOT-30M-200	0.00680	0.0342	0.0449	0.00224	0.0171
DPOT-122M-200	0.00427	0.0144	0.0255	0.00218	0.0142
DPOT-500M-200	0.00383	0.0117	0.0235	0.00197	0.0158
DPOT-7M-500	0.00580	0.0367	0.0520	0.00241	0.0148
DPOT-30M-500	0.00437	0.0237	0.0322	0.00235	0.0129
DPOT-122M-500	0.00335	0.0126	0.0229	0.00227	0.0103
DPOT-500M-500	0.00323	**0.0104**	**0.0213**	0.00170	0.00739
MCANOP-17M-500	**0.0022**	0.0108	0.0258	**0.00078**	**0.00358**

tively propagate information along physically meaningful scan paths, enabling efficient modeling of long-range dependencies and complex flow structures such as vortical interactions.

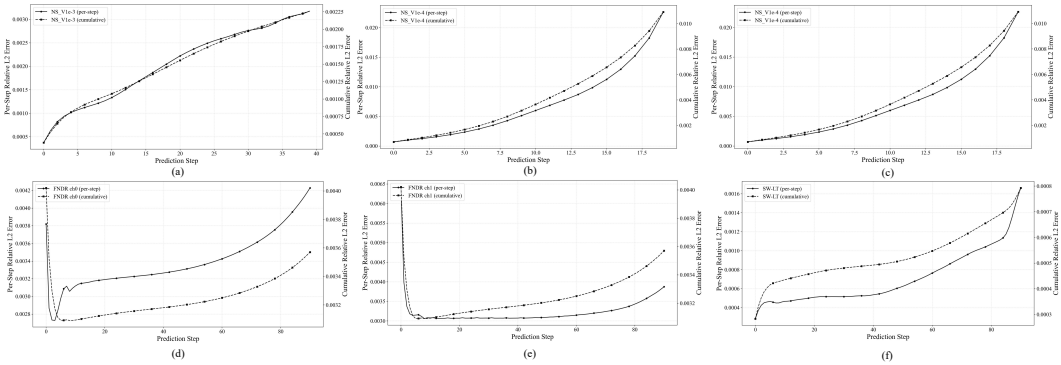

Figure 4: Cumulative error curves of the fine-tuned MCANOP-17M model on five datasets over time.

For the SW-LT dataset, MCANOP demonstrates the strongest advantage. The zero-shot MCANOP-17M achieves an L2RE of 0.0012, outperforming all Transformer-based baselines, including DPOT-1.03B (0.00199) and MPP-400M (0.00220). After fine-tuning, MCANOP further reduces the error to 0.00078, significantly surpassing DPOT-500M-500 (0.00170). Compared with standard Transformer attention, which treats all spatial tokens uniformly, the dynamic scan mechanism in MCANOP can better preserve directional propagation patterns and wave-like structures commonly observed in shallow-water dynamics.

For the FNDR dataset, MCANOP achieves zero-shot and fine-tuned L2REs of 0.0169 and 0.00358, respectively. Although MPP-400M obtains slightly better zero-shot performance (0.0098), MCANOP substantially outperforms other models with comparable or smaller sizes and achieves the best fine-tuning accuracy among all compared methods. This demonstrates that MCANOP can effectively adapt to nonlinear pattern formation and multi-scale interactions in reaction-diffusion systems.

Overall, these results demonstrate that MCANOP achieves a significantly better parameter-performance trade-off than existing Transformer-based PDE foundation models. The superior efficiency originates from the joint design of architecture and pretraining strategy. Unlike Transformer-based approaches that rely on global attention to model spatiotemporal dependencies, MCANOP

explicitly separates temporal evolution modeling and spatial interaction learning through TAM and SA-Mamba. TAM provides a lightweight temporal lifting mechanism to expose local evolution patterns, while SA-Mamba captures global physical interactions with adaptive and content-aware propagation.

Furthermore, random sub-trajectory sampling, teacher-forcing-based autoregressive pretraining, and RevIN-based normalization enable stable learning across heterogeneous PDE systems with substantial distribution shifts. During fine-tuning, the selective parameter initialization strategy preserves the learned physical evolution patterns in SA-Mamba layers while allowing newly initialized temporal and projection layers to adapt to dataset-specific variables. Together, these designs allow a 17M-parameter MCANOP model to achieve accuracy comparable to Transformer-based models with hundreds of millions to over one billion parameters.

Cumulative error analysis of MCANOP over time. In the field of spatiotemporal prediction, the significant accumulation of errors over time in autoregressive forecasting has long been a challenging issue. We present the cumulative error curves of the fine-tuned MCANOP-17M model on five datasets, as shown in Figure 4. Overall, MCANOP exhibits stable long-term prediction behavior, with gradually increasing cumulative relative L_2 errors over extended rollout horizons.

Interestingly, the FNDR (reaction-diffusion) datasets exhibit a different error evolution pattern from the other benchmarks, where the per-step relative L_2 errors first decrease and then increase during rollout. This phenomenon does not indicate a temporary improvement of prediction accuracy, but is mainly attributed to the intrinsic nonlinear evolution characteristics of reaction-diffusion systems and the definition of relative error. Specifically, FNDR dynamics typically undergo three stages: pattern initiation, nonlinear amplification, and saturation. During the early rollout stage, instability-driven pattern formation rapidly increases the magnitude of the ground-truth solution. Although the absolute prediction discrepancy may remain or gradually increase due to spatial phase shifts and pattern localization errors, the growth of the denominator in the relative L_2 error, dominates the metric behavior, resulting in a decreasing relative error. As the reaction-diffusion patterns approach a saturated regime, the magnitude of the solution becomes relatively stable, while autoregressive prediction errors continue accumulating due to small structural deviations. Consequently, the relative error gradually increases in later rollout steps. A similar behavior is observed across different FNDR channels, suggesting that this trend originates from the physical dynamics of reaction-diffusion systems rather than model instability.

In contrast, the Navier-Stokes and shallow-water datasets mainly exhibit monotonic error growth, as their solution magnitudes evolve differently and do not experience the same rapid pattern-amplification process. These results demonstrate that MCANOP maintains robust long-horizon forecasting capability across diverse dynamical regimes.

To further evaluate the generalization capability of the pretrained MCANOP-17M model, we conduct zero-shot out-of-distribution (OOD) experiments on two unseen CFD datasets, including the compressible Navier–Stokes (CNS) equations from PDEBench and the incompressible Navier–Stokes equations from PDEArena. These datasets are not involved during pretraining, providing a challenging evaluation of whether MCANOP can transfer the learned spatiotemporal representations to different physical systems and governing equations.

Specifically, we select the $M = 0.1, \eta = 0.1$ configuration of the compressible Navier–Stokes dataset from PDEBench CNS, which describes compressible fluid dynamics involving coupled density, velocity, and pressure fields. Unlike the PDEs used during pretraining, this dataset introduces additional physical variables (four coupled fields, including density ρ, two velocity components v_x and v_y, and pressure p) and compressibility effects, requiring the model to capture complex interactions among multiple physical quantities. We also evaluate MCANOP on PDEArena-NS1/2, which models two-dimensional incompressible Navier–Stokes flows and requires predicting velocity, pressure, and density fields over long-term temporal evolution. This experiment further examines the capability of MCANOP to adapt to PDEs with different physical constraints and multi-variable outputs.

As shown in Table **??**, MCANOP-17M consistently achieves the best performance among all compared neural operators on both OOD benchmarks. On PDEBench CNS, MCANOP obtains an L2 relative error of 0.0112, outperforming most existing operator learning approaches and achieving

comparable performance with much larger pretrained models. On PDEArena, MCANOP achieves an error of only 0.0372, significantly reducing the prediction error compared with previous neural operators.

Notably, MCANOP reaches this performance on PDEArena after only 80 training epochs, while other approaches typically require substantially longer optimization schedules. This rapid convergence demonstrates the effectiveness of the proposed Mamba-based architecture and the designed temporal- and spatial-awareness enhancement modules in capturing long-range dependencies and complex evolution patterns in dynamical systems. The results indicate that the pretrained MCANOP model learns transferable spatiotemporal representations from diverse PDEs, enabling efficient adaptation to unseen fluid dynamics problems and accelerating convergence when applied to new physical systems.

Overall, these OOD evaluations demonstrate that MCANOP is not merely memorizing specific PDE trajectories during pretraining, but instead learns generalizable representations associated with the underlying spatiotemporal dynamics. Such transferability allows MCANOP to effectively extend across different physical regimes, variable configurations, and governing equations.

H H. Supplementary Visualization Results

Due to the limited space in the main manuscript, we provide additional visualization results in the following pages, including the long-term prediction results and error maps of MCANOP, as well as the qualitative comparisons between MCANO and other baselines on both regular and irregular geometry datasets.

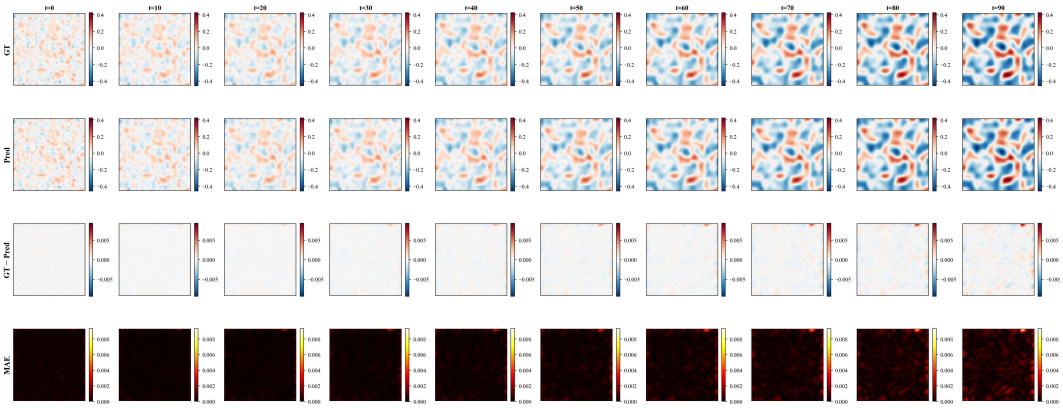

Figure 5: Visualization of MCANOP after fine-tuning on FNDR (Channel 0), showing from top to bottom: ground truth, model predictions, difference between ground truth and predictions, and MAE visualization.

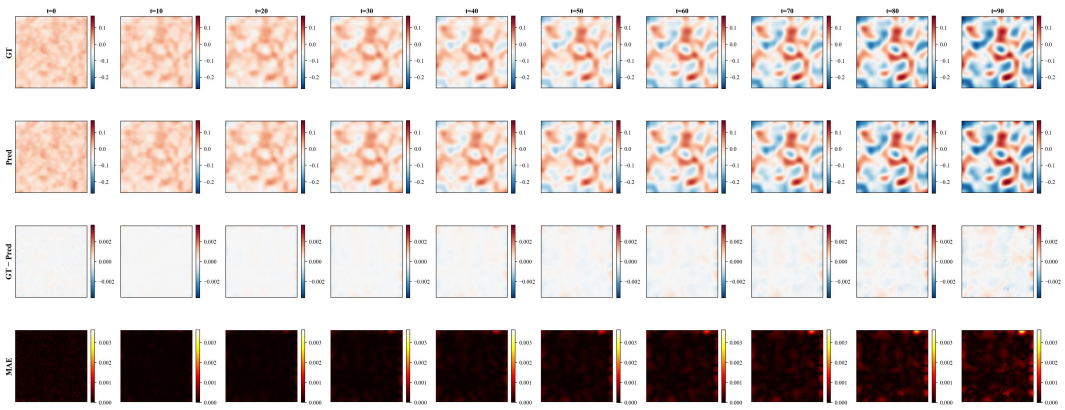

Figure 6: Visualization of MCANOP after fine-tuning on FNDR (Channel 1), showing from top to bottom: ground truth, model predictions, difference between ground truth and predictions, and MAE visualization.

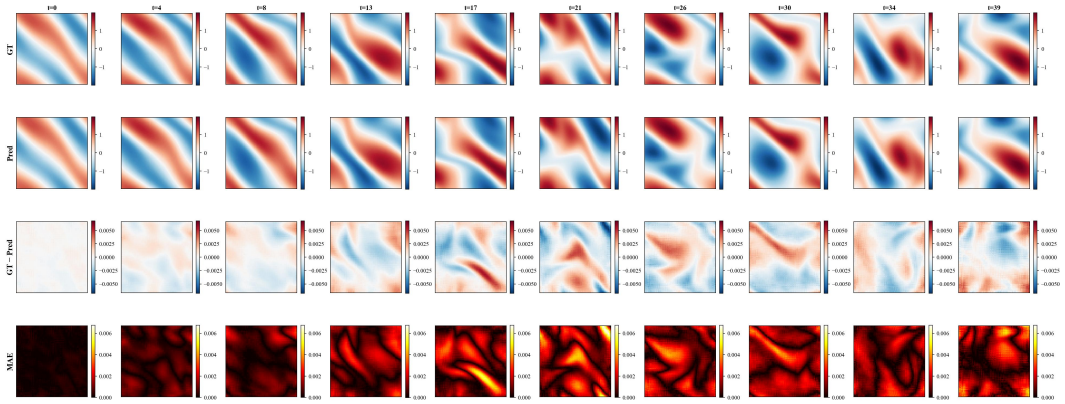

Figure 7: Visualization of MCANOP after fine-tuning on NS $\nu = 1\text{e-}3$, showing from top to bottom: ground truth, model predictions, difference between ground truth and predictions, and MAE visualization.

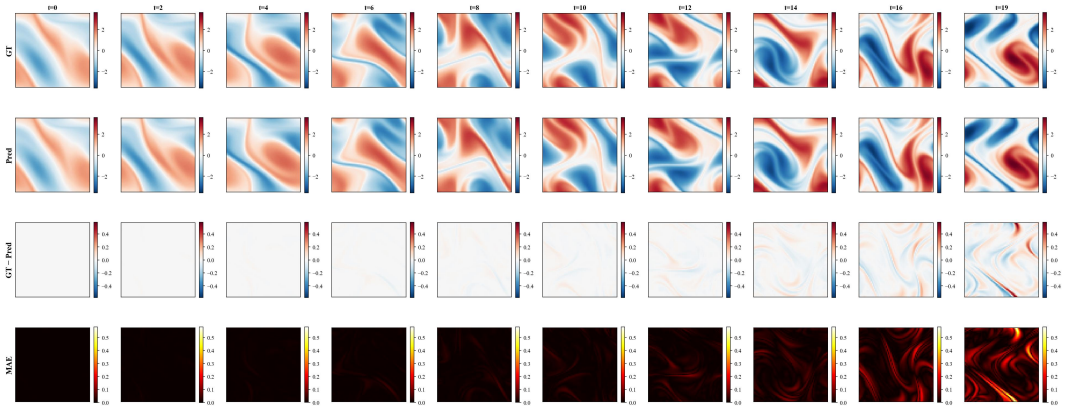

Figure 8: Visualization of MCANOP after fine-tuning on NS $\nu = 1\text{e-}4$, showing from top to bottom: ground truth, model predictions, difference between ground truth and predictions, and MAE visualization.

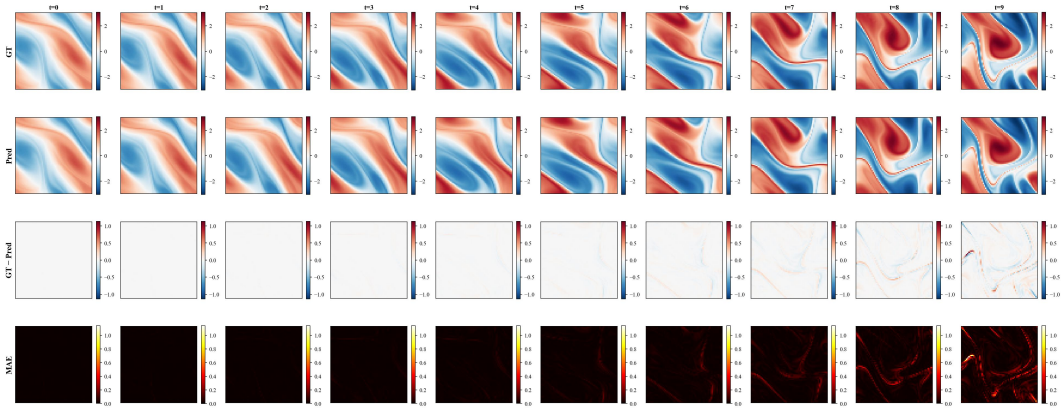

Figure 9: Visualization of MCANOP after fine-tuning on NS $\nu = 1\text{e-}5$, showing from top to bottom: ground truth, model predictions, difference between ground truth and predictions, and MAE visualization.

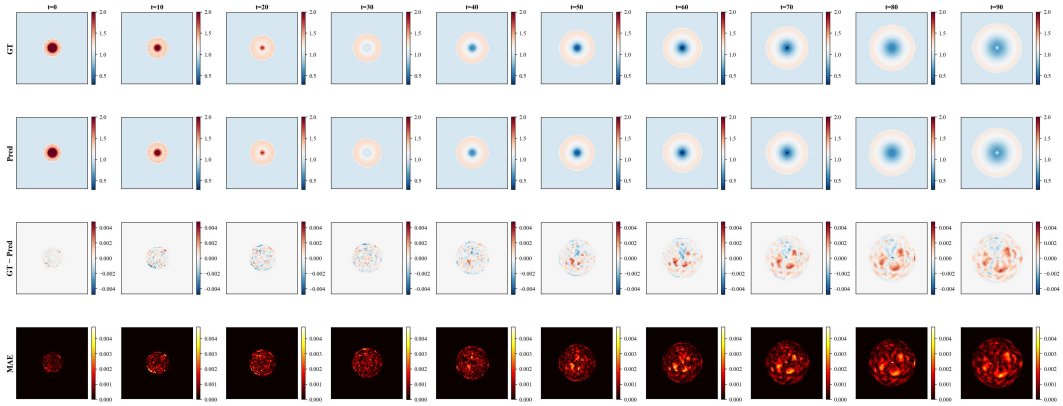

Figure 10: Visualization of MCANOP after fine-tuning on SW-LT, showing from top to bottom: ground truth, model predictions, difference between ground truth and predictions, and MAE visualization.

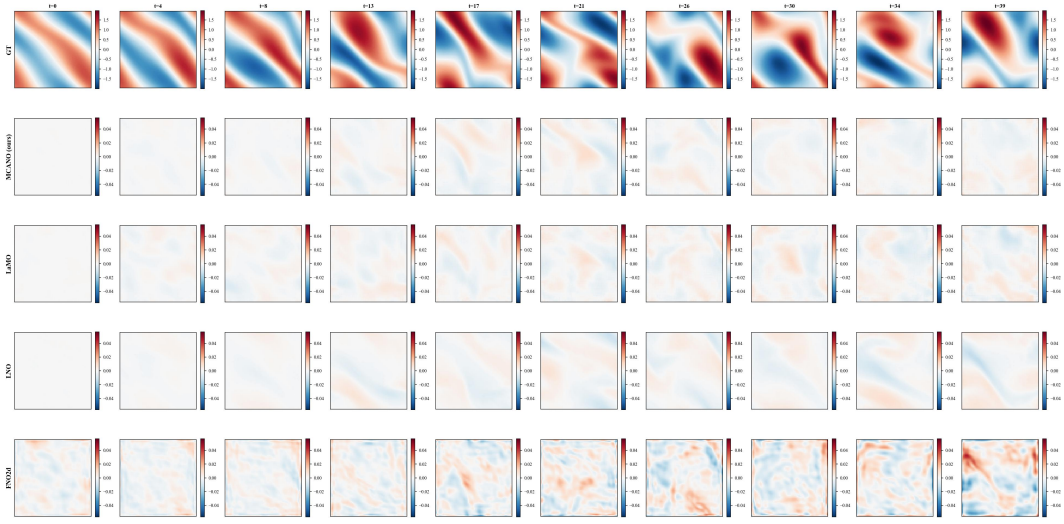

Figure 11: Performance comparison of different models on the regular NS $\nu = 1\text{e-}3$ dataset.

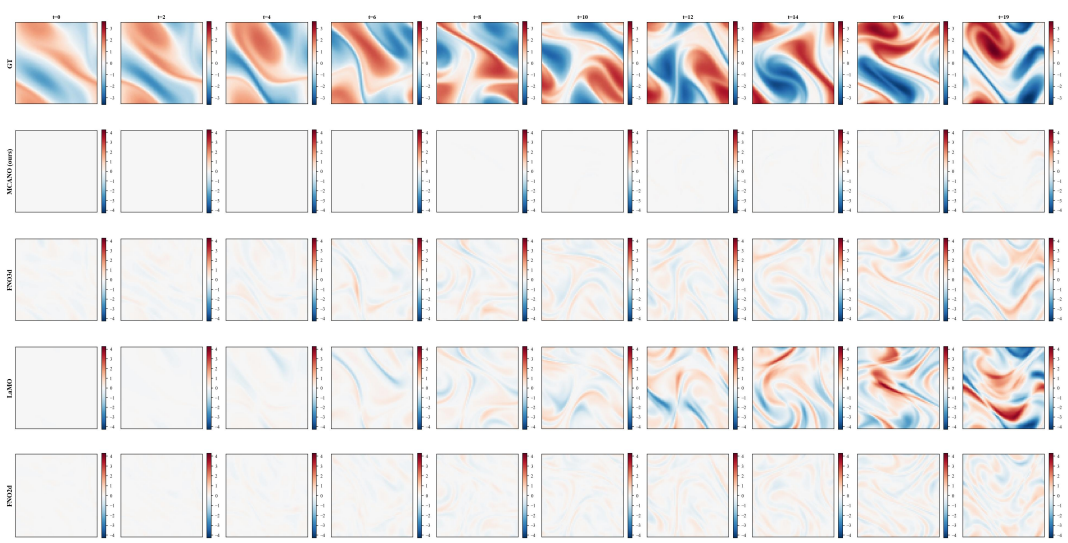

Figure 12: Performance comparison of different models on the regular NS $\nu = 1\text{e-}4$ dataset.

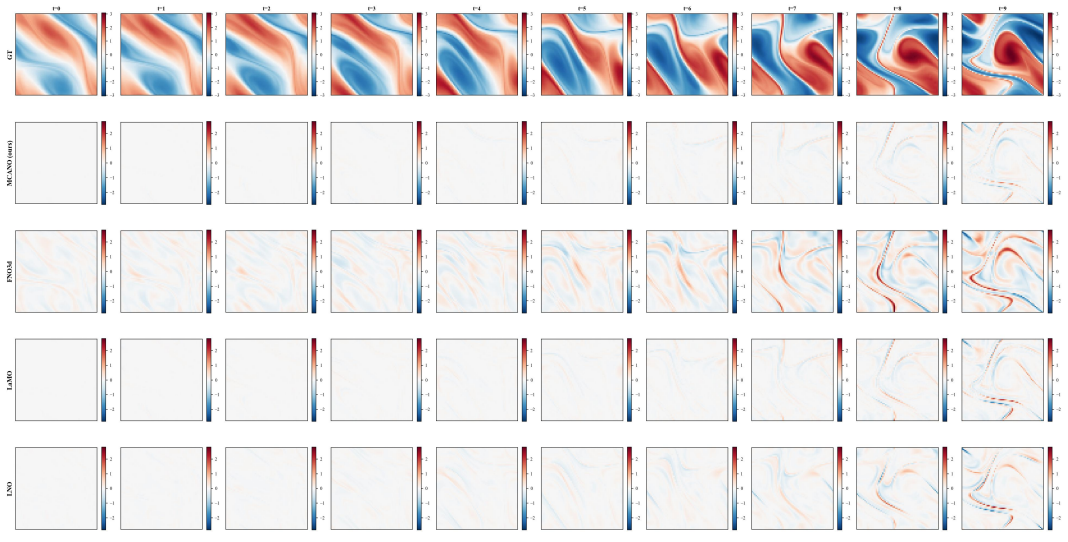

Figure 13: Performance comparison of different models on the regular NS $\nu = 1\text{e-}5$ dataset.

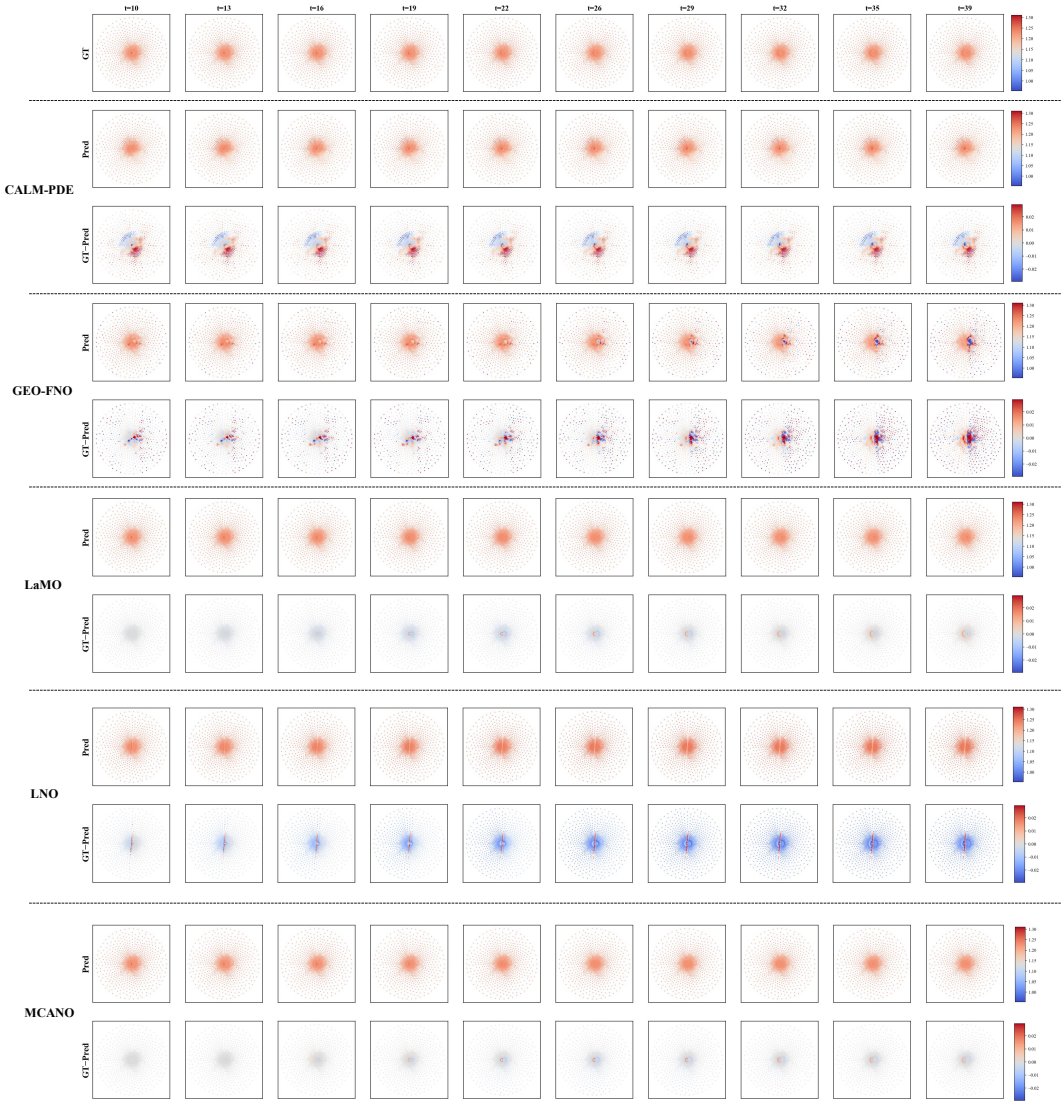

Figure 14: Performance comparison of different models on the irregular Airfoil dataset across density.

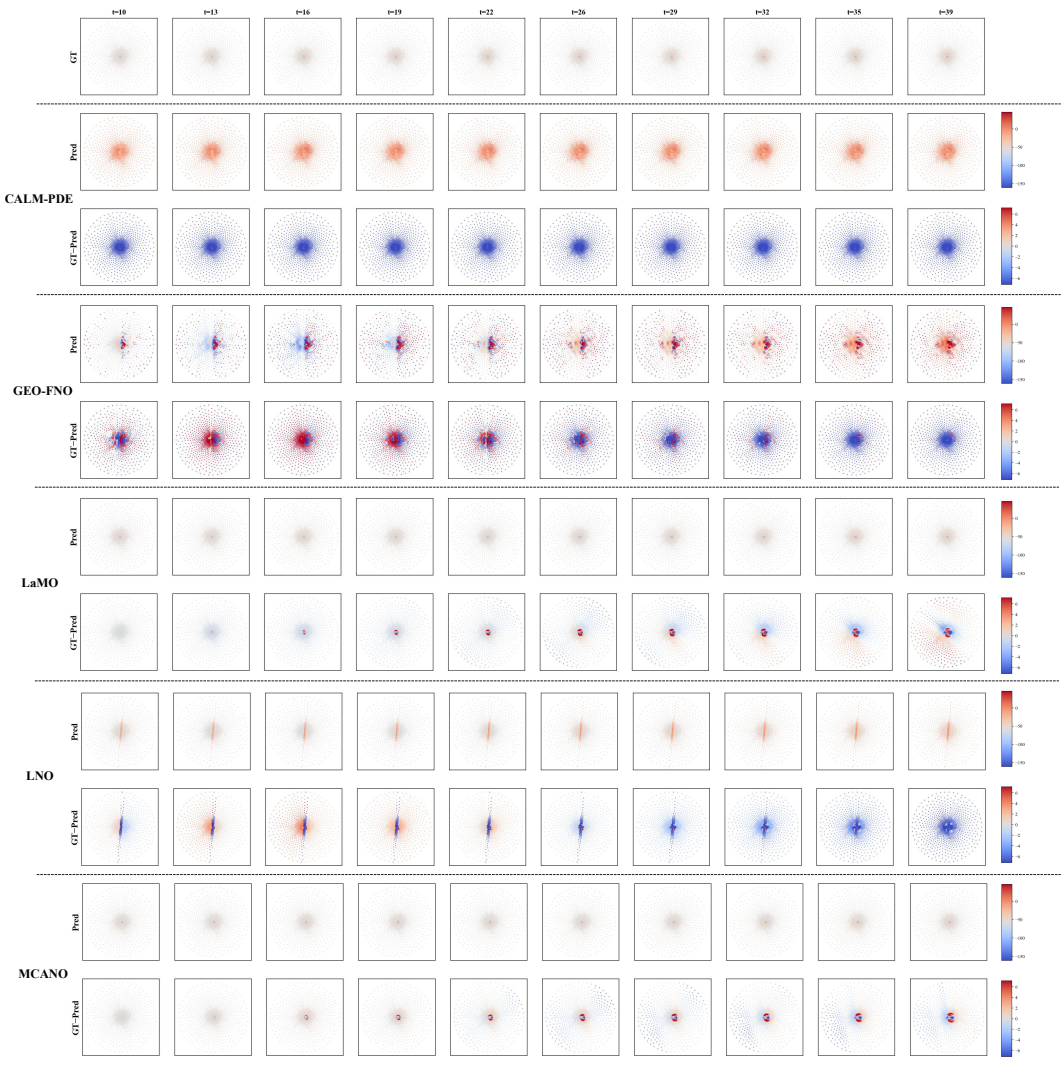

Figure 15: Performance comparison of different models on the irregular Airfoil dataset across y-direction velocity.