ADE-SOLVER: OBSERVATION-ANCHORED DISENTANGLED EVOLUTION FOR SPATIOTEMPORAL PDE SOLVING

Anonymous authors
Paper under double-blind review

ABSTRACT

Spatiotemporal partial differential equation (PDE) solving is fundamental to the efficient modeling of dynamic systems whose states evolve across space and time. However, many existing methods reconstruct complete target fields from latent representations without explicitly retaining spatial structures present in the observations, requiring the model to recover existing information while learning its future changes. Moreover, different evolution effects often coexist but are typically modeled jointly, obscuring how spatial redistribution and physical value variation should contribute to target construction. To address these challenges, we propose ADE-Solver, an observation-anchored spatiotemporal PDE solver that retains the latest observation as a shared physical anchor and models future evolution through spatial transfer and state variation. Specifically, we expose changes between adjacent observations to provide explicit evolution cues, learn representations of the two evolution factors, and functionally differentiate them during solution construction: spatial transfer modifies the sampling coordinates of the anchor, while state variation adjusts its physical values. The resulting evolution factors are applied to the shared anchor to generate all target states in parallel, shifting target generation from complete field reconstruction toward evolution from observed physical states. Experiments on four PDE benchmarks demonstrate that ADE-Solver achieves average improvements of 33.81% in RMSE and 33.06% in relative L_2 error over the strongest baselines. The code is available at `https://anonymous.4open.science/r/ADE-Solver`.

1 INTRODUCTION

Spatiotemporal partial differential equations (PDEs) describe the evolution of physical fields in fluid dynamics, reaction-diffusion systems, and many other scientific applications Takamoto et al. (2022). Efficiently solving these equations is essential for scientific simulation, design optimization, and system control. However, classical numerical solvers often require fine spatial discretization and repeated temporal integration, leading to substantial computational costs at high resolution or for repeated simulations. Learning-based PDE solvers provide an appealing alternative by learning solution mappings from observed trajectories and amortizing the cost of repeated numerical computation Azizzadenesheli et al. (2024); Lu et al. (2021); Kovachki et al. (2023); Li et al. (2021).

Learning-based PDE solvers have progressively advanced from general operator learning toward richer modeling of spatiotemporal dynamics. DeepONet Lu et al. (2021) learns nonlinear operators through branch and trunk networks, while FNO Li et al. (2021) parameterizes integral operators in Fourier space. Later methods strengthen multiscale modeling through U-FNO Wen et al. (2022), U-NO Rahman et al. (2022), and M2NO Li et al. (2026). Recent attention-based solvers further improve spatiotemporal interaction modeling, including Transolver Wu et al. (2024), HPM Yue et al. (2025), and SAOT Zhou et al. (2026). Related studies have explored other forms of spatial structure preservation. Geo-FNO Li et al. (2023b) learns coordinate deformations for irregular geometries, while AROMA Serrano et al. (2024) preserves spatial structure in latent PDE representations. However, these developments mainly improve how geometry or spatial information is represented, whereas many spatiotemporal solvers still construct future states from learned latent features rather than directly using an observed physical state as the reference for

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.

target generation. Consequently, spatial structures already present in the latest observation must be encoded and reproduced through the learned mapping instead of being directly reused. *This raises the first challenge: how to retain observed physical information as an explicit reference when constructing future states.* More importantly, once the observed state is retained, its evolution toward future states involves different forms of change. A useful physical perspective is provided by the material derivative, which distinguishes local temporal variation from spatial transport Batchelor (2000); related advection-diffusion-reaction formulations similarly separate transport from local changes induced by reaction or source terms Hundsdorfer et al. (2003). This suggests that spatial redistribution and physical value variation should play different roles in target construction rather than being absorbed into a single latent update. Building on this perspective, we formulate future state construction around a shared observed anchor, where spatial transfer relocates observed structures and state variation modifies their physical values. However, simply predicting separate latent factors does not by itself establish their distinct roles in solution construction. *The second challenge is therefore how to separate complementary evolution effects and, more importantly, assign them distinct functional roles when transforming the observed state.*

To address these challenges, we propose **ADE-Solver**, an shared-observation-anchored disentangled evolution framework for spatiotemporal PDE solving. To directly preserve and reuse observed physical information during target construction, ADE-Solver retains the latest observation as a shared anchor for all target states. We characterize target evolution through **spatial transfer** and **state variation**, which play distinct functional roles during solution construction: spatial transfer modifies the sampling coordinates of the anchor to model spatial redistribution, while state variation adjusts its physical values. To better expose temporal changes in the observed sequence, we explicitly introduce differences between adjacent observations and learn distinct representations of these evolution factors. The learned factors are then applied to the shared observation anchor to construct all target states in parallel, shifting target generation from complete field reconstruction toward evolution from observed physical states.

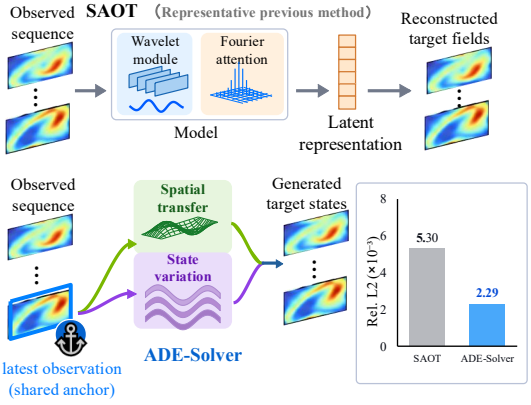

Figure 1: Comparison of target construction paradigms and relative L_2 error on Diffusion-Reaction.

We summarize our contributions below:

- We propose ADE-Solver, a spatiotemporal PDE solver that integrates temporal-difference augmentation, multiscale spatiotemporal representation learning, and parallel target generation within a unified observation-anchored framework.
- The core of ADE-Solver is an anchor-based functionally disentangled evolution mechanism: the latest observation is retained as a shared physical anchor, while spatial transfer and state variation act on its sampling coordinates and physical values, respectively, to construct future states.
- We conduct extensive experiments on simulated and real-world spatiotemporal PDE benchmarks. The results show that ADE-Solver consistently outperforms the strongest baseline, with up to 57.43% and 56.78% improvements in RMSE and relative L_2 error, respectively.

2 Problem Formulation and Preliminaries

Consider a spatiotemporal PDE defined on a spatial domain $\Omega \subset \mathbb{R}^{C_g}$, discretized into N spatial nodes, where C_g denotes the spatial dimension. At each discrete time step t, the physical state is represented as $\mathbf{u}_t \in \mathbb{R}^{N \times C_u}$, where C_u is the number of physical variables.

Given an observation window of H consecutive states $\mathbf{U}_{1:H} = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_H] \in \mathbb{R}^{H \times N \times C_u}$, the goal of spatiotemporal PDE solving is to generate the subsequent K states. We formulate this task as:

$$\mathcal{F}_\theta : \mathbf{U}_{1:H} \mapsto \widehat{\mathbf{U}}_{H+1:H+K}, \tag{1}$$

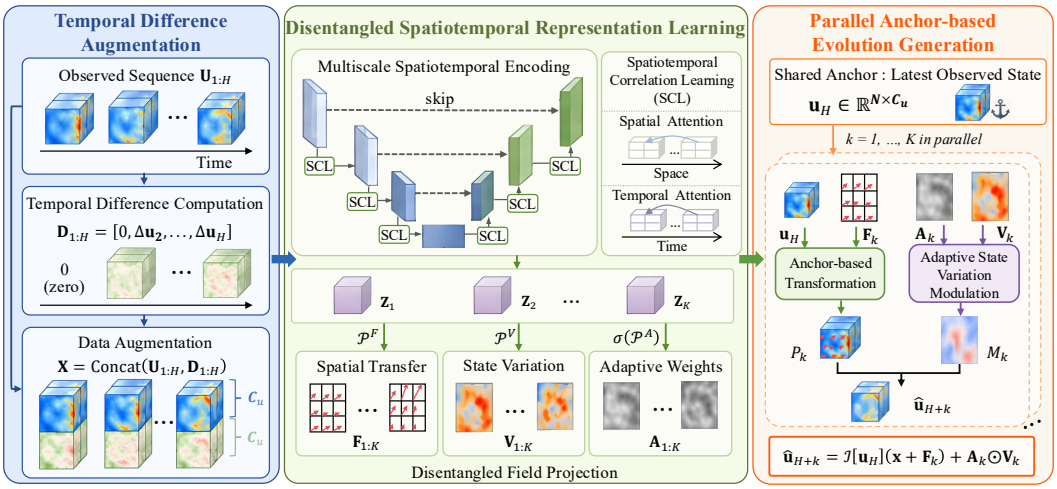

Figure 2: Overall framework of the proposed ADE-Solver.

where $\mathcal{F}_\theta$ denotes the parameterized spatiotemporal PDE solver and $\widehat{\mathbf{U}}_{H+1:H+K} \in \mathbb{R}^{K \times N \times C_u}$ denotes the generated target sequence.

3 METHOD

Figure 2 illustrates the overall framework of ADE-Solver, which comprises three core components: (1) **Temporal Difference Augmentation**, (2) **Disentangled Spatiotemporal Representation Learning**, and (3) **Parallel Anchor-based Evolution Generation**. At its core, ADE-Solver adopts an observation-anchored formulation in which spatial transfer and state variation play complementary roles in target construction.

3.1 TEMPORAL DIFFERENCE AUGMENTATION

The observed states mainly describe the fields at individual time steps, while the changes between consecutive states remain implicit. Such change information is important for our disentangled evolution modeling, since both spatial transfer and state variation are reflected in how the observed fields vary over time. Therefore, we augment the observation sequence with temporal differences, providing evolution cues that complement the spatial structures preserved in the original states.

Temporal Difference Computation. For the input sequence $\mathbf{U}_{1:H}$, we compute the temporal differences as follows:

$$\Delta\mathbf{u}_t = \mathbf{u}_t - \mathbf{u}_{t-1}, \qquad t = 2, \ldots, H. \tag{2}$$

To preserve temporal alignment with $\mathbf{U}_{1:H}$, we prepend a zero tensor for the first observation, forming $\mathbf{D} = [\mathbf{0}, \Delta\mathbf{u}_2, \ldots, \Delta\mathbf{u}_H] \in \mathbb{R}^{H \times N \times C_u}$.

Data Augmentation. We concatenate the observed states and their temporal differences along the physical variable dimension:

$$\mathbf{X} = \text{Concat}\left(\mathbf{U}_{1:H}, \mathbf{D}_{1:H}\right), \tag{3}$$

where $\mathbf{X} \in \mathbb{R}^{H \times N \times 2C_u}$. This parameter free augmentation preserves the original observations while supplementing them with explicit local changes, providing dynamic cues for subsequent disentangled evolution modeling.

3.2 DISENTANGLED SPATIOTEMPORAL REPRESENTATION LEARNING

Spatiotemporal PDE evolution involves different forms of change, including spatial transfer of structural patterns and variation in physical states. To better capture these complementary evolution factors, we employ a U shaped multiscale encoder to obtain target-indexed spatiotemporal representations for subsequent evolution field projection. At each scale, we use standard spatial and temporal attention, termed SCL for convenience, to capture correlations across space and target indices.

The augmented data $\mathbf{X}$ are first embedded into latent features and expanded along the temporal dimension to form K representations indexed by the target steps, denoted as $\mathbf{S}^{(0)}$. Then, we use multiscale spatiotemporal encoding to integrate spatial structures at fine scales with contextual patterns at coarse scales while modeling dependencies across target indices. We further use disentangled field projection to transform the resulting representation $\mathbf{Z}$ into functionally differentiated evolution fields that characterize spatial transfer and state variation for each target step.

Multiscale Spatiotemporal Encoding. We process $\mathbf{S}^{(0)}$ using a multiscale U-Net architecture with skip connections. The U-Net serves as the default backbone in our implementation, while other spatiotemporal feature extractors that produce target-indexed representations of the same form may be substituted without changing the subsequent field projection and anchor-based evolution generation. At each scale, Spatiotemporal Correlation Learning (SCL) models dependencies across spatial locations and target indices by sequentially applying spatial and temporal attention mechanisms. Given query ($\mathbf{Q}_a$), key ($\mathbf{K}_a$), and value ($\mathbf{V}_a$) projected from the input features, the basic attention operation in SCL is defined as:

$$\mathrm{Attn}_a = \mathrm{Softmax}\left(\frac{\mathbf{Q}_a\mathbf{K}_a^\top}{\sqrt{d}}\right)\mathbf{V}_a, \tag{4}$$

where $a \in \{s, t\}$ denotes spatial or temporal attention and d is the feature dimension. To capture relative temporal relations among target indices, temporal attention further incorporates a learnable relative temporal position bias into the attention scores before the softmax operation. With residual connections applied to both attention operations, the encoded representation is:

$$\mathbf{Z} = \mathcal{E}\left(\mathbf{S}^{(0)}\right) \in \mathbb{R}^{K\times N\times C_z}, \tag{5}$$

where $\mathcal{E}$ denotes the multiscale spatiotemporal encoding module and C_z is the latent feature dimension. Each slice $\mathbf{Z}_k$ is subsequently used to derive the evolution fields for target k.

Disentangled Field Projection. The material derivative provides an intuitive distinction between local temporal change and advective spatial change Batchelor (2000):

$$\frac{D\mathbf{u}}{Dt} = \underbrace{\frac{\partial \mathbf{u}}{\partial t}}_{\text{local temporal change}} + \underbrace{(\mathbf{v}\cdot\nabla)\mathbf{u}}_{\text{advective spatial change}}, \tag{6}$$

where $\mathbf{u}$ denotes the physical state field, D/Dt denotes the material derivative, $\mathbf{v}$ is a generic transport velocity, and ∇ denotes the spatial gradient operator. Inspired by this distinction, we use the term disentangled to describe their distinct functional roles in target construction: spatial transfer acts through coordinate transformation, whereas state variation acts through value modulation. The material derivative motivates the functional distinction rather than imposing an exact physical decomposition. Given $\mathbf{Z}$, independent pointwise projections derive the spatial transfer fields, state variation fields, and adaptive weighting matrices while preserving the temporal dimension.

Spatial transfer. We derive these fields as follows:

$$\mathbf{F}_{1:K} = \mathcal{P}^F(\mathbf{Z}), \tag{7}$$

where $\mathbf{F}_{1:K} \in \mathbb{R}^{K\times N\times C_g}$, and $\mathcal{P}^F$ denotes a convolutional operation. Each $\mathbf{F}_k$ specifies a sampling offset from the anchor in grid point units, describing spatial transfer toward target k.

State variation. We derive these fields as follows:

$$\mathbf{V}_{1:K} = \mathcal{P}^V(\mathbf{Z}), \tag{8}$$

where $\mathbf{V}_{1:K} \in \mathbb{R}^{K\times N\times C_u}$, and $\mathcal{P}^V$ is a convolutional operation. Each $\mathbf{V}_k$ represents changes in physical variable values rather than spatial displacement.

Adaptive weights. To account for spatially varying contributions of state variation, the adaptive weighting matrices are derived as follows:

$$\mathbf{A}_{1:K} = \sigma\left(\mathcal{P}^A(\mathbf{Z})\right), \tag{9}$$

where $\mathbf{A}_{1:K} \in [0,1]^{K\times N\times C_u}$, σ is the sigmoid function, and $\mathcal{P}^A$ is a convolutional operation. Each $\mathbf{A}_k$ adaptively gates state variation across spatial locations and physical variables.

Although $\mathbf{F}_k$ and $\mathbf{V}_k$ are derived through separate projections, their disentangled roles are further established through the subsequent evolution construction. Specifically, $\mathbf{F}_k$ governs spatial transfer by modifying the sampling coordinates of the observation anchor, whereas $\mathbf{V}_k$ captures state variation by adjusting physical variable values. This functional differentiation allows the learned representations to correspond to complementary aspects of PDE evolution.

3.3 Parallel Anchor-based Evolution Generation

To provide a common physical reference for evolution, all target states are directly constructed from the latest observed state $\mathbf{u}_H$, which serves as a shared anchor.

Anchor-based Transformation. To transfer spatial structures from the shared anchor to target index k, ADE-Solver establishes a sampling correspondence between each target location and the observation anchor: The transformation is defined as follows:

$$\mathbf{P}_k(\mathbf{x}) = \mathcal{I}\left[\mathbf{u}_H\right](\mathbf{x} + \mathbf{F}_k(\mathbf{x})), \qquad k = 1, \ldots, K, \tag{10}$$

where $\mathbf{P}_k$ denotes the transformed anchor state for target k, $\mathcal{I}$ denotes bilinear interpolation, $\mathbf{x}$ denotes a spatial location on the discretized grid, and $\mathbf{F}_k(\mathbf{x})$ is the corresponding sampling offset in grid point units. Note that $\mathbf{F}_k$ specifies a backward sampling offset from each target location to the anchor. Here, $F_k(x)$ is a backward sampling offset: each target location x samples the anchor at $y = x + F_k(x)$. For visualization, we reverse the arrow connecting this same pair of locations, drawing it from the anchor location y to the target location x. The displayed vector is therefore $x - y = -F_k(x)$, attached to y and indexed by the corresponding target location x. Since the shifted sampling coordinates may fall outside the spatial domain, we handle them according to the underlying boundary condition: periodic coordinates are wrapped to the correspondin on the opposite boundary, while nearest boundary padding is used for non-periodic boundaries. The transformations for all target indices can be evaluated in parallel.

Adaptive State Variation Modulation. To account for spatially and variable-dependent contributions of state variation, we modulate each state variation field using its corresponding adaptive weighting matrix:

$$\mathbf{M}_k(\mathbf{x}) = \mathbf{A}_k(\mathbf{x}) \odot \mathbf{V}_k(\mathbf{x}), \qquad k = 1, \ldots, K, \tag{11}$$

where $\odot$ denotes element-wise multiplication, and $\mathbf{M}_k \in \mathbb{R}^{N \times C_u}$ denotes the adaptively modulated state variation. This modulation allows the contribution of state variation to adapt across spatial locations and physical variables.

Parallel Solution Construction. The target solution is constructed by combining the spatially transferred anchor with the modulated state variation:

$$\widehat{\mathbf{u}}_{H+k}(\mathbf{x}) = \mathbf{P}_k(\mathbf{x}) + \mathbf{M}_k(\mathbf{x}), \qquad k = 1, \ldots, K. \tag{12}$$

The offset F_k is defined relative to the shared observation u_H, rather than to the preceding target state. Although the target representations interact through temporal attention, their final construction does not require sequential evaluation of predicted states.

3.4 Model Learning

The spatial transfer and state variation fields are learned without direct supervision, while the solution loss only constrains the generated target states. Therefore, we introduce regularization terms to directly constrain the learned evolution factors, encouraging smooth spatial transfer and moderate state variation.

Solution Loss. To supervise target solution generation, we define the solution loss as the mean squared error between predicted and reference states:

$$\mathcal{L}_{\text{sol}} = \text{Mean}\left[\left(\widehat{\mathbf{U}}_{H+1:H+K} - \mathbf{U}_{H+1:H+K}\right)^2\right]. \tag{13}$$

Spatial Transfer Regularization. We regularize the spatial transfer fields using smoothness and magnitude terms:

$$\mathcal{L}_{\text{smooth}} = \text{Mean}\left[\left(\nabla_x \mathbf{F}_{1:K}\right)^2 + \left(\nabla_y \mathbf{F}_{1:K}\right)^2\right], \tag{14}$$

$$\mathcal{L}_{\text{mag}} = \text{Mean}\left[\mathbf{F}_{1:K}^2\right]. \tag{15}$$

Here, ∇_x and ∇_y denote first-order spatial finite differences. Without regularization, learned offsets may exhibit abrupt spatial variations or excessive sampling displacements. Assuming coherent spatial evolution, $\mathcal{L}_{\text{smooth}}$ penalizes squared spatial offset gradients, encouraging neighboring locations to exhibit locally consistent transfers. Meanwhile, $\mathcal{L}_{\text{mag}}$ penalizes large offset magnitudes to discourage unnecessary sampling displacements, while allowing larger shifts when supported by the solution loss.

State Variation Regularization. To discourage unnecessarily large state variations, we introduce state variation regularization:

$$\mathcal{L}_{\text{var}} = \text{Mean}\left[\mathbf{V}_{1:K}^2\right]. \tag{16}$$

By penalizing the squared magnitude of $\mathbf{V}_{1:K}$, $\mathcal{L}_{\text{var}}$ imposes a quadratically increasing penalty on larger state variations. This discourages the variation field from capturing spurious local fluctuations through excessive state modifications, while preserving necessary value changes through target solution supervision.

Overall Objective. The complete training objective is:

$$\mathcal{L} = \mathcal{L}_{\text{sol}} + \lambda_s \mathcal{L}_{\text{smooth}} + \lambda_m \mathcal{L}_{\text{mag}} + \lambda_v \mathcal{L}_{\text{var}}, \tag{17}$$

where λ_s, λ_m, and λ_v control the spatial smoothness, transfer magnitude, and state variation regularization, respectively.

4 Experiments

We conduct extensive experiments to examine whether preserving observed structures and functionally disentangling evolution effects improve solution accuracy, and whether the learned factors exhibit their intended roles during target generation. In addition to comparisons with representative baselines, we conduct controlled studies involving persistence, anchor residual prediction, direct reconstruction, branch specific ablations, and sequential anchor generation, together with temporal, spatial, qualitative, and hyperparameter analyses.

4.1 Experiment Settings

Benchmarks. We evaluate ADE-Solver on four representative spatiotemporal PDE benchmarks spanning both simulated and real-world dynamics. The three simulated benchmarks, Diffusion-Reaction, Shallow Water, and Incompressible NS, are taken from PDEBench and divided into training, validation, and test sets with a 6:2:2 ratio Takamoto et al. (2022). For the real-world benchmark, Cylinder Flow is obtained from RealPDEBench and follows its predefined data split Hu et al. (2026). Further details of all benchmarks are provided in Appendix A.3.

Baselines. We compare ADE-Solver with two groups of representative baselines. (1) **Established methods**: DeepONet Lu et al. (2021), FNO Li et al. (2021), and CNO Raonić et al. (2023). (2) **Recent methods**: Transolver Wu et al. (2024), WDNO Hu et al. (2025), M2NO Li et al. (2026), HPM Yue et al. (2025), 3D-Unet Hu et al. (2026), and SAOT Zhou et al. (2026). For a fair comparison, all baselines are trained and evaluated under the same experimental protocol using their official implementations. Further details of the baseline methods are provided in Appendix A.4.

Implementations. As in previous works, we use the relative L_2 error and root mean square error (RMSE) to quantify the performance. For all methods, we use identical physical variables and data splits, preprocessing, observation and target windows, and evaluation samples. Each method takes 20 observed states as input and directly predicts the subsequent 20 target states. Each experiment is repeated seven times, and the average results for all methods are reported. Our evaluation focuses on observation conditioned PDE solving under standard benchmark settings, broader generalization to unseen parameters, resolutions, geometries, and longer horizons remains future work. All experiments are conducted on a machine with an AMD EPYC 7402 24-core CPU and an NVIDIA GeForce RTX 4090 (24 GB) GPU. Implementation details are provided in Appendix A.5.

4.2 Main Results and Analysis

Table 1 summarizes our experimental results across four challenging PDE benchmarks, with the best performance highlighted in **bold** and the second best underlined. ADE-Solver achieves the best performance on all benchmarks and metrics, with RMSE improvements over the strongest baselines ranging from 12.72% to 57.43%. On **Diffusion-Reaction**, ADE-Solver reduces RMSE

Table 1: Performance comparison on four PDE benchmarks. Marker * indicates the results are statistically significant (t-test with p-value < 0.01).

Method	Diffusion-Reaction		Shallow Water		Incompressible NS		Cylinder Flow	
	RMSE	Rel. L_2	RMSE	Rel. L_2	RMSE	Rel. L_2	RMSE	Rel. L_2
DeepONet	7.730e-2	8.628e-1	4.913e-3	4.589e-3	8.332e-2	9.630e-1	7.130e-2	1.528e-1
FNO	2.949e-3	3.900e-2	1.091e-3	9.720e-4	1.181e-2	1.320e-1	5.400e-2	9.083e-2
CNO	1.265e-3	1.639e-2	3.475e-3	3.283e-3	6.346e-3	6.465e-2	4.240e-2	8.970e-2
Transolver	2.660e-3	3.540e-2	1.172e-2	1.118e-2	1.030e-2	9.729e-2	1.093e-1	1.887e-1
WDNO	7.096e-2	7.843e-1	2.025e-2	1.937e-2	6.255e-2	7.493e-1	5.130e-2	1.666e-1
HPM	1.788e-3	2.292e-2	5.949e-3	5.631e-3	3.661e-3	3.523e-2	9.670e-2	9.209e-2
3D-Unet	5.400e-4	7.040e-3	1.360e-3	1.230e-3	1.685e-3	1.999e-2	6.141e-2	7.418e-2
M2NO	4.252e-3	4.982e-2	8.960e-4	8.420e-4	1.977e-3	2.210e-2	2.787e-2	9.545e-2
SAOT	4.040e-4	5.298e-3	1.175e-3	1.090e-3	1.141e-3	1.152e-2	9.747e-2	7.986e-2
ADE-Solver	**1.720e-4***	**2.290e-3***	**5.410e-4***	**4.930e-4***	**8.500e-4***	**9.111e-3***	**2.432e-2***	**6.444e-2***
Improvement	57.43%	56.78%	39.60%	41.40%	25.50%	20.94%	12.72%	13.13%

and relative L_2 error by 57.43% and 56.78% over SAOT, respectively, while on **Shallow Water** the corresponding improvements over M2NO are 39.60% and 41.40%. Both systems involve evolving spatial structures together with substantial changes in physical values, and the results support explicitly modeling spatial transfer and state variation during target construction. On **Incompressible NS**, ADE-Solver improves RMSE and relative L_2 error by 25.50% and 20.94% over SAOT, while on the real-world **Cylinder Flow** benchmark it achieves 12.72% and 13.13% improvements over the strongest baselines. These flow systems contain continuously evolving vortical or wake structures, and the consistent gains support evolving observed spatial patterns from a shared physical anchor rather than reconstructing each target field solely from latent representations. Notably, 3D-Unet shares the same backbone and nearly identical model size (22.96M vs. 23.00M) but directly reconstructs targets, further supporting the benefit of observation-anchored evolution.

Temporal continuity check. To examine whether the gains mainly arise from temporal continuity and direct reuse of the latest observation, we additionally evaluate a simple *Copy Last* reference with $\hat{\mathbf{u}}_{H+k} = \mathbf{u}_H$. As shown in Table 2, Copy Last produces substantially larger errors than ADE-Solver. This establishes that persistence alone is insufficient. Section 4.3 additionally compares against a learned residual predictor, separating the proposed evolution construction from the simpler strategy of retaining the observation and learning an additive update.

Table 2: Performance of the Copy Last reference.

	DR	SW	NS	CF
RMSE	1.7×10^{-2}	6.3×10^{-2}	3.7×10^{-3}	4.3×10^{-2}
Rel. L_2	2.1×10^{-1}	6.0×10^{-2}	3.8×10^{-2}	1.8×10^{-1}

4.3 Ablation Study

To evaluate the contribution of each component in ADE-Solver, we construct variants by removing different components. ***w/o* TDA** removes Temporal Difference Augmentation. ***w/o* DRL** retains $\mathbf{u}_H$ as the anchor but removes the two factor evolution representation, directly predicting a unified residual $\mathbf{R}_k$: $\hat{\mathbf{u}}_{H+k} = \mathbf{u}_H + \mathbf{R}_k$. ***w/o* AEG** retains separate spatial transfer and state variation predictions but removes their functionally differentiated anchor-based operations, directly combining the two components to construct each target state. ***w/o* DA** removes both DRL and AEG and directly reconstructs each target field from the latent representation: $\hat{\mathbf{u}}_{H+k} = \mathcal{P}_{\text{out}}(\mathbf{Z}_k)$. ***w/o* PA** replaces shared anchor parallel construction with sequential evolution using the previous prediction as the next anchor. As shown in Figure 3, ADE-Solver achieves the lowest RMSE on both benchmarks. The ***w/o* DA** variant exhibits the largest degradation, with RMSE increases of 98% and 81% on Diffusion Reaction and Incompressible NS, respectively. This variant returns to a reconstruction-based generation paradigm, where future fields are predicted solely from latent representations, demon-

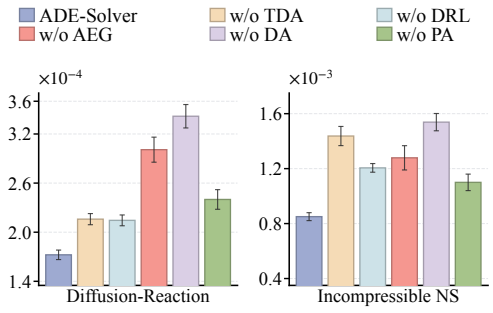

Figure 3: Ablation study results.

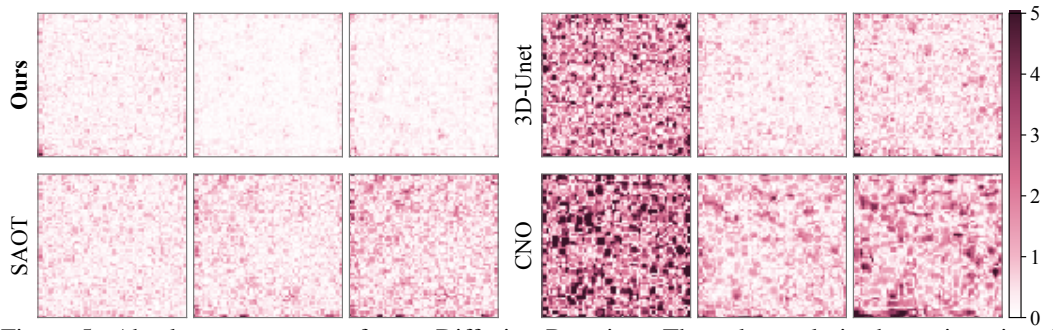

Figure 5: Absolute error maps of u on Diffusion-Reaction. The color scale is shown in units of 10^{-3}. For each method, the three columns correspond to $k = 1, 15$, and 20 from left to right.

strating the importance of observation-anchored evolution. The ***w/o* AEG** variant causes substantial degradation, with RMSE increases of 75% and 50%, respectively. This indicates that anchor-based generation is essential for applying learned evolution factors to transform the latest observed state. The ***w/o* DRL** variant produces smaller but consistent increases (22% and 42%), shows that simply retaining the latest observation and directly predicting relative changes is insufficient, supporting the benefit of disentangled evolution modeling. The ***w/o* TDA** variant increases RMSE by 22% and 70%, showing that temporal differences provide useful evolution cues for learning from observed sequences. The ***w/o* PA** variant consistently degrades performance on both benchmarks and supports the proposed shared-anchor parallel construction over sequential evolution. Appendix A.7 further isolates the two functional pathways by removing spatial transfer or state variation individually. Both removals degrade performance, supporting their complementary roles in target construction. Diffusion-Reaction is more sensitive to removing state variation (50.85% vs. 19.79%), consistent with its pronounced value changes. These results support functional differentiation rather than a unique physical decomposition. Removing adaptive weighting increases RMSE on both benchmarks, indicating that adaptive weighting further improves the modulation of state variation.

4.4 Performance Across Target Steps

To examine model's performance across target steps, we report stepwise RMSE on Diffusion-Reaction. As shown in Figure 4, the RMSE of ADE-Solver remains consistently lower than all baselines across the entire target window. While most baselines exhibit larger errors or increasing trends at longer target steps, ADE-Solver maintains a relatively stable error profile. This stable behavior is consistent with the observation-anchored evolution formulation. By constructing each target state from the same latest observation anchor, ADE-Solver preserves a consistent physical reference when modeling different evolution stages. Moreover, Diffusion Reaction involves continuous changes in physical values driven by reaction and diffusion processes. The explicit temporal cues and state variation modeling help ADE-Solver characterize such evolving dynamics.

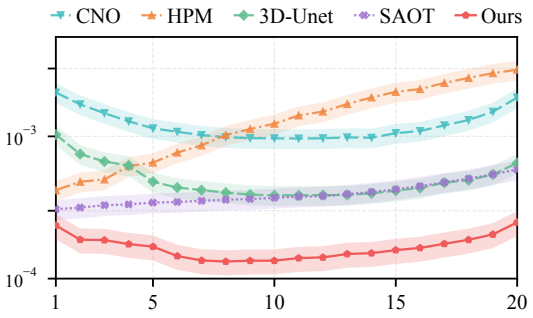

Figure 4: RMSE at each target step.

4.5 Spatial Error Visualization

As shown in Figure 5, ADE-Solver maintains consistently sparse and low-magnitude error patterns across all target steps under the same color scale. In contrast, SAOT exhibits more distributed errors as the prediction horizon increases, while 3D-Unet and CNO show stronger local errors with larger high error regions, especially at early target steps. These observations indicate that ADE-Solver achieves improved spatial fidelity throughout the target window rather than only reducing the averaged error. This behavior is consistent with the observation-anchored evolution formulation, where future states are generated by evolving the observed physical state instead of reconstructing complete fields solely from latent representations. Consequently, ADE-Solver better preserves spatial structures during target generation and produces more accurate solution fields.

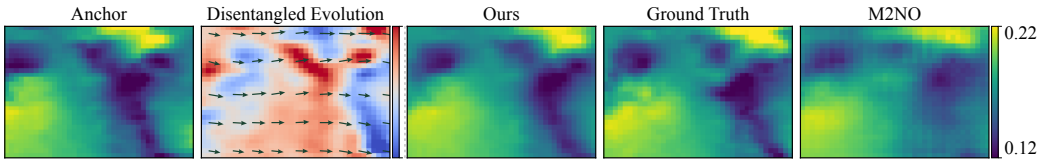

Figure 6: Learned factors in a downstream Cylinder Flow wake. Arrows point from anchor sampling locations $y = x + F_k(x)$ to their corresponding target locations x, with vectors $-F_k(x)$. The background shows the state variation.

4.6 Qualitative Analysis of Learned Evolution Factors

Figure 6 visualizes a representative downstream wake region of Cylinder Flow, where the observed structure undergoes both spatial displacement and local value changes. Compared with the anchor, the ground truth target shows clear changes in the position, shape, and intensity of the wake structures. For intuitive visualization, we reverse each target-to-anchor sampling arrow: for a target location x, we draw an arrow from its anchor sampling location $x+F_k(x)$ to x, with vector $-F_k(x)$. These arrows connect the same corresponding locations used in Eq. (10), but point from the anchor toward the target. Their overall rightward orientation is consistent with the displacement of the main wake structure. Meanwhile, the state variation field presents contrasting positive and negative regions around the evolving wake, indicating localized adjustments of physical values complementary to spatial transfer. After applying both factors to the shared anchor, ADE-Solver reproduces the main low-value wake structure and the upper-right high-value region more faithfully than M2NO, which exhibits stronger smoothing and weaker local structural contrast. Overall, these observations provide qualitative evidence that the two learned factors exhibit the intended complementary functional roles during target construction.

4.7 Hyperparameter Sensitivity

We evaluate the sensitivity of ADE-Solver to the regularization weights λ_s and λ_v on Diffusion Reaction. As shown in Figure 7, both metrics exhibit a consistent trend of first decreasing and then increasing, with the defaults achieving the best performance. Small regularization weights provide insufficient constraints, whereas overly large values limit necessary evolution changes. These results show appropriate regularization balances the flexibility and stability of the learned evolution representations.

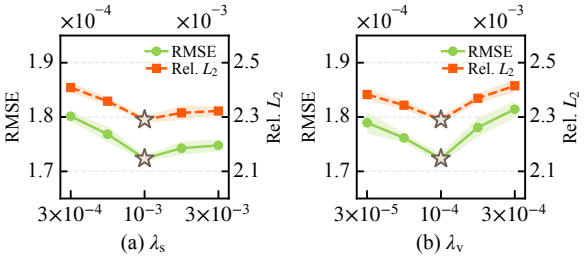

Figure 7: Hyperparameter sensitivity.

5 Conclusion

In this work, we propose ADE-Solver, an observation-anchored spatiotemporal PDE solver that constructs future states by evolving the latest observed physical state. The model functionally differentiates evolution into spatial transfer and state variation, which act on sampling coordinates and physical values, respectively. Experiments on simulated and real-world PDE benchmarks show consistent improvements over the evaluated baselines, while controlled comparisons with persistence, anchor residual prediction, direct reconstruction, branch specific variants, and sequential anchor generation further support the proposed formulation.

6 Related Work

Physics-informed neural networks incorporate governing equations and physical constraints into training Raissi et al. (2019); Jagtap & Karniadakis (2020); Yu et al. (2022). **Data-driven PDE solvers** learn solution mappings from observations through operator learning Lu et al. (2021); Li et al. (2021), multiscale modeling Raonić et al. (2023); Hu et al. (2025); Li et al. (2026), and attention-based representations Wu et al. (2024); Zhou et al. (2026); Yue et al. (2025). DeepLag Ma et al. (2024) incorporates tracked particle dynamics into Eulerian representations, while HelmFluid Xing et al. (2023) models fluid evolution through Helmholtz decomposition. See Appendix A.1 for further details.

AI USE STATEMENT

In this work, we used large language models (LLMs) solely for language polishing and improving the clarity and readability of the manuscript. LLMs were not used for research ideation, method design, experimental design, code generation, data analysis, or interpretation of results. All AI-assisted revisions were carefully reviewed by the authors to ensure technical accuracy and consistency with the intended meaning. We take full responsibility for the final content of this work.

ETHICS STATEMENT

This work focuses on methodological research for spatiotemporal PDE solving and does not involve human subjects, personally identifiable information, or sensitive personal data. All experiments are conducted on publicly available benchmark datasets. We are not aware of any direct ethical risks arising from the proposed method or the experimental setting. The authors are committed to research integrity and responsible reporting of the presented results.

REPRODUCIBILITY STATEMENT

We provide detailed descriptions of the proposed method, model architecture, and training objectives in the main paper. The experimental section specifies the datasets, evaluation metrics, baseline settings, and training configurations, while additional implementation details are provided in the Appendix. We also report the data preprocessing and dataset split strategies used in our experiments. These materials are intended to support reproducibility of the reported results.

REFERENCES

Kamyar Azizzadenesheli, Nikola Kovachki, Zongyi Li, Miguel Liu-Schiaffini, Jean Kossaifi, and Anima Anandkumar. Neural operators for accelerating scientific simulations and design. *Nature Reviews Physics*, 6(5):320–328, 2024.

G. K. Batchelor. *An Introduction to Fluid Dynamics*. Cambridge Mathematical Library. Cambridge University Press, 2000.

Johannes Brandstetter, Daniel Worrall, and Max Welling. Message passing neural pde solvers. *arXiv preprint arXiv:2202.03376*, 2022.

Zhongkai Hao, Zhengyi Wang, Hang Su, Chengyang Ying, Yinpeng Dong, Songming Liu, Ze Cheng, Jian Song, and Jun Zhu. GNOT: A general neural operator transformer for operator learning. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 12556–12569, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/hao23c.html.

Peiyan Hu, Rui Wang, Xiang Zheng, Tao Zhang, Haodong Feng, Ruiqi Feng, Long Wei, Yue Wang, Zhi-Ming Ma, and Tailin Wu. Wavelet diffusion neural operator. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=FQhDIGuaJ4.

Peiyan Hu, Haodong Feng, Hongyuan Liu, Tongtong Yan, Wenhao Deng, Tianrun Gao, Rong Zheng, Haoren Zheng, Chenglei Yu, Chuanrui Wang, Kaiwen Li, Zhi-Ming Ma, Dezhi Zhou, Xingcai Lu, Dixia Fan, and Tailin Wu. Realpdebench: A benchmark for complex physical systems with real-world data. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=y3oHMcoItR.

Willem Hundsdorfer, Jan Verwer, et al. *Numerical Solution of Time-Dependent Advection-Diffusion-Reaction Equations [electronic resource]*. Berlin, Heidelberg: Springer Berlin Heidelberg: Imprint: Springer,, 2003.

Ameya D Jagtap and George Em Karniadakis. Extended physics-informed neural networks (xpinns): A generalized space-time domain decomposition based deep learning framework for nonlinear partial differential equations. *Communications in Computational Physics*, 28(5), 2020.

Ameya D Jagtap, Ehsan Kharazmi, and George Em Karniadakis. Conservative physics-informed neural networks on discrete domains for conservation laws: Applications to forward and inverse problems. *Computer Methods in Applied Mechanics and Engineering*, 365:113028, 2020.

Ehsan Kharazmi, Zhongqiang Zhang, and George Em Karniadakis. hp-vpinns: Variational physics-informed neural networks with domain decomposition. *Computer Methods in Applied Mechanics and Engineering*, 374:113547, 2021.

Nikola Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Learning maps between function spaces with applications to pdes. *Journal of Machine Learning Research*, 24(89):1–97, 2023. URL http://jmlr.org/papers/v24/21-1524.html.

Zhihao Li, Zhilu Lai, Xiaobo Zhang, and Wei Wang. M2no: An efficient multi-resolution operator framework for dynamic multi-scale pde solvers. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 1*, pp. 771–782, 2026.

Zijie Li, Kazem Meidani, and Amir Barati Farimani. Transformer for partial differential equations' operator learning. *Transactions on Machine Learning Research*, 2023a. ISSN 2835-8856. URL https://openreview.net/forum?id=EPPqt3uERT.

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

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, 2023b.

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.

Qilong Ma, Haixu Wu, Lanxiang Xing, Shangchen Miao, and Mingsheng Long. Deeplag: Discovering deep lagrangian dynamics for intuitive fluid prediction. *Advances in Neural Information Processing Systems*, 37:122833–122866, 2024.

Md Ashiqur Rahman, Zachary E Ross, and Kamyar Azizzadenesheli. U-no: U-shaped neural operators. *arXiv preprint arXiv:2204.11127*, 2022.

M. Raissi, P. Perdikaris, and G.E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019. ISSN 0021-9991. doi: https://doi.org/10.1016/j.jcp.2018.10.045. URL https://www.sciencedirect.com/science/article/pii/S0021999118307125.

Bogdan Raonić, Roberto Molinaro, Tim De Ryck, Tobias Rohner, Francesca Bartolucci, Rima Alaifari, Siddhartha Mishra, and Emmanuel de Bézenac. Convolutional neural operators for robust and accurate learning of pdes. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NeurIPS '23, 2023.

Louis Serrano, Thomas X Wang, Etienne Le Naour, Jean-Noël Vittaut, and Patrick Gallinari. Aroma: Preserving spatial structure for latent pde modeling with local neural fields. *Advances in Neural Information Processing Systems*, 37:13489–13521, 2024.

Makoto Takamoto, Timothy Praditia, Raphael Leiteritz, Daniel MacKinlay, Francesco Alesiani, Dirk Pflüger, and Mathias Niepert. Pdebench: An extensive benchmark for scientific machine learning. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 1596–1611, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/0a9747136d411fb83f0cf81820d44afb-Paper-Datasets_and_Benchmarks.pdf.

Tapas Tripura and Souvik Chakraborty. Wavelet neural operator for solving parametric partial differential equations in computational mechanics problems. *Computer Methods in Applied Mechanics and Engineering*, 404:115783, 2023.

Sifan Wang, Yujun Teng, and Paris Perdikaris. Understanding and mitigating gradient flow pathologies in physics-informed neural networks. *SIAM Journal on Scientific Computing*, 43(5):A3055–A3081, 2021.

Gege Wen, Zongyi Li, Kamyar Azizzadenesheli, Anima Anandkumar, and Sally M. Benson. U-fno—an enhanced fourier neural operator-based deep-learning model for multiphase flow. *Advances in Water Resources*, 163:104180, 2022. ISSN 0309-1708. doi: https://doi.org/10.1016/j.advwatres.2022.104180. URL https://www.sciencedirect.com/science/article/pii/S0309170822000562.

Haixu Wu, Huakun Luo, Haowen Wang, Jianmin Wang, and Mingsheng Long. Transolver: A fast transformer solver for PDEs on general geometries. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 53681–53705, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/wu24r.html.

Lanxiang Xing, Haixu Wu, Yuezhou Ma, Jianmin Wang, and Mingsheng Long. Helmfluid: Learning helmholtz dynamics for interpretable fluid prediction. *arXiv preprint arXiv:2310.10565*, 2023.

Jeremy Yu, Lu Lu, Xuhui Meng, and George Em Karniadakis. Gradient-enhanced physics-informed neural networks for forward and inverse pde problems. *Computer Methods in Applied Mechanics and Engineering*, 393:114823, 2022.

Xihang Yue, Yi Yang, and Linchao Zhu. Holistic physics solver: learning pdes in a unified spectral-physical space. In *Proceedings of the 42nd International Conference on Machine Learning*, ICML'25, 2025.

Chenhong Zhou, Jie Chen, and Zaifeng Yang. Saot: An enhanced locality-aware spectral transformer for solving pdes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 28928–28936, 2026.

A APPENDIX

A.1 RELATED WORK

Physics-Informed Neural Networks. Physics-informed neural networks incorporate governing equations together with initial and boundary conditions into the training objective, enabling neural networks to approximate PDE solutions under explicit physical constraints Raissi et al. (2019). Subsequent studies improve their scalability and optimization through domain decomposition, stronger differential constraints, and conservative or hierarchical formulations. For example, XPINN Jagtap & Karniadakis (2020) decomposes the spatiotemporal domain into multiple subdomains, while gPINN Yu et al. (2022) augments PDE residuals with gradient information. Other variants explore conservative formulations Jagtap et al. (2020), adaptive loss balancing Wang et al. (2021), and hierarchical domain decomposition Kharazmi et al. (2021). Despite these advances, such methods generally require access to governing equations and associated physical constraints, limiting their applicability when only observed solution trajectories are available.

Data-Driven PDE Solvers. In contrast, data-driven PDE solvers learn mappings between observed inputs and solution fields directly from data. DeepONet Lu et al. (2021) establishes a general operator learning framework through branch and trunk networks, while FNO Li et al. (2021) models global interactions in Fourier space. Later methods improve representation capacity through convolutional, wavelet, and multiscale modeling, including CNO Raonić et al. (2023), WNO Tripura & Chakraborty (2023), WDNO Hu et al. (2025), U-NO Rahman et al. (2022), and M2NO Li et al. (2026). Attention-based solvers further enhance physical interaction modeling: OFormer Li et al. (2023a) and GNOT Hao et al. (2023) employ attention-based operator architectures, Transolver Wu et al. (2024) introduces physics attention over adaptive physical states, and SAOT Zhou et al. (2026) combines Fourier and wavelet attention for global and local dependencies. HPM Yue et al. (2025) further integrates spectral and physical space representations for multiscale solution modeling. These methods substantially improve how complex PDE dynamics are represented. However, many still generate complete target fields from learned latent representations, requiring spatial structures already present in the observations to be represented and recovered through the learned feature space rather than explicitly retained as the basis of target generation. Moreover, different forms of evolution are typically modeled jointly during solution construction without assigning distinct functional roles to spatial movement and physical value changes.

Structured dynamics modeling for PDE prediction. Explicitly structuring spatiotemporal dynamics provides useful inductive biases for learning PDE evolution. DeepLag Ma et al. (2024) infers the motion of adaptively sampled Lagrangian particles and incorporates their dynamics into Eulerian representations to guide fluid prediction. Its emphasis is on enriching evolving representations through particle–grid interaction. ADE-Solver instead uses target-specific sampling offsets to directly transform the latest observed physical field, making spatial transfer an explicit operation in solution construction. HelmFluid Xing et al. (2023)structures fluid dynamics through a Helmholtz decomposition into curl-free and divergence-free components and integrates the learned dynamics over time. Our decomposition addresses a different modeling level: spatial transfer and state variation act on sampling coordinates and physical values, respectively. This formulation does not require identifying the learned factors with Helmholtz components and is evaluated across both fluid and reaction–diffusion systems. Message Passing Neural PDE Solvers Brandstetter et al. (2022) introduce temporal bundling to predict multiple future states jointly, reducing the number of autoregressive solver calls. While temporal bundling specifies how predictions are grouped across time, ADE-Solver additionally specifies how each target is constructed: all target states are generated from a shared observed anchor through target-specific spatial transformations and gated value corrections. Together, these distinctions position ADE-Solver as an observation-anchored framework that combines explicit coordinate and value updates with parallel multi-horizon prediction.

Overall, prior work advances PDE prediction through operator representations, structured dynamics, and multi step forecasting. ADE-Solver brings these concerns together at solution construction: it retains the latest observation as a shared anchor and applies target-specific spatial transfers and gated state variations to generate all target states in parallel.

Table 3: Summary of the main notation used in this paper.

Symbol	Description
Ω	Spatial domain of the PDE.
N	Number of discretized spatial nodes.
C_g	Spatial dimension of the domain.
C_u	Number of physical variables at each spatial node.
$\mathbf{u}_t$	Physical state at time index t, where $\mathbf{u}_t \in \mathbb{R}^{N \times C_u}$.
$\mathbf{U}_{1:H}$	Observed sequence containing H historical states.
H	Number of observed time steps.
K	Number of target time steps.
$\widehat{\mathbf{U}}_{H+1:H+K}$	Predicted solution sequence for the K target steps.
$\mathcal{F}_\theta$	Learned solution operator parameterized by θ.
$\Delta\mathbf{u}_t$	Temporal difference between two adjacent observed states, $\mathbf{u}_t - \mathbf{u}_{t-1}$.
$\mathbf{D}_{1:H}$	Temporally aligned difference sequence, $[\mathbf{0}, \Delta\mathbf{u}_2, \ldots, \Delta\mathbf{u}_H]$.
$\mathbf{X}$	Augmented input formed by concatenating $\mathbf{U}_{1:H}$ and $\mathbf{D}_{1:H}$.
$\mathbf{S}^{(0)}$	Initial target-indexed latent representations after input embedding and temporal expansion.
$\mathbf{Z}$	Encoded spatiotemporal representation, $\mathbf{Z} \in \mathbb{R}^{K \times N \times C_z}$.
C_z	Latent feature dimension.
$\mathbf{Q}_a, \mathbf{K}_a, \mathbf{V}_a$	Query, key, and value features used in spatial or temporal attention.
$\mathbf{F}_{1:K}$	Spatial transfer fields for all target steps, $\mathbf{F}_{1:K} \in \mathbb{R}^{K \times N \times C_g}$.
$\mathbf{V}_{1:K}$	State variation fields for all target steps, $\mathbf{V}_{1:K} \in \mathbb{R}^{K \times N \times C_u}$.
$\mathbf{A}_{1:K}$	Adaptive weighting matrices for all target steps, $\mathbf{A}_{1:K} \in [0,1]^{K \times N \times C_u}$.
$\mathbf{u}_H$	Latest observed state used as the shared observation anchor.
$\mathbf{x}$	Spatial location on the discretized grid.
$\mathcal{I}[\mathbf{u}_H]$	Bilinear interpolation operator applied to the shared anchor.
$\mathbf{P}_k$	Spatially transformed anchor state for target step k.
$\mathbf{M}_k$	Adaptively modulated state variation, $\mathbf{M}_k = \mathbf{A}_k \odot \mathbf{V}_k$.
$\widehat{\mathbf{u}}_{H+k}$	Predicted solution state at target step k.

Table 4: Dataset configurations and sample construction used in our experiments.*For Cylinder Flow, temporal windows are sampled with a stride of 20 within temporally partitioned trunks of length 250.

Benchmark	Raw Size	Resolution	Split Strategy	Train/Val/Test Samples	Stride
Diffusion-Reaction	1000×101	128×128	trajectory-wise	2400 / 800 / 800	20
Shallow Water	1000×101	128×128	trajectory-wise	2400 / 800 / 800	20
Incompressible NS	4×1000	256×256	time-wise	2244 / 644 / 644	1
Cylinder Flow	92×3990	128×256	time-wise	9063 / 4820 / 4827	20*

A.2 Notation

For clarity, we summarize the main notation used throughout this paper in Table 3. Unless otherwise specified, bold uppercase and lowercase letters denote tensors and vectors, respectively, while calligraphic symbols represent operators or mappings.

A.3 Benchmark Datasets

We evaluate ADE-Solver on four spatiotemporal PDE benchmarks, including three simulated datasets from PDEBench and one real-world dataset from RealPDEBench. Table 4 summarizes the original benchmark characteristics and the sample construction used in our experiments.

Diffusion-Reaction. The Diffusion-Reaction benchmark from PDEBench describes the coupled evolution of two scalar fields, namely an activator u and an inhibitor v. It is governed by a reaction-diffusion system on a two-dimensional spatial domain, where nonlinear local reactions interact with spatial diffusion to produce complex spatiotemporal patterns. The benchmark contains 1000 trajectories, with 101 temporal frames per trajectory and an original spatial resolution of 128×128.

We divide the 1000 trajectories sequentially into 600 training, 200 validation, and 200 test trajectories, with no overlap across the three subsets. For each trajectory, we extract four temporal windows using 20 observation states and the subsequent 20 target states, with starting indices 0, 20, 40, and 60. This results in 2400, 800, and 800 training, validation, and test samples, respectively.

Shallow Water. The Shallow Water benchmark from PDEBench corresponds to a two-dimensional radial dam-break problem. Although the shallow-water equations generally involve water depth and velocity components, the released data used in our experiments contain a single scalar field corresponding to water depth. The benchmark contains 1000 trajectories, with 101 temporal frames per trajectory and an original spatial resolution of 128×128.

The trajectories are sequentially divided into 600 training, 200 validation, and 200 test trajectories. For each trajectory, four temporal windows are extracted using 20 observation states and the subsequent 20 target states, with starting indices 0, 20, 40, and 60. The resulting training, validation, and test sets contain 2400, 800, and 800 samples, respectively.

Incompressible NS. The Incompressible NS benchmark from PDEBench models two-dimensional incompressible flow with spatially varying forcing. We use the two velocity components, denoted by u_x and u_y, as the solution variables. The benchmark contains four trajectories, with 1000 temporal frames per trajectory and an original spatial resolution of 256×256.

Unlike Diffusion-Reaction and Shallow Water, this benchmark is split along the temporal dimension rather than the trajectory dimension. The same four trajectories are used across the three subsets, while their temporal ranges are disjoint: $[0, 600)$ for training, $[600, 800)$ for validation, and $[800, 1000)$ for testing. Temporal windows consist of 20 observation states and the subsequent 20 target states and are sampled with a stride of 1. This produces 2244, 644, and 644 training, validation, and test samples, respectively.

Cylinder Flow. Cylinder Flow is a real-world benchmark from RealPDEBench that describes wake dynamics behind a stationary cylinder. The flow fields are obtained from particle image velocimetry measurements and contain two velocity components, u and v. The benchmark contains 92 trajectories, with 3990 temporal frames per trajectory and an original spatial resolution of 128×256.

The dataset split follows the predefined JSON index files rather than a simple trajectory-level partition. Temporal windows are sampled within temporally partitioned trunks, and each sample contains 20 observation states followed by 20 target states. For regular trajectories, each trunk contains 250 temporal frames, and windows are sampled with a stride of 20. Although the same physical trajectory may appear in different subsets, the temporal ranges assigned to the training, validation, and test sets are mutually disjoint, so no sample overlap exists across subsets. This construction yields 9063 training samples, 4820 validation samples, and 4827 test samples.

Sample Construction. For all benchmarks, each sample consists of $H = 20$ consecutive observation states and the subsequent $K = 20$ states as targets. For Diffusion-Reaction and Shallow Water, samples are constructed trajectory-wise using four temporal windows per trajectory, with starting indices 0, 20, 40, and 60. For Incompressible NS, samples are constructed from disjoint temporal segments of the same trajectories using a stride of 1. For Cylinder Flow, samples are generated according to predefined temporal indices, with windows sampled within temporally partitioned trunks.

Normalization and Preprocessing. For all benchmarks, Gaussian normalization is applied channel-wise. The normalization statistics are computed from the training set only and then applied to the corresponding validation and test data.

A.4 BASELINES

We compare ADE-Solver with nine representative PDE solvers spanning branch and trunk operator learning, spectral and convolutional operators, multiresolution modeling, attention-based operators, and generative approaches.

DeepONet. DeepONet learns nonlinear operators through a branch and trunk architecture. The branch network encodes the input function sampled at a set of sensors, while the trunk network encodes the coordinates at which the output function is evaluated; their latent representations are combined to construct the solution field.

FNO. Fourier Neural Operator (FNO) learns mappings between function spaces by parameterizing integral operators directly in Fourier space. Spectral convolution enables efficient global information mixing and provides a resolution-independent formulation for learning PDE solution operators.

CNO. Convolutional Neural Operator (CNO) extends convolutional architectures to operator learning while preserving the continuous nature of input and output functions under discrete computation. It employs convolutional processing across multiple resolutions to capture both local and multiscale structures in PDE solutions.

HPM. Holistic Physics Mixer (HPM) integrates domain-level spectral structure and point-wise physical states within a unified holistic spectral space. This formulation combines spectral priors with point-wise adaptivity, enabling spectral representations to vary according to local physical features.

M2NO. Multiwavelet-based Algebraic Multigrid Neural Operator (M2NO) combines multiwavelet transformations with algebraic multigrid to learn PDE operators across multiple resolutions. Its multiresolution analysis decomposes solution representations through high- and low-pass components, capturing global structures and localized details at different spatial scales.

Transolver. Transolver introduces Physics-Attention to learn intrinsic physical states from discretized PDE domains. It adaptively groups mesh points with similar physical states into learnable slices and performs attention over the resulting physics-aware tokens to model global physical interactions efficiently.

3D-Unet. 3D-Unet models the complete spatiotemporal volume using three-dimensional convolutional encoding and decoding, allowing temporal and spatial features to be jointly processed across multiple resolutions.

SAOT. Spectral Attention Operator Transformer (SAOT) combines locality-aware Wavelet Attention with global Fourier-based Attention. The two spectral representations are integrated through gated fusion, allowing the model to jointly capture localized multiscale structures and long-range dependencies in PDE solution fields.

WDNO. Wavelet Diffusion Neural Operator (WDNO) performs diffusion-based generative modeling in the wavelet domain for PDE solution trajectories. By modeling the entire trajectory in a multiresolution wavelet representation, WDNO captures long-term dependencies and fine-scale solution structures, while multi-resolution training supports generalization across spatial resolutions.

A.5 Implementation Details

Unified experimental protocol. For a controlled comparison, all methods are trained and evaluated using the same data splits, physical variables, preprocessing procedures, observation and target windows, and evaluation samples. Each method receives $H = 20$ observed states and directly predicts the subsequent $K = 20$ target states. All baselines are reproduced using their official implementations while retaining their original model architectures. Dataset-specific model capacities, learning rates, and batch sizes are selected according to validation performance. No test-set performance is used for hyperparameter selection. Each experiment is repeated seven times, and the average test performance is reported. All experiments are conducted on the same computing platform with an AMD EPYC 7402 24-core CPU and an NVIDIA GeForce RTX 4090 GPU with 24 GB memory. We additionally compute the standard deviation across the seven runs and use the run-level results for significance testing. All methods use the same $20 \rightarrow 20$ prediction setting, data split, Gaussian normalization, spatial preprocessing, and evaluation protocol. Optimization hyperparameters are selected separately for each method based on validation performance and architecture-specific requirements.

Baseline implementations. We preserve the characteristic architecture of each baseline and use its official implementation. DeepONet retains its branch–trunk operator-learning formulation, FNO uses spectral convolution for spatiotemporal operator learning, and CNO employs multiresolution

Table 5: Common experimental protocol used for ADE-Solver and all baselines.

Setting	Configuration
Observation window	$H = 20$
Target window	$K = 20$
Prediction setting	Direct $20 \rightarrow 20$ mapping
Data splits	Identical across all methods
Preprocessing	Identical across all methods
Physical variables	Identical across all methods
Hyperparameter selection	Validation performance
Evaluation samples	Identical across all methods
Number of runs	7
Hardware	NVIDIA RTX 4090 (24 GB)

Table 6: Number of trainable parameters on Diffusion-Reaction.

Method	Parameters
DeepONet	3.63M
FNO	254.87M
CNO	17.83M
Transolver	18.85M
HPM	1.10M
WDNO	102.93M
3D-Unet	22.96M
SAOT	6.91M
ADE-Solver	23.00M

convolutional operator learning. 3D-Unet jointly processes spatial and temporal dimensions using three-dimensional convolutions. HPM and M2NO retain their respective spectral and multiresolution operators, while Transolver and SAOT preserve their attention-based interaction mechanisms. WDNO performs conditional trajectory generation in the wavelet domain. For each benchmark, model capacity and optimization-related hyperparameters are selected using the validation set under the same evaluation protocol.

Training and evaluation. ADE-Solver is optimized using Adam with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$, together with a cosine learning-rate schedule. The learning rate is set to 10^{-3} for Diffusion-Reaction and Shallow Water, and 10^{-4} for Incompressible NS and Cylinder Flow. The regularization weights are set to $\lambda_{\mathrm{s}} = 10^{-3}$, $\lambda_{\mathrm{m}} = 10^{-4}$, and $\lambda_{\mathrm{v}} = 10^{-4}$. All methods are evaluated using RMSE and relative L_2 error over the complete target sequence. Let $\hat{\mathbf{Y}} = \hat{\mathbf{U}}_{H+1:H+K}$ and $\mathbf{Y} = \mathbf{U}_{H+1:H+K}$ denote the predicted and reference target sequences. The metrics are defined as

$$\mathrm{RMSE} = \left(\frac{1}{KNC_u} \left\| \hat{\mathbf{Y}} - \mathbf{Y} \right\|_2^2 \right)^{1/2}, \tag{18}$$

and

$$\text{Rel. } L_2 = \frac{\left\| \hat{\mathbf{Y}} - \mathbf{Y} \right\|_2}{\left\| \mathbf{Y} \right\|_2}. \tag{19}$$

Both metrics are computed over all target steps, spatial locations, and physical variables, with lower values indicating better solution accuracy.

Model size. We additionally report the number of trainable parameters on Diffusion-Reaction to provide a reference for model capacity. ADE-Solver contains 23.0M parameters, which is comparable to 3D-Unet (23.0M) and substantially smaller than FNO and WDNO. Notably, ADE-Solver and 3D-Unet have nearly identical parameter counts (23.00M vs. 22.96M), while ADE-Solver achieves substantially lower errors. Together with the fact that all methods use the same direct $20 \rightarrow 20$ supervision, this suggests that the observed gains cannot be explained solely by model capacity or multi-horizon supervision.

A.6 Time Complexity Analysis

We analyze the computational complexity of the three core components of ADE-Solver. Let L denote the number of scales in the multiscale encoder, N_ℓ the number of spatial nodes at scale ℓ, and C_ℓ the corresponding feature dimension. At the full spatial resolution, $N_1 = N$, and the final encoded feature dimension is denoted by C_z.

Temporal Difference Augmentation. Temporal Difference Augmentation computes the differences between adjacent observations: $\Delta \mathbf{u}_t = \mathbf{u}_t - \mathbf{u}_{t-1}, \ldots, H$. Computing these differences over H states, N spatial nodes, and C_u physical variables requires $\mathcal{O}(HNC_u)$. Concatenating the original

Table 7: Implementation configurations on Shallow Water.

Method	Core Architecture Configuration	LR	Train Batch	Updates
DeepONet	Latent dimension 128	1×10^{-3}	4	10k
FNO	6 layers; width 96; Fourier modes $(12, 12, 12)$	5×10^{-4}	4	10k
CNO	2 layers; channel multiplier 48; 1 residual block; 6 bottleneck residual blocks	5×10^{-4}	4	10k
Transolver	3 layers; hidden dimension 128; 8 attention heads; 32 slices	5×10^{-4}	4	10k
HPM	3 layers; hidden dimension 128; 4 attention heads; frequency count 32	1×10^{-3}	4	10k
WDNO	Base dimension 256; channel multipliers $[1, 2]$; bior1.3 wavelet; 20 sampling steps	5×10^{-5}	4	10k
3D-Unet	U shaped backbone; channel multipliers $[1, 2, 4]$	1×10^{-3}	4	10k
M2NO	$k = 2$; $c = 16$; 3 layers; grid levels $[1]$; Legendre basis	1×10^{-3}	4	10k
SAOT	5 layers; hidden dimension 256; 8 attention heads; mode fraction 0.5	5×10^{-4}	4	10k
ADE-Solver	U shaped backbone; channel multipliers $[1, 2, 4]$	1×10^{-3}	4	10k

states and temporal differences has the same linear order and introduces no additional learnable parameters.

Disentangled Spatiotemporal Representation Learning. Before multiscale encoding, the augmented input is embedded into latent features and expanded to form K target-indexed representations. For a pointwise embedding with output dimension C_1, the embedding cost is linear in the input size, while temporal expansion introduces linear memory and indexing overhead. We denote these costs collectively by $\mathcal{O}(HNC_uC_1 + KNC_1)$. At each scale, Spatiotemporal Correlation Learning applies spatial attention across spatial nodes and temporal attention across the K target indices. For spatial attention at scale ℓ, attention is evaluated over N_ℓ spatial nodes for each of the K target indices. The feature projections require $\mathcal{O}(KN_\ell C_\ell^2)$, while computing and applying the spatial attention matrix requires $\mathcal{O}(KN_\ell^2 C_\ell)$. Therefore, the spatial attention complexity is $\mathcal{O}\left(KN_\ell C_\ell^2 + KN_\ell^2 C_\ell\right)$. Temporal attention is evaluated across K target indices independently at each of the N_ℓ spatial locations. Its feature projections require $\mathcal{O}(N_\ell K C_\ell^2)$, and the temporal attention operation requires $\mathcal{O}(N_\ell K^2 C_\ell)$. The resulting temporal attention complexity is $\mathcal{O}\left(N_\ell K C_\ell^2 + N_\ell K^2 C_\ell\right)$. Including the multiscale feature transformations, which have the same order as the feature projection terms, the overall encoding complexity across all scales is $\mathcal{O}\left(\sum_{\ell=1}^{L}\left[KN_\ell C_\ell^2 + KN_\ell^2 C_\ell + N_\ell K^2 C_\ell\right]\right)$. The final representation $\mathbf{Z} \in \mathbb{R}^{K\times N\times C_z}$ is projected into the spatial transfer fields $\mathbf{F}_{1:K}$, state variation fields $\mathbf{V}_{1:K}$, and adaptive weighting matrices $\mathbf{A}_{1:K}$. Since their output dimensions are C_g, C_u, and C_u, respectively, the field projection complexity is $\mathcal{O}\left(KNC_z(C_g + 2C_u)\right)$.

Parallel Anchor-Based Evolution Generation. For every target index, bilinear interpolation samples the shared anchor at N locations for each of the C_u physical variables. Because bilinear interpolation uses a constant number of neighboring grid points, anchor-based transformation requires $\mathcal{O}(KNC_u)$. Adaptive state variation modulation and final solution construction are element-wise operations with the same complexity. Thus, the total complexity of anchor-based evolution generation is $\mathcal{O}(KNC_u)$.

Overall Complexity. Combining the three components, the overall time complexity is

$$\mathcal{O}\Big(HNC_uC_1 + KNC_1 + \sum_{\ell=1}^{L} KN_\ell C_\ell(C_\ell + N_\ell + K) + KNC_z(C_g + 2C_u) + KNC_u\Big). \tag{20}$$

Under standard full attention, the spatial attention term $KN_\ell^2 C_\ell$ is quadratic in the number of spatial nodes at each scale. The multiscale architecture reduces this cost at deeper scales because N_ℓ decreases as the spatial resolution is reduced. Temporal attention is quadratic in the target length K, whereas Temporal Difference Augmentation, field projection, and anchor-based evolution generation scale linearly with their corresponding sequence and spatial dimensions. The dominant computational cost comes from the representation learning backbone within ADE-Solver, while the proposed evolution generation introduces only linear additional overhead.

Although temporal attention models dependencies among target-indexed representations, the final anchor-based solution construction does not use previously predicted target states. All K target states are therefore constructed from the shared observation anchor and their corresponding evolution fields without sequential target-to-target evaluation.

A.7 Fine Grained Ablation Analysis

To further examine the roles of the evolution factors and adaptive weighting, we conduct fine-grained ablations by separately removing spatial transfer, state variation, and adaptive weighting. Specifically, *w/o Spatial Transfer* removes the spatial sampling offsets, reducing the generation rule to $\hat{\mathbf{u}}_{H+k} = \mathbf{u}_H + \mathbf{A}_k \odot \mathbf{V}_k$. This variant retains the latest observation as the anchor and directly learns a residual state change, thereby isolating the benefit of spatial transfer. ***w/o* State Variation** removes the additive state variation term, yielding $\hat{\mathbf{u}}_{H+k} = \mathcal{I}[\mathbf{u}_H](\mathbf{x} + \mathbf{F}_k)$, which isolates anchor-based spatial transformation without value modulation. *w/o Adaptive Weight* fixes the adaptive weighting matrix to $\mathbf{A}_k = \mathbf{1}$ for all target steps, spatial locations, and physical variables.

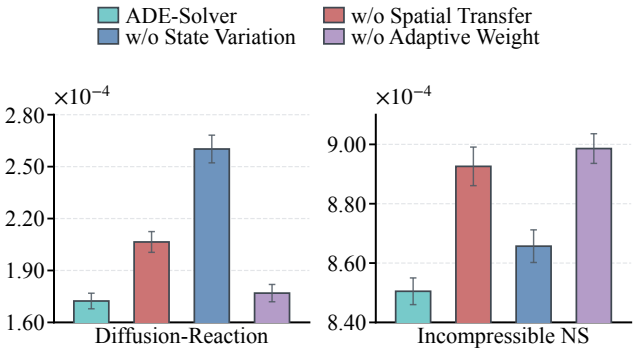

Figure 8: Fine grained ablation of spatial transfer, state variation, and adaptive weighting on two benchmarks in terms of RMSE.

As shown in Figure 8, removing either evolution factor degrades performance on both benchmarks, confirming that spatial transfer and state variation both contribute to target solution construction. On Diffusion-Reaction, removing state variation causes a substantially larger increase in RMSE than removing spatial transfer (50.85% vs. 19.79%), suggesting that state variation is particularly important for dynamics with pronounced changes in physical values.

In contrast, on Incompressible NS, removing spatial transfer causes a larger performance degradation than removing state variation (4.95% vs. 1.79%), indicating the greater importance of spatial redistribution for evolving flow structures. Removing adaptive weighting further increases RMSE by 2.65% and 5.72% on Diffusion-Reaction and Incompressible NS, respectively. The consistent degradation indicates that adaptive weighting provides additional refinement by modulating the contribution of state variation.

Overall, the relative importance of spatial transfer and state variation varies across different dynamics, while both consistently contribute to target construction. Adaptive weighting provides a complementary refinement, further supporting the effectiveness of functionally differentiated evolution modeling.

A.8 Frequency-Resolved Analysis

We further evaluate the prediction accuracy of different methods in the frequency domain using the original frequency-resolved RMSE (fRMSE) implementation provided by RealPDEBench. This metric applies a three-dimensional discrete Fourier transform to the target sequence and measures the prediction error within different frequency bands. Let $\widehat{\mathbf{U}} = \widehat{\mathbf{U}}_{H+1:H+K}$ and $\mathbf{U} = \mathbf{U}_{H+1:H+K}$

denote the predicted and reference target sequences, respectively. For a physical-variable channel c, the Fourier-domain error is defined as

$$\mathbf{E}_c = \mathrm{FFT}_{3D}\left(\widehat{\mathbf{U}}_c\right) - \mathrm{FFT}_{3D}\left(\mathbf{U}_c\right), \tag{21}$$

where the transform is applied over the target-time and two spatial dimensions.

For a frequency band $\mathcal{B}$, the corresponding fRMSE can be expressed as

$$\mathrm{fRMSE}_c(\mathcal{B}) = \left(\frac{1}{|\mathcal{B}|}\sum_{\boldsymbol{\xi}\in\mathcal{B}} |\mathbf{E}_c(\boldsymbol{\xi})|^2\right)^{1/2}, \tag{22}$$

where $\boldsymbol{\xi}$ denotes a frequency index and $|\mathcal{B}|$ denotes the number of Fourier coefficients within the band. The reported results are aggregated over the test samples and physical variables following the official RealPDEBench implementation. We use the same low-, mid-, and high-frequency partition as RealPDEBench.

As shown in Figure 9, ADE-Solver achieves the lowest fRMSE in all three frequency bands. Compared with the strongest baseline within each band, ADE-Solver improves fRMSE by 48.2%, 61.7%, and 38.8% in the low-, mid-, and high-frequency bands, respectively. The largest improvement is obtained in the mid-frequency band, while the consistent gains in the low- and high-frequency bands indicate that the performance advantage is maintained across different spectral ranges. These results further demonstrate that ADE-Solver accurately represents both broad spectral structures and finer variations in the target solutions.

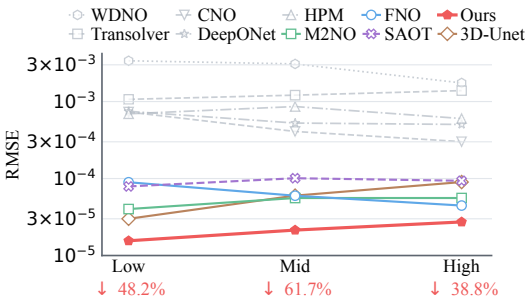

Figure 9: Frequency-resolved fRMSE on Shallow Water. Percentages indicate improvement over the strongest baseline.

A.9 Limitations

Our current experiments focus on fixed benchmark settings with grid-based data representations. The observation-anchored formulation itself is not restricted to a particular discretization and could be combined with geometry-aware interpolation for irregular domains, although this extension is not evaluated here. The learned evolution factors are functionally differentiated through distinct generation operations rather than guaranteed to provide a unique physical decomposition. Broader evaluation under unseen PDE parameters, resolutions, geometries, conservation-sensitive boundary settings, and longer target horizons remains for future work.