GRID–TOKEN NEURAL OPERATOR: ASYMMETRIC SPATIOTEMPORAL COUPLING FOR PDE FORECASTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Neural operators offer an efficient surrogate for repeated numerical simulation of partial differential equations (PDEs), but jointly representing spatial detail and temporal dependencies remains challenging. Applying temporal attention at every fine-grid location is costly, whereas compressing the entire evolving state places spatial reconstruction and temporal modeling behind the same bottleneck. To address this trade-off, we propose the Grid–Token Neural Operator (GTNO), which maintains two complementary representations throughout the network: a structured grid for spatial refinement and compact tokens for causal temporal modeling. Each layer couples time-shared local grid updates with token-wise temporal attention through bidirectional interaction. Local grid-to-token routing refreshes the temporal representation, while token-conditioned spectral corrections feed temporal context back to the grid. This asymmetric design concentrates expensive causal sequence modeling on compact tokens while retaining a dedicated spatial pathway. Across five time-dependent PDE benchmarks, GTNO consistently outperforms eight neural operator baselines, demonstrating the effectiveness and efficiency of combining fine spatial evolution with compact temporal modeling.

1 INTRODUCTION

Time-dependent partial differential equations (PDEs) describe the evolution of physical systems over space and time, underlying applications such as fluid dynamics, reaction–diffusion processes, and climate modeling. Their numerical solution typically relies on spatial discretization and time integration by numerical solvers to advance the physical state over time. Although such solvers provide high-fidelity solutions, repeated simulation across many initial conditions and physical parameters can be computationally expensive. To alleviate this repeated computational cost, neural operators provide a data-driven approach by learning mappings between function spaces (Kovachki et al., 2023; Li et al., 2021; Lu et al., 2021), allowing the learned operator to be reused within a family of PDE instances.

Most existing neural operators have focused on exploiting spatial information to learn PDE solution operators. Spectral methods such as the Fourier Neural Operator (FNO) perform frequency-domain transformations to capture global spatial interactions, which is particularly effective on structured grid data (Li et al., 2021; Tran et al., 2023). To accommodate more general spatial discretizations, attention-based operators directly model data-dependent interactions among spatial elements, with several supporting irregular geometries and unstructured data (Li et al., 2023a;b; Wu et al., 2024; Hu et al., 2026). However, operating directly over a large number of spatial elements can become computationally cost-prohibitive. To reduce computational cost, latent operators compress physical fields into a smaller set of latent tokens and process the resulting compact representations (Wang & Wang, 2024; Alkin et al., 2024; Tiwari et al., 2025). Collectively, these methods have primarily advanced the flexibility and efficiency of spatial representations, while their treatment of temporal evolution remains tied to a predefined history length, limiting their ability to adapt the temporal context to different dynamics and time scales.

To better capture temporal information, recent approaches have introduced dedicated mechanisms to complement spatial processing with explicit temporal modeling. Some of these methods aggregate information within a fixed observation window (Hao et al., 2024), while query-based formulations represent time explicitly and support flexible temporal forecasting (Hagnberger et al., 2024;

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.

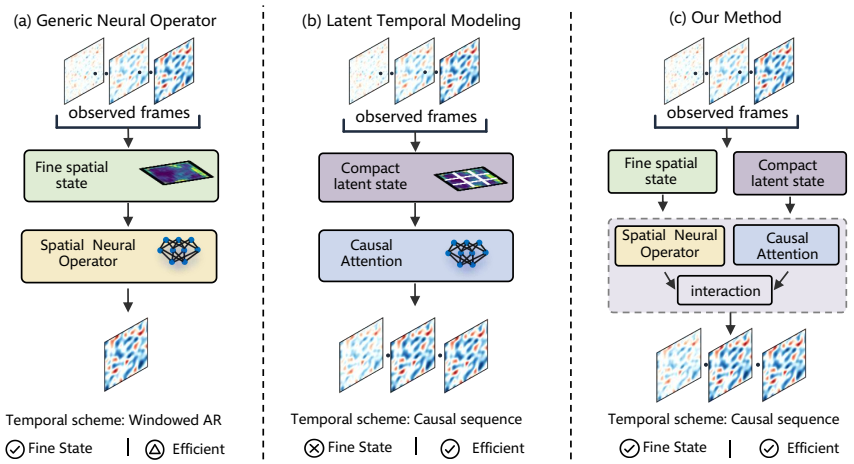


Figure 1: Conceptual comparison of representative design patterns. (a) A spatial neural operator conditions on a fixed history window for autoregressive (AR) forecasting. (b) A latent temporal model processes a compressed state sequence. (c) Our method retains a structured grid alongside compact temporal tokens and couples them with each layer. Causal sequence processing enables parallel next-state prediction during training, while forecasting remains autoregressive.

Hou et al., 2026). Other methods apply causal sequence models to implicit spatial representations, thereby explicitly modeling temporal dependencies while preserving spatial support (Buitrago Ruiz et al., 2025). As an alternative strategy to reduce computational costs, latent sequence models perform temporal modeling over compressed representations (Koupaï et al., 2025), though this necessitates compressing rich spatial information into compact states. A key architectural distinction among these approaches lies in the representation scale at which temporal modeling is performed. Specifically, retaining a fine spatial state supports detailed field evolution, but applying temporal attention at every location is costly. Compressing the evolving state can reduce this cost, yet it may also result in a loss of spatial detail. Therefore, accurate and efficient forecasting calls for an architecture that accommodates both spatial detail and temporal context, as illustrated in Figure 1.

To address this conflict, this paper proposes the Grid–Token Neural Operator (GTNO), which maintains two complementary representations with asymmetric spatial resolutions. Unlike existing works that jointly encode spatial and temporal information in a single representation, GTNO employs a high-resolution structured grid state with lightweight local operators to evolve fine-grained spatial structures without excessive compression, while using a compact token state with causal attention to model temporal dependencies without attending over all spatial elements. However, performing these two states independently risks leaving spatial and temporal evolution largely disconnected, leading to the loss of complementarity. To address this issue, GTNO couples these two states through iterative bidirectional interaction: the temporal state continuously integrates evolving spatial information, while the spatial state is refined by accumulated temporal context. This repeated exchange maintains consistency between spatial and temporal evolution while preserving their specialized roles, thereby mitigating the loss of fine-grained spatial information and improving temporal modeling efficiency through causal attention in a compact token space.

Overall, our main contributions are summarized as follows:

1. We propose a role-separated Grid–Token architecture for time-dependent PDE forecasting, which maintains a high-resolution structured grid for spatial refinement and compact tokens for causal temporal modeling, assigning the two roles different spatial resolutions.
2. We present an asymmetric coupling mechanism with repeated bidirectional interaction, allowing the spatial and temporal representations to remain specialized while continuously exchanging information.
3. We demonstrate the effectiveness of our GTNO on five time-dependent PDE benchmarks, achieving strong predictive accuracy together with a favorable accuracy–efficiency trade-off compared with representative neural operator baselines.

2 Related Work

2.1 Neural Operators for Spatial Representations

Neural operators learn mappings between function spaces and have become a widely used framework for data-driven PDE modeling (Kovachki et al., 2023). A major line of development concerns how physical fields are represented and how interactions are efficiently captured over the spatial domain. Fourier Neural Operator (FNO) introduced global spatial interactions through spectral convolution on structured grids (Li et al., 2021), with subsequent architectures improving efficiency through factorized Fourier transformations (Tran et al., 2023). Attention-based neural operators construct data-dependent interactions among spatial points or nodes, offering greater flexibility for irregular discretizations and complex geometries (Li et al., 2023a;b; Wu et al., 2024; Hu et al., 2026; Cao, 2021; Hao et al., 2023), while hierarchical and multi-resolution architectures capture spatial structures across different scales (Gupta et al., 2021; Liu et al., 2024). Latent operator methods further improve computational efficiency by compressing physical fields before applying the main operator. Representative approaches perform operator learning in a learned latent space, over a fixed set of latent tokens, or through state-space models in the latent domain (Wang & Wang, 2024; Alkin et al., 2024; Tiwari et al., 2025).

2.2 Temporal Modeling for Time-Dependent PDEs

Many neural operators do not introduce dedicated temporal mechanisms and instead handle time-dependent PDEs by conditioning on a fixed window of historical states and autoregressively predicting future states (Li et al., 2021; Lippe et al., 2023). DPOT introduces explicit temporal aggregation to integrate observations within a fixed context window (Hao et al., 2024), while query-based formulations model time explicitly: VCNeF represents solutions as conditional neural fields over spatiotemporal coordinates (Hagnberger et al., 2024), and CFO learns continuous-time PDE dynamics for evaluation at arbitrary temporal locations (Hou et al., 2026). Sequence-based approaches instead model temporal dependencies directly over evolving representations. MemNO applies a sequence model along time to hidden representations at each spatial location, retaining spatial support while introducing explicit memory (Buitrago Ruiz et al., 2025). ENMA instead compresses physical observations into latent representations and applies causal attention over the resulting temporal sequence (Koupaï et al., 2025). GTNO bridges these representation scales by maintaining a fine-grained grid state for spatial evolution while performing causal temporal modeling over compact tokens, with repeated bidirectional interaction coupling the two states.

3 Methods

3.1 Problem Formulation

We consider a family of time-dependent PDEs defined on a spatial domain $\Omega \subset \mathbb{R}^d$. Let $u(x,t) \in \mathbb{R}^m$ denote the physical state at location $x \in \Omega$ and time $t \in [0, T_{\max}]$. Given an initial condition u_0 and physical parameters λ, the system evolves according to $\partial_t u(x,t) = \mathcal{F}[u;\lambda](x,t)$, with $u(x,0) = u_0(x)$. In practice, the physical field is observed at spatial locations $\mathcal{X} = \{x_i\}_{i=1}^N$. We denote the discretized state at the t-th time step by $\mathbf{u}_t = [u(x_1,t), \ldots, u(x_N,t)] \in \mathbb{R}^{N\times m}$ and an observed sequence by $\mathbf{U}_{a:b} = (\mathbf{u}_a, \ldots, \mathbf{u}_b)$. Given T_{in} observed states, we learn a causal neural operator

$$\widehat{\mathbf{u}}_{T_{\text{in}}+1} = \mathcal{N}_\theta(\mathcal{X}, \mathbf{U}_{1:T_{\text{in}}}).$$

For multi-step forecasting, predicted states are fed back autoregressively to generate the subsequent T_{pred} states. Our goal is therefore to learn a spatially structured and temporally causal evolution operator that remains accurate over long rollout horizons.

3.2 Model Overview

Figure 2 illustrates the overall architecture of GTNO. Given spatial coordinates $\mathcal{X}$ and an observed sequence $\mathbf{U}_{1:T}$, the model initializes a structured grid state W^0 and a compact token state Z^0.

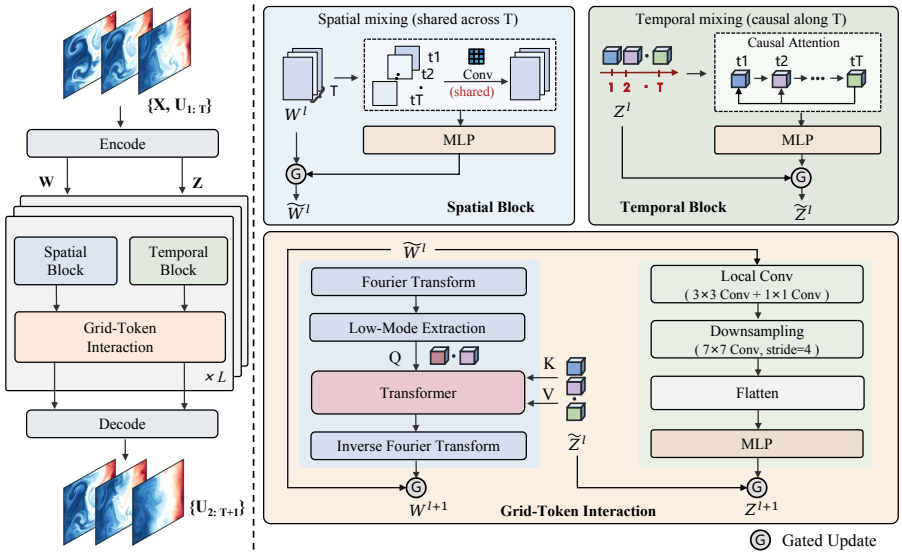


Figure 2: Overview of GTNO. Left: the overall architecture, where a structured grid state W and a compact token state Z co-evolve through L stacked layers. Right: the detailed architecture of each layer, comprising spatial mixing shared across time, causal temporal mixing along time, and bidirectional Grid–Token Interaction.

The two states are then updated through L stacked Grid–Token layers, each following an evolve–then–interact scheme: the states first undergo role-specific spatial and temporal evolution and subsequently exchange information through bidirectional interaction.

Dual-State Initialization Let $M = H_g W_g$ denote the number of latent-grid locations and J the number of compact tokens. The two initial states are

$$W^0 = \mathcal{E}_g\left(\mathcal{X}, \mathbf{U}_{1:T}\right) \in \mathbb{R}^{T\times M\times d_g},$$
$$Z^0 = \mathcal{E}_t\left(\mathcal{X}, W^0\right) \in \mathbb{R}^{T\times J\times d_z}, \qquad J \ll M.$$

Here, $\mathcal{E}_g$ is the physical-to-grid encoder and $\mathcal{E}_t$ is the grid-to-token encoder. Static coordinate features are incorporated into both states. The physical-to-grid encoder and the final grid-to-physical decoder are detailed in Appendix A.1.1.

3.3 Grid–Token Evolution Layer

Each Grid–Token layer consists of two stages. First, the structured grid state and compact token state are processed by their respective spatial and temporal operators. The resulting branch-specific candidates are adaptively incorporated into the corresponding states. Second, the two evolved states exchange information through bidirectional Grid–Token interaction.

Adaptive Gated Fusion Throughout the architecture, we combine two aligned representations using a data-dependent gated fusion operator. Given two features A and B of the same shape, we define

$$\mathrm{GF}_\psi(A, B) = \gamma_\psi(A, B) \odot A + \big(1 - \gamma_\psi(A, B)\big) \odot B,$$

where $\gamma_\psi(A, B)$ is an element-wise gate with values in $[0, 1]$, predicted by a lightweight gating network with parameters ψ, and $\odot$ denotes element-wise multiplication.

3.3.1 Structured Grid Evolution

The structured grid branch evolves the spatial field independently at each temporal position. For the two-dimensional systems considered in this work, we use a lightweight ConvNeXt-style spatial

operator (Liu et al., 2022), consisting of a depthwise convolution for local spatial mixing and a pointwise MLP for channel transformation. Specifically, we compute

$$\widehat{W}^l = W^l + \Phi_{\mathrm{s}}^l\left(\mathrm{LN}\left(\mathrm{DWConv}^l(W^l)\right)\right), \qquad \widetilde{W}^l = \mathrm{GF}_{\mathrm{s}}^l\left(W^l, \widehat{W}^l\right).$$

Here, DWConv and Φ_{s}^l perform local spatial mixing and channel transformation, respectively. The operator is shared across temporal slices, so this branch updates spatial structure without explicitly mixing time; temporal dependencies are handled by the token branch.

3.3.2 Causal Temporal Token Evolution

The compact token branch models temporal dependencies along the observed sequence. For each token j, causal self-attention is applied along its temporal trajectory $Z_{:,j}^l \in \mathbb{R}^{T\times d_z}$, independently of the other token trajectories. The same temporal operator is shared across all J tokens. We compute

$$H^l = Z^l + \mathrm{CausalAttn}^l\left(\mathrm{LN}(Z^l)\right),$$
$$\widehat{Z}^l = H^l + \Phi_{\mathrm{t}}^l\left(\mathrm{LN}(H^l)\right), \qquad \widetilde{Z}^l = \mathrm{GF}_{\mathrm{t}}^l\left(Z^l, \widehat{Z}^l\right).$$

where Φ_{t}^l denotes a feed-forward network. The causal mask ensures that the representation at temporal position t depends only on positions $1, \ldots, t$. Thus, all observed positions can be processed in parallel during training without exposing future states. Moreover, temporal attention is performed over J compact tokens rather than the M high-resolution grid locations, where $J \ll M$.

3.3.3 Bidirectional Grid–Token Interaction

After role-specific evolution, the two states exchange information through asymmetric bidirectional interaction. Both pathways are computed from the same evolved states $(\widetilde{W}^l, \widetilde{Z}^l)$: the grid-to-token pathway refreshes the compact temporal state with newly evolved spatial features, while the token-to-grid pathway introduces temporally conditioned global corrections into the structured grid.

Grid-to-token We map the evolved grid state to the token resolution using a learnable spatial router:

$$Z_{\mathrm{g}\to\mathrm{t}}^l = \mathcal{R}_{\downarrow}^l\left(\widetilde{W}^l\right), Z^{l+1} = \mathrm{GF}_{\mathrm{g}\to\mathrm{t}}^l\left(\widetilde{Z}^l, Z_{\mathrm{g}\to\mathrm{t}}^l\right).$$

The router combines local spatial refinement, learnable downsampling, and feature projection to convert the M grid locations into J compact tokens. This pathway is applied independently at each temporal position, continually refreshing the temporal state with the latest spatial information.

Token-to-grid Rather than expanding the compact tokens to reconstruct the complete high-resolution field, we use them to condition a compact set of low-frequency Fourier modes of the evolved grid. We first form the spectral queries $S^l = \Phi_{\mathrm{spec,in}}^l(\mathcal{F}_{\mathrm{low}}(\widetilde{W}^l))$. They are then updated as $\widehat{S}^l = \mathcal{C}_{\mathrm{spec}}^l(S^l; \widetilde{Z}^l)$, where $\mathcal{C}_{\mathrm{spec}}^l$ denotes a residual cross-attention block followed by a feed-forward network. The spectral features S^l provide the queries, while the temporally evolved tokens $\widetilde{Z}^l$ provide the keys and values. The updated spectral representation predicts a residual correction over the retained modes:

$$\Delta W_{\mathrm{t}\to\mathrm{g}}^l = \mathcal{F}_{\mathrm{low}}^{-1}\left(\Phi_{\mathrm{spec,out}}^l(\widehat{S}^l)\right),$$
$$W^{l+1} = \mathrm{GF}_{\mathrm{t}\to\mathrm{g}}^l\left(\widetilde{W}^l, \widetilde{W}^l + \Delta W_{\mathrm{t}\to\mathrm{g}}^l\right).$$

Here, $\mathcal{F}_{\mathrm{low}}^{-1}$ places the predicted corrections at the retained Fourier modes and sets the remaining modes to zero before the inverse transform. Each retained mode contributes across the spatial grid, allowing compact tokens to provide global corrections without being expanded to every grid location. Meanwhile, the grid branch continues to carry fine-scale spatial information. Appendix A.1.2 separates the error of the pre-gating spectral correction into Fourier truncation error and learned-coefficient error: for a spatially smooth target correction, the former decreases as more modes are retained. Gated fusion then combines this correction with the evolved grid state.

3.4 Prediction and Autoregressive Rollout

Prediction Head After the final layer, we project the token state back to the grid resolution as $Z_{\mathrm{g}}^{L} = \mathcal{R}_{\uparrow}(Z^{L})$ and fuse it with the structured state as $R_{\mathrm{g}}^{L} = \mathrm{GF}_{\mathrm{out}}(W^{L}, Z_{\mathrm{g}}^{L})$. The fused representation is queried at the physical locations and mapped to the predicted physical variables: $\widehat{\mathbf{U}}_{2:T+1} = \Phi_{\mathrm{pred}}(\mathcal{Q}_{\mathcal{X}}(R_{\mathrm{g}}^{L}))$. Here, $\mathcal{R}_{\uparrow}$ maps the compact tokens to the structured grid resolution, $\mathcal{Q}_{\mathcal{X}}$ maps grid features back to the physical locations, and Φ_{pred} denotes the output projection.

Causal Training and Autoregressive Inference During training, the input sequence $\mathbf{U}_{1:T}$ is paired with the shifted target sequence $\mathbf{U}_{2:T+1}$, such that the prediction at temporal position t is supervised by $\mathbf{u}_{t+1}$. Since temporal attention is causally masked and the remaining operations are applied independently at each temporal position, all next-state predictions can be computed in parallel without accessing future observations. During inference, the final prediction $\widehat{\mathbf{u}}_{T+1}$ is fed back as the input for the next step, and this procedure is repeated for T_{pred} steps. We cache the keys and values of each temporal attention layer, allowing every rollout step to process only the newly generated state while reusing the preceding temporal context.

4 Experiments

4.1 Experimental Setup

Datasets We evaluate GTNO on five time-dependent PDE benchmarks drawn from three benchmark suites: NS2D from FNO (Li et al., 2021); Diffusion–Reaction (DR2D), Compressible Navier–Stokes (CNS2D), and Shallow Water Equation (SWE2D) from PDEBench (Takamoto et al., 2022); and NS2D-PA from PDEArena (Gupta & Brandstetter, 2022). These benchmarks cover incompressible and compressible fluid dynamics, diffusion–reaction systems, and shallow-water flows, with trajectories varying in their initial conditions or physical parameters. Dataset statistics, temporal horizons, and train/test splits are provided in Appendix A.2.2.

Baselines We compare GTNO with eight representative neural operators spanning diverse architectural families. These include the spectral neural operators FNO (Li et al., 2021) and F-FNO (Tran et al., 2023), the convolutional U-Net baseline (Ronneberger et al., 2015), and the attention-based operator learners Transolver (Wu et al., 2024) and LinearNO (Hu et al., 2026). We further consider DPOT (Hao et al., 2024), an autoregressive operator Transformer designed for time-dependent PDEs; LaMO (Tiwari et al., 2025), which models PDE dynamics using state-space models in a latent space; and SAOT (Zhou et al., 2026), which combines wavelet-based local representations with global Fourier attention. More details are described in Appendix A.2.1.

Training and Evaluation Methods in the main comparison use the same dataset splits, field preprocessing, and evaluation protocol. Following DPOT (Hao et al., 2024), we use full-rollout relative L_2 error; its definition is given in Appendix A.2.3. We choose GTNO's training budget and learning-rate scale with reference to FNO (Li et al., 2021) and Transolver (Wu et al., 2024). GTNO is trained for 500 epochs with AdamW and a OneCycle schedule whose maximum learning rate is 10^{-3}. At evaluation, each model receives T_{in} observed states and predicts the subsequent T_{pred} states. Experiments are conducted on NVIDIA RTX 3090 GPUs with 24 GB of memory. Additional training settings and baseline implementation details are given in Appendices A.2.4 and A.2.1.

4.2 Main Results

Table 1 compares GTNO with eight representative neural operator baselines on five time-dependent PDE benchmarks. GTNO achieves the lowest mean relative L_2 error on all five datasets. Compared with the strongest baseline on each dataset, GTNO reduces the error by 27.4%, 30.0%, 3.0%, 28.6%, and 8.7% on NS2D, DR2D, CNS2D, SWE2D, and NS2D-PA, respectively. These results indicate that GTNO improves performance across different PDE families rather than being specialized to a single equation. The improvement is smaller on benchmarks where existing methods already perform strongly, such as CNS2D. Detailed mean and standard-deviation statistics from multiple runs are reported for GTNO and the strongest non-GTNO baseline on each benchmark in Appendix A.2.5, Table 6.

Table 1: Mean full-rollout relative L_2 errors on benchmarks, averaged over multiple independent runs. Results on SWE2D are reported in units of 10^{-3}. Smaller values indicate better performance. The best and second-best results are highlighted in **bold** and underlined, respectively.

	FNO	PDEBench			PDEArena
METHOD	**NS2D**	**DR2D**	**CNS2D**	**SWE2D** $[10^{-3}]$	**NS2D-PA**
U-Net (2015)	0.180	0.014	0.381	2.255	0.367
FNO (2021)	0.087	0.033	0.178	44.861	0.456
F-FNO (2023)	0.078	<u>0.010</u>	0.132	8.735	0.382
DPOT (2024)	0.106	0.373	0.211	1.911	0.421
Transolver (2024)	0.087	0.061	0.332	95.448	0.420
LaMO (2025)	<u>0.062</u>	0.186	0.202	3.679	0.530
LinearNO (2026)	0.083	0.536	0.172	8.006	0.440
SAOT (2026)	0.072	0.016	<u>0.101</u>	<u>0.970</u>	<u>0.309</u>
GTNO	**0.045**	**0.007**	**0.098**	**0.693**	**0.282**

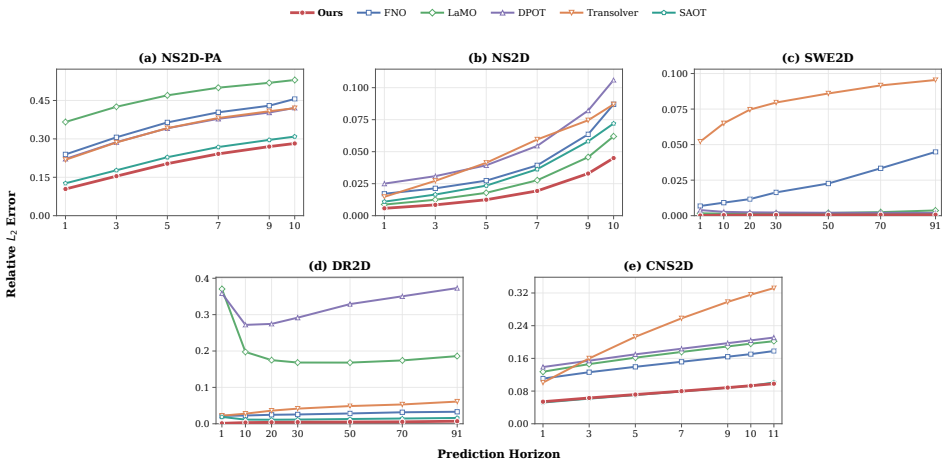


Figure 3: Horizon-wise relative L_2 error on five benchmark datasets: **(a)** NS2D-PA, **(b)** NS2D, **(c)** SWE2D, **(d)** DR2D, and **(e)** CNS2D. Lower error indicates better forecasting performance.

4.3 Long-Horizon Forecasting

We evaluate horizon-wise forecasting stability and find that GTNO's accuracy advantage is sustained throughout the rollout rather than being confined to the final prediction horizon. Table 1 reports errors over the full rollout, while Figure 3 examines the error evolution at intermediate horizons. Across all five benchmarks, GTNO generally maintains the lowest error over the evaluated horizons and exhibits slower error growth on datasets with accumulating prediction errors. Although several baselines on DR2D show non-monotonic error profiles, GTNO remains consistently accurate throughout the rollout.

4.4 Accuracy–Efficiency Trade-off

We evaluate the trade-off between prediction accuracy and training cost on NS2D. GTNO achieves the lowest relative L_2 error while maintaining a training time close to that of one of the fastest baselines and a moderate peak memory requirement. Figure 4(a) shows that GTNO has substantially lower error than FNO, although its per-epoch training time is comparable. Figure 4(b) further shows that its peak training memory is comparable to that of Transolver and lower than that of the more memory-intensive baselines. Causal masking allows multiple temporal positions to be processed in parallel during training, which helps keep the training time moderate. However, storing intermediate temporal representations introduces additional memory overhead. Overall, GTNO provides a favor-

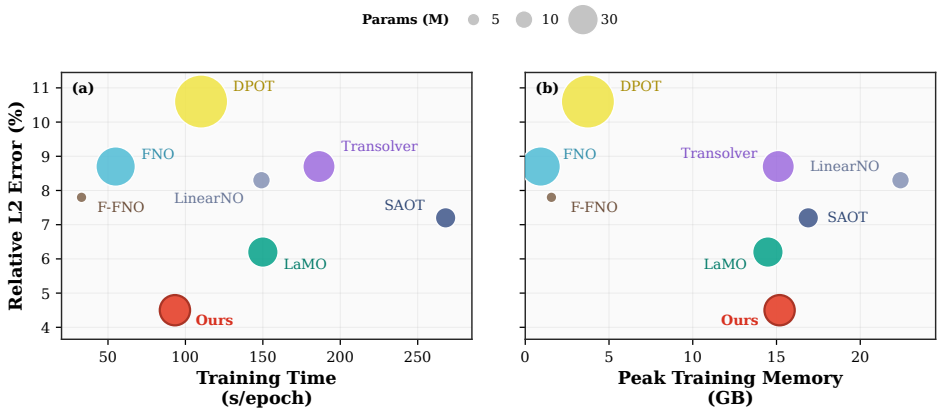


Figure 4: Prediction accuracy versus training cost on NS2D. Relative L_2 error (%) is plotted against (a) average training time per epoch and (b) peak GPU memory usage during training. Marker area indicates the number of parameters. Lower values on both axes are preferred.

Table 2: Ablation study of the Grid–Token architecture. We report the relative L_2 error (↓). Results on SWE2D are reported in units of 10^{-3}. Cross denotes repeated cross-state interaction within the Grid–Token blocks, and N.C. denotes runs that failed to converge.

Variant	Grid	Token	Cross	NS2D	NS2D-PA	SWE2D [10^{-3}]	DR2D	CNS2D
Grid-only	✓	✗	✗	0.289	0.355	8.040	0.0068	0.437
Token-only	✗	✓	✗	0.424	0.564	*N.C.*	*N.C.*	0.440
w/o Cross	✓	✓	✗	0.296	0.398	3.489	0.0069	0.402
Full	✓	✓	✓	**0.045**	**0.282**	**0.693**	**0.0067**	**0.098**

able accuracy–training-cost trade-off without requiring the highest training memory. Additional inference-time efficiency and inference-window sensitivity results are reported in Appendix A.2.6.

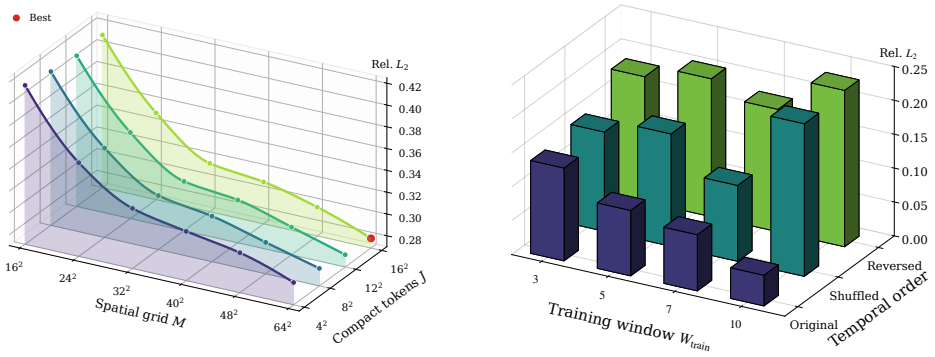
4.5 Ablation Study

To disentangle the contributions of the two representation branches and their cross-state interaction, we construct three variants. *Grid-only* removes the compact token branch, including causal temporal attention and the Grid–Token interactions. Since this variant does not contain a causal temporal module, it is trained with a conventional fixed-window autoregressive objective. *Token-only* removes the structured grid branch and relies solely on the compact temporal representation; its depth is increased to approximately match the parameter count of the Full model. Finally, *w/o Cross* retains both branches but removes their bidirectional interaction within each Grid–Token block, while preserving the final fusion at the decoding stage. Further protocol details are given in Appendix A.2.7.

The Full model achieves the lowest error on all five benchmarks. Removing the token branch and its associated causal training pathway substantially degrades performance on NS2D, NS2D-PA, SWE2D, and CNS2D, while producing a result close to the Full model on DR2D. Removing the structured grid branch leads to larger errors on NS2D, NS2D-PA, and CNS2D, and fails to converge on SWE2D and DR2D, even after adjusting its depth to match the model capacity of the Full model. Removing the repeated cross-state interaction also degrades performance on most benchmarks, although its effect on DR2D is negligible. Overall, the results support the importance of both complementary states and their repeated interaction for accurate PDE forecasting.

4.6 Analysis of Grid–Token Resolution and Temporal Ordering

Grid–Token Resolution Figure 5(a) evaluates the effect of the structured-grid resolution M and token resolution J on PDEArena. The prediction error is more sensitive to M than to J. With $J = 16^2$, increasing M from 16^2 to 64^2 reduces the relative L_2 error from 0.410 to 0.282, corresponding



(a) Grid–token resolution **(b)** Temporal-order intervention

Figure 5: Analysis of representation resolution and temporal ordering in GTNO. **(a)** Sensitivity to the grid (M) and token (J) resolutions on NS2D-PA. **(b)** Temporal-order intervention on NS2D. Models trained with different history lengths W_{train} are evaluated with original, shuffled, or reversed histories while keeping the latest observation fixed. Lower relative L_2 error is better.

to a 31.2% reduction. In contrast, with $M = 64^2$, increasing J from 4^2 to 16^2 reduces the error from 0.290 to 0.282, a smaller 2.8% reduction. This result indicates that the grid resolution has a stronger effect on spatial representation quality, whereas the token state can remain relatively compact while modeling temporal dynamics. We therefore use a higher-resolution grid together with a compact token representation in the main experiments.

Temporal-Order Intervention We further examine how GTNO uses historical states by perturbing the ordering of earlier frames while keeping the latest observation fixed. As shown in Figure 5(b), the effect of disrupting temporal order becomes stronger as the training window increases. At $W_{\text{train}} = 10$, the relative L_2 error increases from 0.045 under the original ordering to 0.225 after shuffling and 0.231 after reversal, corresponding to increases of approximately 400% and 413%, respectively. In contrast, models trained with shorter windows exhibit weaker sensitivity to order perturbations. This pattern indicates that longer training windows encourage GTNO to use historical states as an ordered temporal sequence, whereas models trained with shorter windows rely less on the ordering of earlier frames. Additional intervention results and permutation statistics are provided in Appendix A.2.7.

5 Conclusion

In this work, we introduced GTNO, a Grid–Token Neural Operator for time-dependent PDE prediction. GTNO maintains two complementary representations: a structured grid state for preserving fine-grained spatial information and a compact causal token state for modeling temporal dependencies. Through repeated interaction within Grid–Token blocks, the two states jointly capture spatial evolution and temporal dynamics while retaining their respective representational advantages. Experiments on five time-dependent PDE benchmarks demonstrate that GTNO achieves consistently strong predictive accuracy with a favorable accuracy–training-cost trade-off. Ablation and analysis further validate the importance of the complementary grid and token states, their repeated interaction, and the use of temporally ordered information. Overall, our results suggest that spatial and temporal dynamics need not be represented in the same form or at the same resolution; instead, coupling a high-resolution spatial state with a compact temporal state provides an effective way to model time-dependent physical systems.

AI USE STATEMENT

We used generative AI tools for three limited purposes. First, we used them to assist with manuscript writing and revision, including language polishing, correction of grammatical and writing errors, and checking LaTeX formatting and presentation. Second, we used them to help clarify the assumptions and limiting conditions in the appendix proof, in particular the interpretation of $K \to \infty$ in the corollary and the conditions required for the convergence statement. Third, we used them to reorganize and simplify the research code for release by removing deprecated and redundant components and improving its structure and documentation.

Generative AI tools were not used to generate data, run experiments, choose random seeds, select reported results, or develop the GTNO method. The authors reviewed and verified all AI-assisted text, mathematical statements, formatting changes, and code modifications. The authors take full responsibility for the final manuscript, code, mathematical claims, experimental results, and all other contents of this work.

REPRODUCIBILITY STATEMENT

We support reproducibility through the method description in the main text, the experimental details and proof in the appendix, and an anonymized code repository. Appendix A.2.2 specifies the datasets, splits, and forecasting horizons; Appendices A.2.3 and A.2.4 describe the evaluation metric and training configurations. Baseline implementation choices and run-to-run statistics are documented in Appendices A.2.1 and A.2.5, respectively. The assumptions and proof of the Fourier-truncation result are given in Appendix A.1.2. The anonymized implementation and experimental scripts are available at https://anonymous.4open.science/r/GTNO_test-20A4.

REFERENCES

Benedikt Alkin, Andreas Fürst, Simon Schmid, Lukas Gruber, Markus Holzleitner, and Johannes Brandstetter. Universal physics transformers: A framework for efficiently scaling neural operators. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Ricardo Buitrago Ruiz, Tanya Marwah, Albert Gu, and Andrej Risteski. On the benefits of memory for modeling time-dependent pdes. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 54972–55002, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/89379d5fc6eb34ff98488202fb52b9d0-Paper-Conference.pdf.

Shuhao Cao. Choose a transformer: Fourier or galerkin. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 24924–24940. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/d0921d442ee91b896ad95059d13df618-Paper.pdf.

Gaurav Gupta, Xiongye Xiao, and Paul Bogdan. Multiwavelet-based operator learning for differential equations. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 24048–24062. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/c9e5c2b59d98488fe1070e744041ea0e-Paper.pdf.

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

Jan Hagnberger, Marimuthu Kalimuthu, Daniel Musekamp, and Mathias Niepert. Vectorized Conditional Neural Fields: A framework for solving time-dependent parametric partial differential equations. 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. 17189–17223. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/hagnberger24a.html.

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. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/hao23c.html.

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. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 17616–17635. PMLR, 2024.

Xianglong Hou, Xinquan Huang, and Paris Perdikaris. CFO: Learning continuous-time PDE dynamics via flow-matched neural operators. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=IQhaeSzyup.

Wenjie Hu, Sidun Liu, Peng Qiao, Zhenglun Sun, and Yong Dou. Transolver is a linear transformer: Revisiting Physics-Attention through the lens of linear attention. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 408–416, 2026. doi: 10.1609/aaai.v40i1.37003.

Armand Kassaï Koupaï, Lise Le Boudec, Louis Serrano, and Patrick Gallinari. ENMA: Tokenwise autoregression for continuous neural PDE operators. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=3CYXSMFv55.

Nikola Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew M. 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.

Zijie Li, Kazem Meidani, and Amir Barati Farimani. Transformer for partial differential equations' operator learning. *Transactions on Machine Learning Research*, 2023a.

Zijie Li, Dule Shu, and Amir Barati Farimani. Scalable transformer for PDE surrogate modeling. In *Advances in Neural Information Processing Systems*, volume 36, pp. 28010–28039, 2023b.

Zongyi Li, Nikola 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.

Phillip Lippe, Bas Veeling, Paris Perdikaris, Richard Turner, and Johannes Brandstetter. Pde-refiner: Achieving accurate long rollouts with neural pde solvers. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 67398–67433. Curran Associates, Inc., 2023. doi: 10.52202/075280-2946. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/d529b943af3dba734f8a7d49efcb6d09-Paper-Conference.pdf.

Xinliang Liu, Bo Xu, Shuhao Cao, and Lei Zhang. Mitigating spectral bias for the multiscale operator learning. *Journal of Computational Physics*, 506:112944, 2024. ISSN 0021-9991. doi: https://doi.org/10.1016/j.jcp.2024.112944. URL https://www.sciencedirect.com/science/article/pii/S0021999124001931.

Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11976–11986, 2022.

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. doi: 10.1038/s42256-021-00302-5.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention*, pp. 234–241. Springer, 2015. doi: 10.1007/978-3-319-24574-4_28.

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. Curran Associates, Inc., 2022. doi: 10.52202/068431-0117. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/0a9747136d411fb83f0cf81820d44afb-Paper-Datasets_and_Benchmarks.pdf.

Karn Tiwari, Niladri Dutta, N. M. Anoop Krishnan, and A. P. Prathosh. Latent mamba operator for partial differential equations. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 59639–59667. PMLR, 2025.

Alasdair Tran, Alexander Mathews, Lexing Xie, and Cheng Soon Ong. Factorized fourier neural operators. In *International Conference on Learning Representations*, 2023.

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

Haixu Wu, Huakun Luo, Haowen Wang, Jianmin Wang, and Mingsheng Long. Transolver: A fast transformer solver for PDEs on general geometries. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 53681–53705. PMLR, 2024.

Chenhong Zhou, Jie Chen, and Zaifeng Yang. SAOT: An enhanced locality-aware spectral transformer for solving PDEs. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(34): 28928–28936, 2026. doi: 10.1609/aaai.v40i34.40128.

A Appendix

A.1 Method Details

A.1.1 Local Encoding and Decoding

GTNO uses local window attention to map features between the physical discretization and the structured latent grid. Let q_i denote a query feature and let $\{h_j\}_{j\in\mathcal{N}(i)}$ denote its neighboring source features. The neighborhood $\mathcal{N}(i)$ is constructed according to the normalized query and source coordinates. For each query–source pair, we define their squared spatial distance as

$$d_{ij} = \|\widetilde{s}_j - \widetilde{p}_i\|_2^2 . \tag{1}$$

For a physical point x_n and a latent-grid location g_m, the corresponding instance is denoted by d_{mn}. Within the local neighborhood, the attention score is

$$a_{ij} = \frac{Q_\theta(q_i)^\top K_\theta(h_j)}{\sqrt{d_{\mathrm{a}}}} - \lambda d_{ij}, \tag{2}$$

where Q_θ and K_θ are the query and key projections, d_{a} is the attention dimension, and λ controls the spatial-distance bias. The attention weights are normalized within the local neighborhood as $\alpha_{ij} = \mathrm{Softmax}_{j\in\mathcal{N}(i)}(a_{ij})$. The neighboring source features are aggregated as

$$z_i = \sum_{j\in\mathcal{N}(i)} \alpha_{ij} V_\theta(h_j), \tag{3}$$

where V_θ is the value projection. The unified local attention operator is then defined by the residual refinement

$$\mathcal{A}_\theta\left(q_i; \{h_j, d_{ij}\}_{j\in\mathcal{N}(i)}\right) = z_i + O_\theta\left(\mathrm{LN}(z_i)\right), \tag{4}$$

where O_θ is a lightweight output MLP.

Encoder Let $f_{t,n}$ denote the physical feature at point x_n and time t, and let c_m denote the query feature at latent-grid location g_m. Each latent-grid location attends to nine neighboring physical points. We additionally obtain an initial grid feature $B_{t,m}$ through bilinear interpolation, in which each physical feature is distributed to its surrounding grid locations using coordinate-dependent interpolation weights. The initial grid state is

$$W^0_{t,m} = B_{t,m} + \gamma_{\mathrm{enc}}\, \mathcal{A}_{\theta_{\mathrm{enc}}}\left(c_m; \{f_{t,n}, d_{mn}\}_{n\in\mathcal{N}_{\mathrm{enc}}(m)}\right), \tag{5}$$

where $\mathcal{N}_{\mathrm{enc}}(m)$ denotes the physical points neighboring g_m, and γ_{enc} is the learnable residual scale.

Decoder Let $R^L_{t,m}$ denote the final feature at latent-grid location g_m after L Grid–Token layers. Each physical output point x_n queries a local 5×5 window on the latent grid. The predicted physical state is

$$\widehat{u}_{t+1,n} = P_{\theta_{\mathrm{dec}}}\left[\mathcal{A}_{\theta_{\mathrm{dec}}}\left(e_n; \{R^L_{t,m}, d_{mn}\}_{m\in\mathcal{N}_{\mathrm{dec}}(n)}\right)\right], \tag{6}$$

where e_n is the query feature at physical point x_n, $\mathcal{N}_{\mathrm{dec}}(n)$ denotes its local latent-grid window, and $P_{\theta_{\mathrm{dec}}}$ maps the decoded feature to the predicted physical variables.

Dimensionality The coordinate-based local attention is not restricted to two-dimensional coordinates, but our implementation uses bilinear interpolation and two-dimensional neighborhoods in the encoder and decoder. Extending GTNO to three-dimensional fields would also require corresponding three-dimensional spectral operations. All experiments in this work are two-dimensional and we leave the implementation and evaluation of the three-dimensional extension to future work.

A.1.2 Fourier Truncation Error of the Spectral Bridge

Let $\Omega = \mathbb{T}^d$ denote the d-dimensional torus, and let $w \in H^s(\Omega;\mathbb{R}^{C_{\mathrm{g}}})$. We write $Z = \{z_j\}_{j=1}^J$ for the token states. For $k = (k_1,\dots,k_d) \in \mathbb{Z}^d$, let $\|k\|_2 = (\sum_{i=1}^d k_i^2)^{1/2}$ denote the Euclidean magnitude of the Fourier frequency. We denote the ideal token-conditioned spatial correction by

$$\mathcal{R}(w,Z) \in H^s(\Omega;\mathbb{R}^{C_{\mathrm{g}}}),$$

and its Fourier coefficient at frequency k by $\widehat{\mathcal{R}}(w,Z)(k)$. The Fourier coefficient of w is denoted by $\widehat{w}(k)$. For each retained mode $k \in \mathcal{K}$, the spectral bridge computes

$$\begin{aligned}
s_k &= E_\theta\left(\rho(\widehat{w}(k))\right),\\
\alpha_{kj} &= \frac{\exp\left(Q_\theta(s_k)^\top K_\theta(z_j)/\sqrt{d_{\mathrm{a}}}\right)}{\sum_{r=1}^{J}\exp\left(Q_\theta(s_k)^\top K_\theta(z_r)/\sqrt{d_{\mathrm{a}}}\right)},\\
h_k &= \sum_{j=1}^{J}\alpha_{kj}V_\theta(z_j),\\
\beta_{\theta,k}(\widehat{w}(k),Z) &= \rho^{-1}\left[D_\theta\left(\Psi_\theta(s_k+h_k)\right)\right],
\end{aligned} \tag{7}$$

where ρ separates the real and imaginary components and Ψ_θ denotes the residual feed-forward refinement. The resulting correction is

$$\mathcal{B}_{\theta,\mathcal{K}}(w,Z)(x) = \sum_{k\in\mathcal{K}}\beta_{\theta,k}(\widehat{w}(k),Z)e^{2\pi \mathrm{i}k\cdot x}. \tag{8}$$

For a cutoff radius $K > 0$, define the retained Fourier set

$$\mathcal{K}_K = \left\{k \in \mathbb{Z}^d : \|k\|_2 \le K\right\}. \tag{9}$$

The theorem below uses this radial set for notational simplicity. In the implementation, Fourier modes may instead be retained using a rectangular per-axis cutoff. The same decomposition applies to that choice, with the corresponding truncation set.

Theorem 1. *Let $s > 0$ and suppose that $\mathcal{R}(w, Z) \in H^s(\Omega; \mathbb{R}^{C_g})$. Define the retained-mode coefficient error by*

$$\varepsilon^2_{\theta,K}(w, Z) = \sum_{k \in \mathcal{K}_K} \left\| \widehat{\mathcal{R}}(w, Z)(k) - \beta_{\theta,k}(\widehat{w}(k), Z) \right\|_2^2. \tag{10}$$

Then

$$\begin{aligned} \|\mathcal{R}(w, Z) - \mathcal{B}_{\theta,\mathcal{K}_K}(w, Z)\|^2_{L^2} \le (1 + K^2)^{-s} \|\mathcal{R}(w, Z)\|^2_{H^s} \\ + \varepsilon^2_{\theta,K}(w, Z). \end{aligned} \tag{11}$$

Proof. Let Π_K denote the orthogonal projection onto the Fourier modes in $\mathcal{K}_K$. Since $\mathcal{B}_{\theta,\mathcal{K}_K}(w, Z)$ contains only modes in $\mathcal{K}_K$, orthogonality and Parseval's identity give

$$\begin{aligned} &\|\mathcal{R} - \mathcal{B}_{\theta,\mathcal{K}_K}\|^2_{L^2} \\ &= \|(I - \Pi_K)\mathcal{R}\|^2_{L^2} + \|\Pi_K\mathcal{R} - \mathcal{B}_{\theta,\mathcal{K}_K}\|^2_{L^2} \\ &= \sum_{\|k\|_2 > K} \left\| \widehat{\mathcal{R}}(k) \right\|_2^2 + \varepsilon^2_{\theta,K} \\ &\le (1 + K^2)^{-s} \sum_{\|k\|_2 > K} (1 + \|k\|_2^2)^s \left\| \widehat{\mathcal{R}}(k) \right\|_2^2 + \varepsilon^2_{\theta,K} \\ &\le (1 + K^2)^{-s} \sum_{k \in \mathbb{Z}^d} (1 + \|k\|_2^2)^s \left\| \widehat{\mathcal{R}}(k) \right\|_2^2 + \varepsilon^2_{\theta,K} \\ &= (1 + K^2)^{-s} \|\mathcal{R}\|^2_{H^s} + \varepsilon^2_{\theta,K}, \end{aligned} \tag{12}$$

where the arguments (w, Z) are omitted for brevity. The inequality follows from $\|k\|_2 > K$, which implies

$$(1 + \|k\|_2^2)^{-s} \le (1 + K^2)^{-s}.$$

□

Corollary 1. *Let $\mathcal{C}$ be a compact set of admissible input pairs (w, Z). For each cutoff K, let θ_K denote the parameters of a spectral bridge trained using the mode set $\mathcal{K}_K$. Suppose that*

$$\sup_{(w,Z)\in\mathcal{C}} \|\mathcal{R}(w, Z)\|_{H^s} \le C_{\mathcal{R}} \tag{13}$$

for some constant $C_{\mathcal{R}}$, and that the learned coefficient error satisfies

$$\sup_{(w,Z)\in\mathcal{C}} \varepsilon^2_{\theta_K,K}(w, Z) \longrightarrow 0 \qquad \text{as } K \longrightarrow \infty. \tag{14}$$

Then

$$\sup_{(w,Z)\in\mathcal{C}} \|\mathcal{R}(w, Z) - \mathcal{B}_{\theta_K,\mathcal{K}_K}(w, Z)\|_{L^2} \longrightarrow 0 \qquad \text{as } K \longrightarrow \infty. \tag{15}$$

The bound separates the total approximation error into two terms: the error caused by omitting high-frequency Fourier modes and the error in predicting the coefficients of the retained modes. For a correction with higher spatial regularity, the Fourier coefficients decay more rapidly, so fewer modes are needed to capture most of its spatial structure. As the cutoff K increases, the truncation error therefore decreases, provided that the retained-mode coefficients can also be learned accurately. The limit $K \to \infty$ refers to a sequence of increasing mode budgets and, for discrete implementations, sufficiently fine spatial grids.

A.2 Experimental Details

A.2.1 Baseline Implementations

We use publicly available baseline implementations. Unless stated otherwise, we retain the released architecture settings used for the FNO Navier–Stokes benchmark and apply our common training and data protocol. For FNO, we use the implementation provided by DPOT (Hao et al., 2024). For

Table 3: Exploratory ENMA result on NS2D-PA. Entries are physical-space relative L_2 errors over the first T predicted frames.

Method	$T = 1$	$T = 5$	$T = 10$
ENMA	0.439	0.530	0.570

Table 4: Summary of the benchmark datasets used in our experiments. N denotes the spatial resolution, N_t denotes the total number of temporal frames, and $T_{\text{in}} \rightarrow T_{\text{pred}}$ denotes the input context length and prediction horizon. Dim denotes the number of physical variables at each spatial location.

Benchmark	**Dataset**	**Dim**	N	N_t	$T_{\text{in}} \rightarrow T_{\text{pred}}$	**Train**	**Test**
FNO	NS2D	1	64^2	20	$10 \rightarrow 10$	1000	200
PDEBench	DR2D	2	64^2	101	$10 \rightarrow 91$	900	100
PDEBench	CNS2D	4	64^2	21	$10 \rightarrow 11$	900	100
PDEBench	SWE2D	1	64^2	101	$10 \rightarrow 91$	900	100
PDEArena	NS2D-PA	3	64^2	14	$4 \rightarrow 10$	1000	250

DPOT, we use the approximately 30M-parameter variant and train it from scratch under the same training configuration as the other evaluated methods. The original SAOT configuration exceeded the memory capacity of our 24 GB GPUs; we reduce its depth from 8 to 6 and hidden width from 256 to 192.

Scope of the baseline comparison MemNO (Buitrago Ruiz et al., 2025) and ENMA (Koupaï et al., 2025) are related temporal modeling approaches discussed in the main text. We did not evaluate MemNO because a runnable official implementation was not available to us when these experiments were conducted. For ENMA, we adapted its official implementation to the PDEArena NS2D setting.

ENMA experiment on NS2D-PA We used the official ENMA implementation and retained its model and generation settings, while adapting the data pipeline to the NS2D-PA benchmark. The experiment uses four observed frames to predict ten future frames on 64^2 grids, with 1,000 training and 250 test trajectories. Following ENMA's two-stage procedure, we first trained its VAE and then trained the latent-space autoregressive model; each stage ran for 500 epochs with batch size 4. The reported errors were obtained by decoding the predictions into physical space. ENMA's separate VAE training and flow-matching generation differ from GTNO's training procedure. We therefore provide it as an additional reference in Table 3.

A.2.2 Benchmark Datasets

This appendix provides the dataset-level details omitted from the main text. Table 4 summarizes the spatial resolution, number of physical variables, temporal length, input–prediction horizons, and train/test split sizes used in our experiments. The governing equations and prediction targets are described afterward for completeness.

NS2D The NS2D dataset is the two-dimensional Navier–Stokes vorticity benchmark introduced in the FNO work. The quantity of interest is the vorticity $w(\mathbf{x}, t)$ on $\Omega = [0, 1]^2$, which satisfies

$$\partial_t w + \mathbf{u} \cdot \nabla w = \nu \Delta w + f(\mathbf{x}), \qquad \nabla \cdot \mathbf{u} = 0. \tag{16}$$

DR2D DR2D is the two-variable diffusion–reaction benchmark from PDEBench (Takamoto et al., 2022). It describes an activator u and an inhibitor v on the physical domain $\Omega = (-1, 1)^2$:

$$\begin{aligned} \partial_t u &= D_u \Delta u + u - u^3 - k - v, \\ \partial_t v &= D_v \Delta v + u - v. \end{aligned} \tag{17}$$

CNS2D CNS2D corresponds to the compressible Navier–Stokes dataset in PDEBench. The velocity, pressure, and density fields $\mathbf{u}(\mathbf{x}, t)$, $p(\mathbf{x}, t)$, and $\rho(\mathbf{x}, t)$ are predicted on $\Omega = [0, 1]^2$. The

governing equations are

$$\partial_t \rho + \nabla \cdot (\rho \mathbf{u}) = 0, \tag{18}$$

$$\rho \left(\partial_t \mathbf{u} + \mathbf{u} \cdot \nabla \mathbf{u}\right) = -\nabla p + \eta \Delta \mathbf{u} + \left(\zeta + \frac{\eta}{3}\right) \nabla(\nabla \cdot \mathbf{u}), \tag{19}$$

$$\partial_t \left(\frac{3}{2} p + \frac{\rho \|\mathbf{u}\|^2}{2}\right) = -\nabla \cdot \left[\left(\varepsilon + p + \frac{\rho \|\mathbf{u}\|^2}{2}\right) \mathbf{u} - \mathbf{u} \cdot \boldsymbol{\sigma}'\right]. \tag{20}$$

SWE2D SWE2D is the shallow-water benchmark from PDEBench (Takamoto et al., 2022), using a radial dam-break configuration on $\Omega = [-2.5, 2.5]^2$. We predict the water-depth field $h(\mathbf{x}, t)$. With horizontal velocity $\mathbf{u} = (u_x, u_y)$, the governing equations are

$$\partial_t h + \nabla \cdot (h\mathbf{u}) = 0, \tag{21}$$

$$\partial_t (h\mathbf{u}) + \nabla \cdot \left(h\mathbf{u} \otimes \mathbf{u} + \frac{1}{2} g h^2 I\right) = -gh \nabla b. \tag{22}$$

NS2D-PA NS2D-PA is the unconditioned two-dimensional Navier–Stokes smoke benchmark from PDEArena (Gupta & Brandstetter, 2022). The prediction channels are the transported scalar field u and the velocity components v_x and v_y. Writing $\mathbf{v} = (v_x, v_y)$, the coupled dynamics have the form

$$\begin{aligned} \partial_t u + \mathbf{v} \cdot \nabla u &= 0, \\ \partial_t \mathbf{v} + \mathbf{v} \cdot \nabla \mathbf{v} &= -\nabla p + \mu \Delta \mathbf{v} + \beta u \mathbf{e}_y, \\ \nabla \cdot \mathbf{v} &= 0. \end{aligned} \tag{23}$$

A.2.3 Evaluation Metric

Following the full-rollout evaluation protocol of DPOT, we evaluate prediction accuracy using the relative L_2 error. Given T_{in} observed states, each model autoregressively predicts the subsequent T_{pred} states. We denote the predicted and reference trajectories by

$$\widehat{\mathbf{Y}}, \mathbf{Y} \in \mathbb{R}^{B \times N \times T_{\text{pred}} \times C},$$

where B is the number of evaluation samples, N is the number of spatial locations, and C is the number of physical variables. For sample b and physical variable c, we compute:

$$e_{b,c} = \frac{\left\| \widehat{\mathbf{Y}}_{b,:,:,c} - \mathbf{Y}_{b,:,:,c} \right\|_2}{\|\mathbf{Y}_{b,:,:,c}\|_2}, \qquad \text{Rel } L_2 = \frac{1}{BC} \sum_{b=1}^{B} \sum_{c=1}^{C} e_{b,c}.$$

Here, the L_2 norm is taken jointly over space and the full evaluated time window, separately for each sample and physical variable, before averaging over samples and variables. The length of this window is the number of supervised training steps for the training objective and T_{pred} for full-rollout evaluation. For the horizon-wise analysis at horizon h, the metric is evaluated over the first h predicted steps.

A.2.4 Training Configuration

GTNO uses rotary positional embeddings (RoPE) in temporal attention. The token-to-grid spectral correction is restricted to the retained modes before spatially varying gated fusion; the truncation analysis concerns that pre-gating correction. For all datasets, GTNO is trained for 500 epochs with AdamW and a OneCycle schedule whose maximum learning rate is 10^{-3}. Model configurations are summarized in Table 5.

Training subsequences For trajectories longer than the required training sequence, we sample a contiguous subsequence from the full training trajectory. Let W denote the training history length, set to T_{in} in Table 4. GTNO takes a segment of $W + 1$ frames and pairs its first W frames with the last W frames as shifted targets; the final frame is a label, not an additional input observation. For a fixed-window model with history length W and training rollout length H, a segment contains $W + H$ frames. The first W initialize the history, and the next H are predicted autoregressively

Table 5: Training and model configurations of GTNO. L is the number of Grid–Token Evolution Layers; M and J are the spatial resolutions of the grid and token representations, respectively; and d_g and d_z are their feature dimensions. Modes gives the Fourier truncation along the two spatial axes (e.g., 16^2 denotes 16 retained modes per axis).

Dataset	Loss	Epochs	Optimizer	Max LR	Batch	L	M	J	d_g	d_z	Modes
NS2D	Rel. L_2	500	AdamW	10^{-3}	4	6	64^2	16^2	64	256	16^2
NS2D-PA	Rel. L_2	500	AdamW	10^{-3}	4	6	64^2	16^2	64	256	16^2
DR2D	Rel. L_2	500	AdamW	10^{-3}	2	6	64^2	16^2	64	256	16^2
SWE2D	Rel. L_2	500	AdamW	10^{-3}	2	6	64^2	16^2	64	256	16^2
CNS2D	Rel. L_2	500	AdamW	10^{-3}	2	8	64^2	16^2	64	256	16^2

Table 6: Full-horizon relative L_2 errors over several independent runs, reported as mean $\pm$ standard deviation. For readability, only the strongest non-GTNO baseline is shown for each benchmark, with its name given below the corresponding entry. SWE2D results are reported in units of 10^{-3}. Smaller values indicate better performance.

Model	**FNO**	**PDEBench**			**PDEArena**
	NS2D	**DR2D**	**CNS2D**	**SWE2D** $[10^{-3}]$	**NS2D-PA**
Best competing baseline	$\underline{0.062}\pm 0.00224$ (LaMO)	$\underline{0.010}\pm 0.00161$ (F-FNO)	$\underline{0.101}\pm 0.00156$ (SAOT)	$\underline{0.970}\pm 0.0340$ (SAOT)	$\underline{0.309}\pm 0.0217$ (SAOT)
GTNO	$\mathbf{0.045}\pm 0.00417$	$\mathbf{0.007}\pm 0.000504$	$\mathbf{0.098}\pm 0.00139$	$\mathbf{0.693}\pm 0.160$	$\mathbf{0.282}\pm 0.00162$

with feedback of the model's outputs. In the fixed-window training protocol used here, $H = W$, requiring $2W$ frames. All predicted training frames are concatenated along time for the common relative L_2 loss. For the PDEArena fixed-window baselines, $W = 4$ history frames and $H = 4$ supervised autoregressive targets are used during training. At evaluation, the same models receive four observed frames and are rolled out for ten future frames, i.e., the reported protocol is $4 \to 10$. GTNO is evaluated on the same physical rollout, while its KV cache may grow to ten entries from generated states.

Shared settings Unless explicitly varied, ablations and analysis experiments use the corresponding main-experiment settings, including preprocessing, optimizer, training duration, and evaluation protocol.

A.2.5 Results across Random Seeds

We repeat the experiments with different random seeds (e.g., 0, 1, and 2) and aggregate the resulting errors. This subsection concerns the main deterministic experiments. For a configuration with evaluated seed set $\mathcal{S}$ and $R = |\mathcal{S}|$, let E_s be its full-horizon relative L_2 error. We report the sample mean and standard deviation:

$$\overline{E} = \frac{1}{R}\sum_{s\in\mathcal{S}} E_s, \qquad \sigma_E = \sqrt{\frac{1}{R-1}\sum_{s\in\mathcal{S}} \left(E_s - \overline{E}\right)^2}. \tag{24}$$

Table 6 provides detailed run-to-run statistics for GTNO across the benchmarks. For readability, we also report the corresponding statistics for the strongest non-GTNO baseline on each benchmark, identified by its mean error in Table 1. The main table provides the complete mean comparison with all baselines.

A.2.6 Inference-Time Efficiency and Window Sensitivity

Figure 6 compares the inference accuracy and computational cost of GTNO and the baselines on NS2D. GTNO achieves the lowest relative L_2 error among the compared methods, but incurs higher inference time and peak memory than several lightweight baselines.

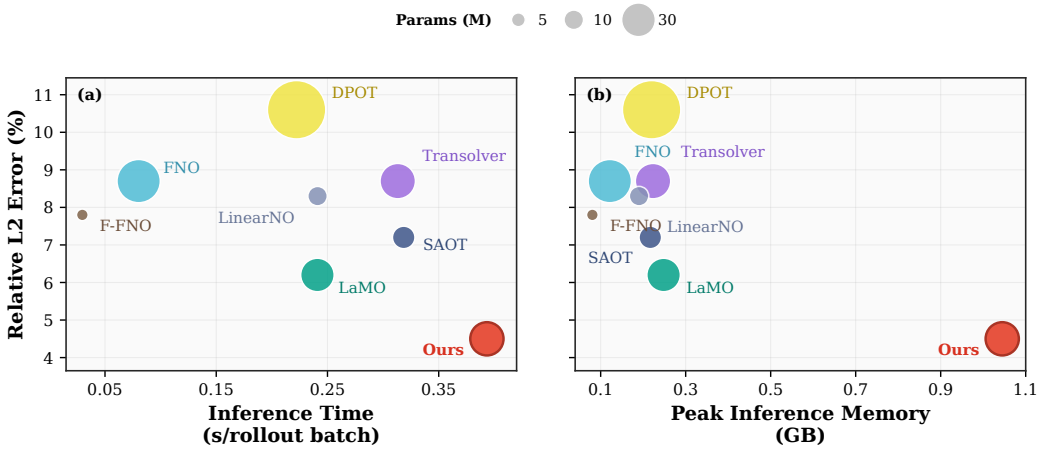


Figure 6: Inference-time accuracy and computational cost on NS2D. Relative L_2 error is compared with inference time per rollout batch (left) and peak inference memory (right). Marker area indicates the number of model parameters. GTNO achieves the lowest prediction error among the compared methods, while its inference cost is higher than that of several lightweight baselines.

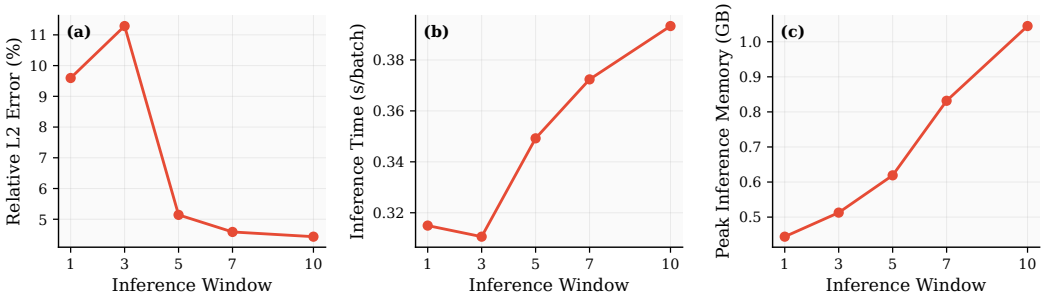


Figure 7: Sensitivity of GTNO to the inference window on NS2D. The three panels show the relative L_2 error, inference time per rollout batch, and peak inference memory as functions of the retained KV-cache window W_{test}.

Figure 7 illustrates the accuracy–cost trade-off associated with the inference window. Overall, larger values of W_{test} tend to reduce the relative L_2 error, while increasing inference time and peak memory because more key–value states are retained. When the cache exceeds its capacity, the oldest entries are discarded. We use $W_{\text{test}} = 10$ in the main experiments to favor predictive accuracy; smaller windows offer a more memory-efficient alternative when computational resources are limited.

A.2.7 Additional Ablation and Robustness Studies

Component ablation protocol All component variants use the main-experiment data splits and evaluation metric. Grid-only removes the token branch, including its causal temporal attention, and both within-layer interaction pathways. Since this removes the explicit temporal module, a fixed-window autoregressive formulation supplies temporal conditioning. Token-only removes the structured grid branch; its depth is increased to approximately match the parameter count of Full. The w/o Cross variant retains both branches and the final decoding fusion but removes their repeated within-layer interaction. These comparisons test the complete systems under their stated temporal formulations; they do not isolate the effect of temporal conditioning while holding every other operation fixed.

Directional Grid–Token Interaction We remove either the grid-to-token router or the token-to-grid pathway to evaluate the two interaction directions on NS2D and SWE2D. The spectral bridge

implements the token-to-grid pathway, so removing this direction means removing the bridge as a whole. This ablation evaluates the contribution of the implemented pathway. Removing grid-to-token changes the full-rollout errors from 0.045 to 0.047 on NS2D and from 0.693 to 0.831 on SWE2D. Removing token-to-grid increases them to 0.422 and 1.899, respectively, with SWE2D errors in units of 10^{-3}. The larger degradation supports the importance of the token-to-grid spectral pathway in these tasks. The results are shown in Table 7.

Table 7: Directional interaction ablations on NS2D and SWE2D. Entries are relative L2 errors over the first T predicted steps; SWE2D values are in units of 10^{-3}. The full rollout length is T = 10 for NS2D and T = 91 for SWE2D. Change denotes the full-rollout error increase relative to Full, calculated from the displayed values.

Dataset	Variant	Spectral bridge	$T = 1$	$T = 5$	Final T	Change
NS2D	Full	Yes	**0.006**	**0.012**	**0.045**	–
	No grid-to-token	Yes	0.007	0.014	0.047	+4.4%
	No token-to-grid	No	0.022	0.186	0.422	+837.8%
SWE2D	Full	Yes	**0.148**	**0.292**	**0.693**	–
	No grid-to-token	Yes	0.178	0.362	0.831	+19.9%
	No token-to-grid	No	0.257	0.534	1.899	+174.0%

Resolution-Dependent Component Ablation The Full model achieves the lowest error for both datasets at both resolutions, consistent with the component-ablation results in Table 2. Removing either state or the repeated Grid–Token interaction increases the error, and this pattern persists when the resolution is increased from 64^2 to 128^2. These results suggest that the benefits of the complete Grid–Token architecture are not limited to the default resolution.

Details of the Temporal-Order Intervention The intervention is applied once to the initial observed history: all earlier frames are reversed or shuffled, while the latest observed frame is kept fixed. The usual positional-index and RoPE computation is retained, with no special positional remapping introduced for this experiment. Subsequent rollout proceeds autoregressively without further permutations or replacement by ground-truth states. Each intervention generates its own predicted trajectory; later numerical predictions are not constrained to equal those of the clean run. Table 9 reports the resulting errors. The standard deviation for Shuffle-old is computed across five permutations, not across independently trained models.

Spatial-Resolution Generalization We study resolution transfer on the PDEArena NS2D-PA benchmark. Models trained at 16^2, 32^2, and 64^2 are evaluated at 64^2 and 128^2. We also include native $128^2 \rightarrow 128^2$ training and evaluation as a reference. For transfer to a higher resolution, predictions are decoded on the target query grid and downsampled only for autoregressive feedback.

At both target resolutions, models trained on higher-resolution inputs achieve lower errors. For example, at the 128^2 target resolution, the $T = 10$ error decreases from 0.607 to 0.420 and 0.289 as the training resolution increases from 16^2 to 32^2 and 64^2; native 128^2 training further reduces it to 0.252. This trend suggests that access to finer spatial information during training improves prediction when transferring to a higher-resolution grid. The results are shown in Table 10.

Spectral-Mode Truncation Table 11 examines the number of Fourier modes retained by the spectral bridge on NS2D. When only a few low-frequency modes are retained, the bridge has limited capacity to represent the spatial correction, and the resulting errors remain high. Increasing the mode budget to 8×8 substantially reduces the error, with a further improvement at 16×16. Overall the results show that retaining a sufficient range of modes is important for the spectral bridge to contribute effectively.

A.3 Visual Comparisons with Ground Truth

Figures 8–15 compare GTNO predictions with the corresponding ground-truth fields at selected rollout horizons. Each figure also includes a spatial relative-error map. GTNO preserves the main

Table 8: Component ablation at two spatial resolutions. We report relative L_2 errors over the predicted steps on NS2D-PA and CNS2D. Lower values indicate better performance.

				NS2D-PA		CNS2D	
Variant	Grid	Token	Cross	64^2	128^2	64^2	128^2
Grid-only	✓	✗	✗	0.355	0.342	0.437	0.418
Token-only	✗	✓	✗	0.564	0.565	0.440	0.437
w/o Cross	✓	✓	✗	0.398	0.389	0.402	0.416
Full	✓	✓	✓	**0.282**	**0.252**	**0.098**	**0.076**

Table 9: Temporal-order interventions on NS2D at horizon $T = 10$. Reverse-old reverses all historical frames except the latest observed frame. Shuffle-old reports the mean $\pm$ standard deviation over five fixed permutations, with the latest observed frame kept fixed. Δ_{rev} and Δ_{shuf} denote the relative error changes from the clean result; positive values indicate higher error.

W_{train}	Clean	Reverse-old	Δ_{rev} (%)	Shuffle-old	Δ_{shuf} (%)
3	0.137	0.184	+34.3	0.146 ± 0.019	+6.6
5	0.097	0.204	+110.3	0.166 ± 0.025	+71.1
7	0.083	0.181	+118.1	0.112 ± 0.027	+34.9
10	0.045	0.231	+413.3	0.225 ± 0.045	+400.0

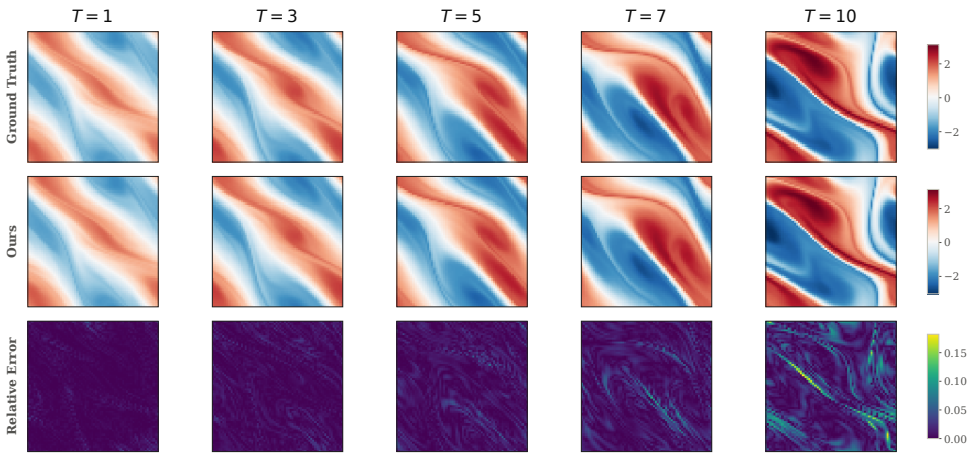


Figure 8: Comparison between GTNO predictions and ground truth for NS2D vorticity. Columns correspond to rollout steps $T = 1$, 3, 5, 7, and 10; rows show the ground truth, GTNO prediction, and relative-error map.

spatial structures over the rollout, while the error maps show the remaining local differences. These visual comparisons complement the quantitative results reported in the main text.

Table 10: Spatial-resolution generalization on NS2D-PA. Entries report relative L_2 error over the first T predicted steps. For each target resolution, Δ_{10} is the percentage increase in $T = 10$ error relative to training and evaluation at that same resolution, calculated from the displayed values. Lower errors are better.

	Target: 64^2				Target: 128^2			
Train	$T = 1$	$T = 5$	$T = 10$	Δ_{10} (%)	$T = 1$	$T = 5$	$T = 10$	Δ_{10} (%)
16^2	0.495	0.543	0.582	+106.4	0.531	0.571	0.607	+140.9
32^2	0.256	0.320	0.379	+34.4	0.323	0.368	0.420	+66.7
64^2	**0.104**	**0.202**	**0.282**	–	0.155	0.221	0.289	+14.7
128^2	–	–	–	–	**0.089**	**0.176**	**0.252**	–

Table 11: Sensitivity to Fourier-mode truncation in the spectral bridge on NS2D. Modes are specified per spatial axis. Changes are relative to 16×16 and calculated from the displayed full-horizon relative L_2 errors. Lower errors are better.

Retained modes	2×2	4×4	8×8	16×16
Relative L_2 error (↓)	0.137	0.166	0.074	**0.045**
Change vs. 16×16 (%)	+204.4	+268.9	+64.4	–

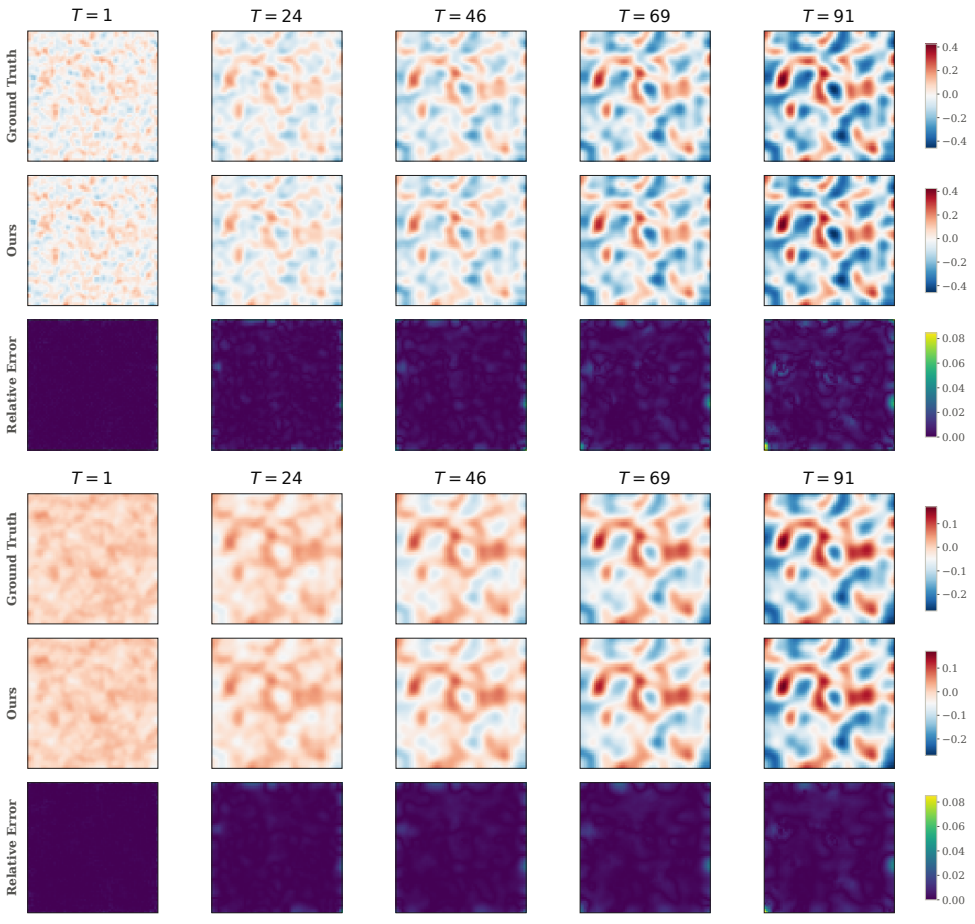


Figure 9: Comparison between GTNO predictions and ground truth for the DR2D variables u and v. Each panel shows the ground truth, GTNO prediction, and relative-error map at rollout steps $T = 1$, 24, 46, 69, and 91.

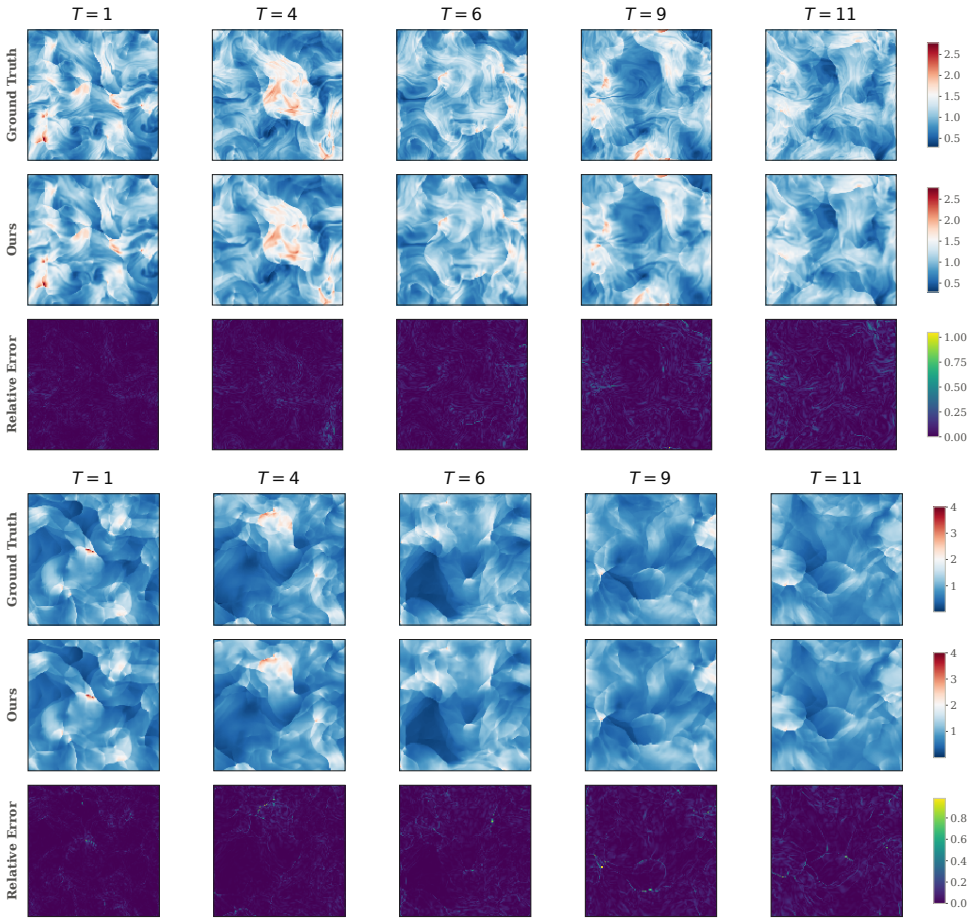


Figure 10: Comparison between GTNO predictions and ground truth for CNS2D density and pressure. Each panel shows the ground truth, GTNO prediction, and relative-error map at rollout steps $T = 1$, 4, 6, 9, and 11.

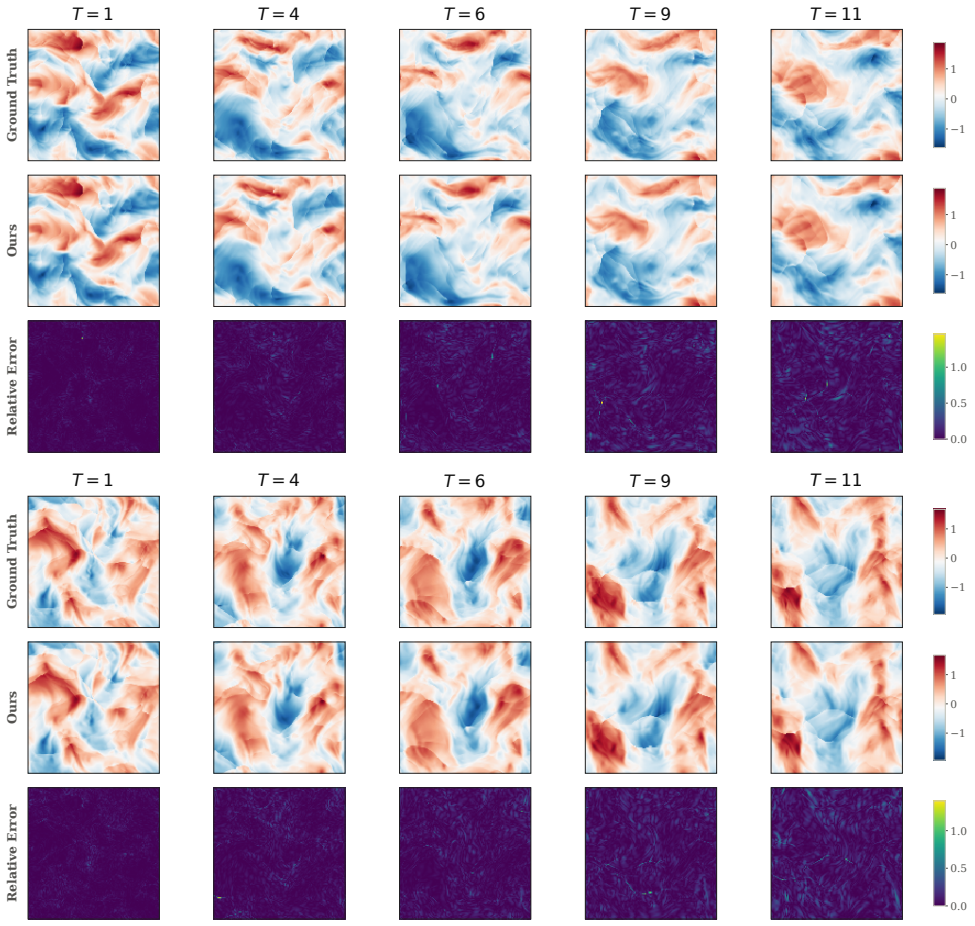


Figure 11: Comparison between GTNO predictions and ground truth for the CNS2D velocity components v_x and v_y. Each panel shows the ground truth, GTNO prediction, and relative-error map at rollout steps $T = 1$, 4, 6, 9, and 11.

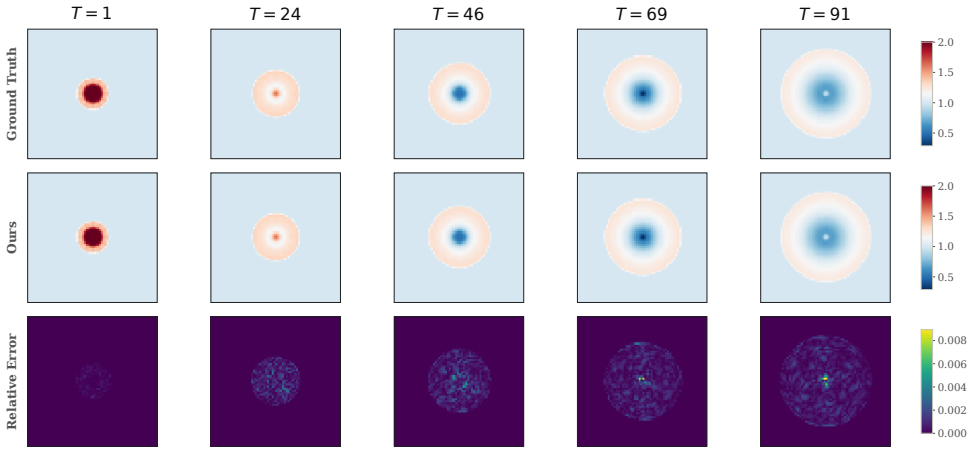


Figure 12: Comparison between GTNO predictions and ground truth for the SWE2D water-height field. Columns correspond to rollout steps $T = 1$, 24, 46, 69, and 91; rows show the ground truth, GTNO prediction, and relative-error map.

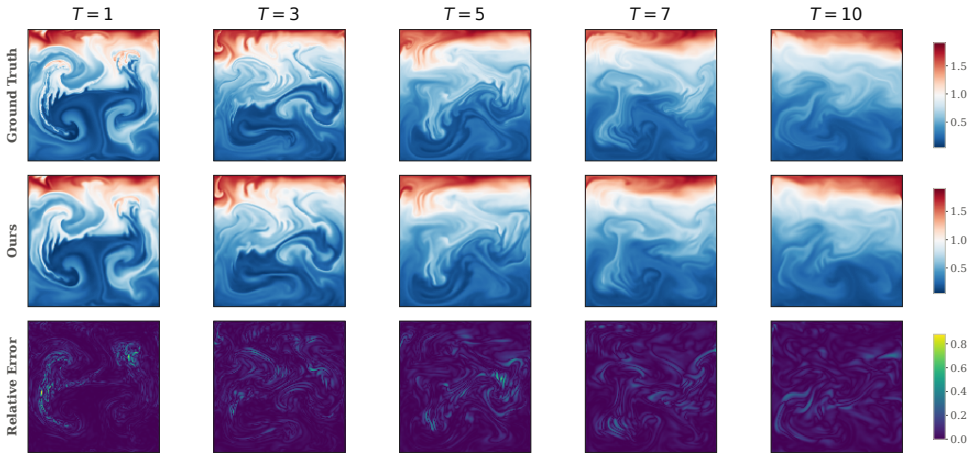


Figure 13: Comparison between GTNO predictions and ground truth for the NS2D-PA transported scalar field u. Columns correspond to rollout steps $T = 1$, 3, 5, 7, and 10; rows show the ground truth, GTNO prediction, and relative-error map.

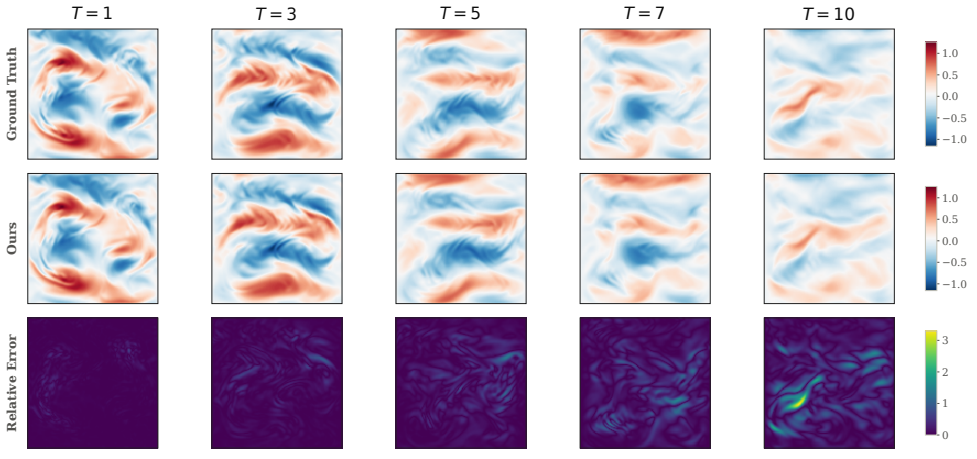


Figure 14: Comparison between GTNO predictions and ground truth for the NS2D-PA channel v_x. Columns correspond to rollout steps $T = 1$, 3, 5, 7, and 10; rows show the ground truth, GTNO prediction, and relative-error map.

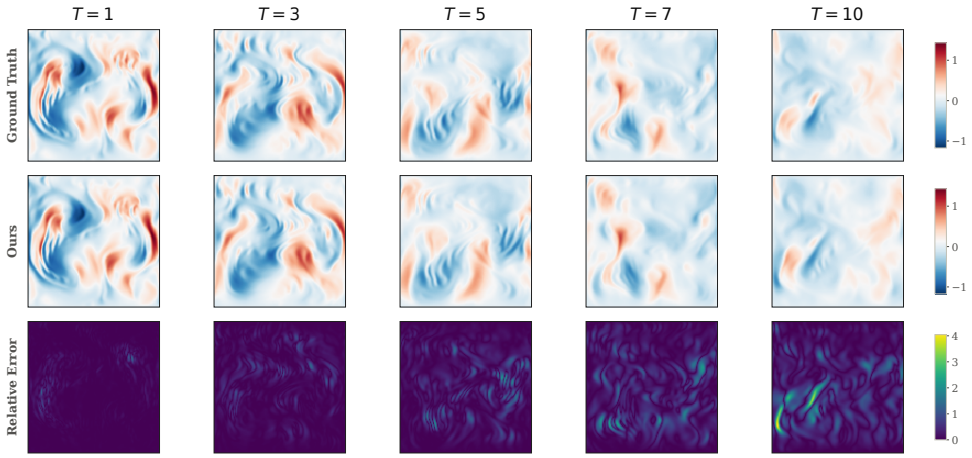


Figure 15: Comparison between GTNO predictions and ground truth for the NS2D-PA channel v_y. Columns correspond to rollout steps $T = 1$, 3, 5, 7, and 10; rows show the ground truth, GTNO prediction, and relative-error map.