COAST: Intelligent Time-Adaptive Neural Operators

Zhikai Wu [1 2] **Shiyang Zhang** [2] **Sizhuang He** [2] **Sifan Wang** [3] **Min Zhu** [4] **Anran Jiao** [4] **Lu Lu** [4]
David van Dijk [2 5]

Abstract

Operator learning for time-dependent partial differential equations (PDEs) has seen rapid progress, enabling efficient modeling of complex spatiotemporal dynamics. However, most existing approaches use fixed time step sizes during rollout, limiting their ability to adapt to varying temporal complexity and leading to error accumulation. We introduce COAST (Causal Operator with Adaptive Solver Transformer), a novel operator learning framework that integrates causal attention with adaptive time stepping. COAST jointly predicts the next step size and the corresponding future system state. The learned step sizes dynamically adapt both across and within trajectories—assigning smaller step sizes to regions with rapidly changing dynamics and larger steps to smoother transitions. We evaluate the COAST across a range of dynamical systems, which consistently outperforms state-of-the-art methods in both accuracy and efficiency, demonstrating the potential of causal transformers for adaptive operator learning in time-dependent systems.

1. Introduction

Partial differential equations (PDEs) underpin a vast spectrum of scientific models, from fluid mechanics to quantum physics (Evans, 1998). Classical numerical schemes—finite differences, finite volumes, and finite elements—form the bedrock of scientific computing (LeVeque, 2007), yet they struggle when confronted with multi-scale phenomena, irregular geometries, or stringent real-time constraints. Adaptive solvers have emerged as a powerful solution, dynamically adjusting spatial and temporal discretization based on local error estimates (Rannacher, 2003). These methods significantly improve computational efficiency by concentrating computational resources where needed most. However, they face inherent limitations: high computational overhead for frequent refinement, careful parameter tuning requirements, and challenges with stiff problems (Hairer et al., 1993).

Recent advances in operator learning have opened new avenues for PDE solving (Li et al., 2020a). Instead of traditional numerical approximations, these approaches learn mappings between function spaces, enabling rapid solution prediction for entire families of PDEs. Several architectures have demonstrated remarkable success: Fourier Neural Operators (FNO) leverage the spectral domain for efficient learning (Li et al., 2020a), DeepONet employs the universal approximation theorem for operators (Lu et al., 2021), and transformer-based models like Oformer (Li et al., 2022), DPOT (Hao et al., 2024a), CViT (Wang et al., 2024b), Transolver (Wu et al., 2024a) adapt attention mechanisms for PDE solving. These methods have achieved impressive results across various applications, from fluid dynamics (Azizzadenesheli et al., 2024), solid mechanics (Wang et al., 2024a), to heat transfer (Roy et al., 2024), often leading to reasonable accuracy and great computational efficiency.

Despite these advances, current machine learning approaches for PDE solving face a significant limitation: they typically operate with fixed time steps. This constraint becomes particularly problematic when dealing with problems exhibiting multiple time scales or rapid temporal variations. To address this limitation, we design Causal Operator with Adaptive Solver Transformer (COAST) as shown in Figure 1, which dynamically adjusts temporal resolution. Our contributions are:

- **Causal Operator with Adaptive Solver Transformer (COAST).** A causal-attention transformer that jointly predicts physical states and adaptive time steps, enabling efficient continuous-time inference through interpolation.
- **State-of-the-art performance.** COAST outperforms prior operator learning models in accuracy and efficiency

[1]Yuanpei College, Peking University [2]Department of Computer Science, Yale University [3]Institute for Foundations of Data Science, Yale University [4]Department of Statistics and Data Science, Yale University [5]School of Medicine, Yale University. Correspondence to: Lu Lu <lu.lu@yale.edu>, David van Dijk <david.vandijk@yale.edu>.

The second AI for MATH Workshop at the 42^{nd} International Conference on Machine Learning, Vancouver, Canada.

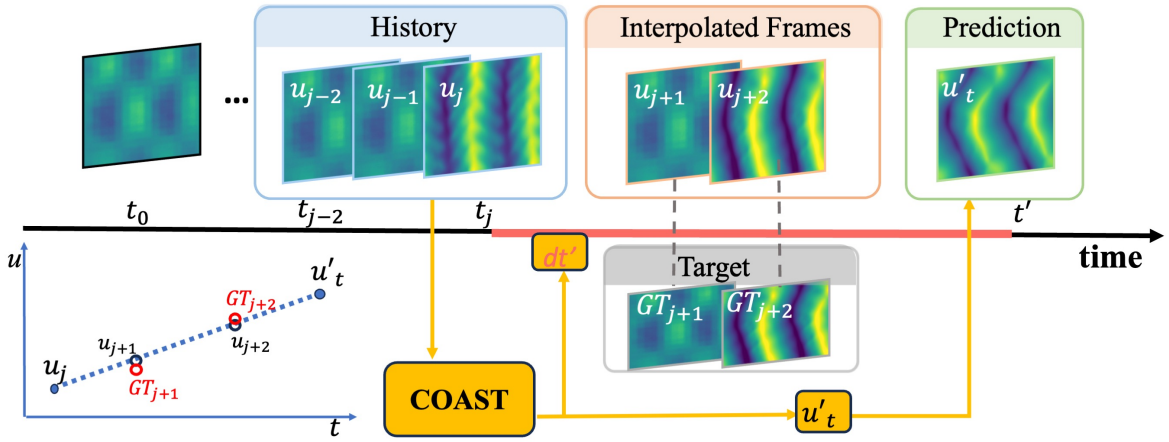

Figure 1. **Overview of the COAST framework for continuous operator learning.** COAST is an intelligent, time-adaptive neural operator that predicts both the system's future evolution and the optimal time step. It takes historical frames $(\ldots, u_{j-2}, u_{j-1}, u_j)$ as input and predicts the next state u'_t and adaptive step size dt. Intermediate frames $(u_{j+1}, u_{j+2}, \ldots)$ are reconstructed via interpolation to align with ground truth samples $(GT_{j+1}, GT_{j+2}, \ldots)$. By adaptively selecting time steps, COAST achieves continuous-time prediction with fewer function calls while preserving accuracy. A complete schematic of model architecture can be found in Appendix Figure 9.

across challenging PDE benchmarks.

- **Interpretable adaptivity.** Learned step sizes correlate with dynamical complexity, revealing the causal-attention model's ability to reason about physics.

2. Method

Conventional operator learning methods use uniform time discretization, which fails to adapt to the uneven temporal complexity inherent in many physical systems. This inefficiency leads to unnecessary computations during periods of gradual change and insufficient resolution during rapid transitions. To overcome these limitations, we propose Causal Operator with Adaptive Solver Transformer (COAST), a framework that enables temporally continuous prediction with adaptive time stepping informed by the underlying dynamics.

Given spatial fields at arbitrary times, COAST: (a) encodes frames with a spatio-temporal encoder, (b) processes tokens through a causal transformer, (c) infers an adaptive step size dt and modifies embedding, and (d) decodes to the next spatial state, permitting interpolation in $[0, dt]$.

2.1. Architecture

Spatio-Temporal Encoder. Following the neural operator paradigm (Kovachki et al., 2024), we first discretize the continuous field $\mathbf{u}$ on a spatio-temporal grid, resulting in an input tensor $\mathbf{u}_{\text{in}} \in \mathbb{R}^{T\times H\times W\times D}$. This tensor represents an ordered sequence of T spatial frames $\mathbf{S}_i \in \mathbb{R}^{H\times W\times D}$, each containing D physical channels.

The COAST encoder processes the spatio-temporal input tensor $\mathbf{u}_{\text{in}}$. It also takes a corresponding sequence of relative timestamps $\mathbf{T}_{\text{seq}} \in \mathbb{R}^T$. When input frames are evenly spaced, $\mathbf{T}_{\text{seq}}$ can be omitted since it is implicitly defined.

We define the most recent frame $\mathbf{S}_T$ to be at $t = 0$, such that $\mathbf{u}(0) = \mathbf{S}_T$. Each spatial frame is independently tokenized using a CNN with overlapping kernels, resulting in patchified inputs $\mathbf{u}_p \in \mathbb{R}^{T\times \frac{H}{P}\times \frac{W}{P}\times C}$, where P is the patch size and C is the embedding dimension.

We add learnable 2D spatial positional embeddings $\mathbf{PE}_s$ and modulate each frame using FiLM Layer ((Perez et al., 2017), implementation details in Appendix B.2) to embed its corresponding temporal information:

$$\mathbf{u}_s = \mathbf{u}_p + \mathbf{PE}_s,\ \mathbf{PE}_s \in \mathbb{R}^{1\times \frac{H}{P}\times \frac{W}{P}\times C};$$

$$\mathbf{u}_{st} = \text{FiLM}(\mathbf{T}_{seq}, \mathbf{u}_s).$$

Causal Transformer Processor. We use a causal attention Transformer block to estimate the future state at the next time step. The processor has an embedding size of C. The input $\mathbf{u}_{st}$ is reshaped to $\mathbb{R}^{(\frac{H}{P}\cdot\frac{W}{P})\times T\times C}$, where each spatial patch is treated as a temporal sequence of length T.

These sequences are processed by the Transformer block. To obtain the latent representation of the future state at the next time step, we extract the final token from each sequence output by the Transformer block, yielding $\mathbf{z} \in \mathbb{R}^{(\frac{H}{P}\cdot\frac{W}{P})\times 1\times C}$. Here, $\mathbf{z}_k$ denotes the element corresponding to the k-th spatial patch, representing the predicted state of that patch at the next time step.

Interpret-Modify Mechanism. This module serves two complementary purposes: (i) it infers an adaptive prediction

time step dt, and (ii) it conditions the latent tokens $\mathbf{z}$ on this value. For the time step size, we apply a lightweight MLP to each token in the latent representation $\mathbf{z}$ and average the resulting scalars to obtain a global inferred time step dt:

$$dt = \frac{1}{|\mathbf{z}|} \sum_k \mathrm{MLP}(\mathbf{z}_k).$$

To avoid a degenerate estimate of the step size, we constrain dt with upper and lower bounds and introduce a regularization loss term penalizing overly small values. We use a piecewise power-exponential function as our regularization loss term:

$$\mathcal{L}_{dt} = \begin{cases} (1 + \varepsilon - dt)^m, & dt \leq 1 + \varepsilon, \\ 0, & dt > 1 + \varepsilon, \end{cases}$$

with $0 < \varepsilon < 1$ and $m \geq 1$. A detailed ablation study of the hyper-parameters m and ε is provided in Section 4.6.

We then modify each token in $\mathbf{z}$ via a FiLM conditioning (see Appendix B.2) on dt and reshape them back to the spatial grid:

$$\mathbf{z} = \mathrm{FiLM}(dt, \mathbf{z}) \quad \in \quad \mathbb{R}^{\frac{H}{P} \times \frac{W}{P} \times 1 \times C}.$$

Interpolation Decoder. To reconstruct the predicted state $\tilde{\mathbf{u}}(dt) \in \mathbb{R}^{H \times W \times D}$ from its latent representation $\mathbf{z}$, we employ a Transposed-CNN decoder with overlapping kernels:

$$\tilde{\mathbf{u}}(dt) = \mathrm{TransCNN}(\mathbf{z}).$$

For prediction at any time point $t \in [0, dt]$, COAST computes $\tilde{\mathbf{u}}(t) \in \mathbb{R}^{H \times W \times D}$ via linear interpolation between $\mathbf{u}(0)$ and the decoder output $\tilde{\mathbf{u}}(dt)$:

$$\tilde{\mathbf{u}}(t) = \mathbf{u}(0) + \frac{t}{dt}(\tilde{\mathbf{u}}(dt) - \mathbf{u}(0)).$$

This interpolation yields temporally continuous predictions within the interval $[0, dt]$ without re-executing the network. For horizons beyond dt, COAST operates autoregressively, concatenating the input with previously predicted states and rerunning the model with the new sequence.

2.2. Detailed Explanations

Causal Attention & Decoder. Causal attention Section 2.1 prevents future information leakage, aligning information flow with physical time. This enhances temporal reasoning and step-size reliability. The decoder Section 2.1 uses final tokens that aggregate full-sequence context via causal attention, ensuring dt reflects global dynamics.

Patch Continuity. Spatial continuity is preserved via overlapping kernels in the encoder/decoder CNNs, mitigating boundary artifacts.

3. Related Work

Neural Operators. Neural operators aim to learn mappings between infinite-dimensional function spaces and have emerged as powerful data-driven surrogates for solving partial differential equations (PDEs). Foundational architectures such as DeepONet (Lu et al., 2021) and the Fourier Neural Operator (Li et al., 2020a) (FNO) introduced end-to-end frameworks for approximating nonlinear operators, inspiring a series of architectural extensions. Variants including GNO (Li et al., 2020b), UNO (Rahman et al., 2023), and AFNO (Guibas et al., 2022) have enhanced this paradigm by improving expressivity, scalability, and generalization across diverse PDE types. Transformer-based neural operators—such as GK-Transformer (Cao, 2021), OFormer (Li et al., 2023b), and CViT (Wang et al., 2025)—further advance this line of work by leveraging self-attention to capture global dependencies and long-range spatiotemporal dynamics more effectively.

To address PDEs defined on irregular domains or complex geometries, models like Transolver (Wu et al., 2024b), UPT (Alkin et al., 2025), and GINO (Li et al., 2023a) incorporate geometric priors or mesh-based representations. In parallel, foundation models for PDEs—such as DPOT (Hao et al., 2024a), Poseidon (Herde et al., 2024), MPP (McCabe et al., 2024), and PDEformer (Ye et al., 2025)—seek to generalize across a wide range of operators and boundary conditions via large-scale pretraining. Another complementary direction integrates physics-based inductive biases into operator learning: hybrid methods like PI-DeepONet (Wang et al., 2021) and PINO (Li et al., 2023c) embed governing physical laws through physics-informed objectives, leading to improved accuracy and data efficiency in low-data or extrapolative regimes.

Transformers as Neural Operators. The distinction of neural operators lies in learning mappings between infinite-dimensional function spaces. As defined in (Kovachki et al., 2024), neural operators approximate operators through compositions of integral kernels and nonlinear activations.

Paper (Kovachki et al., 2024) shows that *transformers are special cases of neural operator*. COAST aligns with this framework:

- Function space setting: While inputs/outputs are discretized as tensors $\mathbb{R}^{H \times W \times D}$ for numerical implementation, the model inherently learns a continuous operator. The tensor represents functions analogous to FNOs that discretize functions on grids but maintain resolution in-

variance.

- Operator structure: The self-attention mechanism implicitly parameterizes integral kernels. Each attention head computes a kernel $\kappa(x, y)$ over coordinates, akin to the integral operators in neural operators $Kv(x) = \int \kappa(x, y)v(y)dy$. Nonlinear activations and residual connections further mirror the neural operator's iterative kernel integrations.

- Discretization invariance: The model's position encoding and attention are coordinate-based, enabling evaluation at arbitrary resolutions, a hallmark of neural operator.

Thus, COAST satisfies neural operator criteria (Kovachki et al., 2024); its tensor output is a discretization for practicality, not a conceptual limit.

COAST vs. Traditional Adaptive Solvers. At first glance COAST mimics an adaptive time-stepping scheme, yet the two paradigms diverge fundamentally. So the comparison between operator learning models (i.e., COAST) and adaptive solvers is fundamentally misaligned:

- Resolution & efficiency. Traditional solvers such as RK45 target strict local-error tolerances, often forcing $\Delta t \approx 10^{-3}$–10^{-5} for stiff PDEs and thus $> 10^3$ steps for modest horizons. Neural operators learn coarse-to-fine mappings with much larger Δt (10^{-2}–10^{-1}) (Li et al., 2020a), yielding 10^2–$10^3\times$ fewer evaluations. Direct comparison ignores this inherent efficiency gap (Krishnapriyan et al., 2021).

- Error Accumulation: Adaptive solvers minimise local truncation error but still accrue global error over long rollouts. Sequence-to-sequence training lets operator learners optimise the multi-step error directly, reporting 3–5$\times$ lower ℓ^2 error at $T = 10$ (Brandstetter et al., 2023). This aligns with COAST's focus on PDE surrogate modeling, not local step-wise precision.

- Baseline Alignment: COAST is evaluated under the standard neural-operator protocol (FNO (Li et al., 2020a), DeepONet (Lu et al., 2021)), where advances are most meaningfully measured. Comparing to high-fidelity adaptive solvers would parallel comparing symbolic surrogates to finite-element solvers—a mismatch well argued in (Krishnapriyan et al., 2021; Brunton et al., 2020).

In short, COAST targets fast, accurate surrogate modelling, not drop-in replacement of high-order solvers; the learned adaptivity simply narrows the gap between data-driven surrogates and physics-driven integrators.

4. Experiments

We benchmark COAST on four demanding PDE datasets, compare it with state-of-the-art neural operators and strong computer-vision surrogates, and analyse its error accumulation, inference efficiency, adaptive step-size behaviour, scalability, and ablation studies.

Datasets. All experiments use the **The-Well** collection (Ohana et al., 2024), which provides high-resolution rollouts for:

- **Active matter** (*AM*) (Maddu et al., 2024): Active matter systems, composed of energy-transforming agents that generate orientation-dependent viscosity and transmit forces, exhibit complex nonlinear spatiotemporal dynamics in viscous fluids.

- **Turbulent radiative layers** (*TR*) (Fielding et al., 2020): Turbulent mixing between cold dense gas clumps and hot ambient gas generates rapidly cooling intermediate-temperature regions, where the competition between radiative energy loss and turbulent velocity fields nonlinearly regulates cold phase growth or dissolution.

- **Viscoelastic fluids** (*VF*) (Beneitez et al., 2024): viscoelastic FENE-P fluid flow in wall-bounded geometries, resolving coupled Navier-Stokes and nonlinear conformation tensor dynamics to study multiscale elasto-inertial phenomena.

- **Rayleigh–Bénard convection** (*RB*) at five Rayleigh numbers $Ra \in \{10^6, 10^7, 10^8, 10^9, 10^{10}\}$(Burns et al., 2020): A buoyancy-driven turbulent flow arising from thermally induced density gradients in fluid layers bounded by contrasting thermal boundary conditions, exhibits nonlinear multiscale transport phenomena critical to geophysical, astrophysical, and engineered systems.

The full details on the underlying equations, dataset generation and problem setup for each case are provided in Appendix B.6.

Training and evaluation. All models receive input data $\mathbf{u}_{\text{in}} \in \mathbb{R}^{T\times H\times W\times D}$ with $T = 4$ history time steps, and predict sequences $\mathbf{u}_{\text{pred}} \in \mathbb{R}^{T'\times H\times W\times D}$ for the next T' time steps. During training, we use single-step prediction, comparing $\mathbf{u}_{\text{pred}}$ with the ground truth $\mathbf{u}_{\text{true}}$ by minimizing MSE loss across predicted frames. For COAST, the additional $\mathcal{L}_{dt}$ for regularizing dt is also included. For evaluation, following standard practice (Li et al., 2021), we perform autoregressive rollouts with $T' = 8$ and report VRMSE (Variance Scaled Root Mean Squared Error, recommended in (Ohana et al., 2024)) between $\mathbf{u}_{\text{pred}}$ and $\mathbf{u}_{\text{true}}$. Full details of the training and evaluation procedures are provided in B.1.

Table 1. VRMSE (lower is better) of 8-step rollouts on four benchmark PDEs (*AM*, *TR*, *VF*, and *RB* with various Rayleigh numbers Ra). The best result in each column is bolded, and the second-best result is underlined. COAST ranks first in nearly every setting, placing second in only one column.

MODEL	AM	TR	VF	RB				
				RA=1E6	RA=1E7	RA=1E8	RA=1E9	RA=1E10
FNO	1.663	0.765	0.711	0.628	0.664	0.674	0.706	0.710
DILATED RESNET	0.848	0.514	0.636	0.263	0.408	0.360	0.496	0.570
CNEXTU-NET	0.573	0.614	0.561	<u>0.205</u>	<u>0.316</u>	<u>0.361</u>	**0.379**	<u>0.485</u>
AVIT	0.903	<u>0.513</u>	<u>0.526</u>	0.348	0.549	0.516	0.556	0.604
DPOT	<u>0.521</u>	0.538	0.655	0.213	0.351	0.393	0.463	0.513
COAST (OURS)	**0.376**	**0.441**	**0.334**	**0.117**	**0.228**	**0.278**	<u>0.404</u>	**0.386**

Baselines. We benchmark COAST against leading operator learning frameworks and vision-based models adapted for operator learning that achieve strong performance, including Fourier Neural Operator (FNO) (Li et al., 2021), Dilated ResNet (Stachenfeld et al., 2022), ConvNeXt U-Net (CNextU-Net) (Liu et al., 2022), Attention Vision Transformer (AViT) (Du et al., 2024), and Denoising Pre-training Operator Transformer (DPOT) (Hao et al., 2024b). We place particular emphasis on comparing with DPOT (Hao et al., 2024b), which has been shown to outperform prior baselines across a wide range of PDE benchmarks. All baseline implementations follow the configurations recommended in their respective papers. Implementation details are provided in Appendix B.4.

4.1. Rollout accuracy

Table 1 presents a comprehensive comparison of our COAST models against competitive baselines. Our proposed method achieves the lowest VRMSE on nearly all benchmarks. The best results are shown in bold and the next best results are underlined. Additional visualizations of our models are shown in Figure 2 and Appendix B.7.

4.2. Error accumulation

Error accumulation is a central challenge in operator learning when performing autoregressive rollout. The models iteratively feed their predictions back as input for future steps. Consequently, errors compound over time—early inaccuracies propagate and amplify, degrading long-horizon forecasts. Therefore, understanding this error propagation mechanism is crucial for evaluating model capabilities and explains why time-adaptive approaches outperform fixed-step methods.

To quantify error accumulation behavior, we compare the temporal error trajectories of models on each benchmark.

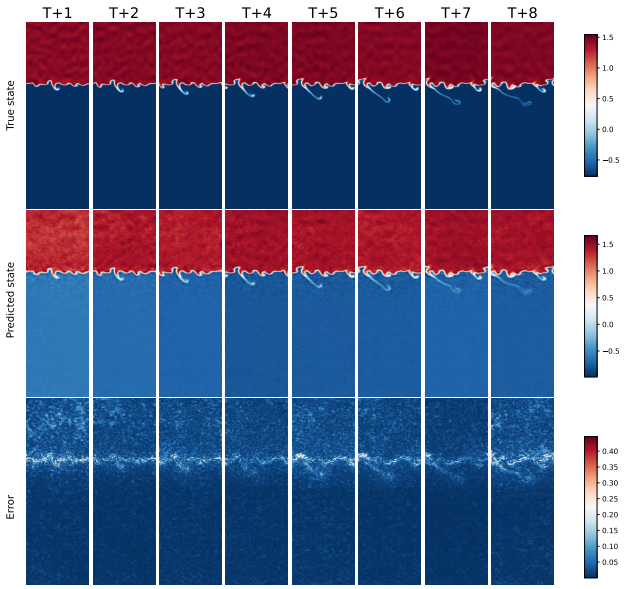

Figure 2. Turbulent radiative layer benchmark. Representative COAST rollout prediction of the $density$ field, and point-wise error against the ground truth.

Specifically, we compute the VRMSE at each rollout step ($T' = 8$). The averaged trajectories, presented in Figure 3, visualise the temporal evolution of error.COAST not only achieves the lowest overall error across all benchmarks but also exhibits the smallest cumulative growth in error. Although COAST may incur slightly higher initial errors on some tasks, it quickly stabilizes and maintains the lowest cumulative error. These results demonstrate the superior stability of time-adaptivity in long-term predictions compared to fixed-step alternatives and other baseline methods.

4.3. Inference time

Inference speed is another key consideration in operator learning, especially for long rollouts. Our time-adaptive

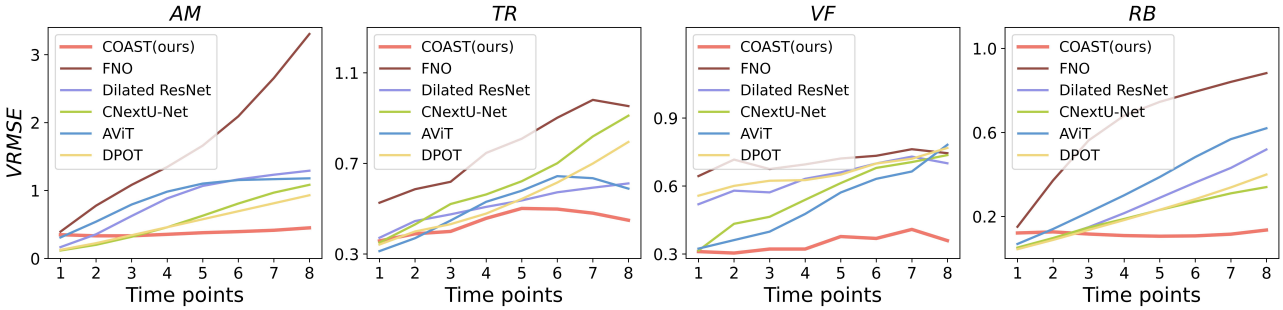

Figure 3. VRMSE (y-axis) at each of 8 rollout time points (x-axis) for four PDE benchmarks (*AM*, *TR*, *VF*, and *RB*). COAST (red) shows the lowest average error across all time steps and the minimum cumulative error compared to the other baseline methods.

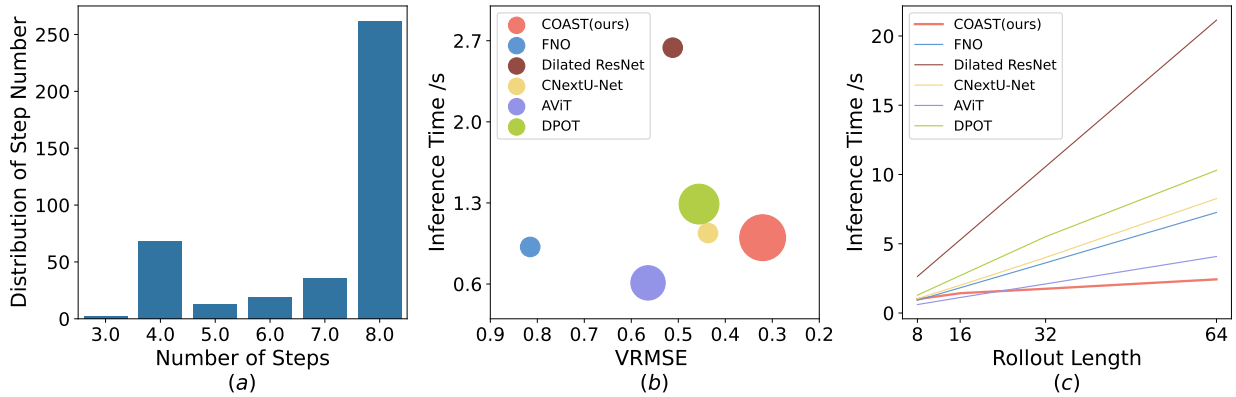

Figure 4. (a) Distribution of the number of time steps taken by COAST to roll out over all benchmarks. (b) Average rollout inference time of COAST and other baselines for the next 8 moments versus their VRMSE; bubble size indicates the number of parameters. (c) Average inference time of COAST and other baselines at rollout lengths 8, 16, 32, and 64. Notably, in (b) COAST attains lower error (higher accuracy) yet remains as fast as some baselines with larger errors and similar parameter counts, while in (c) COAST's inference time grows only modestly across longer rollouts, unlike the steeper curves for other methods.

approach leverages the adaptive time step dt to reduce the number of prediction steps required during rollout when $\tilde{r}_t > 1$, potentially enhancing inference speed without sacrificing accuracy. The distribution of the number of steps used by COAST to perform rollout prediction on all benchmarks is shown in Figure 4 (a), where the rollout length $T' = 8$.

We compared the average rollout inference time of COAST and other baseline models over the 4 benchmarks when the rollout length is 8. The results are presented in Figure 4 (b). We find that COAST achieves the lowest VRMSE while maintaining inference speeds comparable to or faster than baselines.

A horizon of 8, however, underutilizes COAST's adaptive advantage. Because our interpolation scheme often produces step sizes that exceed the interpolation window, the fraction of these "over-reaching" steps is higher in shorter rollouts. Longer horizons, in contrast, give COAST more opportunities to generate larger adaptive steps. To evaluate efficiency more thoroughly, we therefore extend the rollout length T' to 16, 32, and 64, recomputing average inference times; see Figure 4(c). COAST's runtime grows only mildly with the rollout length, whereas the baselines exhibit markedly steeper growth. Consequently, the speed gap between COAST and other models widens as the horizon increases.

4.4. Adaptive step-size analysis

As shown in Figure 4 (a), COAST uses a variable time step size, dt, which adapts dynamically across rollout steps. We analyze how these variations relate to underlying physical system properties and temporal evolution.

Adaptivity across system parameters. Figure 5 shows the distributions of time step sizes given by the model when rolling out inferences for subdatasets with different parameters. Here the step sizes are averaged over each rollout trajectory. It can be seen that under the same dataset, different subdatasets can be distinguished according to the

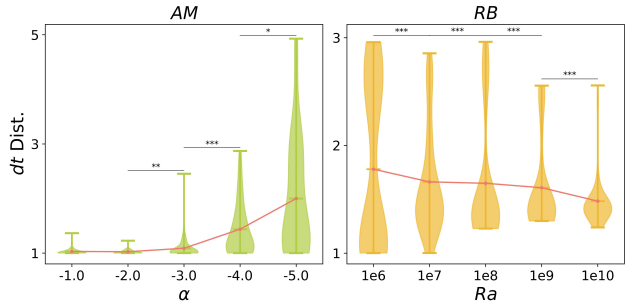

Figure 5. Violin plots of COAST's predicted time-step distributions across different parameters in two benchmarks: (a) Active Matter (*AM*) and (b) Rayleigh–Bénard convection (*RB*). The p-values from a Mann-Whitney U test (indicated as ***, **, *) confirm that step sizes differ significantly between adjacent parameter settings. Notably, as the parameters vary, COAST's predicted step sizes shift accordingly, illustrating that the model learns and adapts its temporal resolution to the underlying dynamical complexity of each system.

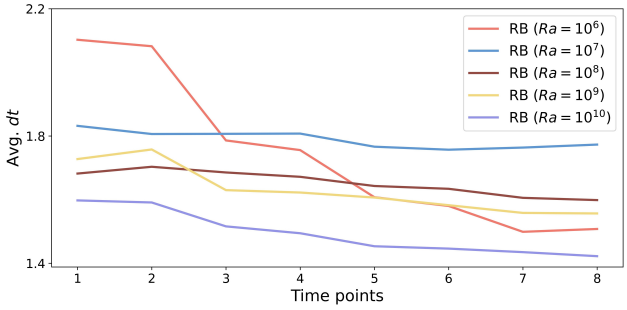

Figure 6. Average predicted time-step size from COAST across eight rollout points in Rayleigh–Bénard convection (*RB*) for different parameter values. Each line shows how COAST adaptively adjusts its temporal resolution over the course of the simulation. As the predictive horizon extends and the system's evolution becomes more complex, COAST gradually decreases the step size.

distribution of step sizes predicted by COAST.

We begin by examining how dt varies across different subsets of the same dataset, each defined by different physical parameters. These PDE parameters are significantly correlated with the complexity of the system. For example, in Active Matter (*AM*) benchmark, larger $|\alpha|$ correspond to simpler dynamics. In contrast, in Rayleigh–Bénard Convection (*RB*) benchmark, higher Rayleigh numbers (Ra) indicate more complex flows. Details on parameter impacts and complexity correlations are provided in Appendix Appendix B.6.1 and Appendix B.6.4.

As shown in Figure 5, the distribution of dt values differs significantly between parameter regimes across all four benchmarks. Statistical tests (Mann–Whitney U, Appendix B.3; annotated as ***, **, *) confirm that COAST meaningfully distinguishes between subsets based on system dynamics. As system complexity increases, COAST predicts smaller dt, indicating more conservative step sizes. Conversely, in simpler regimes, larger dt reflects more confident, longer-step predictions. These patterns suggest that COAST implicitly learns to adjust its temporal resolution based on the local complexity of the governing dynamics.

Temporal adaptivity within trajectories. We further analyze COAST's adaptive behavior across different temporal regions within the same dynamical system. We select the initial stages of Rayleigh–Bénard convection (*RB*) to perform rollouts of length $T' = 8$. In the *RB* system, dynamics evolve from initial stabilization to growing perturbations, with complexity gradually increasing (visualized in Appendix B.7).

Figure 6 shows the average dt at each time point during rollout. COAST gradually reduces its predicted dt as prediction proceeds. As system complexity increases, COAST becomes more conservative to maintain accuracy. This demonstrates that COAST not only predicts appropriate time steps for different system types but also dynamically adjusts step sizes during rollout. Such adaptivity enables it to balance accuracy and efficiency, particularly in long-horizon rollouts where system behavior evolves significantly.

4.5. Scalability

Here we verify the effect of the model's scalability on the Turbulent Radiative Layer (*TR*) benchmark.

We evaluated COAST at three parameter scales, and the rollout validation accuracies are shown in Figure 7(a). The corresponding hyper-parameter settings are summarised in Table 2. Prediction accuracy positively correlates with parameter count, confirming the strong scalability of our approach.

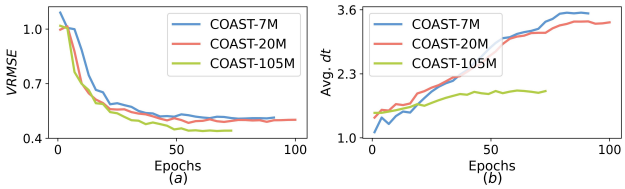

Figure 7. (a) Convergence of test rollout errors for COAST at three different model sizes (7 M, 20 M, and 105 M parameters). As the model size grows, the error decreases more rapidly and converges to a lower value. (b) Corresponding average rollout step sizes during training. The larger, more accurate model can afford to take fewer steps in each rollout—thus using larger time steps—while still maintaining lower error.

A complementary trend appears in Figure 7(b): models with fewer parameters adopt noticeably larger average time-step sizes. Our training objective consists of two terms—a spatial reconstruction loss, $\mathcal{L}_{\text{spatial}}$, and a step-size regularisation loss, $\mathcal{L}_{dt}$, which penalises overly small steps. When a com-

Table 2. Details of COAST model variants

MODEL	EMBEDDING DIM	BLOCKS	HEADS	# PARAMS
COAST-S	256	8	6	7.3M
COAST-M	384	12	8	20M
COAST-L	768	12	12	105M

pact model can no longer meaningfully reduce $\mathcal{L}_{\text{spatial}}$, it can still decrease the total loss by enlarging the step size, thereby lowering $\mathcal{L}_{dt}$. This compensatory behaviour explains the inverse correlation between parameter count and the chosen time step.

4.6. Ablation Studies

We perform ablation studies on key structures and hyper-parameters on benchmarks. Results are summarized in Figure 8.

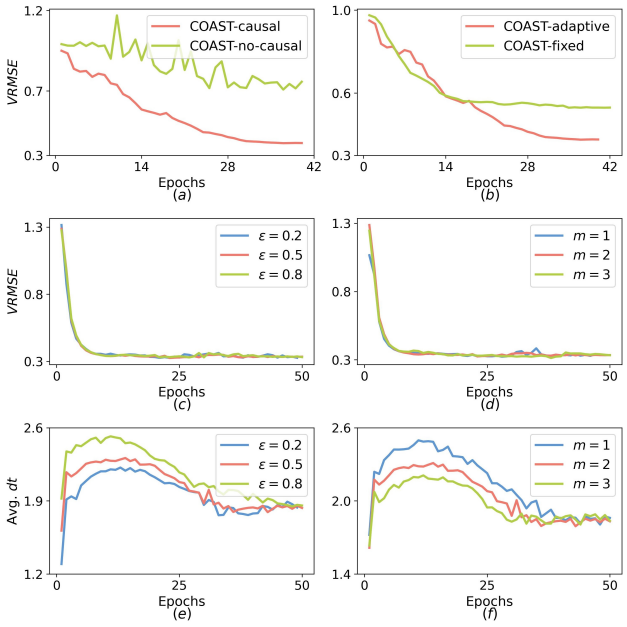

Figure 8. Ablation studies for COAST. Convergence of validation errors for: (a) two types of attention (with/without causal attention mask); (b) two step-types (adaptive/fixed framework); (c) different ε in $\mathcal{L}_{dt}$; (d) different power m in $\mathcal{L}_{dt}$. Convergence of dt in validation for: (e) different ε in $\mathcal{L}_{dt}$; (f) different power m in $\mathcal{L}_{dt}$. Results obtained using COAST with $\varepsilon = 0.5$ and $m = 2$, varying each hyper-parameter of interest while keeping others fixed. Figures (a-b) are obtained on the active matter (*AM*) benchmark and Figures (c-f) are obtained on the viscoelastic instability fluid (*VF*) benchmark.

We begin by evaluating the effect of type of attention mask on model performance. Causal attention is key for the causal inference. If we remove the causal mask, the model will see information from all time steps including the "future" within the same layer, undermining the consistency between the direction of attention and the direction of time evolution. Figure 8(a) shows bad performance without the causal attention mask. Then we show that COAST with fixed time step achieves higher rollout loss though $dt = 1$ as in Figure 8(b). This is because each prediction step accumulates errors. Over an extended rollout, errors compound more severely than adaptive COAST.

We investigate the impact of the regularization term $\mathcal{L}_{dt}$, introduced in Section 2.1 to penalize small values of dt. Specifically, we ablate its two hyper-parameters: the threshold ε and the exponent m. Figure 8(b–e) show that varying ε and m has little effect on predictive accuracy or convergence of dt, suggesting that COAST is robust to these choices. Overall, these studies validate our default hyper-parameter settings and highlight practical design considerations for future time-adaptive operator learning models.

5. Discussion

Summary. This work introduces COAST, a new neural operator architecture that utilizes causal attention transformer at its core to address the challenges of learning complex physical systems. COAST combines the strengths of causal attention transformer and adaptive solution methods to achieve state-of-the-art accuracy and minimal error accumulation behavior on challenging benchmarks in energy transformation, fluid dynamics, and thermodynamic processes. Our approach demonstrates the potential of employing advanced causal attention transformer to develop more flexible and accurate machine learning models for the physical sciences. Key innovations of our work include: (a) an efficient solver on continuous time for autonomous decision prediction step sizes, (b) a rational method for evaluating time-adaptive solvers, and (c) an exploration of the deep understanding of PDE systems embodied in causal attention transformer for operator learning.

Our empirical results in various PDE benchmark tests show that COAST's time-adaptive approach endows it with higher solving efficiency in prediction over longer sequences. This time-adaptive behavior helps to build more general solvers. In addition, the step size distributions predicted by COAST when confronted with different systems and states can also be used to explore some of the more intrinsic properties of dynamical systems. The broader impact of this work based on COAST is that it has the potential to accelerate scientific discovery by more efficiently and accurately modeling complex physical systems over longer time horizons, with applications ranging from energy transformation modeling to engineering design.

Limitations & Future Work. While COAST advances neural operator capabilities, several limitations need atten-

tion. First, current experiments focus on systems with regular geometries and uniform grids, leaving performance on complex geometries (e.g., fractured porous media, turbulent multiphase flows) unexplored. Second, while the architecture shows empirical stability in moderate rollout lengths, its error propagation behavior under extended autoregressive prediction horizons remains unexamined. Third, the current implementation operates as a specialized solver rather than a generalizable framework, limiting direct applicability to PDE systems requiring coupled multi-physics modeling.

Future research should prioritize extending COAST's framework toward multimodal PDE foundation models capable of unifying diverse physical systems under a single architecture. This could involve integrating physical constraints via hybrid symbolic-neural frameworks that enforce various physical laws. Another particularly promising direction lies in coupling COAST with LLMs—such integration could enable cross-modal reasoning where textual system descriptions guide dynamics prediction, or conversely, where learned physical representations enhance LLMs' capacity for quantitative scientific reasoning. We believe that addressing these challenges will enable the synergistic integration of physics-informed machine learning and foundation models, paving the way for next-generation computational tools across scientific domains.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Alkin, B., Fürst, A., Schmid, S., Gruber, L., Holzleitner, M., and Brandstetter, J. Universal physics transformers: A framework for efficiently scaling neural operators, 2025. URL https://arxiv.org/abs/2402.12365.

Azizzadenesheli, K., Kovachki, N., Li, Z., Liu-Schiaffini, M., Kossaifi, J., and Anandkumar, A. Neural operators for accelerating scientific simulations and design. *Nature Reviews Physics*, pp. 1–9, 2024.

Beneitez, M., Page, J., Dubief, Y., and Kerswell, R. R. Multistability of elasto-inertial two-dimensional channel flow. *Journal of Fluid Mechanics*, 981:A30, 2024.

Brandstetter, J., Worrall, D., and Welling, M. Message passing neural pde solvers, 2023. URL https://arxiv.org/abs/2202.03376.

Brunton, S. L., Noack, B. R., and Koumoutsakos, P. Machine learning for fluid mechanics. *Annual Review of Fluid Mechanics*, 52(1):477–508, January 2020. ISSN 1545-4479. doi: 10.1146/annurev-fluid-010719-060214. URL http://dx.doi.org/10.1146/annurev-fluid-010719-060214.

Burns, K. J., Vasil, G. M., Oishi, J. S., Lecoanet, D., and Brown, B. P. Dedalus: A flexible framework for numerical simulations with spectral methods. *Physical Review Research*, 2(2):023068, 2020.

Cao, S. Choose a transformer: Fourier or galerkin, 2021. URL https://arxiv.org/abs/2105.14995.

Du, S., Bayasi, N., Hamarneh, G., and Garbi, R. Avit: Adapting vision transformers for small skin lesion segmentation datasets, 2024. URL https://arxiv.org/abs/2307.13897.

Evans, L. C. *Partial Differential Equations*, volume 19 of *Graduate Studies in Mathematics*. American Mathematical Society, 1998.

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

Guibas, J., Mardani, M., Li, Z., Tao, A., Anandkumar, A., and Catanzaro, B. Adaptive fourier neural operators: Efficient token mixers for transformers, 2022. URL https://arxiv.org/abs/2111.13587.

Hairer, E., Norsett, S., and Wanner, G. *Solving Ordinary Differential Equations I: Nonstiff Problems*, volume 8. 01 1993. ISBN 978-3-540-56670-0. doi: 10.1007/978-3-540-78862-1.

Hao, Z., Su, C., Liu, S., Berner, J., Ying, C., Su, H., Anandkumar, A., Song, J., and Zhu, J. Dpot: Auto-regressive denoising operator transformer for large-scale pde pretraining. *arXiv preprint arXiv:2403.03542*, 2024a.

Hao, Z., Su, C., Liu, S., Berner, J., Ying, C., Su, H., Anandkumar, A., Song, J., and Zhu, J. Dpot: Auto-regressive denoising operator transformer for large-scale pde pretraining, 2024b. URL https://arxiv.org/abs/2403.03542.

Hendrycks, D. and Gimpel, K. Gaussian error linear units (gelus), 2023. URL https://arxiv.org/abs/1606.08415.

Herde, M., Raonić, B., Rohner, T., Käppeli, R., Molinaro, R., de Bézenac, E., and Mishra, S. Poseidon: Efficient foundation models for pdes, 2024. URL https://arxiv.org/abs/2405.19101.

Kovachki, N., Li, Z., Liu, B., Azizzadenesheli, K., Bhattacharya, K., Stuart, A., and Anandkumar, A. Neural operator: Learning maps between function spaces, 2024. URL https://arxiv.org/abs/2108.08481.

Krishnapriyan, A. S., Gholami, A., Zhe, S., Kirby, R. M., and Mahoney, M. W. Characterizing possible failure modes in physics-informed neural networks, 2021. URL https://arxiv.org/abs/2109.01050.

LeVeque, R. J. *Finite Difference Methods for Ordinary and Partial Differential Equations: Steady-State and Time-Dependent Problems*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2007.

Li, Z., Kovachki, N., Azizzadenesheli, K., Liu, B., Bhattacharya, K., Stuart, A., and Anandkumar, A. Fourier neural operator for parametric partial differential equations. *arXiv preprint arXiv:2010.08895*, 2020a.

Li, Z., Kovachki, N., Azizzadenesheli, K., Liu, B., Bhattacharya, K., Stuart, A., and Anandkumar, A. Neural operator: Graph kernel network for partial differential equations, 2020b. URL https://arxiv.org/abs/2003.03485.

Li, Z., Kovachki, N., Azizzadenesheli, K., Liu, B., Bhattacharya, K., Stuart, A., and Anandkumar, A. Fourier neural operator for parametric partial differential equations, 2021. URL https://arxiv.org/abs/2010.08895.

Li, Z., Meidani, K., and Farimani, A. B. Transformer for partial differential equations' operator learning. *arXiv preprint arXiv:2205.13671*, 2022.

Li, Z., Kovachki, N. B., Choy, C., Li, B., Kossaifi, J., Otta, S. P., Nabian, M. A., Stadler, M., Hundt, C., Azizzadenesheli, K., and Anandkumar, A. Geometry-informed neural operator for large-scale 3d pdes, 2023a. URL https://arxiv.org/abs/2309.00583.

Li, Z., Meidani, K., and Farimani, A. B. Transformer for partial differential equations' operator learning, 2023b. URL https://arxiv.org/abs/2205.13671.

Li, Z., Zheng, H., Kovachki, N., Jin, D., Chen, H., Liu, B., Azizzadenesheli, K., and Anandkumar, A. Physics-informed neural operator for learning partial differential equations, 2023c. URL https://arxiv.org/abs/2111.03794.

Liu, Z., Mao, H., Wu, C.-Y., Feichtenhofer, C., Darrell, T., and Xie, S. A convnet for the 2020s, 2022. URL https://arxiv.org/abs/2201.03545.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization, 2019. URL https://arxiv.org/abs/1711.05101.

Lu, L., Jin, P., Pang, G., Zhang, Z., and Karniadakis, G. E. Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. *Nature machine intelligence*, 3(3):218–229, 2021.

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

McCabe, M., Blancard, B. R.-S., Parker, L. H., Ohana, R., Cranmer, M., Bietti, A., Eickenberg, M., Golkar, S., Krawezik, G., Lanusse, F., Pettee, M., Tesileanu, T., Cho, K., and Ho, S. Multiple physics pretraining for physical surrogate models, 2024. URL https://arxiv.org/abs/2310.02994.

Ohana, R., McCabe, M., Meyer, L. T., Morel, R., Agocs, F. J., Beneitez, M., Berger, M., Burkhart, B., Dalziel, S. B., Fielding, D. B., Fortunato, D., Goldberg, J. A., Hirashima, K., Jiang, Y.-F., Kerswell, R., Maddu, S., Miller, J. M., Mukhopadhyay, P., Nixon, S. S., Shen, J., Watteaux, R., Blancard, B. R.-S., Rozet, F., Parker, L. H., Cranmer, M., and Ho, S. 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. URL https://openreview.net/forum?id=00Sx577BT3.

Perez, E., Strub, F., de Vries, H., Dumoulin, V., and Courville, A. Film: Visual reasoning with a general conditioning layer, 2017. URL https://arxiv.org/abs/1709.07871.

Rahman, M. A., Ross, Z. E., and Azizzadenesheli, K. U-no: U-shaped neural operators, 2023. URL https://arxiv.org/abs/2204.11127.

Rannacher, R. Adaptive finite element methods for partial differential equations. *arXiv preprint math/0305006*, 2003.

Roy, A., DuPlissis, A., Mishra, B., and Ben-Yakar, A. Deep operator networks for bioheat transfer problems with parameterized laser source functions. *International Journal of Heat and Mass Transfer*, 228:125659, 2024.

Stachenfeld, K., Fielding, D. B., Kochkov, D., Cranmer, M., Pfaff, T., Godwin, J., Cui, C., Ho, S., Battaglia, P., and Sanchez-Gonzalez, A. Learned coarse models for efficient turbulence simulation, 2022. URL https://arxiv.org/abs/2112.15275.

Wang, S., Wang, H., and Perdikaris, P. Learning the solution operator of parametric partial differential equations with physics-informed DeepONets. *Science Advances*, 7(40): eabi8605, 2021. doi: 10.1126/sciadv.abi8605.

Wang, S., Liu, T.-R., Sankaran, S., and Perdikaris, P. Micrometer: Micromechanics transformer for predicting mechanical responses of heterogeneous materials. *arXiv preprint arXiv:2410.05281*, 2024a.

Wang, S., Seidman, J. H., Sankaran, S., Wang, H., Pappas, G. J., and Perdikaris, P. Bridging operator learning and conditioned neural fields: A unifying perspective. *arXiv preprint arXiv:2405.13998*, 2024b.

Wang, S., Seidman, J. H., Sankaran, S., Wang, H., Pappas, G. J., and Perdikaris, P. Cvit: Continuous vision transformer for operator learning, 2025. URL https://arxiv.org/abs/2405.13998.

Wu, H., Luo, H., Wang, H., Wang, J., and Long, M. Transolver: A fast transformer solver for pdes on general geometries. *arXiv preprint arXiv:2402.02366*, 2024a.

Wu, H., Luo, H., Wang, H., Wang, J., and Long, M. Transolver: A fast transformer solver for pdes on general geometries, 2024b. URL https://arxiv.org/abs/2402.02366.

Ye, Z., Huang, X., Chen, L., Liu, H., Wang, Z., and Dong, B. Pdeformer: Towards a foundation model for one-dimensional partial differential equations, 2025. URL https://arxiv.org/abs/2402.12652.

A. Nomenclature

Table 3 summarizes the main symbols and notation used in this work.

Table 3. Summary of the main symbols and notations used in this work.

Notation	Description
Operator Learning	
$\mathbf{u}$	Input function (continuous field)
$\mathbf{u}_{\text{in}}$	Spatio-temporal input tensor
$\mathbf{u}_{\text{pred}}$	Predicted output tensor
$\mathbf{u}_{\text{true}}$	Ground-truth output tensor
$\mathbf{S}_i$	i-th spatial frames of input tensor
$\mathbf{T}_{\text{seq}}$	Sequence of relative timestamps of input tensor
T	Input sequence length / Number of previous time-steps
T'	Output sequence length / Number of rollout length
$H \times W$	Resolution of spatial discretization
COAST	
$\mathbf{u}_p$	Patchified inputs
$\mathbf{u}_s$	Patchified inputs after spatial positional embedding
$\mathbf{u}_{st}$	Patchified inputs after spatial-temporal embedding
$\mathbf{z}$	Representation of predicted result at time point dt
$\mathbf{z}_k$	Predicted representation of the k-th spatial patch
dt	Predicted local time step / Adaptive time step size
$\mathbf{u}(t)$	Ground truth at time point t
$\tilde{\mathbf{u}}(t)$	Predicted result for time point t in $[0, dt]$
$\mathbf{PE}_s$	Spatial positional embedding
$\text{FiLM}(\cdot)$	FiLM layer
$\text{MLP}(\cdot)$	Multi-Layer Perceptron network
$\text{TransCNN}(\cdot)$	Transposed Convolutional Neural Network
$\mathcal{L}_{dt}(\cdot)$	Regularization loss term penalizing overly small dt values
D	The number of physical channels
C	Embedding dimension of Transformer Processor
P	Hyper-parameter: Patch size of Spatio-Temporal Encoder
ϵ, m	Hyper-parameters in $\mathcal{L}_{dt}(\cdot)$

B. Experimental details

B.1. Training and evaluation

Training recipe. We use a unified training recipe for all COAST experiments. We employ AdamW optimizer (Loshchilov & Hutter, 2019) with a weight decay 10^{-5}. Our learning rate schedule includes an initial linear warm up phase of 5 epochs, starting from zero and gradually increasing to 5×10^{-5}, followed by an exponential decay at a rate of 0.9 for every $5,000$ steps. Each model in the main experiments was trained for $75,000$ optimization steps. All experiments are conducted on a single NVIDIA A100 GPU, taking roughly $9 \sim 12$ hours, depending on the model architecture and the particular benchmark.

The spatial loss function is an average Mean Squared Error (MSE) between the model predictions and the corresponding targets at the predicted time steps, average over all coordinates:

$$\text{MSE} = \frac{1}{B}\frac{1}{T}\frac{1}{D}\sum_{i=1}^{B}\sum_{t=1}^{T}\sum_{k=1}^{D}\|\hat{s}_{t,i}^{(k)} - s_{t,i}^{(k)}\|_2,$$

where $s_{t,i}^{k}$ denotes the k-th channel of the t-th time point in the output sequence of the i-th sample in the training dataset, evaluated at all coordinates, and $\hat{s}_{t,i}^{k}$ denotes the corresponding model prediction. All models are trained for the same number of epochs with a equal batch size $B = 4$ on each benchmark.

For COAST, we add a term to the loss function to penalize comparatively small dt in the model outputs. Here we choose to use a piecewise power-exponential function of dt as $\mathcal{L}_{dt}$, where $0 < \varepsilon < 1$ and n is the power that no less than 1:

$$\mathcal{L}_{dt} = \begin{cases} (1 + \varepsilon - dt)^m, & dt \leq 1 + \varepsilon, \\ 0, & dt > 1 + \varepsilon, \end{cases}$$

Based on our ablation study, changing the values of ϵ and n does not affect the training results. Without loss of generality, we take ε to be 0.5 and m to be 2.

Note that during training, we will have the model output on several sequences of dynamics, thus collecting a batch of dt, which is then averaged for $\mathcal{L}_{dt}$'s computation and backpropagation. This method of computing the $\mathcal{L}_{dt}$ on the average of the dt enables the diversity of dt outputs.

Evaluation. After training, we obtain the predicted trajectory by performing an auto-regressive rollout of T' time points on the test dataset. All evaluation workloads (e.g., results of *Inference time* reported in Section 4.3) are executed on the same NVIDIA A100 GPU used for training.

We evaluate model accuracy using VRMSE, following the recommendation in (Ohana et al., 2024):

$$\text{VRMSE} = \frac{1}{T}\frac{1}{D}\sum_{t=1}^{T}\sum_{k=1}^{D}\sqrt{\frac{\|\hat{s}_t^{(k)} - s_t^{(k)}\|_2}{\|s_t^{(k)} - \bar{s}_t^{(k)}\|_2 + \epsilon}}.$$

Note that, since $\text{VRMSE}(s, \bar{s}) \approx 1$, having $\text{VRMSE} > 1$ indicates worse results than an accurate estimation of the spatial mean $\bar{s}$.

B.2. FiLM Layer

FiLM Layer (Feature-wise Linear Modulation layer) (Perez et al., 2017) takes as input a scalar tensor $t \in \mathbb{R}$ and a feature vector $z \in \mathbb{R}^C$. It then computes two learnable transformations, $\gamma(t)$ and $\beta(t)$, both of which are derived from t. The feature vector z is modulated as follows:

$$\hat{z} = \gamma(t) \odot z + \beta(t).$$

Here, $\odot$ represents element-wise multiplication.

In our Section 2.1, when applying the FiLM Layer to $(\mathbf{T}_{seq}, \mathbf{u}_s)$, $\mathbf{T}_{seq}$ is a sequence of T scalar values, and $\mathbf{u}_s$ consists of T corresponding feature tensors. Each pair $(\mathbf{T}_{seq}[i], \mathbf{u}_s[i])$ undergoes the FiLM operation in the same manner as (t, z) above.

While in our Section 2.1, when applying the FiLM Layer to $(dt, \mathbf{z})$, dt is the scalar tensor, and $\mathbf{z} \in \mathbb{R}^{(\frac{H}{P}\cdot\frac{W}{P})\times C}$ is composed of $(\frac{H}{P} \cdot \frac{W}{P})$ subtensors, each of size C. We perform the FiLM operation on each of these subtensors with the same dt.

This design allows a single FiLM module to handle differently shaped inputs (i.e., $\mathbf{T}_{seq}$ and $\mathbf{u}_s$) by applying the same feature-wise modulation process to each pair of scalar and feature tensor.

B.3. Mann-Whitney U test

The Mann-Whitney U test is a non-parametric procedure for assessing whether two independent samples originate from the same continuous distribution without presuming normality. Given two samples of sizes n_1 and n_2, let R_1 and R_2 denote the sums of the ranks assigned after pooling and ordering all observations. The test statistics are:

$$U_1 = n_1 n_2 + \frac{n_1(n_1+1)}{2} - R_1, \quad U_2 = n_1 n_2 + \frac{n_2(n_2+1)}{2} - R_2,$$

And one sets $U = \min(U_1, U_2)$. Under the null hypothesis H_0 (identical distributions), the sampling distribution of U is known exactly for small samples; for $n_1, n_2 \geq 20$ it is well-approximated by a normal deviate:

$$Z = \frac{U - \mu_U}{\sigma_U}, \quad \mu_U = \frac{n_1 n_2}{2}, \quad \sigma_U = \sqrt{\frac{n_1 n_2 (n_1 + n_2 + 1)}{12}},$$

From which a two-sided p-value is obtained. Because it relies only on ranks, the Mann-Whitney U test is robust to outliers and suitable when sample distributions are skewed or ordinal in nature, making it ideal for the comparative analyses reported in this study.

We therefore apply the Mann-Whitney U test in the *Adaptivity Across System Parameters* experiment (Section 4.4), where ample independent trajectories satisfy the test's asymptotic requirements. Conversely, we omit it in the *Temporal Adaptivity Within Trajectories* experiment (Section 4.4), whose limited data of initial stages would yield an under-powered inference.

B.4. Model details

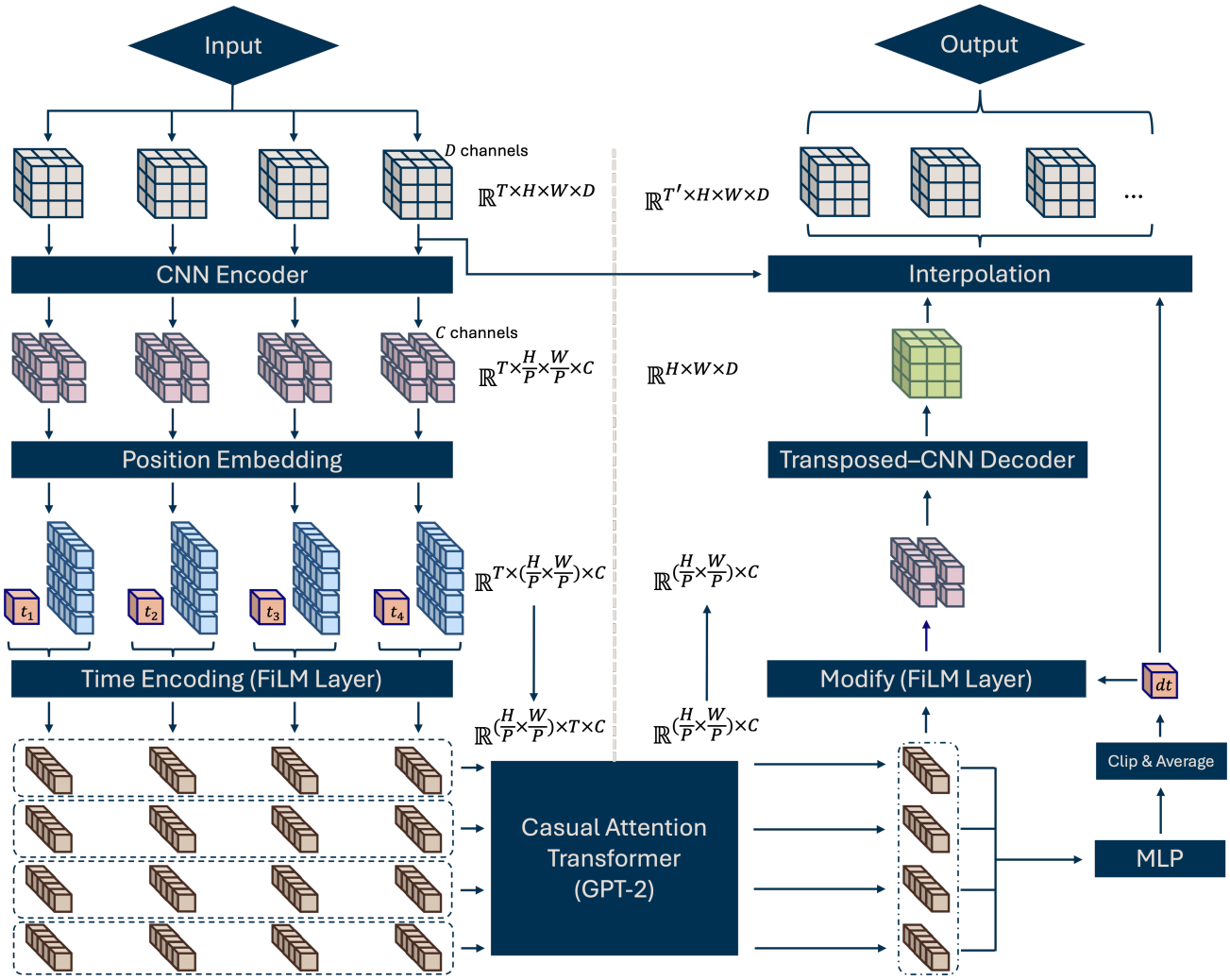

Figure 9. Full architecture of COAST.

COAST. We use a patch size of 32×32 for embedding data. The encoder consists of 3-layer convolutional neural network (CNN) layers with padding and overlapping kernels and so do the decoders. We use a 2-layer MLP to interpret dt from output tokens by Transformer Processor. We use two FiLM Layers in COAST, one is for embedding time series and another is for output tokens modification using output dt. We use Gaussian Error Linear Units (GELU, (Hendrycks & Gimpel, 2023)) as activation function in the structures above. The hyper-parameters ε and m are set to 0.5 and 2 respectively.

FNO. We implement the two–dimensional Fourier Neural Operator (FNO) following (Li et al., 2021) The network consists of an initial lifting layer, 4 spectral–convolution blocks with 128 feature channels and 16 retained Fourier modes in each spatial dimension, followed by point-wise activations and a linear projection head.

Dilated ResNet. We adopt the Dilated ResNet which has a hidden dimension of 96 and 32 residual blocks.

CNextUNet. Our ConvNext-U-Net uses a ConvNext encoder (8 ConvNext blocks per stage, 42 initial filters) as proposed in (Liu et al., 2022). The decoder mirrors the encoder with transposed-convolution up-sampling and skip concatenations; GELU activations and layer scaling are retained from ConvNext.

AViT. We adopt the base Attention Vision Transformer variant from (Du et al., 2024): a vision-transformer encoder of hidden size 768, 6 multi-head Fourier-attention layers, 12 attention heads, and 4 processor blocks, preceded by a 32×32 patchifier and learned positional embeddings.

DPOT. The Denoising Pre-training Operator Transformer (DPOT) is configured according to Hao *et al.* (Hao et al., 2024a). We employ a DPOT with an AFNO as mixing type and Exp-MLP time aggregation. The patch size is 4 and blocks number is 16. For challenging comparison, We set its embedding dimension to 1024, depth to 16 and number of attention head to 16. The out layer of the model has a dimension of 32. The active function in it is also GELU.

Because the official DPOT checkpoints were pre-trained on a 100 k-trajectory multi-PDE corpus rather than on The-Well(Ohana et al., 2024), we re-initialise and train DPOT from scratch to ensure a fair comparison while respecting our compute budget. We omit larger DPOT version (> 500 M parameters) to keep training costs comparable across baselines.

B.5. Dataset and problem setup

We make use of the datasets released by The Well (Ohana et al., 2024). This dataset consists of 15T data of discretized initial conditions on diverse types and parameter-sets.

We compare COAST's performance against several strong baseline models as above. We use different training, validation and testing data split(Ohana et al., 2024).

Problem setup. We modified the problem setup by The Well (Ohana et al., 2024). Our objective is to predict the future solution within 8 timepoints from the previous 4 timepoints.

For the complex systems in **The Well** dataset, this rollout length is not considered short, given that our input sequence length is 4. By comparison, in the DPOT (Hao et al., 2024a) and CViT (Wang et al., 2025) works, their input sequence length is 10, with rollout lengths ranging from 4 to 10 in all of their main experiments.

B.6. Benchmarks

Here we present a detailed description of the benchmarks, which may include the physical settings, governing equations, dataset specifics, and relevant parameters. The discussion also incorporates a complexity analysis of the benchmarks' specific parameters referenced in the main text.

B.6.1. Active Matter (AM)

This dataset comprises simulations of a continuum theory describing the dynamics of N rod-like active particles in a Stokes fluid within a two-dimensional domain of linear size L. The data include 81 time-steps of 256×256 resolution per trajectory, with fields such as concentration (scalar), velocity (vector), orientation tensor, and strain-rate tensor. Simulations explore parameter variations in alignment (ζ), dipole strength (α), and other coefficients, capturing phenomena like energy transfer

across scales, vorticity-orientation coupling, and the isotropic-to-nematic phase transition. Periodic boundary conditions and uniform Cartesian grids are employed, with data stored at 0.25-second intervals over a 20-second timespan. Refer to (Maddu et al., 2024) for details on problem formulation and detailed equations.

Note that α is the dimensionless active dipole strength. Based on the original paper(Maddu et al., 2024), the greater the absolute value of α, the faster the system approaches order/stability. It can be analogous to viscosity in the fluid problem to some extent. So smaller $|\alpha|$ means higher complexity.

B.6.2. Turbulent Radiative Layer (TR)

This dataset explores the dynamics of turbulent radiative layers in astrophysical systems, where hot and cold gases mix, leading to the formation of intermediate-temperature gas that rapidly cools. The simulations model the Kelvin-Helmholtz instability in a 2D domain, with cold, dense gas at the bottom and hot, dilute gas at the top. The data capture key phenomena such as mass flux from the hot to cold phase, turbulent velocities, and the distribution of mass across temperature bins. The dataset includes 101 timesteps of 384×128 resolution for 90 trajectories, varying the cooling time t_{cool} across nine values. Simulations were performed using Athena++ on a uniform Cartesian grid with periodic boundary conditions in the x-direction and zero-gradient in the y-direction. This dataset provides insights into the phase structure, energetics, and dynamics of multiphase gas in astrophysical environments, such as the interstellar and circumgalactic media.

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \vec{v}) = 0, \tag{1}$$

$$\frac{\partial \rho \vec{v}}{\partial t} + \nabla \cdot (\rho \vec{v}\vec{v} + P) = 0, \tag{2}$$

$$\frac{\partial E}{\partial t} + \nabla \cdot ((E + P)\vec{v}) = -\frac{E}{t_{\text{cool}}}, \tag{3}$$

$$E = P/(\gamma - 1)\ \gamma = 5/3, \tag{4}$$

where ρ is the density, $\vec{v}$ is the 2D velocity, P is the pressure, E is the total energy, and t_{cool} is the cooling time.

B.6.3. Viscoelastic Fluids (VF)

This dataset explores the multistability of viscoelastic fluids in a two-dimensional channel flow, capturing four distinct attractors: the laminar state (LAM), a steady arrowhead regime (SAR), Elasto-inertial turbulence (EIT), and a chaotic arrowhead regime (CAR). These states coexist for the same set of parameters, with their emergence dependent on initial conditions. The dataset includes snapshots of these attractors as well as edge states, which lie on the boundary between basins of attraction and provide insight into transitions between flow regimes. The data were generated using direct numerical simulations of the FENE-P model, solving for velocity, pressure, and the conformation tensor fields. Key phenomena include chaotic dynamics in EIT and CAR, as well as the multistability of the system. The dataset, comprising 260 trajectories with 512×512 resolution, is valuable for studying viscoelastic turbulence and evaluating simulators capable of capturing these complex flow behaviors. Simulations were performed using the Dedalus framework, with parameters set to $Re = 1000$, $Wi = 50$, $\beta = 0.9$, $\epsilon = 2 \times 10^{-6}$, and $L_{max} = 70$.

$$\begin{aligned} Re(\partial_t \mathbf{u}^* + (\mathbf{u}^* \cdot \nabla)\mathbf{u}^*) + \nabla p^* &= \beta \Delta \mathbf{u}^* + (1-\beta)\nabla \cdot \mathbf{T}(\mathbf{C}^*), \\ \partial_t \mathbf{C}^* + (\mathbf{u}^* \cdot \nabla)\mathbf{C}^* + \mathbf{T}(\mathbf{C}^*) &= \mathbf{C}^* \cdot \nabla \mathbf{u}^* + (\nabla \mathbf{u}^*)^T \cdot \mathbf{C}^* + \epsilon \Delta \mathbf{C}^*, \\ \nabla \mathbf{u}^* &= 0, \end{aligned}$$

$$\text{with} \quad \mathbf{T}(\mathbf{C}^*) = \frac{1}{\mathbf{Wi}}(f(\text{tr}(\mathbf{C}^*))\mathbf{C}^* - \mathbf{I}), \qquad \text{and} \quad f(s) := \left(1 - \frac{s-3}{L_{max}^2}\right)^{-1},$$

where $\mathbf{u}^* = (u^*, v^*)$ is the streamwise and wall-normal velocity components, p^* is the pressure, $\mathbf{C}^*$ is the positive definite conformation tensor which represents the ensemble average of the product of the end-to-end vector of the polymer molecules. In 2D, 4 components of the tensor are solved: $c^*_{xx}, c^*_{yy}, c^*_{zz}, c^*_{xy}$. $\mathbf{T}(\mathbf{C}^*)$ is the polymer stress tensor given by the FENE-P model.

B.6.4. RAYLEIGH-BÉNARD CONVECTION (RB)

This dataset comprises simulations of two-dimensional, horizontally periodic Rayleigh-Bénard convection, capturing the dynamics of fluid motion driven by thermal gradients. The system consists of a fluid layer heated from below and cooled from above, leading to the formation of convective cells and complex flow patterns. The dataset includes 200 timesteps of 512 $\times$ 128 resolution for 1,750 simulations, varying the Rayleigh number (10^6 to 10^{10}), Prandtl number (0.1 to 10), and initial buoyancy perturbations. Fields such as buoyancy (scalar), pressure (scalar), and velocity (vector) are provided, with periodic boundary conditions horizontally and Dirichlet conditions vertically. The data, generated using the Dedalus framework, offer insights into turbulent eddies, convection cells, and the sensitivity of flow structures to initial conditions. This dataset is valuable for studying thermal convection phenomena and validating numerical models in fluid dynamics.

The time domain problem is formulated as:

$$\begin{aligned} \frac{\partial b}{\partial t} - \kappa \Delta b &= -u \nabla b, \\ \frac{\partial u}{\partial t} - \nu \Delta u + \nabla p - b\vec{e}_z &= -u \nabla u, \end{aligned}$$

with boundary conditions:

$$\begin{aligned} b(z=0) = Lz \quad , \quad b(z=Lz) = 0, \\ u(z=0) = u(z=Lz) = 0. \end{aligned}$$

Note that the Rayleigh number (Ra) satisfies the relation: $(viscosity)\ \nu = (\frac{Ra}{Prandtl})^{-\frac{1}{2}}$. It means that the greater Ra means smaller viscosity, then the system will approach order/stability slower. So greater Ra means higher complexity.

B.7. Visualization

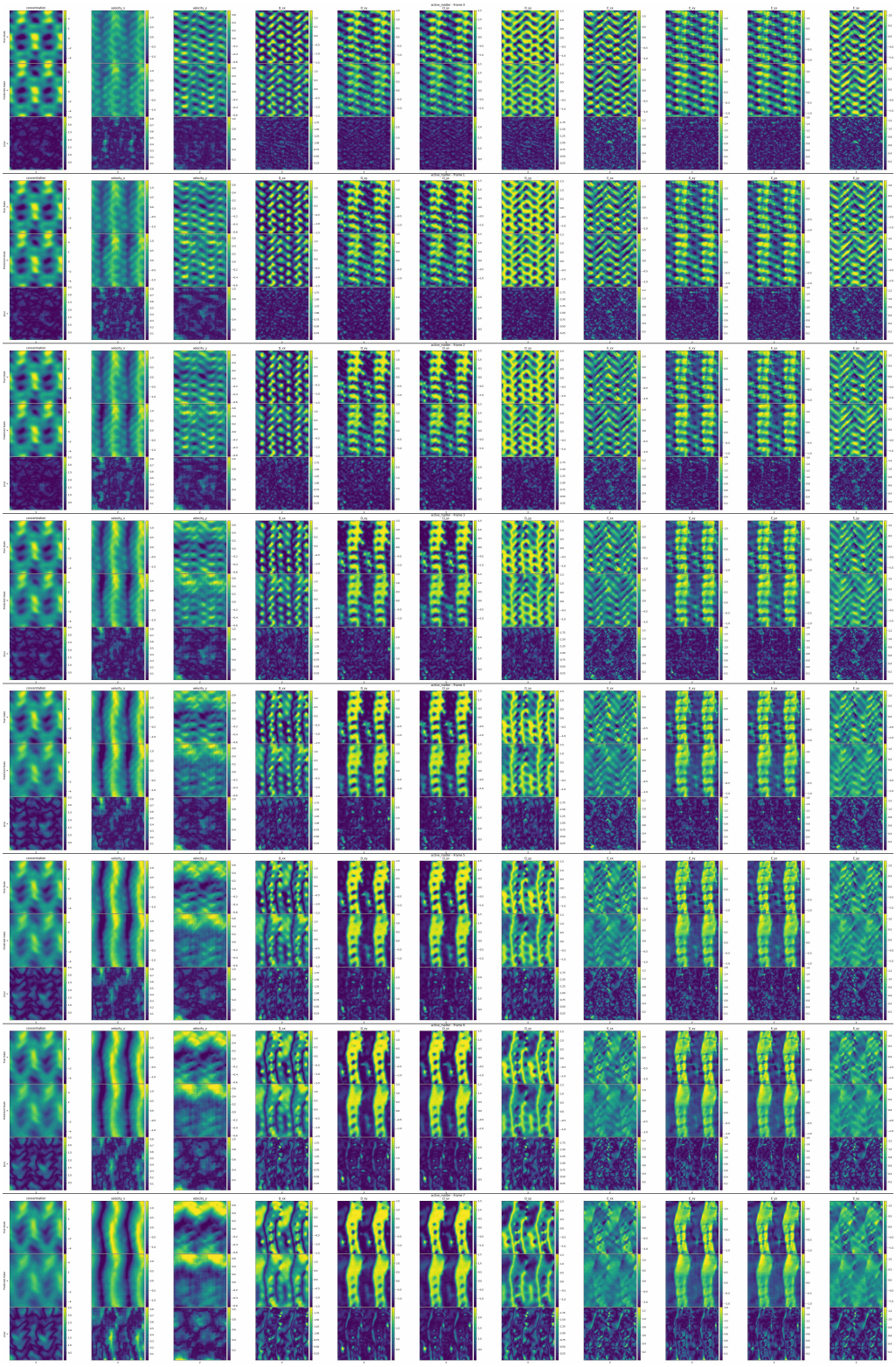

Figure 10. Active Matter (AM). Representative COAST rollout prediction of all the different fields in y direction, and point-wise error against the ground truth.

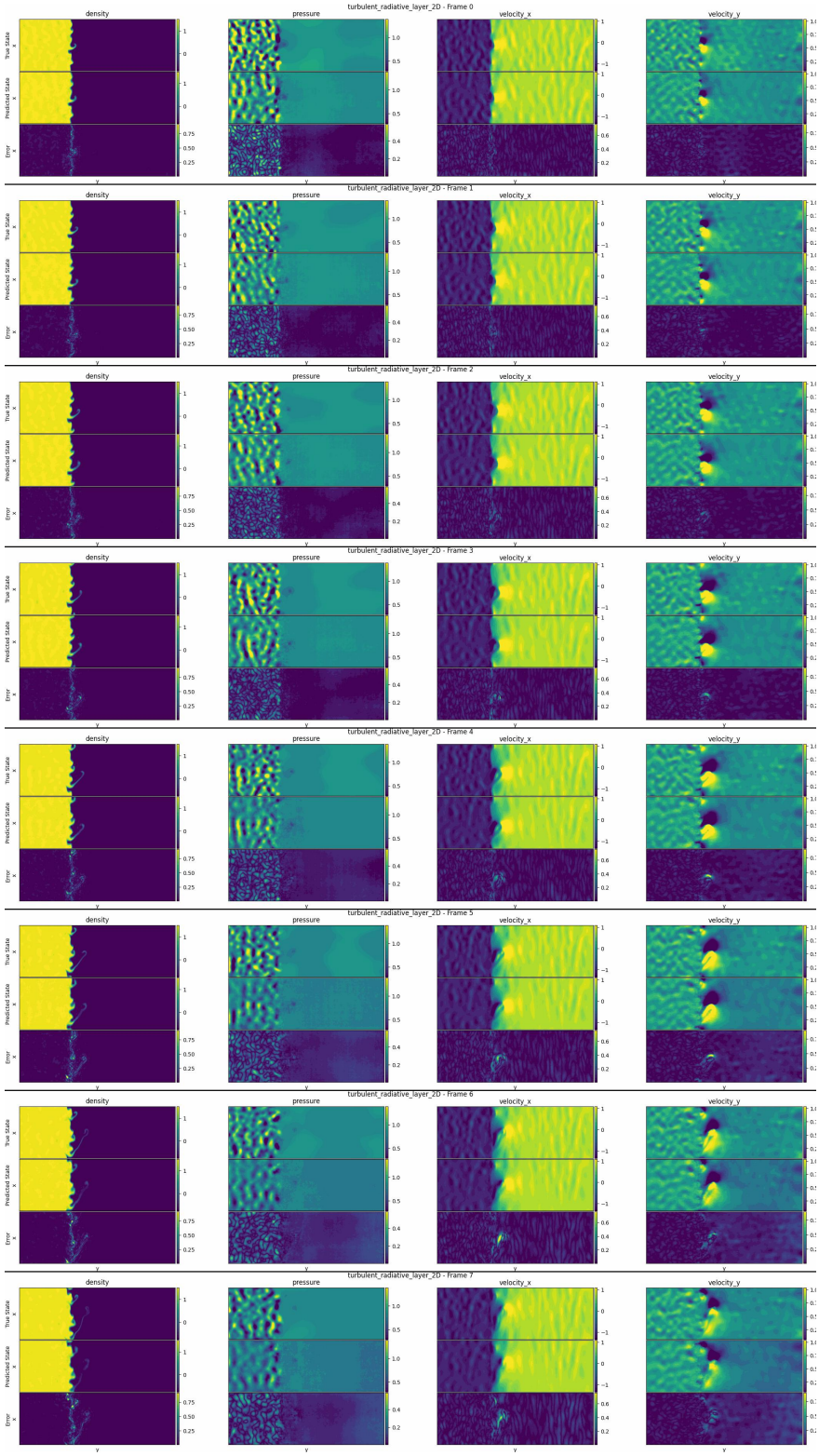

Figure 11. Turbulent Radiative Layer (TR). Representative COAST rollout prediction of all the different fields in y direction, and point-wise error against the ground truth.

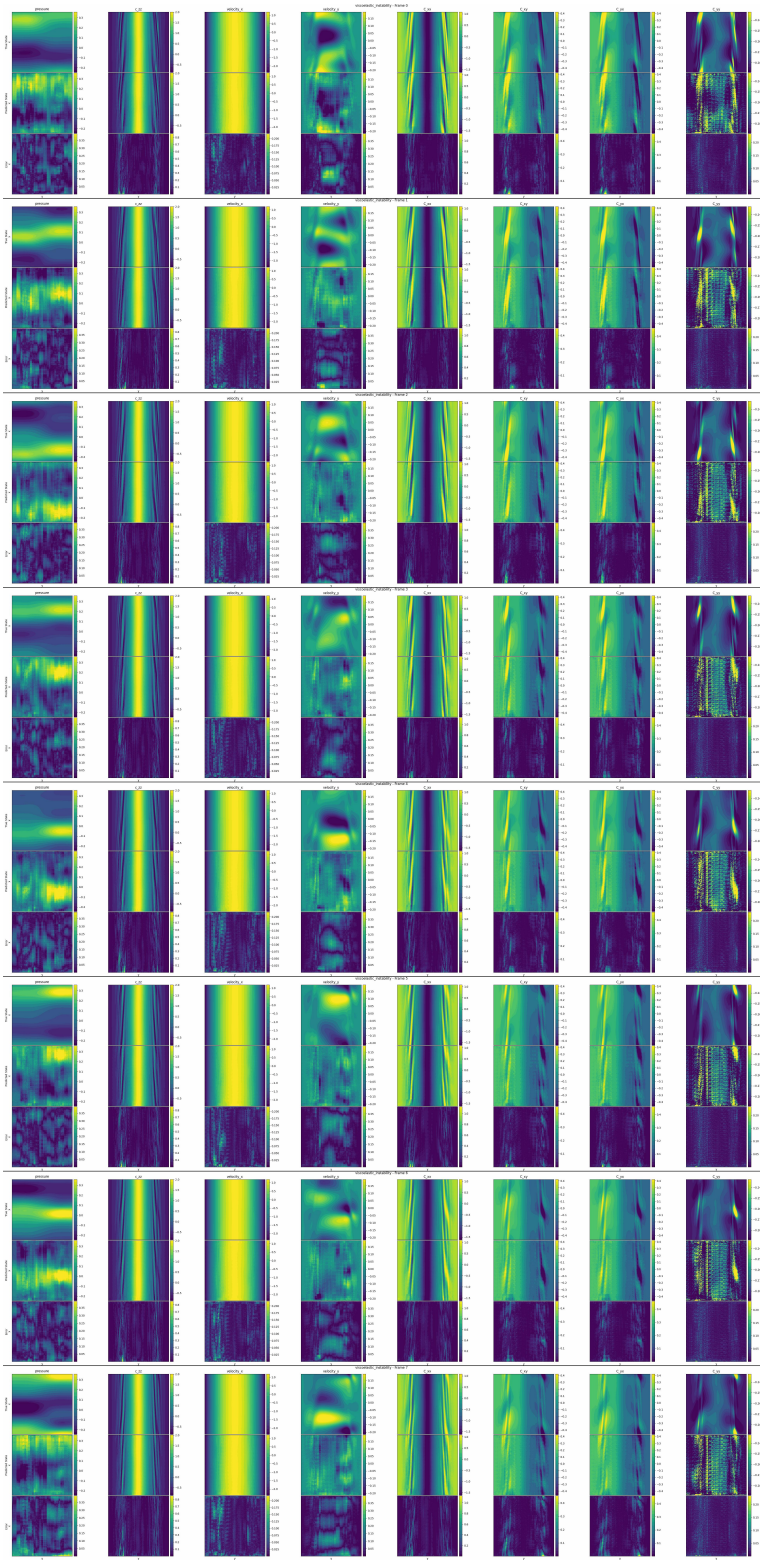

Figure 12. Viscoelastic Fluids (VF). Representative COAST rollout prediction of all the different fields in y direction, and point-wise error against the ground truth.

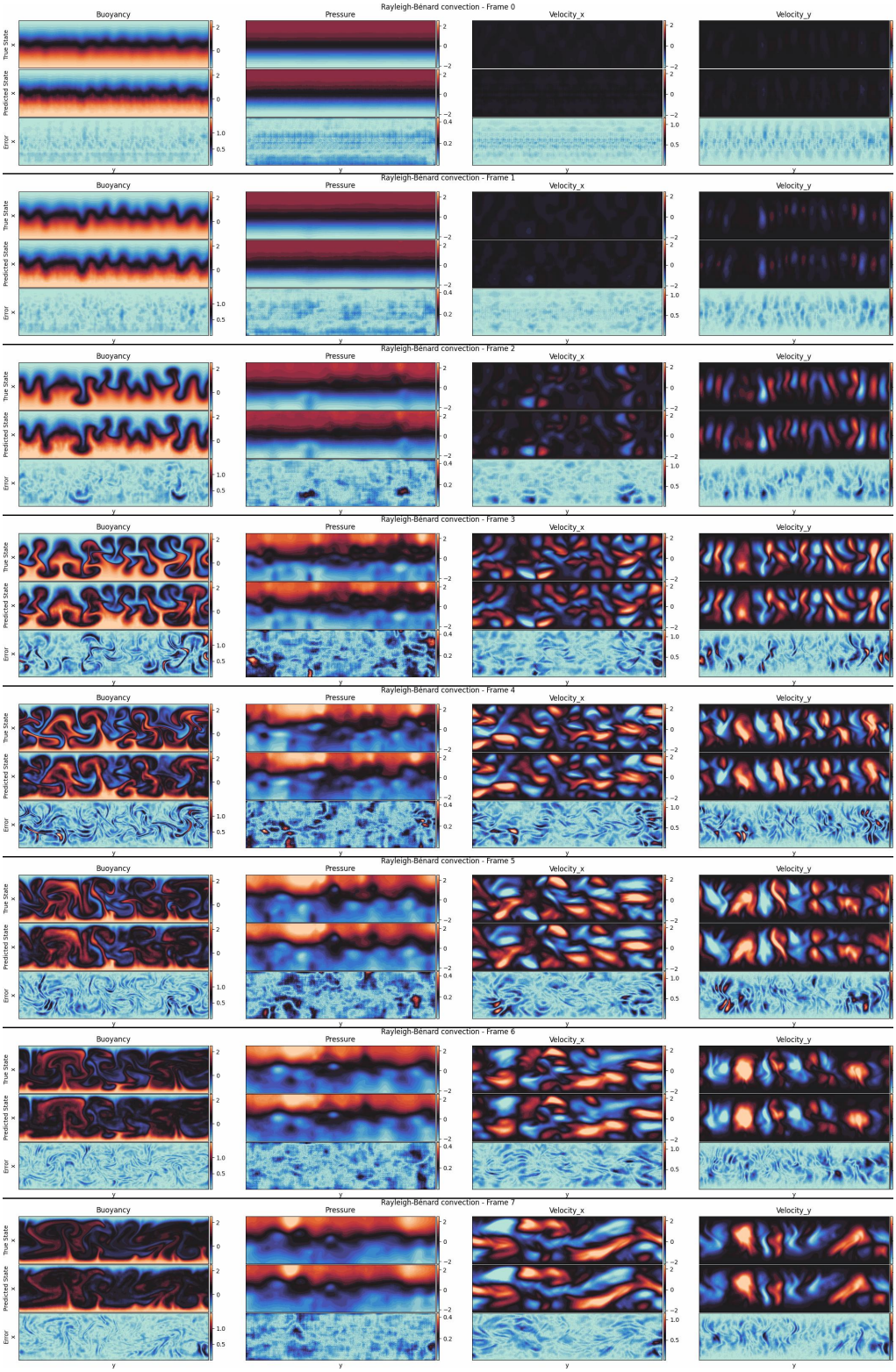

Figure 13. Rayleigh-Bénard Convection (RB). Representative COAST rollout prediction of all the different fields, and point-wise error against the ground truth.