SCALABLE OPERATOR LEARNING VIA TENSORIZED TRANSFORMER

Anonymous authors
Paper under double-blind review

ABSTRACT

Attention-based models are widely used to learn solution operators for partial differential equations (PDEs), but the computational cost of attention grows quadratically with the number of tokens. Windowed attention reduces this cost but limits direct interactions to local neighborhoods. Factorized attention attends along subsets of coordinates, but its approximation properties and practical benefits remain unclear. We study a tensorized Transformer that stacks independently parameterized blocks, each attending along a single coordinate, so that one sweep over all coordinates yields global interaction. Viewing attention as an integral operator, we approximate kernels by finite sums of tensor products of one-dimensional kernels. We then prove that tensorized Transformers approximate continuous solution operators uniformly on compact sets. We compare against full and windowed attention under matched model and training configurations on four two- and three-dimensional benchmarks from The Well. Across three model sizes, tensorized attention reduces VRMSE by 17.5% relative to these alternatives and runs 2.1× faster than full attention.

1 INTRODUCTION

Operator learning approximates mappings between function spaces. It provides data-driven surrogates for partial differential equations (PDEs) when repeated numerical simulation is expensive (Lu et al., 2021; Kovachki et al., 2023; 2024). Learned solution operators generalize across initial conditions, coefficients, and discretizations. They have been applied to weather forecasting (Pathak et al., 2022; Lam et al., 2023; Bodnar et al., 2025), aerodynamics (Li et al., 2023b), solid and fluid mechanics (Liu et al., 2022; Rashid et al., 2022; Wang et al., 2026; Kou et al., 2025), and geoscience (Lee et al., 2024; Taccari et al., 2024; Guo et al., 2024).

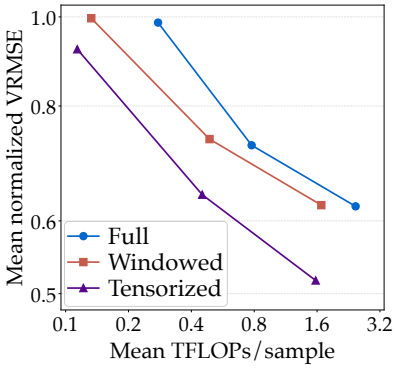

Figure 1: Accuracy versus inference cost at three model sizes. VRMSE is normalized per benchmark by the maximum across models, then averaged.

DeepONet (Lu et al., 2021) and FNO (Li et al., 2021) established end-to-end frameworks for learning nonlinear operators. Attention-based models such as ViTO (Ovadia et al., 2024), CViT (Wang et al., 2025), and Micrometer (Wang et al., 2026) capture long-range interactions. PDE foundation models such as DPOT (Hao et al., 2024) and Walrus (McCabe et al., 2026) pretrain these architectures at scale across operators, geometries, and physical regimes. As token counts grow in three-dimensional and spatiotemporal settings, the choice of attention pattern increasingly determines both accuracy and cost.

Full attention scales quadratically with the number of tokens (Vaswani et al., 2017), which is prohibitive for three-dimensional and spatiotemporal PDE data. Quasi-linear alternatives such as AFNO (Guibas et al., 2022) remain costly at large token counts. **Windowed attention** (Liu et al., 2021), used in Poseidon (Herde et al., 2024) and P3D (Holzschuh et al., 2026), reduces this cost but lets distant tokens interact only indirectly. **Factorized attention** attends along one coordinate at a time, as in axial attention (Ho et al., 2020). PDE models such as FactFormer (Li et al., 2023a),

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.

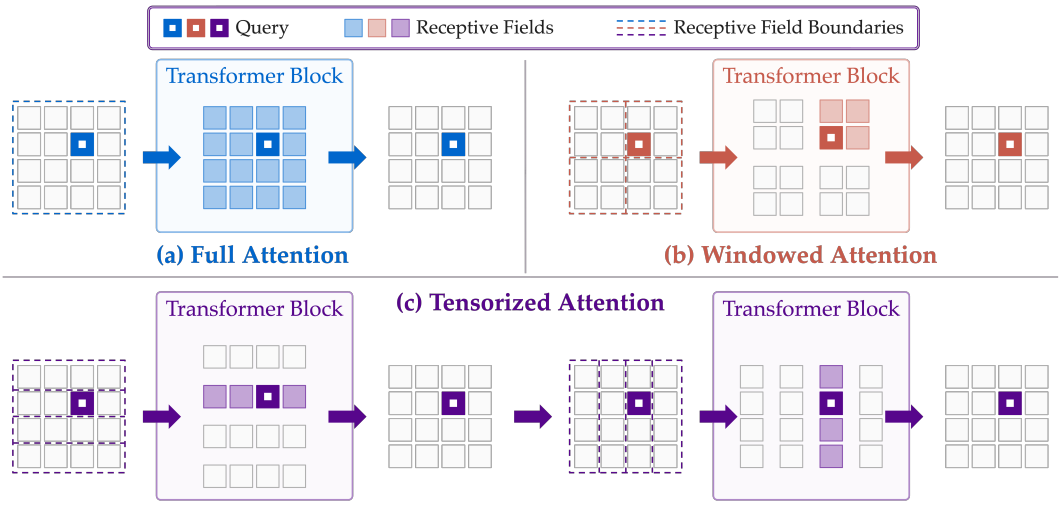

Figure 2: Attention patterns on a 2D token grid. (a) Full attention connects each query to all tokens. (b) Windowed attention restricts interactions to local windows. (c) Tensorized attention alternates blocks along rows and columns, giving global interaction.

AViT (McCabe et al., 2024), and MORPH (Rautela et al., 2026) couple the coordinate attentions within a single block, either by multiplying attention matrices or by summing parallel outputs. It remains unclear, both theoretically and under controlled comparisons, why factorization suits PDEs and how the coordinates should be coupled.

We address this question through *kernel tensorization*. Viewing attention as an integral operator (Kovachki et al., 2023), we approximate multidimensional kernels by sums of products of one-dimensional kernels. For the heat equation, this factorization is exact. This view leads to the **tensorized Transformer**, which follows the sequential structure of the Axial Transformer (Ho et al., 2020). Each block attends along a single coordinate with its own parameters and MLP, and blocks are composed in sweeps over the coordinates (Fig. 2(c)). Unlike a parallel sum, composition produces product kernels that couple all tokens after one sweep. The intermediate MLPs also let each coordinate's attention depend on what earlier coordinates have aggregated. We hypothesize that, under a fixed parameter budget, these one-dimensional factors are easier to learn than a full multi-dimensional kernel. Combined with the lower cost of attending along one coordinate, tensorization would then improve accuracy and efficiency together, and we test this through controlled comparisons. Our contributions are:

- We formulate tensorized attention as an interchangeable component of Transformer frameworks for PDE modeling, enabling controlled comparisons with full and windowed attention.
- We realize tensor-product kernel decompositions using self-attention and pointwise maps, and prove approximation results for continuous solution operators under appropriate assumptions.
- We evaluate attention patterns on two- and three-dimensional PDE benchmarks under matched model and training configurations, achieving an accuracy improvement of 17.5% over alternative attention patterns, an average speedup of $2.1\times$ over full attention, and the best scalability among the evaluated methods.

2 Problem Setup

We consider time-dependent PDEs on the domain $\Omega = \prod_{j=1}^{d} \Omega_j$, with bounded $\Omega_j \subset \mathbb{R}$ and $d \in \{2, 3\}$, of the form

$$\begin{cases} \mathcal{L}_\lambda u(x,t) = f(x,t), & x \in \Omega, \quad t > 0, \\ \mathcal{B}_\lambda u(x,t) = g(x,t), & x \in \partial\Omega, \quad t > 0, \\ u(x,0) = I(x), & x \in \Omega, \end{cases} \tag{1}$$

where the differential operator $\mathcal{L}$ and the forcing term f define the PDE parameterized by λ, the boundary operator $\mathcal{B}$ and boundary condition g define the solution on the boundary, and I is the initial condition.

Let $\mathcal{A}$ denote the function space that contains the input function a, and $\mathcal{U}$ denote the space for the corresponding solution u. The operator $\mathcal{G} : \mathcal{A} \to \mathcal{U}$ is defined by

$$(\mathcal{G}a)(x,t) = u(x,t), \qquad x \in \Omega, \qquad t > 0. \tag{2}$$

Consider evolution problems, where the operator $\mathcal{G}$ predicts $u(x,t)$ for $t = \tau > 0$ from the input function $a(x,t)$. Here, a is observed over a finite history including the initial state $a(x,0) = I(x)$.

3 METHODS

Attention can be viewed as an integral operator over the domain, with a kernel given by normalized query–key products (Kovachki et al., 2023). We first decompose such kernels into one-dimensional factors (Section 3.1). We then realize this decomposition with coordinate-wise Transformer blocks (Section 3.2) and describe the discretization (Section 3.3).

3.1 KERNEL TENSORIZATION

Let $\Omega = \prod_{j=1}^{d} \Omega_j$ carry the product measure $\mu = \mu_1 \otimes \cdots \otimes \mu_d$. Let $\mathcal{K}$ be an integral operator with a continuous matrix kernel $\kappa : \Omega \times \Omega \to \mathbb{R}^{d_u \times d_a}$, mapping d_a-channel fields to d_u-channel fields:

$$(\mathcal{K}v)(x) = \int_{\Omega} \kappa(x,y)\, v(y)\, \mathrm{d}\mu(y). \tag{3}$$

Kernel tensorization approximates κ by a finite sum of products of scalar one-dimensional kernels:

$$\kappa_M(x,y) = \sum_{\ell=1}^{M} C_\ell \prod_{j=1}^{d} k_{\ell j}(x_j, y_j). \tag{4}$$

Here $k_{\ell j} : \Omega_j \times \Omega_j \to \mathbb{R}$, and $C_\ell \in \mathbb{R}^{d_u \times d_a}$ mixes channels. We call M the separation rank. Such sums approximate any continuous kernel uniformly (Lemma C.3). By Fubini's theorem (Folland, 1999), the operator with kernel κ_M is a sum of compositions of one-dimensional integral operators:

$$\mathcal{K}_M = \sum_{\ell=1}^{M} C_\ell \, (\mathcal{K}_{\ell d} \circ \cdots \circ \mathcal{K}_{\ell 1}), \quad (\mathcal{K}_{\ell j} v)(x) = \int_{\Omega_j} k_{\ell j}(x_j, y_j)\, v(x_1, \ldots, y_j, \ldots, x_d)\, \mathrm{d}\mu_j(y_j). \tag{5}$$

Each $\mathcal{K}_{\ell j}$ integrates over coordinate j only, so one d-dimensional integral becomes d successive one-dimensional integrals. Section 3.2 realizes each $\mathcal{K}_{\ell j}$ by attention along coordinate j. The sum over ℓ is not built explicitly. Instead, multiple heads and channels carry it, and residual connections expand compositions of blocks into sums of products.

Example: heat equation. To illustrate, consider the heat equation $\partial_t u = \nu \Delta u$ on $\Omega = \prod_{j=1}^{d} \Omega_j$, where $\Omega_j = [0,1]$, with periodic boundary conditions and constant diffusivity $\nu > 0$. Write its Green's function as $G_\tau(x,y) = G_\nu(x, t+\tau; y, t)$; this kernel propagates an arbitrary current state over an interval τ (Stein & Shakarchi, 2003):

$$u(x, t+\tau) = \int_{\Omega} G_\tau(x,y) u(y,t)\, \mathrm{d}y, \qquad G_\tau(x,y) = \sum_{\mathbf{k} \in \mathbb{Z}^d} e^{-4\pi^2 \nu \tau |\mathbf{k}|^2} e^{2\pi i \mathbf{k} \cdot (x-y)}. \tag{6}$$

Here, the Green's function can be decomposed through

$$G_\tau(x,y) = \prod_{j=1}^{d} \left[1 + 2 \sum_{k_j=1}^{\infty} e^{-4\pi^2 \nu \tau k_j^2} \cos(2\pi k_j (x_j - y_j)) \right] \coloneqq \prod_{j=1}^{d} g_\tau(x_j, y_j). \tag{7}$$

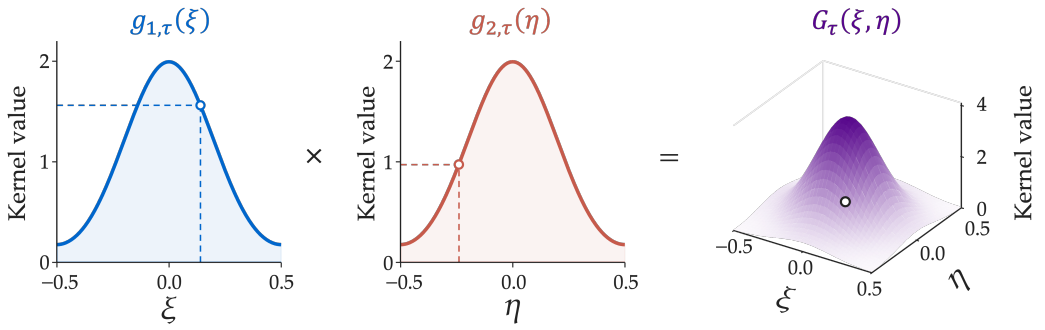

Figure 3: Tensorization of the two-dimensional periodic heat kernel with $\nu = 1$ and $\tau = 0.02$. The one-dimensional factors are shown in displacement coordinates $\xi = x_1 - y_1$ and $\eta = x_2 - y_2$; their product gives the full kernel. Markers highlight corresponding factor and product values.

Fix $\tau > 0$ and let $a = u(\cdot, t)$ be a continuous periodic field. The heat propagator $\mathcal{G}^\tau$ satisfies $\mathcal{G}^\tau a = u(\cdot, t + \tau)$ and admits the representation

$$(\mathcal{G}^\tau a)(x) = \int_\Omega \prod_{j=1}^{d} g_\tau(x_j, y_j) a(y) \mathrm{d}y = \int_{\Omega_d} g_\tau(x_d, y_d) \cdots \int_{\Omega_1} g_\tau(x_1, y_1) a(y) \mathrm{d}y_1 \cdots \mathrm{d}y_d, \quad (8)$$

where the second equality follows from Fubini's theorem. Define the integral operator acting along the jth coordinate; then the propagator factorizes as

$$\mathcal{G}^\tau = \mathcal{K}_d^\tau \circ \cdots \circ \mathcal{K}_1^\tau, \qquad \text{where } (\mathcal{K}_j^\tau v)(x) = \int_{\Omega_j} g_\tau(x_j, y_j) v(x_1, \ldots, y_j, \ldots, x_d) \mathrm{d}y_j. \quad (9)$$

Fig. 3 illustrates this factorization in two dimensions, where multiplying the two one-dimensional kernels yields the full heat kernel. In this example, the global integral operator is realized by composing one-dimensional integral operators with simpler kernels, illustrating the potential of tensorization for operator approximation.

3.2 Tensorized Transformer Architecture

Coordinate attention. Following Kovachki et al. (2023, Proposition 6), we write self-attention along coordinate j as an integral operator over Ω_j. For a feature field $v : \Omega \to \mathbb{R}^C$, let $x[y_j]$ denote the point x with its jth coordinate replaced by y_j. A single attention head computes

$$(\mathcal{K}_j v)(x) = \int_{\Omega_j} \alpha_j(x, y_j)\, W^V v(x[y_j])\, \mathrm{d}y_j, \quad \alpha_j(x, y_j) \propto \exp\Big(\frac{1}{\sqrt{d_h}} \langle W^Q v(x),\, W^K v(x[y_j]) \rangle \Big). \quad (10)$$

Here $\alpha_j(x, \cdot)$ is normalized to integrate to one over Ω_j, and $W^Q, W^K, W^V \in \mathbb{R}^{d_h \times C}$ are the query, key, and value projections. Multi-head attention sums the heads after their output projections. We write $\mathcal{K}_j$ for the resulting multi-head operator. Each head plays the role of one factor $\mathcal{K}_{\ell j}$ in Section 3.1. Unlike those fixed factors, its kernel depends on the input, and through $v(x)$ it also depends on the remaining coordinates of x.

Transformer blocks and sweeps. Each coordinate attention sits in a pre-normalization Transformer block (Vaswani et al., 2017; Xiong et al., 2020):

$$\mathcal{S}_j = (\mathrm{Id} + \mathcal{M} \circ \mathcal{N}) \circ (\mathrm{Id} + \mathcal{K}_j \circ \mathcal{N}). \quad (11)$$

Here $\mathcal{N}$ is layer normalization and $\mathcal{M}$ is a two-layer MLP, both applied pointwise. Query–key normalization is applied inside $\mathcal{K}_j$ (Dehghani et al., 2023). As a result, $\mathcal{S}_j$ is a composition of neural operator layers (Kovachki et al., 2023). A *sweep* $\mathcal{S}_d \circ \cdots \circ \mathcal{S}_1$ applies one block along each coordinate. The tensorized Transformer stacks sweeps until it reaches L blocks. Every block has its own parameters, including repeated blocks along the same coordinate. With time included, each sweep also contains $\mathcal{S}_0$ ($\mathcal{S}_t$ in experiments). Fig. 2(c) shows one sweep in two dimensions.

Why sequential blocks. Freeze the attention weights and omit normalization and MLPs. One two-dimensional sweep then expands as

$$(\mathrm{Id} + \mathcal{K}_2)(\mathrm{Id} + \mathcal{K}_1) = \mathrm{Id} + \mathcal{K}_1 + \mathcal{K}_2 + \mathcal{K}_2\mathcal{K}_1. \tag{12}$$

A parallel sum of coordinate attentions provides only the single-coordinate terms (McCabe et al., 2024; Rautela et al., 2026). The cross term $\mathcal{K}_2\mathcal{K}_1$ composes one-dimensional integral operators, as in Eq. 5, and couples every pair of tokens. More heads and more sweeps add more such terms. The MLPs between blocks add something more: $\mathcal{K}_2$ computes its queries and keys from features already aggregated along coordinate 1. The composite kernel therefore adapts to the input and need not be separable. This mirrors dimensional splitting in numerical PDE solvers (Peaceman & Rachford, 1955; Strang, 1968).

3.3 Discretization

The input a is sampled at T times on a grid with N_j points along coordinate j. This gives $\mathbf{a} \in \mathbb{R}^{T\times N_1\times\cdots\times N_d\times D}$ for D physical fields. The model predicts $\tilde{\mathbf{u}} \in \mathbb{R}^{N_1\times\cdots\times N_d\times D}$, an estimate of $u(\cdot,\tau)$ on the same grid.

Encoding and decoding. A hierarchical CNN with patch factor P embeds $\mathbf{a}$ into features

$$\mathbf{v} \in \mathbb{R}^{T'\times N_1/P\times\cdots\times N_d/P\times C}. \tag{13}$$

With a temporal coordinate, each snapshot is embedded separately and $T' = T$. When the history is stacked into channels, $T' = 1$. The embedding preserves the tensor-product structure of the grid, which coordinate attention requires. We add learnable positional embeddings and apply the tensorized Transformer. We then take the last temporal slice of the output and decode it with a hierarchical transposed CNN. All attention patterns we compare share this encoder, decoder, and positional encoding (Appendix B).

Discrete attention and cost. Replacing the integrals in Eq. 10 by Riemann sums over the grid points along coordinate j recovers standard softmax attention over each grid line. The other axes act as batch dimensions. Suppose there are n latent points per axis. A coordinate block then attends over n^{d-1} lines of length n, which costs $O(n^{d+1})$ per channel. Full attention costs $O(n^{2d})$, and windowed attention with window size w costs $O(n^d w^d)$. One sweep couples every pair of tokens, which windowed attention does not. Attention parameters do not depend on the number of tokens. Models with the same number of blocks therefore have essentially the same parameter count, which is the basis of our matched comparisons.

4 Approximation Theory

We analyze how tensorized Transformers approximate continuous solution operators by combining existing neural-operator approximation results with kernel tensorization. The precise functional setting, assumptions, and complete proof are provided in Appendix C.

Theorem 4.1 (Approximation by Tensorized Transformers)**.** *Let $K \subset \mathcal{A}$ be nonempty and compact, and let $\mathcal{G} : K \to \mathcal{U}$ be continuous. Assume $a \in K$ determines the solution $u = \mathcal{G}(a)$. For every $\epsilon > 0$, there exists a finitely parameterized model $\mathcal{T}_\theta$ built from tensorized Transformers such that*

$$\sup_{a\in K} \|\mathcal{T}_\theta(a) - \mathcal{G}(a)\|_{\mathcal{U}} < \epsilon. \tag{14}$$

Proof. To bound the approximation error, we first choose a finite integral neural operator $\mathcal{I}$, then construct its tensorized counterpart $\mathcal{T}$, and finally approximate $\mathcal{T}$ using tensorized Transformers:

$$\sup_{a\in K} \|\mathcal{T}_\theta(a) - \mathcal{G}(a)\|_{\mathcal{U}} \le \sup_{a\in K} \|\mathcal{I}(a) - \mathcal{G}(a)\|_{\mathcal{U}} + \sup_{a\in K} \|\mathcal{T}(a) - \mathcal{I}(a)\|_{\mathcal{U}} + \sup_{a\in K} \|\mathcal{T}_\theta(a) - \mathcal{T}(a)\|_{\mathcal{U}}. \tag{15}$$

For the first term, Kovachki et al. (2023, Theorem 11) establish, under their stated hypotheses, that finite compositions of continuous integral operators and pointwise nonlinearities approximate

continuous operators uniformly on compact input sets. Accordingly, we use layers with pointwise activation σ, linear projection W, integral operator $\mathcal{K}$, and bias b, of the form

$$(\mathcal{F}_{\sigma,W,\mathcal{K},b}v)(x) = \sigma(Wv(x) + \mathcal{K}(v)(x) + b(x)), \quad \text{where } (\mathcal{K}v)(x) = \int_\Omega \kappa(x,y)v(y)\,\mathrm{d}\mu(y). \tag{16}$$

We fix a finite composition $\mathcal{I}$ of these layers such that

$$\sup_{a\in K} \|\mathcal{I}(a) - \mathcal{G}(a)\|_{\mathcal{U}} < \epsilon/3. \tag{17}$$

Next, we construct $\mathcal{T}$ by tensorizing each kernel in $\mathcal{I}$ as described in Section 3.1, while retaining its pointwise maps in the form of Eq. 16. Lemmas C.3 and C.4, together with the finite-composition stability estimate in Appendix C.3, imply that sufficiently accurate kernel approximations yield

$$\sup_{a\in K} \|\mathcal{T}(a) - \mathcal{I}(a)\|_{\mathcal{U}} < \epsilon/3. \tag{18}$$

Finally, we approximate the fixed operator $\mathcal{T}$ using tensorized Transformers. Calvello et al. (2025, Theorem 22) prove that Transformers approximate continuous (integral) operators uniformly on compact sets of continuous inputs. In the one-dimensional setting, this result allows us to approximate each coordinate integral operator by a finite composition of Transformer blocks and pointwise transformations, yielding

$$\sup_{a\in K} \|\mathcal{T}_\theta(a) - \mathcal{T}(a)\|_{\mathcal{U}} < \epsilon/3. \tag{19}$$

We use the Transformer architecture covered by this theorem, extended to include a standard normalization; Appendix C.4 shows that this extension preserves the approximation guarantee. Combining the three bounds in Eq. 15 establishes the desired solution-operator approximation. □

Remark 4.2. Eq. 16 suggests that full attention can directly act as an integral operator, while tensorized attention requires coordinate sweeps. One may then ask what advantage the tensorized design offers. We remark that the *approximation rate*, reflected in the parameter count required to achieve a prescribed error tolerance, characterizes parameter efficiency. In particular, we hypothesize that, under the same parameter budget, the decomposed kernel factors are easier to approximate than the full kernel. Controlled empirical comparisons are therefore needed to assess its practical benefits, as investigated in the next section.

5 Experiments

We evaluate the predictive accuracy and computational efficiency of tensorized attention through controlled architectural substitutions. We compare full, windowed, and tensorized attention across four PDE benchmarks, two temporal representations, and three model sizes. We also compare our method with existing factorized attention models.

5.1 Baselines, Benchmarks, and Experimental Setup

We compare tensorized attention with full attention and windowed attention. We consider two treatments of the input history: aggregating it into feature channels, or applying temporal attention. Representative models using these representations are summarized in Table 1.

Let $\mathcal{S}_f$ denote a Transformer block that attends over all spatial tokens, and let $\mathcal{S}_w$ denote a block that attends within local spatial windows with window shifts (Liu et al., 2021). We denote blocks attending along time and the spatial coordinates by $\mathcal{S}_t$, $\mathcal{S}_x$, $\mathcal{S}_y$, and $\mathcal{S}_z$. Within each temporal representation, we compare full, windowed, and tensorized attention using the same encoding/decoding procedures and configurations. Additional hyperparameters are specified in Appendix B. The six models are summarized in Table 2.

We evaluate the six models on four benchmarks from *The Well* (Ohana et al., 2024), covering distinct nonlinear dynamics in two and three spatial dimensions. The four benchmarks are Turbulent Radiative Layer **(TRL)** (Fielding et al., 2020), Active Matter **(AM)** (Maddu et al., 2024), Rayleigh–Bénard Convection **(RB)** (Burns et al., 2020), and Compressible Magnetohydrodynamic Turbulence **(MHD)** (Burkhart et al., 2020). We use the provided training, validation, and test splits. Underlying equations and data preprocessing are detailed in Appendices D.1 and D.2.

Table 1: Representative operator learning models grouped by spatial attention pattern and temporal representation. A dash indicates that no example is listed.

Representation	Full attention	Windowed attention	Factorized attention
Single / aggregation	ViTO, DPOT, CViT, Micrometer	Poseidon, P3D	FactFormer
Temporal attention	Walrus	—	MPP-AViT, MORPH

Table 2: Controlled attention arrangements under two temporal representations. All models share the total block count L within each size configuration. d is the number of spatial dimensions, and $(\cdot)$ is included only in 3D. Every block occurrence has independent parameters.

(a) Models with temporal aggregation

Pattern	Block sequence	Abbr.
Full	$[\mathcal{S}_f \circ \mathcal{S}_f(\circ\mathcal{S}_f)]_{\times L/d}$	FF(F)
Windowed	$[\mathcal{S}_w \circ \mathcal{S}_w(\circ\mathcal{S}_w)]_{\times L/d}$	WW(W)
Tensorized	$[\mathcal{S}_x \circ \mathcal{S}_y(\circ\mathcal{S}_z)]_{\times L/d}$	XY(Z)

(b) Models with temporal attention

Pattern	Block sequence	Abbr.
Full	$[\mathcal{S}_t \circ \mathcal{S}_f \circ \mathcal{S}_f(\circ\mathcal{S}_f)]_{\times L/(d+1)}$	TFF(F)
Windowed	$[\mathcal{S}_t \circ \mathcal{S}_w \circ \mathcal{S}_w(\circ\mathcal{S}_w)]_{\times L/(d+1)}$	TWW(W)
Tensorized	$[\mathcal{S}_t \circ \mathcal{S}_x \circ \mathcal{S}_y(\circ\mathcal{S}_z)]_{\times L/(d+1)}$	TXY(Z)

Following *The Well*, each model receives four consecutive snapshots and predicts the next snapshot in the dataset. We train all models by minimizing MSE on normalized fields (McCabe et al., 2026), and evaluate models using VRMSE from *The Well*. We construct small, medium, and large configurations following the official ViT repository (Dosovitskiy et al., 2021). Additional details are provided in Appendices D.3 and D.4.

5.2 Main Results

Table 3 compares three attention patterns under two temporal representations at the large model size. Tensorized attention achieves the lowest one-step VRMSE across all four benchmarks within each temporal representation. For ten-step rollouts, it also achieves the lowest average VRMSE in all settings except one case. These results support the effectiveness of tensorized Transformers for PDE prediction under matched model and training configurations.

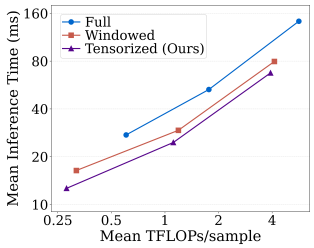

Figure 6: Inference time versus FLOPs across scales.

We next examine how predictive accuracy varies with computational cost across the small, medium, and large configurations. Fig. 4 and Fig. 5 (left) plot one-step VRMSE against inference FLOPs per sample on RB and MHD for both temporal representations. Tensorized attention offers a favorable accuracy–cost trade-off across the evaluated scales.

Inference speed is also important for PDE surrogates. Fig. 6 compares one-step inference time and FLOPs averaged equally over RB and MHD. At each model size, tensorized attention achieves the lowest inference time and FLOPs, with a 2.1× speedup over full attention.

5.3 Related Works

We compare the tensorized Transformer with existing factorized architectures for PDE modeling: FactFormer (Li et al., 2023a), AViT (McCabe et al., 2024), and MORPH (Rautela et al., 2026). We examine their architectural distinctions and empirically compare their performance with our method.

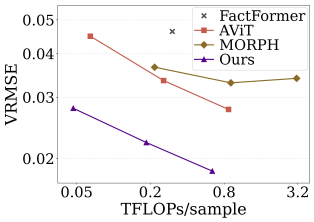

Figure 7: One-step VRMSE versus inference cost across three model sizes.

FactFormer computes attention weights separately along each dimension and applies them successively to the value tensor through matrix multiplications, realizing global attention. **AViT** applies temporal attention followed by parallel attention with shared query, key, and value projections (QKV), and then averages the spatial outputs. **MORPH** follows the same temporal–spatial arrangement but uses separate QKV projections and sums the spatial outputs. Table 4 summarizes the architectural differences between these models and our tensorized Transformer.

Table 3: One-step and mean rollout VRMSE (↓ lower is better) on TRL, AM, RB, and MHD for the six large-model configurations. One-step scores average over all test windows; ten-step scores average over steps $1, \ldots, 10$ and test trajectories, each rolled out autoregressively from its first input window. Underlining marks the best within each group; bold marks the best overall in each column.

Model	TRL		AM		RB		MHD	
	One-step	Ten-step	One-step	Ten-step	One-step	Ten-step	One-step	Ten-step
FF(F)	0.1703	0.2040	0.0130	0.1031	0.0226	0.0665	0.0796	0.4229
WW(W)	0.1845	0.2037	0.0232	0.1215	0.0208	0.0685	0.0896	0.4797
XY(Z) (Ours)	0.1600	0.1992	0.0115	**0.0947**	**0.0184**	0.0633	0.0679	0.4320
TFF(F)	0.1591	0.2050	0.0114	0.1234	0.0239	0.0611	0.0792	0.4177
TWW(W)	0.1755	0.2195	0.0197	0.1363	0.0209	0.0630	0.0865	0.4672
TXY(Z) (Ours)	**0.1519**	**0.1902**	**0.0100**	0.1089	0.0198	**0.0574**	**0.0660**	**0.4108**

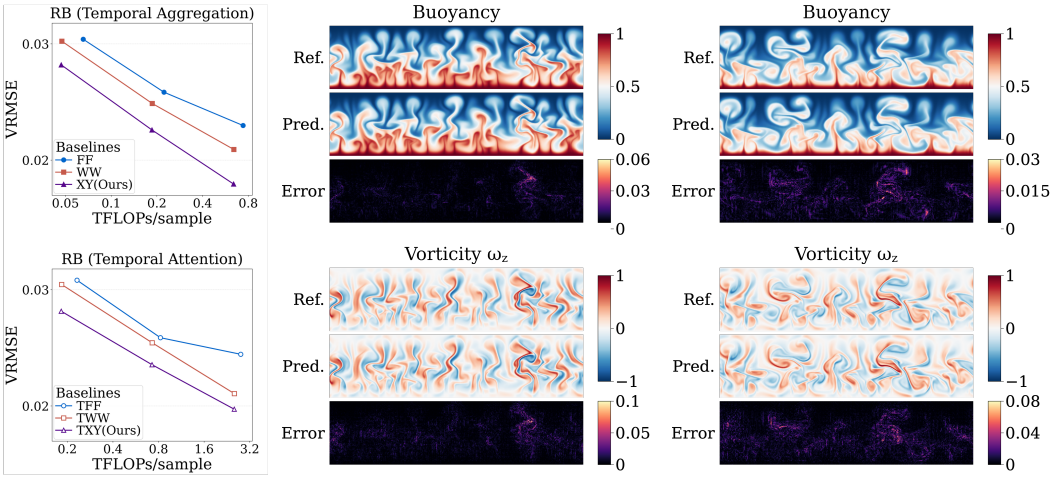

Figure 4: Scaling and qualitative predictions on RB. Left: one-step VRMSE versus inference FLOPs per sample across three model sizes, with temporal aggregation (top) and temporal attention (bottom). Right: reference and predicted buoyancy and vorticity with their pointwise absolute error. Results are obtained using the large tensorized XY model.

We retain the architectural configurations from the original papers and use the same training and evaluation protocol as for our models. We evaluate three configurations each for AViT and MORPH. For FactFormer, we use the only configuration provided in its paper for this kind of benchmark. Implementation details are provided in Appendix D.4. Fig. 7 shows that the tensorized Transformer outperforms the other models across the evaluated scales.

Fig. 7 also provides empirical support for the rationale in Section 3.2. The tensorized Transformer outperforms alternative factorized methods in the evaluated settings, suggesting a practical advantage of composing Transformer blocks over coupled or summed attention.

5.4 Ablation Study

The kernel tensorization in Section 3.1 combines M products of one-dimensional kernels through channel projections C_ℓ (Eq. 4), suggesting a role for multiple attention heads. We therefore vary the head count at fixed feature width in the small tensorized XY and full-attention models on TRL. Fig. 8 shows similar errors with one head. However, increasing the head count improves tensorized attention over a wider range, whereas full attention deteriorates at larger head counts. This trend is consistent with the hypothesis that multiple heads help approximate sums of tensor-product kernels, complementing the channel mixing provided by pointwise MLPs.

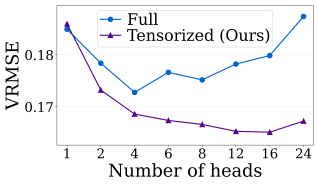

Figure 8: One-step VRMSE of full and tensorized attention on TRL.

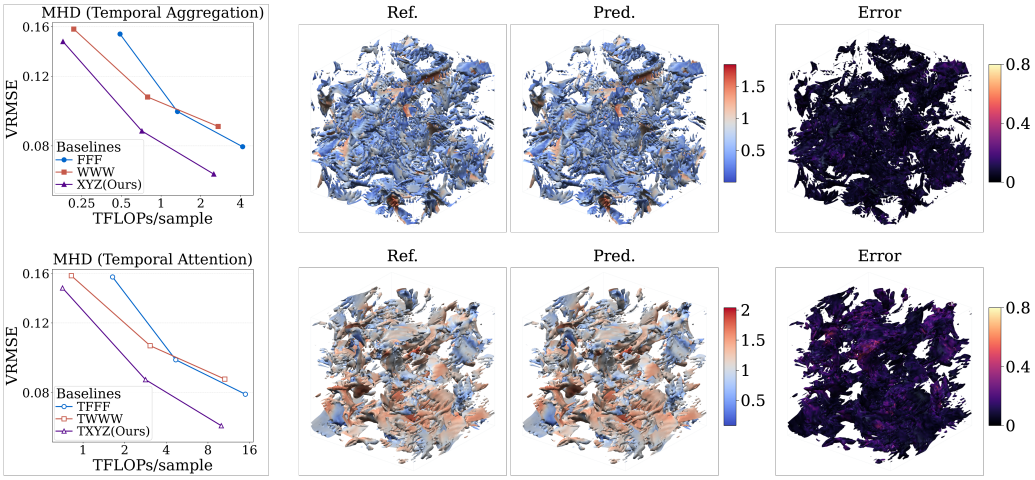

Figure 5: Scaling and qualitative predictions on MHD. Left: one-step VRMSE versus inference FLOPs per sample across three model sizes, with temporal aggregation (top) and temporal attention (bottom). Right: reference and predicted magnetic-curl isosurfaces with their pointwise absolute error. Results are obtained using the large tensorized XYZ model.

Table 4: Comparison of factorized model architectures for PDEs on a two-dimensional domain, denoted by (x, y). Schematics omit residual connections, normalization, and head indices.

Method	Schematic formulation	Attention construction
FactFormer	$W(A_y[v] \cdot A_x[v] \cdot V(v))$	Successive matrix multiplications.
AViT	$W\left(\frac{1}{2}[A_x^{\mathrm{sh}}(w) + A_y^{\mathrm{sh}}(w)]\right)$, where $w = A_t(v)$	Parallel mean; shared spatial QKV (A^{sh}).
MORPH	$W(A_x(w) + A_y(w))$, where $w = A_t(v)$	Parallel sum; separate spatial QKV.
Ours	$(W \circ A_y) \circ (W \circ A_x)(v)$	Sequential independent blocks.

v is the input feature tensor; A_j denotes attention along dimension $j \in \{t, x, y\}$. W denotes a pointwise MLP or linear transformation. For FactFormer, $A_j[v]$ denotes attention weights computed from v, with each matrix acting along its corresponding tensor dimension; V is the value projection.

6 Discussion

We study the tensorized Transformer as an interchangeable module for attention-based PDE models. Through kernel tensorization, we establish uniform approximation of continuous solution operators on compact sets. Controlled comparisons on four PDE benchmarks show improved accuracy and computational efficiency across three model sizes, with faster inference than full attention. These findings support composing independently parameterized blocks along individual coordinates as a practical design for scalable PDE modeling.

Several limitations remain. First, our analysis assumes Cartesian-product domains, and our experiments use structured grids. Extending tensorized attention to irregular domains and complex geometries may require suitable coordinate mappings or structured latent representations, together with corresponding approximation analysis. For domains with obstacles or disconnected components, communication along coordinate directions depends on the representation and attention masks; global interaction after one sweep must therefore be reassessed. Second, our experiments focus on forward prediction. Inverse problems require separate evaluation and appropriate formulations of the observations, conditioning inputs, and reconstruction targets. Third, our approximation guarantees and experiments do not establish transfer to unseen PDEs or long-term rollout stability, leaving these questions open. Finally, our model features preserve the coordinate structure of the physical grid. Whether tensorized attention remains effective in pretrained latent representations that do not preserve this structure requires further investigation.

AI USE STATEMENT

In this work, we used generative AI tools for implementing methods, proposing or refining hypotheses, and assisting in the writing of proofs. We have not used generative AI tools for developing theoretical models or conceptual frameworks, formulating mathematical claims, providing critical ingredients for proving mathematical claims, designing or providing feedback on research methodology or experiments, interpreting results, or supporting qualitative and thematic data analysis. Generating synthetic datasets, assisting with translation, and cleaning or reformatting datasets are not applicable to this work.

Additionally, we used generative AI tools for creating or modifying scientific figures or images, creating or editing software code, summarizing or analyzing existing literature, searching for information, and identifying relevant literature. We have reviewed all AI-assisted work. Two authors verified the LLM-generated code and proof steps for correctness. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work studies operator learning for PDEs using publicly available simulation datasets—*The Well* (Ohana et al., 2024). It does not involve human participants or personal data. We have identified no specific ethical concerns arising from the methods or experiments presented in this work.

REPRODUCIBILITY STATEMENT

Our methods and experimental settings are described in Sections 3 and 5, with additional implementation details in Appendices B and D. The theoretical analysis and supporting proofs are presented in Section 4 and Appendix C. All experiments use publicly available datasets from *The Well*. We will release our source code, training configurations, and evaluation scripts on GitHub upon acceptance.

REFERENCES

Cristian Bodnar, Wessel P. Bruinsma, Ana Lucic, Megan Stanley, Anna Allen, Johannes Brandstetter, Patrick Garvan, Maik Riechert, Jonathan A. Weyn, Haiyu Dong, Jayesh K. Gupta, Kit Thambiratnam, Alexander T. Archibald, Chun-Chieh Wu, Elizabeth Heider, Max Welling, Richard E. Turner, and Paris Perdikaris. A foundation model for the Earth system. *Nature*, 641:1180–1187, 2025. doi: 10.1038/s41586-025-09005-y.

B Burkhart, SM Appel, S Bialy, J Cho, AJ Christensen, D Collins, Christoph Federrath, DB Fielding, D Finkbeiner, AS Hill, et al. The catalogue for astrophysical turbulence simulations (cats). *The Astrophysical Journal*, 905(1):14, 2020.

Keaton J Burns, Geoffrey M Vasil, Jeffrey S Oishi, Daniel Lecoanet, and Benjamin P Brown. Dedalus: A flexible framework for numerical simulations with spectral methods. *Physical Review Research*, 2(2):023068, 2020.

Edoardo Calvello, Nikola B. Kovachki, Matthew E. Levine, and Andrew M. Stuart. Continuum attention for neural operators. *Journal of Machine Learning Research*, 26(300):1–52, 2025. URL https://jmlr.org/papers/v26/24-0879.html.

Mostafa Dehghani, Josip Djolonga, Basil Mustafa, Piotr Padlewski, Jonathan Heek, Justin Gilmer, Andreas Peter Steiner, Mathilde Caron, Robert Geirhos, Ibrahim Alabdulmohsin, Rodolphe Jenatton, Lucas Beyer, Michael Tschannen, Anurag Arnab, Xiao Wang, Carlos Riquelme Ruiz, Matthias Minderer, Joan Puigcerver, Utku Evci, Manoj Kumar, Sjoerd Van Steenkiste, Gamaleldin Fathy Elsayed, Aravindh Mahendran, Fisher Yu, Avital Oliver, Fantine Huot, Jasmijn Bastings, Mark Collier, Alexey A. Gritsenko, Vighnesh Birodkar, Cristina Nader Vasconcelos, Yi Tay, Thomas Mensink, Alexander Kolesnikov, Filip Pavetic, Dustin Tran, Thomas Kipf, Mario Lucic, Xiaohua Zhai, Daniel Keysers, Jeremiah J. Harmsen, and Neil Houlsby. Scaling vision transformers to 22 billion parameters. 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. 7480–7512. PMLR, 2023. URL https://proceedings.mlr.press/v202/dehghani23a.html.

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

James Dugundji. An extension of Tietze's theorem. *Pacific Journal of Mathematics*, 1(3):353–367, 1951. doi: 10.2140/pjm.1951.1.353. URL https://doi.org/10.2140/pjm.1951.1.353.

Drummond B Fielding, Eve C Ostriker, Greg L Bryan, and Adam S Jermyn. Multiphase gas and the fractal nature of radiative turbulent mixing layers. *The Astrophysical Journal Letters*, 894(2): L24, 2020.

Gerald B. Folland. *Real Analysis: Modern Techniques and Their Applications*. John Wiley & Sons, 2nd edition, 1999. ISBN 9780471317166.

John Guibas, Morteza Mardani, Zongyi Li, Andrew Tao, Anima Anandkumar, and Bryan Catanzaro. Adaptive Fourier neural operators: Efficient token mixers for transformers. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=EXHG-A3jlM.

Quan Guo, Yu He, Ming Liu, Yue Zhao, Yuzheng Liu, and Jian Luo. Reduced geostatistical approach with a Fourier neural operator surrogate model for inverse modeling of hydraulic tomography. *Water Resources Research*, 60(6):e2023WR034939, 2024. doi: 10.1029/2023WR034939.

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. URL https://proceedings.mlr.press/v235/hao24d.html.

Maximilian Herde, Bogdan Raonić, Tobias Rohner, Roger Käppeli, Roberto Molinaro, Emmanuel de Bézenac, and Siddhartha Mishra. Poseidon: Efficient foundation models for PDEs. In *Advances in Neural Information Processing Systems*, volume 37, pp. 72525–72624. Curran Associates, Inc., 2024. doi: 10.52202/079017-2311.

Jonathan Ho, Nal Kalchbrenner, Dirk Weissenborn, and Tim Salimans. Axial attention in multidimensional transformers, 2020. URL https://openreview.net/forum?id=H1e5GJBtDr.

Benjamin Holzschuh, Georg Kohl, Florian Redinger, and Nils Thuerey. P3d: Highly scalable 3d neural surrogates for physics simulations with global context. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=8UdCE5nhFl.

Keller Jordan, Yuchen Jin, Vlado Boza, You Jiacheng, Franz Cesista, Laker Newhouse, and Jeremy Bernstein. Muon: An optimizer for hidden layers in neural networks, 2024. URL https://kellerjordan.github.io/posts/muon/.

Chenhui Kou, Yuhui Yin, Min Zhu, Shengkun Jia, Yiqing Luo, Xigang Yuan, and Lu Lu. Neural topology optimization via active learning for efficient channel design in turbulent mass transfer. *Advanced Science*, 12(37):e08386, 2025. doi: 10.1002/advs.202508386.

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 https://jmlr.org/papers/v24/21-1524.html.

Nikola B. Kovachki, Samuel Lanthaler, and Andrew M. Stuart. Operator learning: Algorithms and analysis. In Siddhartha Mishra and Alex Townsend (eds.), *Numerical Analysis Meets Machine Learning*, volume 25 of *Handbook of Numerical Analysis*, pp. 419–467. Elsevier, 2024. doi: 10.1016/bs.hna.2024.05.009.

Remi Lam, Alvaro Sanchez-Gonzalez, Matthew Willson, Peter Wirnsberger, Meire Fortunato, Ferran Alet, Suman Ravuri, Timo Ewalds, Zach Eaton-Rosen, Weihua Hu, Alexander Merose, Stephan Hoyer, George Holland, Oriol Vinyals, Jacklynn Stott, Alexander Pritzel, Shakir Mohamed, and Peter Battaglia. Learning skillful medium-range global weather forecasting. *Science*, 382(6677):1416–1421, 2023. doi: 10.1126/science.adi2336.

Jonathan E. Lee, Min Zhu, Ziqiao Xi, Kun Wang, Yanhua O. Yuan, and Lu Lu. Efficient and generalizable nested Fourier-DeepONet for three-dimensional geological carbon sequestration. *Engineering Applications of Computational Fluid Mechanics*, 18(1):2435457, 2024. doi: 10.1080/19942060.2024.2435457.

Zijie Li, Dule Shu, and Amir Barati Farimani. Scalable Transformer for PDE surrogate modeling. In *Advances in Neural Information Processing Systems*, volume 36, 2023a.

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. URL https://openreview.net/forum?id=c8P9NQVtmnO.

Zongyi Li, Nikola Kovachki, Chris Choy, Boyi Li, Jean Kossaifi, Shourya Otta, Mohammad Amin Nabian, Maximilian Stadler, Christian Hundt, Kamyar Azizzadenesheli, and Animashree Anandkumar. Geometry-informed neural operator for large-scale 3D PDEs. In *Advances in Neural Information Processing Systems*, volume 36, pp. 35836–35854. Curran Associates, Inc., 2023b. doi: 10.52202/075280-1556.

Burigede Liu, Nikola Kovachki, Zongyi Li, Kamyar Azizzadenesheli, Anima Anandkumar, Andrew M. Stuart, and Kaushik Bhattacharya. A learning-based multiscale method and its application to inelastic impact problems. *Journal of the Mechanics and Physics of Solids*, 158:104668, 2022. doi: 10.1016/j.jmps.2021.104668.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin Transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10012–10022, 2021. doi: 10.1109/ICCV48922.2021.00986.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

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*, 3:218–229, 2021. doi: 10.1038/s42256-021-00302-5. URL https://www.nature.com/articles/s42256-021-00302-5.

Suryanarayana Maddu, Scott Weady, and Michael J Shelley. Learning fast, accurate, and stable closures of a kinetic theory of an active fluid. *Journal of Computational Physics*, 504:112869, 2024.

Michael McCabe, Bruno Régaldo-Saint Blancard, Liam Parker, Ruben Ohana, Miles Cranmer, Alberto Bietti, Michael Eickenberg, Siavash Golkar, Geraud Krawezik, Francois Lanusse, Mariel Pettee, Tiberiu Tesileanu, Kyunghyun Cho, and Shirley Ho. Multiple physics pretraining for spatiotemporal surrogate models. In *Advances in Neural Information Processing Systems*, volume 37, pp. 119301–119335. Curran Associates, Inc., 2024. doi: 10.52202/079017-3791.

Michael McCabe, Payel Mukhopadhyay, Tanya Marwah, Bruno Régaldo-Saint Blancard, François Rozet, Cristiana Diaconu, Lucas Meyer, Kaze W. K. Wong, Hadi Sotoudeh, Alberto Bietti, Irina Espejo, Rio Fear, Siavash Golkar, Tom Hehir, Keiya Hirashima, Geraud Krawezik, Francois Lanusse, Rudy Morel, Ruben Ohana, Liam Parker, Mariel Pettee, Jeff Shen, Kyunghyun Cho, Miles Cranmer, and Shirley Ho. Walrus: A cross-domain foundation model for continuum dynamics. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2511.15684.

Ruben Ohana, Michael McCabe, Lucas Thibaut Meyer, Rudy Morel, Fruzsina Julia Agocs, Miguel Beneitez, Marsha Berger, Blakesley Burkhart, Stuart B. Dalziel, Drummond Buschman Fielding, Daniel Fortunato, Jared A. Goldberg, Keiya Hirashima, Yan-Fei Jiang, Rich Kerswell, Suryanarayana Maddu, Jonah M. Miller, Payel Mukhopadhyay, Stefan S. Nixon, Jeff Shen, Romain Watteaux, Bruno Régaldo-Saint Blancard, François Rozet, Liam Holden Parker, Miles Cranmer, and Shirley Ho. The Well: a large-scale collection of diverse physics simulations for machine learning. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024.

Oded Ovadia, Adar Kahana, Panos Stinis, Eli Turkel, Dan Givoli, and George Em Karniadakis. ViTO: Vision transformer-operator. *Computer Methods in Applied Mechanics and Engineering*, 428:117109, 2024. doi: 10.1016/j.cma.2024.117109.

Jaideep Pathak, Shashank Subramanian, Peter Harrington, Sanjeev Raja, Ashesh Chattopadhyay, Morteza Mardani, Thorsten Kurth, David Hall, Zongyi Li, Kamyar Azizzadenesheli, Pedram Hassanzadeh, Karthik Kashinath, and Animashree Anandkumar. FourCastNet: A global data-driven high-resolution weather model using adaptive Fourier neural operators. *arXiv preprint arXiv:2202.11214*, 2022. doi: 10.48550/arXiv.2202.11214. URL https://arxiv.org/abs/2202.11214.

Donald W Peaceman and Henry H Rachford, Jr. The numerical solution of parabolic and elliptic differential equations. *Journal of the Society for industrial and Applied Mathematics*, 3(1):28–41, 1955.

Allan Pinkus. Approximation theory of the MLP model in neural networks. *Acta Numerica*, 8: 143–195, 1999. doi: 10.1017/S0962492900002919. URL https://doi.org/10.1017/S0962492900002919.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. URL https://www.jmlr.org/papers/v21/20-074.html.

Meer Mehran Rashid, Tanu Pittie, Souvik Chakraborty, and N. M. Anoop Krishnan. Learning the stress-strain fields in digital composites using Fourier neural operator. *iScience*, 25(11):105452, 2022. doi: 10.1016/j.isci.2022.105452.

Mahindra Singh Rautela, Alexander Most, Siddharth Mansingh, Bradley C. Love, Ayan Biswas, Diane Oyen, and Earl Lawrence. MORPH: Shape-agnostic PDE foundation models, 2026. URL https://openreview.net/forum?id=e2uwm88WFG.

Walter Rudin. *Principles of Mathematical Analysis*. McGraw-Hill, New York, 3 edition, 1976.

Elias M. Stein and Rami Shakarchi. *Fourier Analysis: An Introduction*, volume 1 of *Princeton Lectures in Analysis*. Princeton University Press, 2003. ISBN 9780691113845.

Gilbert Strang. On the construction and comparison of difference schemes. *SIAM journal on numerical analysis*, 5(3):506–517, 1968.

Maria Luisa Taccari, He Wang, Somdatta Goswami, Mario De Florio, Jonathan Nuttall, Xiaohui Chen, and Peter K. Jimack. Developing a cost-effective emulator for groundwater flow modeling using deep neural operators. *Journal of Hydrology*, 630:130551, 2024. doi: 10.1016/j.jhydrol.2023.130551.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

Sifan Wang, Jacob H. Seidman, Shyam Sankaran, Hanwen Wang, George J. Pappas, and Paris Perdikaris. CViT: Continuous vision transformer for operator learning. In *International Conference on Learning Representations*, pp. 24004–24039, 2025. URL https://openreview.net/pdf?id=cRnCcuLvyr.

Sifan Wang, Tong-Rui Liu, Shyam Sankaran, and Paris Perdikaris. Micrometer: Micromechanics transformer for predicting full field mechanical responses of heterogeneous materials. *Computer Methods in Applied Mechanics and Engineering*, 448:118373, 2026. doi: 10.1016/j.cma.2025.118373.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tie-Yan Liu. On layer normalization in the transformer architecture. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 10524–10533. PMLR, 2020. URL https://proceedings.mlr.press/v119/xiong20b.html.

A NOMENCLATURE

Table 5 summarizes the main symbols and notation used in this work.

Table 5: Summary of the main symbols and notation used in this work.

Notation	Description
Operator learning	
$\Omega, \partial\Omega$	Spatial domain and its boundary
d	Number of spatial dimensions
x, y	Spatial evaluation and integration points
t, τ	Time and prediction interval
λ	Parameters specifying the PDE
$\mathcal{L}, \mathcal{B}$	Differential operator and boundary operator
f, g	Forcing term and boundary data
I	Initial condition
$\mathcal{A}, \mathcal{U}$	Input and solution function spaces
a, u	Input function and corresponding solution
$\mathcal{G}$	Solution operator mapping inputs to solutions
Kernel tensorization and approximation	
μ	Product integration measure on the domain
κ	Matrix-valued integral kernel
$\mathcal{K}$	Integral operator
$k_{\ell j}$	Scalar kernel along coordinate j in tensor-product term ℓ
C_ℓ	Constant channel-mixing matrix in tensor-product term ℓ
d_a, d_u	Input and output channel dimensions of the integral kernel
M	Separation rank: number of tensor-product terms
G_τ	Periodic heat kernel (Green's function) over the time interval τ
ν	Diffusivity in the heat-equation example
$\mathcal{I}$	Integral neural operator approximating $\mathcal{G}$
$\mathcal{T}$	Tensorized counterpart of $\mathcal{I}$
$\mathcal{T}_\theta$	Tensorized Transformer approximating the solution operator
Tensorized Transformer	
v	Feature field
C	Number of feature channels
α_j	Normalized attention weight along coordinate j
W^Q, W^K, W^V	Query, key, and value projection matrices
d_h	Feature dimension of an attention head
$\mathcal{S}_j$	Transformer block attending along coordinate j
$\mathcal{M}, \mathcal{N}$	Pointwise MLP and layer normalization
σ	Pointwise activation function
W, b	Linear projection and bias in a neural operator layer
L	Total number of Transformer blocks
T	Number of input snapshots
N_j	Number of spatial grid points along coordinate j
D	Number of physical fields
P	Spatial patch factor of the encoder
n, w	Latent points per axis and window length per axis

B METHOD DETAILS

Our implementation maps a sequence of observed fields to the next snapshot on the same spatial grid. The input is $\mathbf{a} \in \mathbb{R}^{T \times N_1 \times \cdots \times N_d \times D}$, where T is the number of input frames, N_j is the grid size along spatial dimension j, and D is the number of physical variables. We omit the batch dimension and write feature channels last. The main experiments use four consecutive input frames to predict the next stored frame.

Spatial encoder. We embed the input using a stack of three spatial convolutional layers with output widths $C/4$, $C/2$, and C, where C is the Transformer feature width. GELU activations follow the first two layers. With temporal attention, the same CNN processes each snapshot independently. With temporal aggregation, we concatenate the T snapshots into TD channels at each spatial node before applying the CNN. The convolutions reduce each spatial grid size by a total patch factor P, producing

$$\mathbf{v}_p \in \mathbb{R}^{T' \times N_1' \times \cdots \times N_d' \times C}, \qquad N_j' = N_j/P, \qquad T' = \begin{cases} T, & \text{temporal attention,} \\ 1, & \text{temporal aggregation.} \end{cases} \tag{20}$$

We use $P = 8$ in 2D and $P = 4$ in 3D, with successive spatial strides $(2, 2, 2)$ and $(2, 2, 1)$, respectively. Each stage uses its stride as the kernel size along every spatial dimension, yielding nonoverlapping patches. All grid sizes are divisible by P. The resulting features preserve the coordinate structure of the grid at a coarser resolution.

Positional information. We incorporate spatial position through a learnable embedding $\mathbf{PE}_s$, initialized by concatenating one-dimensional sine–cosine features (Vaswani et al., 2017):

$$\mathbf{v}^{(0)} = \mathbf{v}_p + \mathbf{PE}_s, \qquad \mathbf{PE}_s \in \mathbb{R}^{1 \times N_1' \times \cdots \times N_d' \times C}. \tag{21}$$

The same spatial embedding is shared across input frames. When temporal attention is included, each temporal block also adds learned, head-specific T5-style relative-position biases to its attention logits (Raffel et al., 2020). These biases encode differences in frame indices. Temporal attention is noncausal over the supplied history, allowing each temporal token to attend to all observed frames.

Transformer processor. Our tensorized processor consists of L independently parameterized Transformer blocks, arranged into complete coordinate sweeps as described in Section 3.2. Each block applies attention along one coordinate while holding the others fixed and preserves the feature tensor's shape. Under the LayerNorm assumption used in our analysis, the attention branch first applies tokenwise LayerNorm over the C feature channels. The projected queries and keys are further normalized by LayerNorm over their head channels (Dehghani et al., 2023). After the attention residual, a separately parameterized tokenwise LayerNorm and a two-layer MLP with widths $C \to 4C \to C$ and a GELU activation form the second residual branch. Thus, each block updates the features before the next coordinate is processed.

Physical-parameter conditioning. When physical parameters are supplied, each block uses a learned affine projection to produce separate channelwise scale adjustments and shifts for its attention and MLP branches. Specifically, each normalized branch input h is modulated as $(1+\gamma)\odot h+\beta$, where $\gamma, \beta \in \mathbb{R}^C$ are broadcast over all tokens and $\odot$ denotes elementwise multiplication. The projection weights and biases are initialized to zero, so the modulation initially leaves the features unchanged. For RB, the conditioning inputs are Ra and Pr, transformed by $\log_{10}$ and standardized using training-trajectory statistics. For TRL, we use the cooling parameter t_{cool} without transformation. AM and MHD use no physical-parameter conditioning.

Spatial decoder. After the final block, we extract the last temporal slice of $\mathbf{v}^{(L)}$, which is the only slice when $T' = 1$. The resulting spatial features are decoded by three transposed-convolutional layers with output widths $C/2$, $C/4$, and D. GELU activations follow the first two layers. Their spatial strides reverse those of the encoder, restoring the original grid resolution. After reversing the samplewise scaling, we obtain $\tilde{\mathbf{u}} \in \mathbb{R}^{N_1 \times \cdots \times N_d \times D}$, the predicted next state in dataset-standardized units. The main experiments use these same encoding, positional encoding, normalization, conditioning, and decoding procedures for all compared attention patterns within each benchmark.

C Assumptions and Proofs

C.1 Functional Setting and Assumptions

We first specify the setting of Theorem 4.1, then prove the three bounds in Eq. 15 in order: integral approximation, kernel tensorization, and Transformer realization.

Domain and function spaces. Let $\Omega = \prod_{j=1}^{d} \Omega_j$, where $d \in \{2, 3\}$ and $\Omega_j = [a_j, b_j]$ with $a_j < b_j$. The PDE is posed in the interior, with boundary conditions on $\partial\Omega$. We use Lebesgue measure μ_j on each interval and the product measure $\mu = \mu_1 \otimes \cdots \otimes \mu_d$ on Ω, so $\mathrm{d}\mu_j(y_j) = \mathrm{d}y_j$ as in Eq. 10. These measures are finite and positive on every nonempty relatively open set.

For a positive integer r, $C(\Omega; \mathbb{R}^r)$ is the space of continuous fields with r channels. We measure a field by its largest absolute channel value over the domain:

$$\|v\|_{C(\Omega)} = \sup_{x\in\Omega} |v(x)|_\infty, \qquad |z|_\infty = \max_{1\le c\le r} |z_c|. \tag{22}$$

Unless a temporal coordinate is retained explicitly, we take

$$\mathcal{A} = C(\Omega; \mathbb{R}^{d_a}), \qquad \mathcal{U} = C(\Omega; \mathbb{R}^{d_u}), \qquad \|a\|_{\mathcal{A}} = \|a\|_{C(\Omega)}, \quad \|u\|_{\mathcal{U}} = \|u\|_{C(\Omega)}. \tag{23}$$

Here d_a, d_u count input and output channels. The integer C denotes latent feature width and is distinct from the function-space notation $C(\Omega; \mathbb{R}^r)$. All fields and kernels are continuous up to the boundary. Boundary conditions, including periodicity, restrict the admissible data and solutions; errors are measured in the uniform norm above.

Fix the prediction time τ from Section 2. The input a must determine $\mathcal{G}(a) = u(\cdot, \tau)$. When history is stored in channels, T snapshots of D physical fields give $d_a = TD$. Any other varying data needed for the prediction, such as coefficients, forcing, or boundary values, are continuously encoded in additional input channels. All remaining PDE data are fixed.

Assume $K \subset \mathcal{A}$ is nonempty and compact in the uniform norm, and $\mathcal{G} : K \to \mathcal{U}$ is single-valued and continuous. Thus convergence of inputs in this norm implies convergence of their solutions. Compactness also gives

$$\sup_{a\in K} \|a\|_{\mathcal{A}} < \infty, \qquad \sup_{a\in K} \|\mathcal{G}(a)\|_{\mathcal{U}} < \infty. \tag{24}$$

These assumptions concern the chosen PDE problem and input set. The solution map only needs to be defined on K; Appendix C.2 explains how to apply the established approximation theorem in this case.

When time is an integration coordinate, add a nondegenerate compact history interval Ω_0 containing the observation times and ending at $t_{\mathrm{last}} < \tau$. Use Lebesgue measure μ_0 and the input space $C(\Omega \times \Omega_0; \mathbb{R}^{d_a})$ with its uniform norm. The output remains a spatial field in $\mathcal{U}$. The index 0 denotes time, and d still counts spatial dimensions. Measures and sweeps include all these coordinates. Attention is unmasked over each full interval, and any learnable relative-position bias may be set to zero. To use the same domain for intermediate inputs and outputs, repeat the target along the history coordinate and read out its final slice:

$$(\widehat{\mathcal{G}}(a))(x, t) = (\mathcal{G}(a))(x), \qquad (\mathcal{E}_{\mathrm{last}} w)(x) = w(x, t_{\mathrm{last}}). \tag{25}$$

Then $\mathcal{E}_{\mathrm{last}} \widehat{\mathcal{G}} = \mathcal{G}$ and $\|\mathcal{E}_{\mathrm{last}} w\|_{C(\Omega)} \le \|w\|_{C(\Omega\times\Omega_0)}$. The proof therefore applies on the enlarged product domain, and taking the final slice cannot increase the error. This continuous-history formulation assumes the whole history field is available as input; a fixed collection of snapshots is covered by the channel representation above.

Integral layers and tensor factors. The integral operator $\mathcal{I}$ uses continuous matrix kernels and continuous biases. The following norms make the kernel error estimates compatible with our field norm. For $B \in \mathbb{R}^{r'\times r}$, set

$$\|B\|_{\infty\to\infty} = \max_{1\le i\le r'} \sum_{c=1}^{r} |B_{ic}|, \qquad \|\kappa\|_\infty = \sup_{x,y\in\Omega} \|\kappa(x, y)\|_{\infty\to\infty},$$
$$\|v\|_{L^1} = \int_\Omega |v(y)|_\infty \, \mathrm{d}\mu(y). \tag{26}$$

The operator norm $\|\mathcal{K}\|_{L^1\to C}$ measures the largest uniform output norm over inputs with L^1 norm at most one. For continuous v, $\|v\|_{L^1}\leq \mu(\Omega)\|v\|_{C(\Omega)}$. Thus the estimate in Lemma C.4 applies to our continuous latent fields.

In Eq. 4, $k_{\ell j}$ is a continuous scalar kernel on (x_j, y_j), C_ℓ mixes channels, and M counts the chosen tensor terms. No minimality is required. The factors may have either sign and any integration mass. To form $\mathcal{T}$, replace each integral layer by its sum of complete linear sweeps, retaining Wv, b, and the subsequent activation from Eq. 16.

Pointwise maps and finite model capacity. Fix a scalar nonlinear activation $\sigma \in C^\infty(\mathbb{R})$ that is nonpolynomial and globally Lipschitz: there is a finite L_σ such that

$$|\sigma(s) - \sigma(t)|\leq L_\sigma|s - t| \qquad (s, t \in \mathbb{R}). \tag{27}$$

Here C^∞ means that derivatives of every order exist and are continuous. The activation acts on each channel; ρ in Eq. 46 denotes the same function. Affine and output layers may use the identity. These conditions suffice for both Kovachki et al. (2023, Theorem 11) and Calvello et al. (2025, Theorem 22).

The model has finite pointwise neural lifting and output projection. Positional features supply every integration coordinate, including time when present. Channels may store input data, coordinates, constants, and intermediate results. Blocks have the form in Eq. 11, with independent parameters and two-layer pointwise MLPs. Width C, head dimension d_h, head count, MLP width, and depth L may grow with the required accuracy, but remain finite; parameter magnitudes are unrestricted. Blocks follow the prescribed coordinate sweeps. Setting both residual-branch outputs to zero makes a block the identity.

At each stage, all heads attend along the active coordinate. Value and output projections can keep separate channel groups through a sweep and combine them afterwards. The theorem concerns this continuum model class with selectable lifting, readout, and capacity; its scope is distinct from a fixed sampled grid or experimental model size in Section 3.3.

Normalization and retained features. We use the pre-normalization residual ordering of Eq. 11. At each point, standard LayerNorm acts on $z \in \mathbb{R}^C$ as

$$\mathrm{LN}_{\gamma,\beta}(z) = \gamma \odot \frac{z - \bar{z}\mathbf{1}}{\sqrt{C^{-1}\sum_{c=1}^{C}(z_c - \bar{z})^2 + \varepsilon_{\mathrm{LN}}}} + \beta, \qquad \bar{z} = C^{-1}\sum_{c=1}^{C} z_c, \tag{28}$$

where $\varepsilon_{\mathrm{LN}} > 0$ is fixed, γ, β are selectable channel scales and shifts, and $\odot$ denotes componentwise multiplication. The two LayerNorms in a block may have separate parameters. Query–key normalization acts on the d_h channels of each projected query or key. It may use the centered formula above, with $C = d_h$, or the uncentered form (Dehghani et al., 2023)

$$z \longmapsto \gamma \odot \frac{z}{\sqrt{d_h^{-1}\sum_{c=1}^{d_h} z_c^2 + \varepsilon_{\mathrm{QK}}}} + \beta, \qquad \varepsilon_{\mathrm{QK}} > 0. \tag{29}$$

Its stabilizer is positive and its scales are selectable. Shifts may be absent or set to zero. With zero query and key projections and zero positional bias, both forms give uniform attention: $\alpha_j(x, y_j) = 1/\mu_j(\Omega_j)$.

For the normalization proof, we may store each feature together with its negative and add two constant reference channels:

$$v(x) = (c_{\mathrm{ref}}, -c_{\mathrm{ref}}, z_1(x), -z_1(x), \ldots, z_r(x), -z_r(x)), \qquad c_{\mathrm{ref}} > 0. \tag{30}$$

Here z_i stores data, coordinates, or intermediate values, and c_{ref} is finite and selectable. An affine lifting creates this representation. Zero output rows and biases preserve stored channels through the residual paths, while opposite output rows and biases preserve each pair. Appendix C.4 explains how this permitted use of the existing channels controls the effect of LayerNorm.

C.2 Integral Operator Approximation

Two results of Kovachki et al. (2023) provide the starting point. Proposition 6 expresses attention as an integral layer. Theorem 11 then gives the integral neural operator $\mathcal{I}$ needed for the first term of Eq. 15.

Proposition C.1 (Attention as an integral layer; Kovachki et al., 2023, Proposition 6, rephrased)**.** *Self-attention is a special case of a neural-operator integral layer whose kernel depends on the input feature field. Finite sums of attention heads, followed by a linear output projection, have the same integral form.*

For a field $v \in C(\Omega; \mathbb{R}^C)$ and projections $W^Q, W^K, W^V \in \mathbb{R}^{d_h \times C}$, this representation is explicit. As in Eq. 10, we normalize over key locations for each fixed query:

$$\begin{aligned} \alpha[v](x,y) &= \frac{\exp(\langle W^Q v(x), W^K v(y)\rangle/\sqrt{d_h})}{\displaystyle\int_\Omega \exp(\langle W^Q v(x), W^K v(z)\rangle/\sqrt{d_h})\,\mathrm{d}\mu(z)}, \\ (\mathcal{K}(v))(x) &= \int_\Omega \kappa[v](x,y)v(y)\,\mathrm{d}\mu(y), \qquad \kappa[v](x,y) = \alpha[v](x,y)W^V. \end{aligned} \tag{31}$$

The denominator is positive, and equal-weight quadrature recovers discrete softmax attention. Output projections and sums of heads are absorbed into the matrix kernel. Integrating along Ω_j with values $v(x[y_j])$ gives the coordinate attention in Eq. 10.

For operator approximation, the kernels can be fixed functions of (x, y), with pointwise activations providing the nonlinearity. We use the following specialization to continuous fields.

Theorem C.2 (Integral neural-operator approximation; Kovachki et al., 2023, Theorem 11, specialized)**.** *Let $\mathcal{A}, \mathcal{U}$ be the continuous-field spaces in Eq. 23, and let the activation satisfy the assumptions in Appendix C.1. If $\widetilde{\mathcal{G}} : \mathcal{A} \to \mathcal{U}$ is continuous, then for every nonempty compact $K \subset \mathcal{A}$ and every $0 < \delta \leq 1$, there exists a finitely parameterized integral neural operator $\mathcal{I}$ such that*

$$\sup_{a\in K} \|\mathcal{I}(a) - \widetilde{\mathcal{G}}(a)\|_{\mathcal{U}} \leq \delta. \tag{32}$$

The approximant can be chosen as a finite composition of layers of Eq. 16, with $W = 0$, continuous matrix-valued kernels and continuous biases represented by finite neural networks. Each layer uses either the chosen activation σ or the identity, and the final layer uses the identity. Intermediate channel dimensions may vary and remain finite.

Section 9.1 of Kovachki et al. (2023) uses this subclass of Eq. 16, with zero local linear terms and identity lifting and projection. Our rectangular domain has Lipschitz boundary, and our continuous-field spaces and activation satisfy the theorem's hypotheses. The result also covers finitely many channels; equivalence of finite-dimensional norms allows the maximum norm used here.

The cited theorem assumes a target defined on all of $\mathcal{A}$. To apply it to $\mathcal{G} : K \to \mathcal{U}$, use the Dugundji extension theorem (Dugundji, 1951, Theorem 4.1). The set K is closed in the metric space $\mathcal{A}$, and the Banach space $\mathcal{U}$ is locally convex, so a continuous extension $\widetilde{\mathcal{G}} : \mathcal{A} \to \mathcal{U}$ exists. It agrees with $\mathcal{G}$ on K and requires no PDE solvability outside K. The same extension principle appears in Appendix G of Kovachki et al. (2023).

Choose $0 < \delta < \min\{1, \epsilon/3\}$ and apply Theorem C.2 to this extension. We obtain

$$\sup_{a\in K} \|\mathcal{I}(a) - \mathcal{G}(a)\|_{\mathcal{U}} = \sup_{a\in K} \|\mathcal{I}(a) - \widetilde{\mathcal{G}}(a)\|_{\mathcal{U}} \leq \delta < \epsilon/3. \tag{33}$$

This proves the first bound in Eq. 15. With an explicit temporal coordinate, apply the theorem to $\widehat{\mathcal{G}}$ on $\Omega \times \Omega_0$ and then take the final slice as in Eq. 25; this cannot increase the error. We now fix $\mathcal{I}$ and tensorize its kernels.

C.3 Kernel Tensorization

Starting from the fixed $\mathcal{I}$, we replace each kernel by a finite tensor-product approximation and keep its pointwise maps. The first lemma approximates the kernel; the second expresses the resulting operator as a sum of complete coordinate sweeps. We then bound the error through all layers to prove the second term of Eq. 15. In these lemmas, d_a, d_u count the channels of an individual integral layer, and kernel norms are defined in Eq. 26.

Lemma C.3 (Tensor-product approximation)**.** *For every $\eta > 0$, and for every continuous matrix kernel $\kappa : \Omega \times \Omega \to \mathbb{R}^{d_u \times d_a}$, there are finitely many continuous scalar functions $k_{\ell j} : \Omega_j \times \Omega_j \to \mathbb{R}$ and constant matrices $C_\ell \in \mathbb{R}^{d_u \times d_a}$ such that*

$$\|\kappa - \kappa_M\|_\infty < \eta, \qquad \text{with } \kappa_M(x,y) = \sum_{\ell=1}^{M} C_\ell \prod_{j=1}^{d} k_{\ell j}(x_j, y_j). \tag{34}$$

Here M counts tensor terms.

Proof. Regroup $\Omega \times \Omega$ into the coordinate pairs $\prod_j (\Omega_j \times \Omega_j)$. Finite sums of products of continuous functions on these pairs form an *algebra* containing constants and separating points. Stone–Weierstrass (Rudin, 1976, Theorem 7.32) therefore gives scalar uniform density. Approximate each matrix entry within η/d_a and combine its finite expansion using matrix units as the C_ℓ; the maximum row-sum error is then $< \eta$. □

For each tolerance $\eta > 0$ in the next lemma, choose M, C_ℓ, and $k_{\ell j}$ by Lemma C.3. Each product of kernels then becomes a sequence of coordinate integrations.

Lemma C.4 (Coordinate composition)**.** *Let $\mathcal{K}$ be the integral operator on Ω with kernel κ, and let $\mathcal{K}_{\ell j}$ be the coordinate integral operator defined in Eq. 5, with scalar kernel $k_{\ell j}$ from Eq. 34. For every $\eta > 0$,*

$$\mathcal{K}_M = \sum_{\ell=1}^{M} C_\ell (\mathcal{K}_{\ell d} \circ \cdots \circ \mathcal{K}_{\ell 1}), \quad \text{and } \|\mathcal{K} - \mathcal{K}_M\|_{L^1 \to C} \le \|\kappa - \kappa_M\|_\infty < \eta. \tag{35}$$

Each completed integral layer maps $L^1(\Omega)$ continuously into $C(\Omega)$.

Proof. The identity follows by repeated integration, as in Eq. 8. For integrable v, Fubini's theorem (Folland, 1999, Theorem 2.37(b)) identifies the completed sweep with

$$\begin{aligned}(\mathcal{K}_M v)(x) &= \int_\Omega \kappa_M(x,y) v(y)\, \mathrm{d}\mu(y) = \sum_{\ell=1}^{M} C_\ell \int_\Omega \prod_{j=1}^{d} k_{\ell j}(x_j, y_j) v(y)\, \mathrm{d}\mu(y) \\ &= \sum_{\ell=1}^{M} C_\ell \left(\mathcal{K}_{\ell d} \circ \cdots \circ \mathcal{K}_{\ell 1} v\right)(x).\end{aligned} \tag{36}$$

The integral triangle inequality gives

$$\|(\mathcal{K} - \mathcal{K}_M) v\|_C \le \|\kappa - \kappa_M\|_\infty \|v\|_{L^1}. \tag{37}$$

Continuity of the output follows from

$$|\mathcal{K}v(x) - \mathcal{K}v(x')|_\infty \le \sup_y \|\kappa(x,y) - \kappa(x',y)\|_{\infty \to \infty} \|v\|_{L^1} \tag{38}$$

and uniform continuity of the kernel. Therefore, the inequality in Eq. 35 holds. □

For L^1 inputs, intermediate fiber integrals are understood almost everywhere; the full kernel integral gives the continuous output of a completed sweep. Our latent fields are continuous, so every intermediate coordinate integration is also continuous.

The sum has a simple channel interpretation. Copy the input into M groups, apply $\mathcal{K}_{\ell j}$ to group ℓ at coordinate j, and keep the groups separate through the sweep. The linear output projection $[C_1 \ \cdots \ C_M]$ then mixes and sums them. This is the structure carried by the heads and channels in Section 3.1. These ideal sweeps are linear; Appendix C.4 realizes them with attention and pointwise Transformer maps.

Write $\mathcal{I} = \mathcal{F}_n \circ \cdots \circ \mathcal{F}_1$ with finite depth $n \ge 1$. For $0 < \eta \le 1$, approximate each kernel κ_i within η, using a finite number M_i of tensor terms. This defines

$$\mathcal{T} = \widehat{\mathcal{F}}_n \circ \cdots \circ \widehat{\mathcal{F}}_1, \qquad \widehat{\mathcal{F}}_i(v) = \sigma_i(W_i v + \mathcal{K}_{i,M_i} v + b_i). \tag{39}$$

Here $\mathcal{K}_{i,M_i}$ is the sum of complete sweeps from Lemma C.4. The original W_i, b_i, and activation $\sigma_i \in \{\sigma, \mathrm{Id}\}$ are unchanged. The matrices $C_{i,\ell}$ combine the completed sweeps before this activation.

It remains to check that small kernel errors give a small output error. Let $v_i(a)$ and $\widehat{v}_i(a)$ be the fields after layer i of $\mathcal{I}$ and $\mathcal{T}$, respectively, starting from $v_0(a) = \widehat{v}_0(a) = a$. The original intermediate sets are continuous images of compact K, so $B = \max_{0\le i<n} \sup_{a\in K} \|v_i(a)\|_{L^1}$ is finite. Put $\beta = \max\{1, L_\sigma\}$. Since $\|\kappa_{i,M_i}\|_\infty \le \|\kappa_i\|_\infty + 1$, a tensorized layer amplifies an input error by at most

$$\Lambda = \beta \max_{1\le i\le n} \left(\|W_i\|_{\infty\to\infty} + \mu(\Omega)(\|\kappa_i\|_\infty + 1)\right). \tag{40}$$

Both B and Λ are independent of η and the tensor ranks. At each layer, the preceding error is amplified by at most Λ, and replacing the kernel at the original input adds at most $\beta B\eta$. Thus, with $e_i = \sup_{a\in K} \|\widehat{v}_i(a) - v_i(a)\|_{C(\Omega)}$,

$$\begin{aligned} e_0 = 0, \qquad e_i \le \Lambda e_{i-1} + \beta B\eta \quad (1 \le i \le n), \\ \sup_{a\in K} \|\mathcal{T}(a) - \mathcal{I}(a)\|_{\mathcal{U}} = e_n \le \eta\, \beta B \sum_{r=0}^{n-1} \Lambda^r. \end{aligned} \tag{41}$$

The last line follows by applying the recurrence through the n layers. Its coefficient $\Gamma = \beta B \sum_{r=0}^{n-1} \Lambda^r$ is fixed and finite. Choosing $0 < \eta < \min\{1, \epsilon/[3(1+\Gamma)]\}$ therefore proves Eq. 18. Only the norms of the completed kernels enter this estimate.

With an explicit temporal coordinate, include Ω_0 in each sweep and retain the final-slice readout, which cannot increase the error. We now fix this finite $\mathcal{T}$ for the last approximation step.

C.4 Transformer Realization

Fix $\mathcal{T}$ from Appendix C.3. We first approximate its coordinate operations using the continuum Transformer theorem, then implement the resulting construction with our normalized residual blocks. Each step uses an error budget of $\epsilon/6$, giving the third bound in Eq. 15.

Theorem C.5 (Continuum Transformer approximation; Calvello et al., 2025, Theorem 22, specialized)**.** *Let $\mathcal{J} : C(\Omega_j; \mathbb{R}^r) \to C(\Omega_j; \mathbb{R}^{r'})$ be continuous and let $H \subset C(\Omega_j; \mathbb{R}^r)$ be compact. For every $\delta > 0$, there is a finite Transformer neural operator $\mathcal{T}_{j,\theta}$, with coordinate-aware pointwise lifting and readout and without layer normalization, such that*

$$\sup_{w\in H} \|\mathcal{T}_{j,\theta}(w) - \mathcal{J}(w)\|_{C(\Omega_j)} < \delta. \tag{42}$$

The activation may be the function σ fixed in Appendix C.1.

This is the case $s = s' = 0$ of the cited theorem; a smaller tolerance gives the strict inequality. Its proof, Eqs. (36)–(37) of Calvello et al. (2025), uses zero queries and keys, so its nonlocal operation is uniform averaging. Pointwise neural maps supply the remaining transformations. On our domain, the coordinate average is

$$(\mathcal{P}_j v)(x) = \frac{1}{\mu_j(\Omega_j)} \int_{\Omega_j} v(x[y_j])\, \mathrm{d}\mu_j(y_j). \tag{43}$$

The source architecture uses learned linear residual maps and omits LayerNorm. We handle our identity residuals and pre-LayerNorm ordering below.

One parameter choice must work on every coordinate slice. For a compact set of fields $\mathcal{H} \subset C(\Omega; \mathbb{R}^r)$, collect these slices in

$$\mathcal{H}_j = \{y_j \mapsto v(x[y_j]) : v \in \mathcal{H},\ x \in \Omega\} \subset C(\Omega_j; \mathbb{R}^r). \tag{44}$$

This set is compact because taking a slice depends continuously on the field and its location. Each coordinate factor of $\mathcal{T}$ is a continuous linear operator on this one-dimensional function space. The theorem therefore gives one approximation valid on every slice. Its vector-valued formulation handles separate channel groups, whose completed outputs are combined by C_ℓ. The target operator $\mathcal{J}$ includes the signs and integration masses of the factors.

Choose the approximations in order through $\mathcal{T}$. At each stage, use the compact image of K under the approximating stages already chosen. The local error bound then covers the inputs actually encountered. Keep local features in separate channels while forming averages, so later pointwise maps can combine both, as required by the source theorem's residual updates.

The exact coordinate factors and pointwise maps are Lipschitz. If the preceding error is e_{i-1}, a stage with Lipschitz constant λ_i and local error δ_i gives an error at most $\lambda_i e_{i-1} + \delta_i$. As in Appendix C.3, choosing sufficiently small tolerances over the finitely many stages gives a fixed construction $\widetilde{\mathcal{T}}$, using pointwise maps, coordinate averages, and retained features, such that

$$\sup_{a \in K} \|\widetilde{\mathcal{T}}(a) - \mathcal{T}(a)\|_{\mathcal{U}} < \epsilon/6. \tag{45}$$

We next implement this construction with the standard pointwise MLP

$$(\mathcal{M}_j v)(x) = W_{j,2}\rho(W_{j,1}v(x) + b_{j,1}) + b_{j,2}, \tag{46}$$

where $\rho = \sigma$, and the matrices and biases are finite and shared across x. Denoting the two LayerNorms by $\mathcal{N}_{j,1}$ and $\mathcal{N}_{j,2}$, our block is

$$z = v + \mathcal{K}_j(\mathcal{N}_{j,1}v), \qquad \mathcal{S}_j v = z + \mathcal{M}_j(\mathcal{N}_{j,2}z). \tag{47}$$

Lemma C.6 (Realization with standard pre-LayerNorm)**.** *Under the assumptions in Appendix C.1, any fixed finite construction using coordinate averages, continuous pointwise maps, and retained feature channels can be approximated uniformly on compact input sets by blocks of Eq. 11. The model uses standard tokenwise LayerNorm, pointwise lifting and output projection, and finitely many complete coordinate sweeps.*

Proof. Store coordinates and intermediate values in separate channel groups. Each operation reads earlier groups and writes its output into a fresh group initialized to zero. Only finitely many groups are needed. The desired intermediate values are bounded on the compact input set. Enlarge their bounding boxes by a positive margin, then use the two-layer approximation theorem (Pinkus, 1999, Theorem 3.1) to choose accurate pointwise MLPs on these boxes. Fix their weights and a sufficient feature width before choosing the normalization parameters.

To control LayerNorm, use the paired representation from Eq. 30, with reference value $R > 0$:

$$E_R(z) = (R, -R, z_1, -z_1, \ldots, z_r, -z_r), \qquad C = 2 + 2r. \tag{48}$$

Here z collects all stored features, including unused zero entries; extra pairs of zeros may be added. Pairing makes the channel mean zero. Large reference values make the normalization denominator nearly independent of the bounded features. Let π select the entries z_i from each pair $(z_i, -z_i)$. With zero shift and gain $s_R = \sqrt{2R^2/C + \varepsilon_{\mathrm{LN}}}$, direct substitution gives

$$\pi\,\mathrm{LN}_{s_R\mathbf{1},0}(E_R(z)) = \frac{z}{\sqrt{1 + 2\|z\|_2^2/(2R^2 + C\varepsilon_{\mathrm{LN}})}} \longrightarrow z \quad \text{as } R \to \infty, \tag{49}$$

uniformly on bounded boxes, where $\|z\|_2^2 = \sum_i z_i^2$. Thus ordinary LayerNorm can preserve the stored features to any required accuracy with a finite choice of R and gain.

For a pointwise operation, turn off attention and let the MLP read the entries selected by π and write into its fresh output group. The fixed MLP is Lipschitz, so taking R large makes the extra normalization error small. For an average along j, set queries, keys, their shifts, and positional biases to zero. Attention then averages the selected normalized entries into a fresh group, with the MLP branch turned off. Averaging has norm one, so the same LayerNorm estimate controls its error. Query–key normalization preserves the zero queries and keys.

Zero output rows and biases preserve the reference and earlier feature groups through the identity residuals. Opposite output rows and biases preserve each pair exactly. Every feature needed later therefore remains available, and a final linear projection selects the output. The source architecture's learned residual maps, lifting, and readout are included among the pointwise maps realized this way.

Choose the pointwise approximation errors first, then a finite R making all normalization errors sufficiently small. Uniform continuity on the enlarged boxes lets us propagate these errors through

the finite construction, keeping intermediate values inside the boxes and making the final error arbitrarily small. Insert identity blocks at inactive coordinates and complete the last sweep if needed. Each average then occurs along its required coordinate, while pointwise operations can occur at any stage. □

Apply the lemma to $\widetilde{\mathcal{T}}$ with error below $\epsilon/6$. Together with Eq. 45, this gives

$$\begin{aligned}\sup_{a\in K}\|\mathcal{T}_\theta(a)-\mathcal{T}(a)\|_{\mathcal{U}} &\le \sup_{a\in K}\|\mathcal{T}_\theta(a)-\widetilde{\mathcal{T}}(a)\|_{\mathcal{U}}+\sup_{a\in K}\|\widetilde{\mathcal{T}}(a)-\mathcal{T}(a)\|_{\mathcal{U}}\\ &< \epsilon/6+\epsilon/6=\epsilon/3.\end{aligned} \tag{50}$$

This proves the third bound in Eq. 15. With an explicit temporal coordinate, include averages along Ω_0 and keep the final-slice readout. Its norm bound preserves the same error in $\mathcal{U}$.

D Experimental Details

D.1 Datasets

We use four datasets from *The Well* (Ohana et al., 2024), covering radiatively cooling gas, active suspensions, thermal convection, and magnetized turbulence. Below, we describe their governing equations, physical fields, and the data used in our experiments. Spatial resolutions follow the stored coordinate order, and trajectory counts refer to the supplied training, validation, and test partitions. We use the supplied spatial resolutions without additional spatial resampling in our data loader.

Turbulent Radiative Layer (TRL) (Fielding et al., 2020). This dataset describes the mixing of hot, dilute gas and cold, dense gas in a two-dimensional shear layer. The Kelvin–Helmholtz instability generates turbulence and populates an intermediate-temperature phase that cools rapidly, transferring gas from the hot phase to the cold phase. The dynamics follow the compressible Euler equations with radiative cooling:

$$\begin{aligned}\partial_t\rho+\nabla\cdot(\rho\mathbf{u})&=0,\\ \partial_t(\rho\mathbf{u})+\nabla\cdot(\rho\mathbf{u}\otimes\mathbf{u}+p\mathbf{I})&=0,\\ \partial_t E+\nabla\cdot((E+p)\mathbf{u})&=-\mathcal{C}(\rho,p;t_{\mathrm{cool}}).\end{aligned} \tag{51}$$

Here, ρ, $\mathbf{u}$, and p are the density, velocity, and pressure; $E=p/(\gamma-1)+\rho|\mathbf{u}|^2/2$ is the total energy density, with $\gamma=5/3$. The net radiative cooling rate $\mathcal{C}$ is controlled by the characteristic cooling time t_{cool}. The simulations use Athena++ on $[-0.5,0.5]\times[-1,2]$, with periodic boundaries in x and zero-gradient boundaries in y. The dataset contains 90 trajectories across nine cooling times from 0.03 to 3.16. Each trajectory contains 101 snapshots on a uniform 128×384 grid, with a nominal sampling interval of 1.59703 in simulation time. We predict density, pressure, and both velocity components, giving four physical channels, and use $72/9/9$ trajectories for training/validation/testing.

Active Matter (AM) (Maddu et al., 2024). This dataset describes rod-like active particles suspended in a two-dimensional Stokes fluid. Particle alignment and active stresses generate collective motion, vortices, and transitions between isotropic and nematic states. The underlying kinetic model represents particle positions $\mathbf{x}$ and orientations $\mathbf{p}\in S^1$ through a distribution $\Psi(\mathbf{x},\mathbf{p},t)$, which satisfies the Smoluchowski equation:

$$\partial_t\Psi+\nabla_{\mathbf{x}}\cdot(\dot{\mathbf{x}}\Psi)+\nabla_{\mathbf{p}}\cdot(\dot{\mathbf{p}}\Psi)=0, \tag{52}$$

where S^1 is the unit circle and $\nabla_{\mathbf{p}}$ is its tangential gradient. The translational and orientational flux velocities are

$$\begin{aligned}\dot{\mathbf{x}}&=\mathbf{u}-d_T\nabla_{\mathbf{x}}\log\Psi,\\ \dot{\mathbf{p}}&=(\mathbf{I}-\mathbf{p}\otimes\mathbf{p})\cdot(\nabla\mathbf{u}+2\zeta\mathbf{D})\cdot\mathbf{p}-d_R\nabla_{\mathbf{p}}\log\Psi.\end{aligned} \tag{53}$$

Here, d_T and d_R are translational and rotational diffusion coefficients, and ζ is the alignment strength. With orientational moments defined by $\langle F\rangle=\int_{S^1}F(\mathbf{p})\Psi(\mathbf{x},\mathbf{p},t)\,\mathrm{d}\mathbf{p}$, the concentration is $c=\langle 1\rangle$, the second moment is $\mathbf{D}=\langle\mathbf{p}\otimes\mathbf{p}\rangle$, and the fourth moment is $\mathbf{S}=\langle\mathbf{p}^{\otimes4}\rangle$. The

distribution is coupled to incompressible Stokes flow through the particle stress:

$$\begin{aligned} -\Delta \mathbf{u} + \nabla p = \nabla \cdot \mathbf{\Sigma}, \qquad \nabla \cdot \mathbf{u} = 0, \\ \mathbf{\Sigma} = \alpha \mathbf{D} + \beta \mathbf{S} : \mathbf{E} - 2\zeta\beta(\mathbf{D} \cdot \mathbf{D} - \mathbf{S} : \mathbf{D}). \end{aligned} \tag{54}$$

Here, α is the active dipole strength, β characterizes particle density, $\mathbf{E} = (\nabla \mathbf{u} + (\nabla \mathbf{u})^\top)/2$ is the strain-rate tensor, and : denotes contraction over two tensor indices. The simulations use a periodic domain $[0, 10]^2$, with $\alpha \in \{-5, -4, -3, -2, -1\}$ and $\zeta \in \{1, 3, 5, 7, 9, 11, 13, 15, 17\}$. The dataset contains 225 trajectories, each with 81 snapshots on a 256×256 uniform grid over $t \in [0, 20]$, sampled every 0.25 time units. We predict concentration, two velocity components, and all four stored components of each of $\mathbf{D}$ and $\mathbf{E}$, giving 11 channels. The training/validation/test partitions contain $175/24/26$ trajectories.

Rayleigh–Bénard Convection (RB) (Burns et al., 2020). This dataset describes a two-dimensional fluid layer heated from below and cooled from above. Buoyancy drives convective cells and turbulent motion, while viscosity and thermal diffusion dissipate momentum and temperature variations. The nondimensional Boussinesq equations are

$$\begin{aligned} \partial_t b + \mathbf{u} \cdot \nabla b = \kappa \Delta b, \\ \partial_t \mathbf{u} + (\mathbf{u} \cdot \nabla)\mathbf{u} = -\nabla p + \nu \Delta \mathbf{u} + b\mathbf{e}_y, \\ \nabla \cdot \mathbf{u} = 0, \end{aligned} \tag{55}$$

where b is buoyancy, $\mathbf{u}$ is velocity, p is pressure, and $\mathbf{e}_y$ is the vertical unit vector. The thermal diffusivity and viscosity are $\kappa = (Ra\,Pr)^{-1/2}$ and $\nu = (Ra/Pr)^{-1/2}$, where Ra and Pr are the Rayleigh and Prandtl numbers. The domain is $[0, 4] \times [0, 1]$, with periodic boundaries in x and fixed-buoyancy, no-slip conditions at the horizontal plates:

$$b(x, 0, t) = 1, \qquad b(x, 1, t) = 0, \qquad \mathbf{u}(x, 0, t) = \mathbf{u}(x, 1, t) = \mathbf{0}. \tag{56}$$

The data were generated with Dedalus, varying Ra from 10^6 to 10^{10}, Pr from 0.1 to 10, and the initial buoyancy perturbations. The dataset contains 1,750 trajectories with 200 stored snapshots each and a nominal sampling interval of 0.25; the actual stored intervals are not exactly uniform. We use the supplied 512×128 fields and predict buoyancy, pressure, and two velocity components. The training/validation/test partitions contain $1{,}400/175/175$ trajectories.

Compressible Magnetohydrodynamic Turbulence (MHD) (Burkhart et al., 2020). This dataset describes three-dimensional compressible turbulence in an isothermal magnetized fluid. Large-scale driving sustains turbulent motions, while magnetic pressure and tension couple the velocity and magnetic fields. The generating simulations solve the ideal MHD equations (Ohana et al., 2024, Appendix C.7):

$$\begin{aligned} \partial_t \rho + \nabla \cdot (\rho \mathbf{u}) = 0, \\ \partial_t (\rho \mathbf{u}) + \nabla \cdot \left[\rho \mathbf{u} \otimes \mathbf{u} + \left(p + \frac{|\mathbf{B}|^2}{8\pi} \right) \mathbf{I} - \frac{\mathbf{B} \otimes \mathbf{B}}{4\pi} \right] = \mathbf{f}, \\ \partial_t \mathbf{B} - \nabla \times (\mathbf{u} \times \mathbf{B}) = 0, \\ \nabla \cdot \mathbf{B} = 0, \qquad p = c_s^2 \rho. \end{aligned} \tag{57}$$

Here, ρ, $\mathbf{u}$, $\mathbf{B}$, and p denote density, velocity, magnetic field, and gas pressure; c_s is the constant isothermal sound speed, and $\mathbf{f}$ is external momentum forcing. The isothermal relation determines pressure, so no separate energy equation is evolved. The simulations use periodic boundaries and continuous solenoidal driving at large spatial scales, covering sonic Mach numbers $\mathcal{M}_s \in \{0.5, 0.7, 1.5, 2, 7\}$ and Alfvén Mach numbers $\mathcal{M}_A \in \{0.7, 2\}$. We use the released `MHD_64` data, obtained by low-pass filtering and downsampling the original 256^3 simulations to 64^3. The stored coordinates span $[0, 1]^3$, and each trajectory contains 100 snapshots separated by 0.01 time units. We predict density, three velocity components, and three magnetic-field components, giving seven channels. The data available to our experiments contain $77/10/10$ training/validation/test trajectories.

D.2 Field Normalization

Following Walrus (McCabe et al., 2026), we apply samplewise root-mean-square normalization to the input fields. We first standardize each physical variable using the dataset statistics. For each sample, we then compute the root-mean-square magnitude of each variable over the input history and all spatial nodes, with a lower bound of 10^{-5}. We divide the input and target by these input-derived scales and multiply the decoded prediction by the same scales. This restores the prediction to the dataset-standardized scale without using future observations for samplewise normalization. We report VRMSE in physical-field space, after reversing the samplewise normalization, rather than in normalized model-output space.

D.3 Training and Evaluation Protocol

Following *The Well* (Ohana et al., 2024), each model receives four consecutive snapshots, represented by $\mathbf{a} \in \mathbb{R}^{T\times N_1\times\cdots\times N_d\times D}$ with $T = 4$, and predicts the next snapshot $\tilde{\mathbf{u}} \in \mathbb{R}^{N_1\times\cdots\times N_d\times D}$. Here, N_j is the grid size along spatial dimension j, and D is the number of physical channels. Longer forecasts are generated autoregressively by appending each prediction to the input history and discarding the oldest snapshot.

Training. During training, models perform one-step prediction and minimize the mean squared error (MSE) on the normalized fields described above (McCabe et al., 2026):

$$\mathcal{L}_{\mathrm{MSE}} = \frac{1}{ND}\sum_{c=1}^{D}\sum_{i=1}^{N}\left(\tilde{u}^{\mathrm{n}}_{i,c} - u^{\mathrm{n}}_{i,c}\right)^2, \tag{58}$$

where $N = \prod_{j=1}^{d} N_j$ is the number of spatial nodes, i indexes these nodes, and c indexes physical channels. The superscript n denotes samplewise normalized values; $\tilde{u}^{\mathrm{n}}_{i,c}$ and $u^{\mathrm{n}}_{i,c}$ are the prediction and target, respectively. The loss is averaged over samples in each training batch.

We use the same training recipe across attention patterns within each benchmark. Muon updates trainable matrix parameters, while AdamW updates the remaining parameters (Jordan et al., 2024; Loshchilov & Hutter, 2019), both with weight decay 10^{-4}. The learning rate increases linearly from 2×10^{-5} to 2×10^{-4} during warm-up, followed by cosine decay toward 2×10^{-5}. We set a target of $100{,}000$ optimizer updates and $5{,}000$ warm-up updates. All models are trained on two NVIDIA H200 GPUs, with a per-GPU batch size of 8 for the 2D benchmarks and 2 for the 3D benchmark MHD. We clip the gradient norm at 1.0 and use dropout 0.1 and random seed 211.

Evaluation. We evaluate the final training checkpoints using the variance-scaled root mean squared error (VRMSE) from *The Well* (Ohana et al., 2024). For each prediction, we compute the spatial error separately for each physical channel and average the resulting scores:

$$\mathrm{VRMSE} = \frac{1}{D}\sum_{c=1}^{D}\sqrt{\frac{\frac{1}{N}\sum_{i=1}^{N}(\tilde{u}_{i,c} - u_{i,c})^2}{s_c^2 + 10^{-7}}}, \tag{59}$$

where $\bar{u}_c = N^{-1}\sum_{i=1}^{N} u_{i,c}$ and $s_c^2 = (N-1)^{-1}\sum_{i=1}^{N}(u_{i,c} - \bar{u}_c)^2$ are the target channel's spatial mean and sample variance. We compute VRMSE in physical-field space after reversing the samplewise normalization, as described above. Vector and tensor components are treated as separate channels with equal weight.

For one-step evaluation, we average scores over all eligible input windows within each test trajectory and then over trajectories. For ten-step evaluation, each rollout starts from the first four eligible ground-truth snapshots of a test trajectory and proceeds without further ground-truth inputs. At each step, the input-derived normalization scales are recomputed from the updated history. We report the mean VRMSE over the ten forecast steps and all test trajectories.

D.4 Model Setup

Tensorized Transformer. We use small, medium, and large configurations following the official ViT repository (Dosovitskiy et al., 2021), as summarized in Table 6. Within each size, the embedding

width, MLP width, number of heads, and total block count are held fixed across attention patterns and temporal representations. Each tensorized block attends along one coordinate, and the blocks are arranged into complete coordinate sweeps as described in Section 3.2. For windowed attention, we use 8×8 windows ($w = 8$) in 2D and $4 \times 4 \times 4$ windows ($w = 4$) in 3D.

Table 6: Shared configurations for full, windowed, and tensorized Transformer modules. Parameter counts are approximate and vary slightly with the dataset and temporal representation.

Size	Embed dim.	MLP dim.	Heads	Blocks	Params (M)
Small	384	1536	6	12	22
Medium	768	3072	12	12	89
Large	1024	4096	16	24	310

FactFormer. We use the paper's single configuration for time-dependent two-dimensional flow, developed for Kolmogorov flow (Li et al., 2023a, Table 5), and adapt its inputs and outputs to RB. It uses a feature width of 128, four factorized-attention layers, eight heads, and a kernel dimension of 128 per head. The value dimension is 64 per head, and the latent propagator uses width 256.

AViT. We evaluate the small, base, and large AViT configurations (McCabe et al., 2024), summarized in Table 7. Each space–time block applies temporal attention followed by parallel spatial axial attention with shared projections and averaged spatial outputs. The MLP hidden width is four times the embedding width, with GELU activation. All three configurations use relative-position biases and a stochastic-depth rate of 0.1.

Table 7: AViT configurations evaluated on RB. Block counts refer to complete space–time blocks; parameter counts include the RB input and output modules.

Size	Embed dim.	MLP dim.	Heads	Blocks	Params (M)
Small	384	1536	6	12	29
Base	768	3072	12	12	116
Large	1024	4096	16	24	407

MORPH. We evaluate the small, medium, and large MORPH configurations (Rautela et al., 2026), summarized in Table 8. Each uses a constant-width processor, with temporal attention followed by parallel spatial axial attention using separate projections and summed spatial outputs. The field encoder uses 32 cross-attention heads at all three scales. For RB, pressure and buoyancy are encoded as separate scalar fields, while the two velocity components form one vector field.

The medium and large configurations overfit during training. We therefore evaluate these MORPH configurations using the checkpoints with the best validation performance; all other models use their final training checkpoints.

Table 8: MORPH configurations evaluated on RB. Head counts apply to each temporal or spatial attention module; block counts refer to complete temporal–spatial processor blocks. Parameter counts include the field encoder and decoder.

Size	Embed dim.	MLP dim.	Heads	Blocks	Params (M)
Small	512	2048	8	4	27
Medium	768	3072	12	8	110
Large	1024	4096	16	16	379

E Additional Results

E.1 Prediction Accuracy with Standard Errors

Table 9 supplements the one-step results in Table 3 with SEM estimates from the same evaluation checkpoints.

Table 9: One-step VRMSE ± standard error of the mean (SEM) for the six large-model configurations on four benchmarks; lower is better. Underlining marks the lowest mean within each three-row group, and bold marks the lowest mean overall in each column.

Model	TRL	AM	RB	MHD
FF(F)	0.17026 ± 0.02364	0.01299 ± 0.00097	0.02257 ± 0.00014	0.07961 ± 0.00763
WW(W)	0.18445 ± 0.02490	0.02325 ± 0.00265	0.02076 ± 0.00013	0.08959 ± 0.00842
XY(Z) (Ours)	$\underline{0.15999} \pm 0.02335$	$\underline{0.01155} \pm 0.00078$	$\mathbf{0.01840} \pm 0.00013$	$\underline{0.06789} \pm 0.00683$
TFF(F)	0.15907 ± 0.02471	0.01141 ± 0.00077	0.02395 ± 0.00014	0.07925 ± 0.00742
TWW(W)	0.17549 ± 0.02501	0.01971 ± 0.00211	0.02088 ± 0.00014	0.08653 ± 0.00832
TXY(Z) (Ours)	$\mathbf{0.15193} \pm 0.02383$	$\mathbf{0.01003} \pm 0.00054$	$\underline{0.01978} \pm 0.00014$	$\mathbf{0.06599} \pm 0.00670$

SEM is computed from per-trajectory mean errors, with physical-parameter stratification for RB. It quantifies test-set uncertainty for a fixed trained model. The test sets contain 9, 26, 175, and 10 trajectories for TRL, AM, RB, and MHD, respectively.

E.2 Autoregressive Rollout Evaluation

Fig. 9 resolves the ten-step mean errors in Table 3 into individual forecast steps. Each rollout starts from the first four eligible ground-truth snapshots of a test trajectory and then advances using only model predictions. TXY(Z) achieves the lowest VRMSE throughout the displayed horizon on TRL and RB, while XY(Z) leads on AM from step 3 onward.

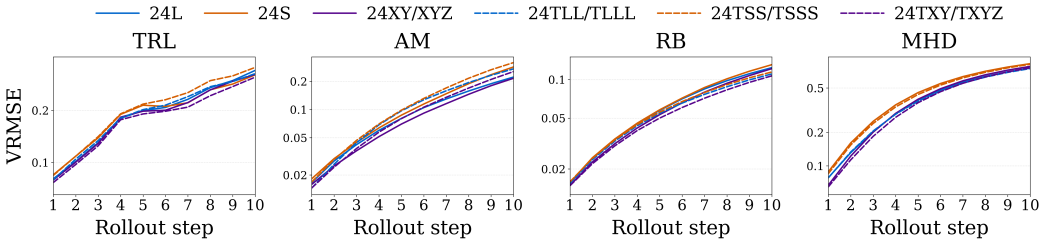

Figure 9: Ten-step autoregressive VRMSE for the six large-model configurations on TRL, AM, RB, and MHD. Scores are averaged over physical channels and test trajectories at each step. Vertical axes are logarithmic. In the legend, L and S denote full and windowed spatial attention, XY/XYZ denote tensorized spatial attention, and 24 indicates the number of blocks. Solid and dashed lines denote temporal aggregation and temporal attention (T), respectively.