ALIGN THE SKIP: SPATIALLY ALIGNED AND TEMPORALLY RESOLVED HIERARCHICAL SURROGATES FOR MULTISCALE TURBULENCE

Anonymous authors
Paper under double-blind review

ABSTRACT

Hierarchical encoder–decoder surrogates efficiently represent multiscale turbulent dynamics, but their skip connections fuse input-time encoder features with future-oriented decoder features at matching grid indices. Under advection, these features need not remain spatially correspondent. We introduce decoder-predicted spatial alignment, which learns where encoder features should be sampled before skip fusion. Across three fluid and plasma benchmarks, alignment substantially improves autoregressive forecasting and supports accurate prediction of multiple states within a larger forecast interval, reducing complete autoregressive model calls while preserving intermediate-time outputs. These intermediate states also partition each outer interval into shorter subintervals for physics-informed training through equation supervision. At a fixed outer interval, increasing the number of internal subintervals from one to eight reduces the 64-step Kolmogorov-flow relative error by more than 40%, bringing equation-supervised performance close to data-supervised accuracy without using future-state targets. We further train full-resolution surrogates using spatially reduced observations of the training initial states together with a persistent prediction-derived unresolved state. This state is iteratively constructed from model predictions and carried across optimization, allowing initially absent fine-scale structure to emerge and adapt. Reducing the effective training observations from 64×64 to only 2×2 block averages, a $1024\times$ reduction in observed spatial values, increases the 64-step error by just 17% when evaluation begins from full-resolution held-out states. Even when the observed training states contain no spatial variation, small white noise seeds unresolved structure that is subsequently developed through equation supervision and persistent model-generated state. These results show that explicit spatial correspondence and temporally resolved equation supervision improve multiscale forecasting while reducing reliance on paired current-to-future states and spatially resolved training initial conditions.

1 INTRODUCTION

Fast surrogate models for unsteady partial differential equations (PDEs) can reduce the cost of repeated simulation in turbulent-flow engineering, weather and climate prediction, and magnetized plasma dynamics (Kochkov et al., 2021; Lam et al., 2023). We consider models that learn, across varying initial conditions, a map from the current state and boundary data to future states (Lu et al., 2021; Li et al., 2021a). Long-horizon prediction remains difficult because multiscale interactions must be represented accurately while autoregressive errors amplify (Kolmogorov, 1962; Lippe et al., 2023). Hierarchical encoder–decoder architectures address these requirements through localized fine-resolution processing, coarse longer-range representations, and equal-resolution skip connections (Ronneberger et al., 2015; Liu et al., 2021).

In dynamical forecasting, however, encoder features describe the current state while decoder features represent a future state. Under advection, equal grid indices may therefore cease to indicate spatial correspondence, causing skip connections to fuse mismatched features (Zhang et al., 2024; Khrabry et al., 2026b). We address this mismatch by letting each future-oriented decoder level predict where its corresponding input-time encoder feature should be sampled before fusion.

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.

Spatial correspondence also depends on forecast time because features generally travel farther as the time increases. Conventional fixed-step autoregressive models couple the interval advanced by one model invocation to the temporal resolution of their outputs. Predicting multiple states within a larger outer interval separates these quantities, reduces the number of complete model invocations needed to generate a trajectory, and retains intermediate-time outputs. Prior work has explored temporal bundling, discrete multi-state output, and explicit temporal queries (Brandstetter et al., 2022; Lu et al., 2021; Diab & Al-Kobaisi, 2025; Cho et al., 2024). We consider both direct multi-state prediction and queried-time conditioning. The intermediate predictions also divide the outer transition into shorter subintervals, providing a finer temporal discretization for equation supervision.

Training surrogate models conventionally requires trajectories containing paired current and future states, but sufficiently resolved trajectories may be expensive, sparsely observed, or unavailable. When the governing equations are known, equation residuals can reduce reliance on future-state targets (Raissi et al., 2019; Li et al., 2021b; Wang et al., 2021; Zhu et al., 2019; Geneva & Zabaras, 2020). Removing future-state targets, however, may still leave training dependent on spatially resolved current states. We therefore also consider training from spatially reduced current observations, using persistent prediction-derived unresolved state to supply missing spatial content as training proceeds.

Our contributions are threefold: 1) We introduce decoder-predicted skip alignment to resolve the spatial mismatch that arises when skip connections fuse input-time encoder features with future-oriented decoder features at equal grid indices in advective systems. This improves prediction accuracy across hierarchical convolutional and Swin-based backbones, particularly when predicting multiple future time steps per model call. For Kolmogorov flow, alignment reduces prediction error by up to a factor of four, and a 16-channel aligned model outperforms a 64-channel unaligned model.

2) We efficiently predict intermediate states within a fixed outer forecast interval, reducing temporal discretization error in equation supervision that uses only current-state inputs without future-state targets. Increasing the number of subintervals from one to eight reduces the Kolmogorov-flow 64-step autoregressive rollout error of our skip-aligned convolutional model by more than 40%, whereas the corresponding unaligned model does not benefit from temporal refinement.

3) To train from low-resolution current-state inputs, we introduce a persistent prediction-derived unresolved state that is added to each reduced observation and updated from model predictions. This allows initially missing fine-scale structure to develop and adapt during training, exposing the model to full-resolution states without spatially resolved training inputs or future-state targets. Reducing the observed Kolmogorov training states from 64×64 to 2×2 block averages increases the best 64-step autoregressive rollout error by only 17%. Even when the training data is reduced to its spatial mean, only a 62% increase in error is observed.

2 Related Work

Our method connects hierarchical PDE forecasting, learned feature alignment, multi-state temporal prediction, and physics-informed learning from incomplete observations. We focus on distinctions most relevant to our contributions; Appendix B provides detailed comparisons.

Hierarchical architectures for PDE forecasting. Convolutional U-Nets and Swin U-Nets suit multiscale forecasting by combining localized processing, progressively longer-range interaction through resolution changes, and equal-resolution skip connections that transfer spatial detail (Ronneberger et al., 2015; Liu et al., 2021; Cao et al., 2023). For fixed widths and kernel or window sizes, convolutional and shifted-window hierarchies scale linearly with the number N of spatial tokens, whereas transformer global attention is quadratic and Fourier operators use $\mathcal{O}(N \log N)$ transforms for spectral mixing (Vaswani et al., 2017; Liu et al., 2021; Li et al., 2021a). Fourier operators provide efficient global communication, although spectral truncation can limit fine-scale information in the global mixing path (Li et al., 2021a; Raonić et al., 2023). See Appendix B.1 for details.

Temporal mismatch and learned skip alignment. Previous work reduces temporal mismatch in hierarchical forecasting by distributing evolution across multiple stages or using alternative hierarchical organizations (Zhang et al., 2024; Khrabry et al., 2026b;a). We retain a single end-to-end encoder–decoder and correct correspondence at its skip connections. Learned displacement fields

have aligned multiresolution features in static vision models, including feature-pyramid networks and a U-Net-like image-completion model (Huang et al., 2021b; Li et al., 2020; Huang et al., 2021a; Yan et al., 2018). These methods address mismatch within one observed image, whereas our decoder resamples an input-time encoder feature for fusion with a representation of an unobserved future state. Related PDE architectures transport the evolving feature stream or use learned warps as the principal within-level spatial interaction (Zakariaei et al., 2024; Muser et al., 2026), rather than aligning a retained input-time skip before future-oriented decoder fusion. Details appear in Appendix B.2.

Multi-state and queried-time prediction. Temporal bundling returns a prescribed block of future states, whereas DeepONet-based, graph-based, and temporal operators condition predictions on requested times (Brandstetter et al., 2022; Lu et al., 2021; Diab & Al-Kobaisi, 2025; Cho et al., 2024; Park et al., 2023). We evaluate both formulations and do not claim either parameterization alone as novel. Their role is to separate outer invocation interval from output and residual resolution within an aligned hierarchy. The experiments use offsets represented during training and therefore do not establish interpolation to unseen times. Further comparisons appear in Appendix B.3.

Physics-informed learning from incomplete observations. Equation-based objectives can train reusable PDE surrogates with reduced or absent future-state supervision (Raissi et al., 2019; Zhu et al., 2019; Geneva & Zabaras, 2020; Li et al., 2021b). Physics-only operator-learning methods may omit future-state targets but require complete high-resolution current states during training. Physics-informed super-resolution and reconstruction can avoid corresponding high-resolution targets, but use coarse or sparse observations from multiple times (Kelshaw et al., 2022; Ren et al., 2023; Kelshaw & Magri, 2024; Page, 2025). We instead train from spatially reduced current states without future observations or complete high-resolution training states, using persistent prediction-derived unresolved state to supply missing spatial content. Evaluation from full-resolution held-out states assesses transition learning rather than test-time reconstruction. See Appendix B.4 for details.

3 METHOD

Figure 1 summarizes the method. Decoder-predicted warping aligns each input-time encoder skip with a future-oriented decoder representation. Direct or queried-time output produces states inside a larger outer interval, and adjacent states define shorter intervals for equation supervision. When training observations are spatially reduced, persistent prediction-derived unresolved state supplies missing spatial content. Secondary variants and implementation details appear in the appendices.

3.1 SPATIALLY ALIGNED HIERARCHICAL SURROGATE

Let E_ℓ and D_ℓ denote encoder and decoder representations at resolution ℓ, with $\ell = 0$ finest. Conventional skips fuse the input-time feature E_ℓ directly with a future-oriented decoder representation. We instead let decoder upsampling produce both the next decoder feature and a bounded displacement,

$$(D_\ell, \delta_\ell) = \mathcal{U}_\ell(D_{\ell+1}), \qquad \widetilde{E}_\ell(\mathbf{x}) = \mathcal{A}(E_\ell, \delta_\ell)(\mathbf{x}) = E_\ell(\mathbf{x} + \delta_\ell(\mathbf{x}))\,, \tag{1}$$

and fuse $\widetilde{E}_\ell$ with D_ℓ before the decoder block. Because the decoder has processed the bottleneck and coarser future-oriented representations, it can infer where relevant input-time information should be retrieved. Principal models predict independent per-channel displacements at every skip. A spatially shared field assumes that all latent channels retrieve information from the same source locations, whereas per-channel fields allow features encoding different structures or attributes to establish distinct correspondences. These fields describe latent-feature correspondence and are not constrained to reproduce a physical flow map. As an ablation, we instead predict $\delta_\ell = h_\ell(E_\ell)$ from the input-time encoder feature, testing whether correspondence can be inferred before the future-oriented decoder representation is available. We also compare spatially shared displacement and alignment restricted to the finest skip. Parameterization, initialization, interpolation, boundary handling, and complete ablation definitions appear in Appendix C.

3.2 TEMPORALLY RESOLVED PREDICTION

Let $u^n = u(t_n)$ denote the current state and Δt_{data} the interval between stored fields. One invocation spans $\Delta T = N_{\mathrm{out}} \Delta t_{\mathrm{data}}$; we separate this outer interval from the temporal spacing of states returned

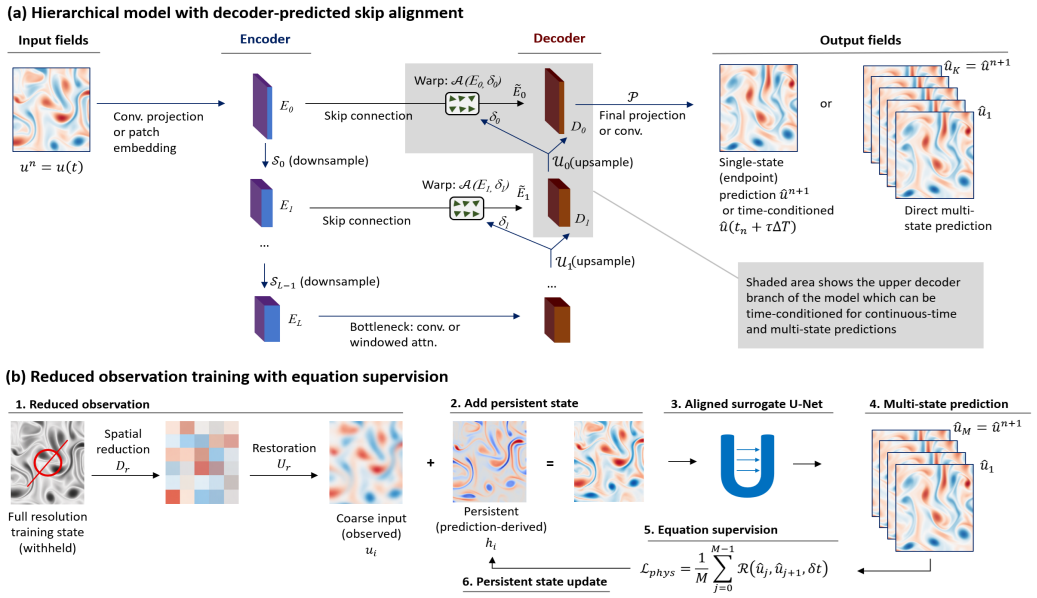

Figure 1: Overview of the spatially aligned and temporally resolved surrogate and its equation-supervised training regimes.

inside it. We consider two output parameterizations:

$$\mathcal{P}(D_0) \longrightarrow [\widehat{u}_1, \ldots, \widehat{u}_K], \qquad \widehat{u}(t_n + \tau\Delta T) = \mathcal{G}_\theta(u^n, \tau), \quad \tau \in (0, 1]. \tag{2}$$

Direct multi-state prediction jointly emits K states at prescribed offsets through one final projection. Queried-time prediction instead returns the state associated with a supplied offset τ. Direct output is efficient when required times are fixed, whereas queried-time conditioning provides one shared parameterization across requested offsets.

Repeating the complete hierarchy for every query would remove much of this computational benefit. We therefore use late conditioning: the encoder, bottleneck, and lower decoder construct one query-independent multiscale representation, while the upper decoder branch is reevaluated for each offset. Multiple queries are processed together by expanding shared features along a query dimension and folding that dimension into the batch. The query modulates normalized upper-decoder features through feature-wise affine conditioning (Huang & Belongie, 2017); convolutional and Swin blocks apply it after Group and Layer Normalization, respectively (Wu & He, 2018; Ba et al., 2016).

At the finest conditioned upsampling stage, separate query-dependent displacement fields warp the encoder skip and upsampled decoder feature. Both representations are therefore aligned for the requested forecast time before fusion. Restricting conditioning to the upper branch preserves reuse of the multiscale trunk while allowing fine-scale placement and correspondence to vary with τ.

For ordered outputs $0 = \tau_0 < \cdots < \tau_K = 1$, with $\widehat{u}_0 = u^n$, only $\widehat{u}_K$ initializes the next autoregressive invocation. Intermediate predictions are retained as trajectory outputs or adjacent states for equation supervision. During training, query intervals may be sampled without replacement to reduce conditioned-branch evaluations; evaluation uses the complete requested sequence. Tensorization, temporal sampling, conditioning variants, and displacement parameterization appear in Appendix D.

3.3 Temporally Resolved Equation Supervision

A residual evaluated across the complete outer interval may not resolve the nonlinear evolution between its endpoints. We therefore use intermediate predictions to evaluate equation consistency over shorter adjacent intervals. With M generated subintervals per stored-data interval,

$$K = N_{\text{out}} M, \qquad \delta t = \frac{\Delta t_{\text{data}}}{M} = \frac{\Delta T}{K}. \tag{3}$$

After prepending the supplied state as $\widehat{u}_0 = u^n$, the objective is

$$\mathcal{L}_{\text{phys}} = \frac{1}{K} \sum_{j=0}^{K-1} \mathcal{R}(\widehat{u}_j, \widehat{u}_{j+1}, \delta t), \tag{4}$$

where $\mathcal{R}$ is a system-specific discrete residual. Increasing M refines the generated trajectory used by the objective while leaving ΔT unchanged. Thus, temporal refinement neither requires more complete encoder–decoder invocations nor turns the model into an ordinary surrogate with a smaller outer step; one invocation represents its internal evolution more explicitly.

For Kolmogorov and Hasegawa–Wakatani dynamics, we use a Crank–Nicolson-type discretization, evaluating temporal differences between adjacent predictions and spatial terms at the midpoint state (Crank & Nicolson, 1947). The smoke objective instead follows a solver-inspired discrete update reflecting advection, buoyancy, viscosity, and incompressibility. Reported experiments use no future-state targets in the objective: trajectories provide current states from which generated evolution begins, while future states are used only for evaluation. Complete residuals appear in Appendix A.

3.4 Training from spatially reduced observations

Equation supervision removes future-state targets but normally still uses representative full-resolution current states. We reduce this information by supplying spatial block averages. Let $\mathcal{D}_r$ denote average pooling by factor r in each spatial direction and $\mathcal{U}_r$ interpolation back to the model grid:

$$u_i^{\text{obs}} = \mathcal{U}_r \mathcal{D}_r(u_i), \qquad P_{\text{unres}}(u) = u - \mathcal{U}_r \mathcal{D}_r(u). \tag{5}$$

Interpolation restores tensor dimensions but not the spatial information removed by pooling. Because many full-resolution turbulent states share the same block averages, the observation alone does not determine the unresolved content used to initialize a full-resolution transition.

For each training example i, identified by trajectory and starting-time indices, we retain a prediction-derived unresolved state h_i and construct

$$u_i^{\text{model}} = u_i^{\text{obs}} + h_i, \qquad h_i \leftarrow (1-\alpha) h_i + \alpha P_{\text{unres}}\left(\widehat{u}_i^{(k_{\text{copy}})}\right). \tag{6}$$

An internal prediction thus supplies the unresolved component used when the same example is presented again. Persistence prevents newly generated spatial structure from being discarded after each optimization step; it becomes part of the effective full-resolution training input and can adapt as the transition model improves. The cache is detached and updated outside the computation graph. It is neither an optimized latent variable nor constructed from the withheld full-resolution current state, a future-state target, or a prescribed fine-scale spectrum. In the spatially uniform limit, a small white-noise perturbation only initiates symmetry breaking; subsequent unresolved structure comes from equation-constrained model predictions. Reduction and restoration, cache-update schedules, temporal curricula, noise, and controls appear in Appendix G.

4 Experiments and Results

4.1 Systems, models, and evaluation

Systems. We evaluate three two-dimensional systems with multiscale advective dynamics. Kolmogorov flow is represented by scalar vorticity on a periodic 256×256 grid and provides independently initialized trajectories of forced turbulence. Modified Hasegawa–Wakatani turbulence is represented by density and electrostatic potential on a periodic 256×256 grid; its train and test units are nonoverlapping temporal segments from one statistically developed simulation rather than independent realizations. Buoyant smoke contains concentration and two velocity components on a wall-bounded 128×128 grid and consists of independent transient trajectories. We report rollout errors for 64, 16, and 8 time steps for the respective systems. Governing equations, numerical solvers, data partitions, and system-specific residuals are given in Appendix A.

Models and baselines. Aligned and unaligned convolutional and Swin U-Net backbones are matched in width, depth, optimization, data partition, temporal output parameterization, and random seed, isolating the effect of skip alignment. Endpoint-only, direct multi-state, and queried-time models

test temporal parameterization. Fourier Neural Operator and vision-transformer baselines provide broader architecture and accuracy–cost comparisons under matched input information and training budgets. Complete details appear in Appendix H.1.

Selection and replication. Learning rates are selected under a common validation protocol separate from final reporting. Reported results use seeds 2, 12, and 22, each determining model initialization and a trajectory or temporal-segment partition shared across directly compared configurations. Variation across the three runs therefore reflects both optimization and partition sensitivity.

Evaluation. Relative L_2 errors are normalized per state channel and averaged across channels and test samples. Forecasts are autoregressive; multi-state models retain all returned states for evaluation, but only each block endpoint initializes the next invocation. Results report the seed mean (filled markers), best-seed (open stars), and sample standard deviation over the three seeds. Long-rollout spectra are computed separately for each held-out trajectory before averaging. Partition records, timing, and complete metrics appear in Appendix H.1.

4.2 Alignment across outer prediction intervals

We first examine how alignment interacts with the outer interval $n_t \Delta t_{\mathrm{data}}$. Increasing n_t reduces the number of complete hierarchy evaluations over a fixed physical horizon but makes each transition harder to learn. An intermediate interval may therefore be optimal, balancing fewer autoregressive calls against greater transition complexity. Both turbulent systems exhibit this trade-off (Fig. 2a). For the 32-channel aligned models, the Kolmogorov seed-mean step-64 error is approximately unchanged between $n_t = 1$ and 2, while the Hasegawa–Wakatani step-16 error improves through $n_t = 4$. Both systems deteriorate at the longest interval, indicating that moderate outer intervals can reduce model calls without sacrificing endpoint accuracy, but that transition difficulty eventually dominates. Full results across widths, outer intervals, and evaluation horizons appear in Appendices H.2 and H.4.

Alignment improves the displayed endpoint seed means at every outer interval for both turbulent systems. For example, at $n_t = 1$, alignment reduces the 32-channel Kolmogorov seed-mean step-64 error by approximately a factor of three. The complete width sweeps show that this advantage persists across model capacities and completed seeds rather than arising from one selected initialization. At $n_t = 4$, aligned U-Nets consequently occupy the strongest measured accuracy–training-cost region among the completed operating points in Fig. 2d, with smaller aligned models outperforming substantially larger unaligned models. The Kolmogorov comparisons at steps 16 and 64 further show that the effect is robust to evaluation horizon and generally becomes clearer after longer autoregressive rollout. Individual seeds, completed-run counts, measured generation costs, and the complete horizon-dependent comparisons appear in Appendices H.2 and H.4.

Matched alignment-design ablations establish that decoder-predicted alignment reduces seed-mean error by 42–71% relative to encoder-predicted alignment for Kolmogorov flow and by 18–25% for Hasegawa–Wakatani. Per-channel displacement provides a smaller consistent improvement, whereas extending alignment from the finest skip to all resolutions is more interval-dependent and becomes most useful for longer outer transitions. Complete matched results appear in Appendix H.3.

The result also extends qualitatively to shifted-window hierarchies (see Fig. 2d). The tested Swin models use a factor-four patch projection followed by one factor-two hierarchy transition, giving two represented patch-grid resolutions; alignment improves most matched operating points. This supports the applicability of decoder-predicted skip alignment beyond convolutional feature processing; the experiment is not intended as a comprehensive comparison between convolutional and attention-based hierarchies. Among the completed configurations, the aligned convolutional model provides the strongest measured accuracy–cost trade-off. Complete Swin, FNO, and ViT comparisons, including long-rollout stability and spectral fidelity, appear in Appendix H.4.

The buoyant smoke system provides a complementary test because each stored-data interval spans four solver steps. At the solver-step interval made accessible by equation supervision, alignment reduces the 32-channel seed-mean eight-step error from 0.489 to 0.229, whereas aligned and unaligned models are much closer at the stored-data interval under data supervision (Fig. 2c). Because both the objective and the transition interval change, this is a qualitative result showing that equation supervision provides access to a finer transition without paired intermediate targets and that alignment is beneficial in this regime. Width-dependent accuracy–cost results appear in Appendix H.4.

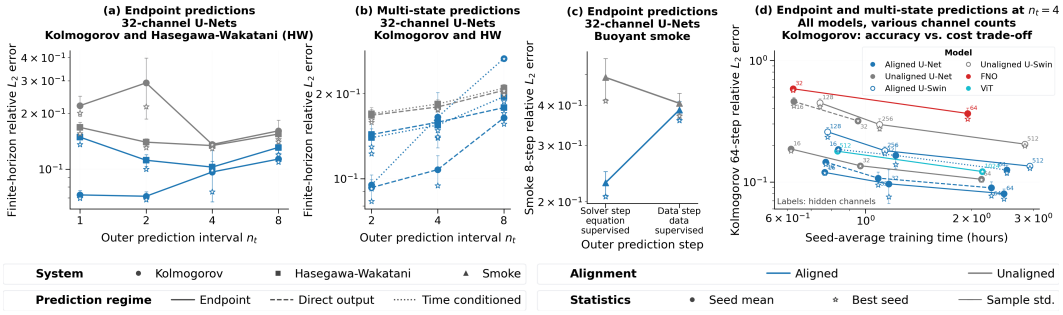

Figure 2: **Alignment and multi-state forecasting.** (a) Endpoint-only 32-channel U-Nets for Kolmogorov flow (step 64) and Hasegawa–Wakatani (HW; step 16) across outer intervals n_t. (b) Direct and queried-time multi-state prediction. (c) Buoyant-smoke endpoint prediction at the solver interval with equation supervision and the stored-data interval with data supervision (qualitative comparison). (d) Kolmogorov step-64 error versus seed-average training time at $n_t = 4$; labels give hidden-channel counts. Complete sweeps and costs appear in Appendices H.2–H.5.

4.3 Multi-state prediction within the outer interval

We next ask whether one complete hierarchy evaluation can return intermediate stored-data states without sacrificing endpoint accuracy. The model returns one prediction per stored-data checkpoint within n_t; all outputs are evaluated, but only the final state initializes the next block. Direct output emits all states through one projection, whereas queried-time prediction evaluates requested offsets while reusing the encoder and lower decoder. The latter therefore reduces complete hierarchy evaluations, although its conditioned upper decoder branch is evaluated for each requested offset.

At $n_t = 4$, the 64-channel aligned direct-output Kolmogorov model obtains a best step-64 error of 0.077, close to 0.073 for the matched endpoint-only model, while returning all four stored-data checkpoints from one complete hierarchy evaluation. Across widths, aligned direct-output models remain close to the endpoint-only accuracy–training-cost frontier in Fig. 2d. Alignment is particularly important in this multi-state setting (Fig. 2b): at $n_t = 2$, the best step-64 error is 0.103 with alignment and 0.623 without it, with clear advantages also at $n_t = 4$ and 8. This supports the proposed mechanism: as requested outputs extend farther from the input, fusion with spatially stale input-time skips becomes increasingly restrictive. Complete results across systems, model widths, outer intervals, and output times appear in Appendix H.5.

The preferred temporal parameterization is system dependent. Direct and queried-time prediction achieve similar finite-horizon accuracy for Hasegawa–Wakatani, whereas direct output becomes increasingly accurate for Kolmogorov flow as n_t grows. Direct output emits all prescribed states through one projection at nearly fixed complete-hierarchy cost, while query conditioning accepts continuous-valued offsets but reevaluates the conditioned upper decoder branch for each query.

Appendix H.5 tests whether these conclusions depend on forecast horizon and whether they persist in long-rollout statistics. For Kolmogorov flow, the alignment advantage is present at both steps 16 and 64 and generally becomes more pronounced at the longer horizon, particularly for short outer intervals and multi-state prediction. This indicates that alignment improves autoregressive stability rather than only immediate forecast accuracy. Long-rollout spatial-Fourier comparisons provide complementary evidence. For Kolmogorov flow, aligned models generally retain favorable spectral fidelity, while several unaligned or baseline operating points become spectrally inaccurate or unstable. For Hasegawa–Wakatani turbulence, spectral performance is more variable across architectures, intervals, and seeds. A representative rollout illustrating the transition from trajectory-level to statistical agreement appears in Appendix H.4.1, while complete horizon-dependent and Fourier comparisons appear in Appendices H.2 and H.4.

Multi-state output therefore serves two roles: it reduces complete hierarchy evaluations while retaining intermediate trajectory states, and it supplies the adjacent-time predictions used to evaluate governing-equation residuals over shorter intervals. The following experiments isolate the latter role by fixing the outer prediction interval and varying its internal temporal resolution.

4.4 TEMPORALLY RESOLVED EQUATION SUPERVISION

We fix the outer prediction interval at one stored-data interval and vary the number M of internal temporal intervals used for target-free equation supervision, as defined in Eqs. (3)–(4). For Kolmogorov and Hasegawa–Wakatani dynamics, the temporal derivative over each interval is approximated from its endpoint states, and the right-hand side of the governing equations is evaluated at their arithmetic midpoint. The resulting residual uses a second-order midpoint discretization in time. Increasing M therefore shortens the interval used to evaluate the residual without changing the outer prediction interval. Appendices H.6.1–H.6.3 report the complete short- and longer-horizon errors, long-rollout Fourier diagnostics, computational costs, and supplemental optimization results.

We compare direct-output and time-conditioned models under a common 96-epoch training budget. For Kolmogorov flow, direct output improves consistently through $M = 8$ (Fig. 3a). Relative to $M = 1$, the best-seed errors at rollout steps 16 and 64 decrease by 66% and 41%, respectively, and the long-rollout spatial Fourier error decreases by 36%; seed means show the same overall trend. Time conditioning instead performs best at moderate refinement and deteriorates at $M = 8$. Further increasing direct-output refinement to $M = 16$ reverses these improvements and increases seed variability. Thus, Kolmogorov flow exhibits an intermediate optimum, with direct output benefiting from finer temporal discretization over a wider range of M than time conditioning. Short- and longer-horizon results are reported in Appendix H.6.1, and long-rollout spectral results in Appendix H.6.2.

Hasegawa–Wakatani turbulence also benefits from temporal refinement, but the output parameterization has a stronger effect on optimization robustness (Fig. 3c). Under the matched 96-epoch budget, both parameterizations improve through $M = 4$. At $M = 8$, time conditioning reduces the seed-mean 16-step error by 54% relative to $M = 1$ and remains consistent across seeds, whereas direct output becomes strongly seed dependent. The corresponding short-horizon and field-specific Fourier results are reported in Appendices H.6.1 and H.6.2.

Supplemental direct-output models trained for 160 epochs with a smaller learning rate recover a competitive best Hasegawa–Wakatani result at $M = 8$, but not a competitive seed mean (Appendix H.6.1). Direct output can therefore represent an accurate refined trajectory, while time conditioning provides substantially more reliable optimization for this system under the tested settings. A possible explanation is that sharing a query-conditioned mapping across internal times supplies a temporal inductive bias that regularizes the predicted trajectory, although the present experiments do not isolate this mechanism independently. The associated computational costs are reported in Appendix H.6.3.

Alignment enables consistent gains from temporal refinement. In the matched Kolmogorov ablation, per-channel alignment gives the lowest seed-mean 64-step error for every $M \leq 8$, whereas the unaligned U-Net deteriorates beyond $M = 2$. Complete results appear in Appendix H.7.

Overall, shortening the residual interval can substantially improve target-free training, but the benefit is not monotonic and depends on the physical system and temporal output parameterization. Direct output is more effective at high refinement for Kolmogorov flow, whereas time conditioning provides more reliable optimization for Hasegawa–Wakatani turbulence. We use the best tested Kolmogorov direct-output refinement, $M = 8$, in the following reduced-observation experiments.

4.5 LEARNING FROM SPATIALLY REDUCED TRAINING STATES

We next ask how much spatial information must be retained in the current-state inputs used for target-free equation supervision. To represent under-resolved simulation or experimental observations, each training state is average pooled to a coarser grid and interpolated back to the original model grid. Interpolation restores the required input dimensions but not the spatial information removed by pooling. The resulting input is supplemented with the persistent prediction-derived unresolved state described in Section 3.4. At evaluation, rollouts begin from full-resolution held-out states. These experiments therefore test whether a full-resolution transition model can be learned from spatially reduced training inputs, rather than whether a particular full-resolution state can be reconstructed from a coarse test observation. All reduced-observation models are trained for 144 epochs.

Forecast accuracy degrades slowly over substantial reductions in the observed training-state resolution (Fig. 3b). Reducing the effective observed grid from 64×64 to 8×8 decreases the number of observed spatial values by a factor of 64 while increasing the best 64-step error by only 4%. Further reducing the observation to 2×2 increases the error by 17% relative to the 64×64 case. Even at

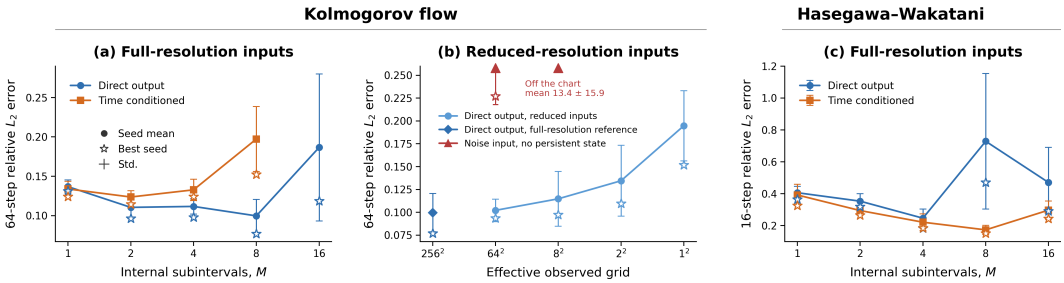

Figure 3: **Equation supervision and reduced observations.** (a) Kolmogorov step-64 error versus internal subintervals M. (b) Step-64 error after training from reduced current states; red markers show the best tested fresh-noise controls without persistent state. The full-resolution reference uses 96 epochs, and reduced-observation models use 144. (c) Hasegawa–Wakatani step-16 error under a matched 96-epoch budget.

1×1, where each supplied input retains only its spatial mean, the learned model produces useful finite-horizon forecasts when initialized from a full-resolution held-out state.

The persistent-state update is adapted to the observation resolution. At mild reduction, an early internal prediction limits spatial displacement between the resolved input structure and the prediction-derived unresolved state. At severe reduction, where the observation retains little spatial organization, the final internal prediction provides a longer interval over which initially missing structure can develop. Because this development is slower under severe reduction, the 2×2 and 1×1 models are initially trained with a larger effective temporal interval in the equation residual, which is gradually reduced to the standard interval during early training (see Appendix H.8.1 for details).

At 1×1, the supplied input is spatially uniform, so decaying Gaussian white noise is added during early training to break spatial symmetry. Equation-constrained predictions then generate spatial structure that is retained and updated through the persistent prediction-derived unresolved state. This shows that the reduction-adapted procedure can construct training states sufficient for learning useful full-resolution dynamics even when the supplied observations contain no spatial variation. Noise and curriculum details are given in Appendix H.8.1.

We also disable the persistent state and add freshly sampled Gaussian noise to the reduced inputs (Fig. 3b). At 64×64, the best tested noise level reduces the seed-mean 64-step error from 1.114 to 0.244, but remains worse than the persistent-state result of 0.102. At 8×8, occasional finite rollouts are accompanied by severe seed-dependent failures, giving a mean error of 13.4. Fresh noise can thus assist mild reduction but does not replace persistent prediction-derived structure under severe information loss. Complete sweeps and seed-level results appear in Appendix H.8.3.

Together, these results show that persistent prediction-derived unresolved structure enables useful full-resolution dynamics to be learned from training inputs containing little or, at 1×1, no spatial variation. A representative 1×1 rollout and spatial spectra appear in Appendix H.8.2.

5 Conclusion

We introduced decoder-predicted skip alignment for hierarchical surrogates of unsteady multiscale systems. By sampling input-time encoder features at locations inferred from future-oriented decoder representations, the method improves spatial correspondence while retaining convolutional or shifted-window processing within each resolution. Across fluid and plasma benchmarks, alignment improves autoregressive forecasting for convolutional and Swin U-Net backbones, with particularly large benefits when one invocation must represent multiple future states.

The intermediate states produced by multi-state prediction also divide the outer transition into shorter intervals for equation supervision. At a fixed outer interval, temporal refinement substantially improves target-free training. A persistent prediction-derived unresolved state further enables training from spatially reduced current observations without future-state targets or complete high-resolution current states, even when the supplied observations initially contain no spatial variation.

Reproducibility statement

We provide the implementation of the reported experiments in an anonymized OSF repository, available at `https://osf.io/wrmxn/overview?view_only=a305611162134defb7670eb2697a2fc9`. The repository includes documentation, dependency specifications, model implementations, training and analysis code, the Hasegawa–Wakatani turbulence dataset, and selected model checkpoints. All reported models were trained and evaluated on a single NVIDIA A100 80 GB GPU. The FNO baselines use `neuraloperator` version 2.0.0.

AI use statement

Generative AI tools were used to assist with language revision, organization of the related-work discussion, and identification of potentially relevant literature. The authors manually reviewed the cited papers, checked the technical claims and bibliography entries, and revised all AI-assisted text. Generative AI was not used to generate experimental data or reported results. The authors take responsibility for the final content of this work, including text and artifacts produced with AI assistance.

References

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Guha Balakrishnan, Amy Zhao, Mert R. Sabuncu, John Guttag, and Adrian V. Dalca. Voxelmorph: A learning framework for deformable medical image registration. *IEEE Transactions on Medical Imaging*, 38(8):1788–1800, 2019.

Johannes Brandstetter, Daniel E. Worrall, and Max Welling. Message passing neural PDE solvers. In *International Conference on Learning Representations*, 2022.

Hu Cao, Yueyue Wang, Joy Chen, Dongsheng Jiang, Xiaopeng Zhang, Qi Tian, and Manning Wang. Swin-Unet: Unet-like pure transformer for medical image segmentation. In *Computer Vision – ECCV 2022 Workshops*, volume 13803 of *Lecture Notes in Computer Science*, pp. 205–218. Springer, 2023. doi: 10.1007/978-3-031-25066-8_9.

Qianying Cao, Somdatta Goswami, and George Em Karniadakis. Laplace neural operator for solving differential equations. *Nature Machine Intelligence*, 6:631–640, 2024. doi: 10.1038/s42256-024-00844-4.

Junfeng Chen, Jonathan Viquerat, and Elie Hachem. U-net architectures for fast prediction of incompressible laminar flows. *arXiv preprint arXiv:1910.13532*, 2019.

Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Sung Woong Cho, Jae Yong Lee, and Hyung Ju Hwang. Learning time-dependent PDE via graph neural networks and deep operator network for robust accuracy on irregular grids. *arXiv preprint arXiv:2402.08187*, 2024.

J. Crank and P. Nicolson. A practical method for numerical evaluation of solutions of partial differential equations of the heat-conduction type. *Mathematical Proceedings of the Cambridge Philosophical Society*, 43(1):50–67, 1947. doi: 10.1017/S0305004100023197.

Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 764–773, 2017.

Waleed Diab and Mohammed Al-Kobaisi. Temporal neural operator for modeling time-dependent physical phenomena. *Scientific Reports*, 15:32791, 2025.

Yongchao Duan, Rui Yang, Ming Zhao, Mingrui Qi, and Sheng-Lung Peng. DAF-UNet: Deformable U-Net with atrous-convolution feature pyramid for retinal vessel segmentation. *Mathematics*, 13 (9):1454, 2025.

Benjamin D. Dudson, Maxim V. Umansky, Xueqiao Q. Xu, Philip B. Snyder, and Howard R. Wilson. BOUT++: A framework for parallel plasma fluid simulations. *Computer Physics Communications*, 180(9):1467–1480, 2009. doi: 10.1016/j.cpc.2009.03.008.

Kai Fukami, Koji Fukagata, and Kunihiko Taira. Super-resolution reconstruction of turbulent flows with machine learning. *Journal of Fluid Mechanics*, 870:106–120, 2019. doi: 10.1017/jfm.2019.238.

Kai Fukami, Koji Fukagata, and Kunihiko Taira. Machine-learning-based spatio-temporal super resolution reconstruction of turbulent flows. *Journal of Fluid Mechanics*, 909:A9, 2021. doi: 10.1017/jfm.2020.948.

Raghudeep Gadde, Varun Jampani, and Peter V. Gehler. Semantic video CNNs through representation warping. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 4463–4472, 2017.

Nicholas Geneva and Nicholas Zabaras. Modeling the dynamics of PDE systems with physics-constrained deep auto-regressive networks. *Journal of Computational Physics*, 403:109056, 2020. doi: 10.1016/j.jcp.2019.109056.

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

Akira Hasegawa and Masahiro Wakatani. Plasma edge turbulence. *Physical Review Letters*, 50(9): 682–686, 1983. doi: 10.1103/PhysRevLett.50.682.

Juncai He, Xinliang Liu, and Jinchao Xu. MgNO: Efficient parameterization of linear operators via multigrid. In *International Conference on Learning Representations*, 2024.

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, 2024.

Philipp Holl, Vladlen Koltun, Kiwon Um, and Nils Thuerey. PhiFlow: A differentiable pde solving framework for deep learning via physical simulations. In *NeurIPS Workshop*, 2020.

Benjamin Holzschuh, Qiang Liu, Georg Kohl, and Nils Thuerey. PDE-transformer: Efficient and versatile transformers for physics simulations. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 23562–23602. PMLR, 2025.

Shihua Huang, Zhichao Lu, Ran Cheng, and Cheng He. FaPN: Feature-aligned pyramid network for dense image prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021a.

Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1501–1510, 2017.

Zilong Huang, Yunchao Wei, Xinggang Wang, Wenyu Liu, Thomas S. Huang, and Humphrey Shi. Alignseg: Feature-aligned segmentation networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021b.

Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial transformer networks. In *Advances in Neural Information Processing Systems*, volume 28, 2015.

Daniel Kelshaw and Luca Magri. Physics-constrained convolutional neural networks for inverse problems in spatiotemporal partial differential equations. *Data-Centric Engineering*, 5:e46, 2024. doi: 10.1017/dce.2024.46.

Daniel Kelshaw, Georgios Rigas, and Luca Magri. Physics-informed CNNs for super-resolution of sparse observations on dynamical systems. In *NeurIPS Workshop on Machine Learning and the Physical Sciences*, 2022.

Alexander I. Khrabry, Edward A. Startsev, Andrew T. Powis, and Igor D. Kaganovich. Deep wave network for modeling multi-scale physical dynamics. *arXiv preprint arXiv:2605.04198*, 2026a.

Alexander I. Khrabry, Edward A. Startsev, Andrew T. Powis, and Igor D. Kaganovich. Hierarchical-embedding autoencoder with a predictor as efficient architecture for learning time-evolution in multi-scale turbulent flows. *Physics of Fluids*, 38(4):045138, 2026b. doi: 10.1063/5.0323107.

Dmitrii Kochkov, Jamie A. Smith, Ayya Alieva, Qing Wang, Michael P. Brenner, and Stephan Hoyer. Machine learning–accelerated computational fluid dynamics. *Proceedings of the National Academy of Sciences*, 118(21):e2101784118, 2021. doi: 10.1073/pnas.2101784118.

A. N. Kolmogorov. A refinement of previous hypotheses concerning the local structure of turbulence in a viscous incompressible fluid at high reynolds number. *Journal of Fluid Mechanics*, 13(1): 82–85, 1962. doi: 10.1017/S0022112062000518.

Remi Lam, Alvaro Sanchez-Gonzalez, Matthew Willson, Peter Wirnsberger, Meire Fortunato, Ferran Alet, Suman Ravuri, Timo Ewalds, Zach Eaton-Rosen, Weihua Hu, et al. Learning skillful medium-range global weather forecasting. *Science*, 382(6677):1416–1421, 2023. doi: 10.1126/science.adi2336.

Xiangtai Li, Ansheng You, Zhen Zhu, Houlong Zhao, Maoke Yang, Kuiyuan Yang, and Yunhai Tong. Semantic flow for fast and accurate scene parsing. In *European Conference on Computer Vision*, 2020.

Zijie Li, Dule Shu, and Amir Barati Farimani. Scalable transformer for PDE surrogate modeling. In *Advances in Neural Information Processing Systems*, volume 36, pp. 28010–28039, 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*, 2021a.

Zongyi Li, Hongkai Zheng, Nikola Kovachki, David Jin, Haoxuan Chen, Burigede Liu, Kamyar Azizzadenesheli, and Anima Anandkumar. Physics-informed neural operator for learning partial differential equations. *arXiv preprint arXiv:2111.03794*, 2021b.

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

Phillip Lippe, Bastiaan S. Veeling, Paris Perdikaris, Richard E. Turner, and Johannes Brandstetter. PDE-Refiner: Achieving accurate long rollouts with neural PDE solvers. *arXiv preprint arXiv:2308.05732*, 2023.

Xin-Yang Liu, Min Zhu, Lu Lu, Hao Sun, and Jian-Xun Wang. Multi-resolution partial differential equations preserved learning framework for spatiotemporal dynamics. *Communications Physics*, 7:31, 2024. doi: 10.1038/s42005-024-01521-z.

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.

Ziwei Liu, Raymond A. Yeh, Xiaoou Tang, Yiming Liu, and Aseem Agarwala. Video frame synthesis using deep voxel flow. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 4463–4471, 2017.

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.

Arda Mavi, Ali Can Bekar, Ehsan Haghighat, and Erdogan Madenci. An unsupervised latent/output physics-informed convolutional-LSTM network for solving partial differential equations using peridynamic differential operator. *Computer Methods in Applied Mechanics and Engineering*, 407: 115944, 2023. doi: 10.1016/j.cma.2023.115944.

Till Muser, Alexandra Spitzer, Matti Lassas, Maarten V. de Hoop, and Ivan Dokmanić. Flowers: A warp drive for neural PDE solvers. In *International Conference on Machine Learning*, 2026. Spotlight; arXiv:2603.04430.

Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8162–8171. PMLR, 2021.

Jacob Page. Super-resolution of turbulence with dynamics in the loss. *Journal of Fluid Mechanics*, 1002:R3, 2025. doi: 10.1017/jfm.2024.1202.

Yesom Park, Jaemoo Choi, Changyeon Yoon, Chang Hoon Song, and Myungjoo Kang. Learning PDE solution operator for continuous modeling of time-series. *arXiv preprint arXiv:2302.00854*, 2023.

T. S. Pedersen, P. K. Michelsen, and J. J. Rasmussen. Lyapunov exponents and particle dispersion in drift wave turbulence. *Physics of Plasmas*, 3(8):2939–2950, 1996. doi: 10.1063/1.871636.

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. FiLM: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, pp. 3942–3951, 2018. doi: 10.1609/aaai.v32i1.11671.

Md Ashiqur Rahman, Zachary E. Ross, and Kamyar Azizzadenesheli. U-NO: U-shaped neural operators. *Transactions on Machine Learning Research*, 2023.

Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.

Chengping Rao, Hao Sun, and Yang Liu. Embedding physics to learn spatiotemporal dynamics from sparse data. *Nature Machine Intelligence*, 5:765–779, 2023. doi: 10.1038/s42256-023-00685-7.

Bogdan Raonić, Roberto Molinaro, Tim De Ryck, Tobias Rohner, Francesca Bartolucci, Rima Alaifari, Siddhartha Mishra, and Emmanuel de Bézenac. Convolutional neural operators for robust and accurate learning of PDEs. In *Advances in Neural Information Processing Systems*, volume 36, 2023. doi: 10.52202/075280-3376.

Pu Ren, Chengping Rao, Yang Liu, Jian-Xun Wang, and Hao Sun. PhyCRNet: Physics-informed convolutional-recurrent network for solving spatiotemporal PDEs. *Computer Methods in Applied Mechanics and Engineering*, 389:114399, 2022. doi: 10.1016/j.cma.2021.114399.

Pu Ren, Chengping Rao, Yang Liu, Zihan Ma, Qi Wang, Jian-Xun Wang, and Hao Sun. PhySR: Physics-informed deep super-resolution for spatiotemporal data. *Journal of Computational Physics*, 492:112438, 2023. doi: 10.1016/j.jcp.2023.112438.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention*, pp. 234–241, 2015.

Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8934–8943, 2018.

Kiwon Um, Robert Brand, Yun Fei, Philipp Holl, and Nils Thuerey. Solver-in-the-loop: Learning from differentiable physics to interact with iterative PDE solvers. In *Advances in Neural Information Processing Systems*, 2020.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

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

Xintao Wang, Kelvin C. K. Chan, Ke Yu, Chao Dong, and Chen Change Loy. EDVR: Video restoration with enhanced deformable convolutional networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2019.

Gege Wen, Zongyi Li, Kamyar Azizzadenesheli, Anima Anandkumar, and Sally M. Benson. U-FNO: An enhanced fourier neural operator-based deep-learning model for multiphase flow. *Advances in Water Resources*, 163:104180, 2022. doi: 10.1016/j.advwatres.2022.104180.

Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European Conference on Computer Vision*, pp. 3–19, 2018. doi: 10.1007/978-3-030-01261-8_1.

Zhaoyi Yan, Xiaoming Li, Mu Li, Wangmeng Zuo, and Shiguang Shan. Shift-net: Image inpainting via deep feature rearrangement. *arXiv preprint arXiv:1801.09392*, 2018.

Lirong Yin, Lei Wang, Tingqiao Li, Siyu Lu, Zhengtong Yin, Xuan Liu, Xiaolu Li, and Wenfeng Zheng. U-net-stn: A novel end-to-end lake boundary prediction model. *Land*, 12(8):1602, 2023.

Niloufar Zakariaei, Siddharth Rout, Eldad Haber, and Moshe Eliasof. Advection augmented convolutional neural networks. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-1110.

Xuan Zhang, Jacob Helwig, Yuchao Lin, Yaochen Xie, Cong Fu, Stephan Wojtowytsch, and Shuiwang Ji. Sinenet: Learning temporal dynamics in time-dependent partial differential equations. In *International Conference on Learning Representations*, 2024.

Anthony Zhou and Amir Barati Farimani. Predicting change, not states: An alternate framework for neural PDE surrogates. *arXiv preprint arXiv:2412.13074*, 2024.

Yinhao Zhu, Nicholas Zabaras, Phaedon-Stelios Koutsourelakis, and Paris Perdikaris. Physics-constrained deep learning for high-dimensional surrogate modeling and uncertainty quantification without labeled data. *Journal of Computational Physics*, 394:56–81, 2019. doi: 10.1016/j.jcp.2019.05.024.

A BENCHMARK SYSTEMS, DATASETS, AND RESIDUALS

We consider Kolmogorov turbulence, buoyant smoke, and Hasegawa–Wakatani plasma turbulence. The following subsections report the governing equations, discretization, boundary conditions, state variables, data-generation procedures, data partitions, and system-specific residuals for all three benchmarks.

A.1 KOLMOGOROV FLOW

We use two-dimensional Kolmogorov flow as the primary test system. This canonical model describes an incompressible viscous fluid driven by a unidirectional, spatially periodic body force on the doubly periodic domain

$$\Omega = (0, 2\pi)^2. \tag{7}$$

The state is represented by the scalar vorticity

$$\omega = \partial_{x_1} u_2 - \partial_{x_2} u_1, \tag{8}$$

where $\mathbf{u} = (u_1, u_2)$ is the two-dimensional divergence-free velocity field. The nondimensional governing equations are

$$\frac{\partial \omega}{\partial t} + \mathbf{u} \cdot \nabla \omega = \frac{1}{\mathrm{Re}} \nabla^2 \omega - n_f \cos(n_f x_2) - \gamma \omega, \tag{9}$$

$$\nabla \cdot \mathbf{u} = 0, \tag{10}$$

with Reynolds number $\mathrm{Re} = 1000$, forcing wavenumber $n_f = 8$, and linear drag coefficient $\gamma = 0.1$ (Li et al., 2023a; Khrabry et al., 2026b;a). The velocity is recovered from the vorticity through the periodic streamfunction ψ,

$$-\nabla^2 \psi = \omega, \qquad \mathbf{u} = (\partial_{x_2} \psi, -\partial_{x_1} \psi)\,. \tag{11}$$

In our array convention, x_1 is the first spatial dimension and x_2 is the second, or last, spatial dimension. The sinusoidal term injects energy at forcing wavenumber n_f, while the linear drag limits the accumulation of energy at the largest spatial scales.

Numerical data. We use the Kolmogorov-flow dataset released with FactFormer (Li et al., 2023a). The source trajectories were generated using the modified pseudo-spectral Kolmogorov-flow solver distributed with the physics-informed neural operator codebase (Li et al., 2021b; 2023a). The underlying numerical simulation was run on a 2048×2048 grid with an integration time step of 10^{-4}. The released learning data are available as vorticity fields on a 256×256 periodic grid and at the larger saved-state interval

$$\Delta t_{\mathrm{data}} = \frac{1}{32} = 0.03125. \tag{12}$$

The fields used here are therefore reduced outputs of a substantially finer numerical simulation. We refer to them as high-resolution fields only relative to the additional spatially reduced observations constructed in our coarse-conditioned experiments.

The released array contains 120 trajectories, each comprising 320 stored vorticity fields,

$$\omega^n \in \mathbb{R}^{1 \times 256 \times 256}. \tag{13}$$

Initial vorticity fields were sampled from a prescribed Gaussian random field.

Train–test split and temporal sampling. We partition the 120 simulated trajectories at the trajectory level before constructing temporal examples. For each model initialization, 10 complete trajectories are sampled uniformly without replacement using a NumPy random-number generator initialized with the run seed; the remaining 110 trajectories form the training split. Both sets of trajectory indices are sorted and saved with the run metadata. This trajectory-level partition prevents temporal windows from the same simulated trajectory from appearing in both the training and test sets.

For a model invocation spanning N_{out} stored-data intervals and using one input state, each trajectory provides

$$N_{\mathrm{start}} = 320 - N_{\mathrm{out}} \tag{14}$$

valid starting indices. Every admissible starting index is first enumerated within its assigned trajectory split. When the number of available examples exceeds the prescribed experimental budget, we sample without replacement at most 4000 training examples and 50 test examples. These subsets are reproducible: the pair-selection random seeds are the run seed plus 10001 for training and plus 20001 for testing. Training examples are shuffled reproducibly between batches, whereas test examples are evaluated without shuffling.

A temporal example begins from ω^n and covers the physical interval

$$\Delta T = N_{\mathrm{out}} \Delta t_{\mathrm{data}} = \frac{N_{\mathrm{out}}}{32}. \tag{15}$$

For queried-time, direct block-prediction, and physics-informed models, the dataset returns the consecutive stored reference states inside this interval. For fixed-horizon endpoint prediction, the target is only the state at offset N_{out}.

Generation of test trajectories. After training, test trajectories are generated independently for all 10 held-out simulated trajectories. Each model rollout is initialized from the first stored vorticity field of the corresponding test trajectory. No later reference fields are supplied to the model during generation. The model is then applied autoregressively: after each invocation, its final predicted state replaces the current state and serves as the input to the next invocation.

The precise autoregressive update depends on the prediction regime. A fixed-horizon endpoint model advances by N_{out} stored intervals per invocation. A queried-time or direct block-prediction model returns N_{out} consecutive predictions in one call; all of these states are retained for evaluation, and the final state of the block initializes the next call. A physics-informed model generates M internal substeps per stored-data interval. Only every Mth internal state is compared with the stored reference data, while the final internal state of the complete block is supplied to the next autoregressive invocation. Thus, all models ultimately produce fields at the same saved-data checkpoints despite potentially using different numbers of complete model evaluations and internal predictions.

The standard test rollout contains 304 forecast states. Prediction step j is compared with stored state ω^j from the same held-out trajectory, while the supplied initial state ω^0 is excluded from forecast-error and trajectory-statistics calculations. The corresponding physical rollout duration is

$$T_{\mathrm{rollout}} = 304 \Delta t_{\mathrm{data}} = 9.5. \tag{16}$$

At each saved-data checkpoint, we compute the relative L_2 error between the generated and reference vorticity fields and average it over the 10 held-out trajectories. Before measuring inference time, three untimed model calls are performed to remove one-time model, accelerator, and memory-allocation overhead. Reported cumulative and incremental generation times are synchronized with the accelerator.

When trajectory statistics are reported, they are computed separately for each held-out rollout and then averaged over the 10 trajectories, rather than concatenating independent trajectories in time. The reported quantities include the time-averaged spatial Fourier spectrum, the spatially averaged temporal Fourier spectrum, and the normalized temporal autocorrelation. An example rollout from the spatially uniform training-observation regime and its spatial spectra appear in Appendix H.8.2.

A.2 Two-dimensional buoyant smoke flow

The third benchmark describes the buoyancy-driven motion of a light scalar species, referred to as smoke, in a closed two-dimensional domain. We use the fixed-force Navier–Stokes dataset introduced by Gupta & Brandstetter (2022). The system couples the incompressible Navier–Stokes equations to a transport equation for the smoke concentration:

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} = -\nabla p + \nu \nabla^2 \mathbf{u} + \mathbf{f}_b, \tag{17}$$

$$\nabla \cdot \mathbf{u} = 0, \tag{18}$$

$$\frac{\partial d}{\partial t} + \mathbf{u} \cdot \nabla d = 0. \tag{19}$$

Here, $\mathbf{u} = (u_x, u_y)$ is the velocity, p is pressure, d is the smoke concentration, and ν is the kinematic viscosity. The smoke is transported through pure advection, with no explicit scalar-diffusion term in the benchmark equations.

The vertically directed buoyancy force is proportional to the local smoke concentration:

$$\mathbf{f}_b(d) = \begin{bmatrix} 0 \\ \eta d \end{bmatrix}, \qquad \eta = 0.5. \tag{20}$$

The smoke concentration is therefore an active scalar rather than a passive tracer. The velocity transports and deforms the smoke field, while the concentration acts back on the velocity through buoyancy. Regions with larger smoke concentration experience stronger upward acceleration, producing rising plumes, shear layers, and vortical structures that subsequently alter scalar transport. This benchmark therefore differs from the single-field turbulent systems considered above by combining nonlinear momentum transport, viscous diffusion, pressure-enforced incompressibility, and two-way coupling between a vector flow field and an advected scalar.

The incompressible formulation assumes that the flow velocity is small relative to the speed of sound, so compressibility effects are neglected. Variations associated with the light scalar species enter the momentum equation through the buoyancy force rather than through a compressible mass-continuity equation.

Numerical configuration and boundary conditions. The equations were solved using the ΦFlow framework (Holl et al., 2020) on the closed square domain

$$\Omega = [0, 32] \times [0, 32]. \tag{21}$$

The fields were saved on a 128×128 Cartesian grid, corresponding to $\Delta x = \Delta y = 0.25$. The viscosity was $\nu = 0.01$, corresponding to $\mathrm{Re} = 100$ under the benchmark normalization, and the fixed buoyancy coefficient was $\eta = 0.5$.

Homogeneous Dirichlet boundary conditions were imposed on the velocity,

$$\mathbf{u}|_{\partial\Omega} = 0, \tag{22}$$

representing stationary walls. Homogeneous Neumann conditions were imposed on the smoke concentration,

$$\left.\frac{\partial d}{\partial n}\right|_{\partial\Omega} = 0, \tag{23}$$

where n denotes the outward boundary-normal direction. Each trajectory was initialized with zero velocity and a randomly sampled smoke-concentration field,

$$\mathbf{u}(\mathbf{x}, 0) = 0, \qquad d(\mathbf{x}, 0) = d_0(\mathbf{x}). \tag{24}$$

The trajectories consequently differ through their initial concentration distributions, while the physical parameters, domain, boundary conditions, buoyancy force, and saved-data interval remain fixed.

Learning dataset and trajectory partition. Each saved state contains the smoke concentration and two velocity components,

$$q(t) = \begin{bmatrix} d(t) \\ u_x(t) \\ u_y(t) \end{bmatrix} \in \mathbb{R}^{3\times128\times128}. \tag{25}$$

Pressure is used internally by the numerical solver to enforce incompressibility but is not retained as a machine-learning channel. The stored HDF5 fields `u`, `vx`, and `vy` correspond to d, u_x, and u_y, respectively.

The saved-data interval is $\Delta t_{\mathrm{data}} = 1.5$. Each learning trajectory contains nine consecutive saved states,

$$q^{(j)} \in \mathbb{R}^{9\times3\times128\times128}. \tag{26}$$

A single state is supplied as model input, consistently with the other systems considered in this work, and the model is evaluated over the following eight autoregressively generated states. The complete evaluated sequence therefore contains the initial state and eight predicted states and spans

$8\Delta t_{\text{data}} = 12$ time units. A representative equation-supervised rollout at stored-data checkpoints is shown in Appendix H.4.2.

Training and test sets are formed at the level of complete trajectories. For each seeded run, ten trajectories are sampled without replacement and held out for testing, while all remaining loaded trajectories are used for training. All temporal prediction pairs from a given trajectory remain in the same partition, preventing overlap between training and test windows. The selected trajectory indices are stored in the run metadata for reproducibility.

ΦFlow-matched physics loss. For physics-informed training, we construct a discrete update-consistency loss designed to approximate the numerical operations used in the ΦFlow data generator more closely than a generic continuous differential-equation residual. Because pressure is not present in the saved state, the residual does not attempt to recover the pressure field directly. Instead, it evaluates scalar advection, momentum advancement before pressure projection, incompressibility of the predicted velocity, and the curl-free character of the remaining velocity correction.

Let $q^m = (d^m, u_x^m, u_y^m)$ and $q^{m+1} = (d^{m+1}, u_x^{m+1}, u_y^{m+1})$ denote two consecutive generated states separated by δt. The cell-centered velocity channels are first converted to the staggered representation used by the discrete update. Smoke concentration and velocity are then advanced by semi-Lagrangian advection,

$$d_{\text{adv}} = \mathcal{A}_{\delta t}(d^m; \mathbf{u}^m), \tag{27}$$

$$\mathbf{u}_{\text{adv}} = \mathcal{A}_{\delta t}(\mathbf{u}^m; \mathbf{u}^m), \tag{28}$$

where $\mathcal{A}_{\delta t}$ denotes the corresponding transport operator with closed-domain boundary handling. Buoyancy is evaluated using the advected concentration and applied to the vertical velocity,

$$\mathbf{u}_b = \mathbf{u}_{\text{adv}} + \delta t \begin{bmatrix} 0 \\ \eta d_{\text{adv}} \end{bmatrix}. \tag{29}$$

The pre-projection velocity is then formed by applying the viscous update,

$$\mathbf{u}_{\text{pre}} = \mathbf{u}_b + \nu \delta t \, \nabla_h^2 \mathbf{u}_b, \tag{30}$$

where ∇_h^2 is a discrete Laplacian with zero-valued velocity outside the physical domain. The prescribed wall values are imposed after the viscous update.

In a projection method, the final incompressible velocity is obtained as

$$\mathbf{u}^{m+1} = \mathbf{u}_{\text{pre}} - \delta t \, \nabla_h p. \tag{31}$$

The pressure correction is therefore curl free, allowing pressure consistency to be tested without explicitly reconstructing p. We define the normalized residual components

$$R_{\text{div}} = \frac{L}{U} \nabla_h \cdot \mathbf{u}^{m+1}, \tag{32}$$

$$R_d = \frac{d^{m+1} - d_{\text{adv}}}{D}, \tag{33}$$

$$R_{\text{proj}} = \frac{L}{U} \nabla_h \times (\mathbf{u}^{m+1} - \mathbf{u}_{\text{pre}}), \tag{34}$$

where L, U, and D are characteristic length, velocity, and concentration scales. The pointwise smoke physics residual combines these terms with penalties enforcing the stationary-wall velocity values,

$$\mathcal{R}_{\text{smoke}} = R_{\text{div}}^2 + R_d^2 + R_{\text{proj}}^2 + R_{\text{wall}}^2. \tag{35}$$

The scalar physics loss is obtained by averaging $\mathcal{R}_{\text{smoke}}$ over the batch and spatial domain.

For one complete saved-data interval, the residual uses $\delta t = 1.5$. When the model generates t_{multiple} internal states within this interval, the residual is evaluated with $\delta t = 1.5/t_{\text{multiple}}$ between successive generated states. The residual is solver inspired rather than an exact replay of every internal operation performed during data generation because the saved states do not expose the intermediate advection, diffusion, and pressure-projection stages. Nevertheless, semi-Lagrangian transport, staggered velocity reconstruction, concentration-dependent buoyancy, viscous advancement, closed-wall handling, discrete incompressibility, and the pressure-projection structure are represented explicitly.

A.3 HASEGAWA–WAKATANI PLASMA TURBULENCE

The Hasegawa–Wakatani system is a reduced model of resistive drift-wave turbulence in a magnetized plasma with a background density gradient perpendicular to the confining magnetic field (Hasegawa & Wakatani, 1983; Khrabry et al., 2026b). We consider the normalized density perturbation $n(x, z, t)$ and electrostatic potential $\phi(x, z, t)$ in a two-dimensional plane perpendicular to the magnetic field. The normalized vorticity is related to the potential by

$$\omega = \nabla_\perp^2 \phi. \tag{36}$$

The production data were generated using the modified Hasegawa–Wakatani equations

$$\frac{\partial n}{\partial t} + [\phi, n] - C + \kappa \frac{\partial \phi}{\partial z} + D_n \nabla_\perp^4 n = 0, \tag{37}$$

$$\frac{\partial \omega}{\partial t} + [\phi, \omega] - C + D_\omega \nabla_\perp^4 \omega = 0, \tag{38}$$

$$\omega = \nabla_\perp^2 \phi, \tag{39}$$

where the resistive coupling is applied only to the nonzonal components,

$$\widetilde{n} = n - \langle n \rangle_z, \tag{40}$$

$$\widetilde{\phi} = \phi - \langle \phi \rangle_z, \tag{41}$$

$$C = \alpha(\widetilde{\phi} - \widetilde{n}). \tag{42}$$

Here, $\langle \cdot \rangle_z$ denotes the average over the periodic z direction, α is the resistive or adiabatic-response coupling parameter, κ is the normalized background density-gradient drive, and D_n and D_ω are density and vorticity hyperdiffusion coefficients. The Poisson bracket follows the coordinate convention used by the numerical solver,

$$[f, g] = \frac{\partial f}{\partial z}\frac{\partial g}{\partial x} - \frac{\partial f}{\partial x}\frac{\partial g}{\partial z}. \tag{43}$$

Although velocity is not evolved as an independent field, the equations contain nonlinear convective transport. In the adopted normalization, the electrostatic potential determines an incompressible $\mathbf{E} \times \mathbf{B}$ drift velocity, up to the coordinate-orientation convention, and the Poisson-bracket terms describe advection of density and vorticity by this potential-generated flow. The background density gradient supplies free energy for the resistive drift-wave instability. After saturation, nonlinear $\mathbf{E} \times \mathbf{B}$ advection redistributes fluctuation energy among modes and spatial scales, including transfer into large-scale, approximately zonally symmetric structures. Because the modified coupling acts on nonzonal drift-wave fluctuations but does not directly damp the zonal components, coherent zonal flows can develop and regulate turbulent cross-field transport. This drift-wave–zonal-flow self-organization is a characteristic mechanism in reduced models of magnetically confined plasma dynamics. The normalized parameters used below characterize the reduced model and should not be interpreted as dimensional operating parameters of a particular confinement device.

Physical parameters. The data used in this work were generated with

$$\alpha = 0.1, \qquad \kappa = 0.5, \qquad D_n = D_\omega = 10^{-4}, \tag{44}$$

using the modified coupling in Eq. (42). Small positive hyperdiffusion coefficients suppress grid-scale accumulation while preserving the multiscale turbulent dynamics.

BOUT++ data generation. The equations were solved using BOUT++ (Dudson et al., 2009) on the doubly periodic physical domain

$$(x, z) \in [0, 64) \times [0, 64). \tag{45}$$

The physical mesh contained 256×256 cells with $\Delta x = \Delta z = 0.25$. The internal BOUT++ mesh contained 260 cells in the distributed x direction because two guard cells were included on each side of the 256 physical cells. The z direction contained 256 periodic cells, and the singleton BOUT++ parallel direction contained one cell. The calculation was distributed over 16 MPI ranks in x. The

machine-learning arrays were reconstructed by ordering the rank-local files by processor index, combining their physical subdomains, and removing local guard cells.

The stored metric coefficients were uniform and Cartesian. The continuous perpendicular Laplacian consequently reduces to $\partial_x^2 + \partial_z^2$, but the numerical operator used by BOUT++, denoted L_B, was a hybrid finite-difference–spectral operator,

$$(L_B f)_{i,k} = \frac{f_{i-1,k} - 2f_{i,k} + f_{i+1,k}}{\Delta x^2} + \mathcal{F}_z^{-1}\left[-k_z^2 \widehat{f}_i(k_z)\right]. \tag{46}$$

Thus, the x contribution used second-order centered finite differences, whereas the periodic z contribution used Fourier-spectral second differentiation. The hyperdiffusion terms were evaluated by applying this same operator twice, $L_B^2 f = L_B(L_B f)$. The calculation therefore used neither a fully spectral two-dimensional Laplacian nor a conventional finite-difference five-point Laplacian.

The density-gradient term used the second-order centered derivative selected by BOUT++,

$$(D_z\phi)_{i,k} = \frac{\phi_{i,k+1} - \phi_{i,k-1}}{2\Delta z}. \tag{47}$$

The nonlinear terms used the conservative Arakawa discretization of the Poisson bracket with the sign convention in Eq. (43). BOUT++ evolved n and ω, while ϕ was obtained at every right-hand-side evaluation by solving the discrete elliptic equation

$$L_B\phi = \omega. \tag{48}$$

Time integration used the adaptive PVODE solver with relative and absolute tolerances of 10^{-5} and 10^{-12}, respectively. The native BOUT++ output interval was

$$\Delta t_{\mathrm{BOUT}} = 0.1. \tag{49}$$

The complete output contained 50,001 saved states from $t = 0$ to approximately $t = 5000$. The first production job reached its wall-time limit, after which a restart-and-append continuation completed the trajectory. The restart did not change the equations, parameters, spatial discretization, or time-integration settings.

Exported fields and temporal segmentation. The exported machine-learning state contains density and electrostatic potential,

$$u(t) = \begin{bmatrix} n(t) \\ \phi(t) \end{bmatrix} \in \mathbb{R}^{2\times 256\times 256}, \tag{50}$$

and the full exported array has shape $(50001, 2, 256, 256)$ with dimension order (t, c, x, z) and single-precision storage. Vorticity is not retained as a separate learning channel and is reconstructed when needed as $\omega = L_B\phi$.

Because a doubly periodic Laplacian has a constant null mode, the BOUT++ inversion imposes a numerical gauge condition at one physical x location. This produces a small, fixed, one-row discrepancy between the independently evolved vorticity and $L_B\phi$. We omit this time-independent discrepancy when reconstructing vorticity because it is an inversion-gauge artifact rather than a physical term in the Hasegawa–Wakatani equations.

The learning dataset was constructed from the late-time portion of this single long simulation. We first retained every fifth BOUT++ output, giving the effective learning interval

$$\Delta t_{\mathrm{data}} = 5\Delta t_{\mathrm{BOUT}} = 0.5. \tag{51}$$

The first 5,000 states of the temporally subsampled sequence were omitted, and the following 5,000 states were retained. In terms of the native BOUT++ output, these data extend from approximately $t = 2500$ to $t = 4999.5$. This removes the initial instability growth, nonlinear saturation, and early transient evolution, restricting the learning data to the late-time turbulent regime.

The retained sequence was divided into ten contiguous, nonoverlapping temporal segments of 500 states each, producing a learning array of shape

$$(10, 500, 2, 256, 256). \tag{52}$$

Each segment spans 499 intervals, or 249.5 normalized time units. These segments are trajectory-like units for the training pipeline, but they are not independently initialized BOUT++ realizations: all ten originate from one continuous statistically developed simulation. Segmenting the long sequence provides nonoverlapping temporal blocks for model training and evaluation while preserving local temporal continuity.

Training and test partition. The ten temporal segments were partitioned before temporal prediction pairs were constructed. For each run, four complete segments were sampled without replacement for testing using a seeded random-number generator, and the remaining six segments were used for training. All windows from a segment remained in the same partition, preventing direct overlap between training and test windows. The selected indices were stored in the run metadata. The particular segment assignment could change with the run seed; details are retained in the run records rather than emphasized in the main comparison.

For example, one representative $n_t = 4$ run used training segments $\{0, 1, 3, 5, 6, 7\}$ and held-out segments $\{2, 4, 8, 9\}$. With one input history state, each 500-state segment contains $500-1-4+1 = 496$ valid starts, giving 2,976 available training pairs and 1,984 available test pairs. All training pairs were used in that run, while 50 test pairs were sampled for epoch-level evaluation. This test-pair cap does not apply to the final autoregressive rollout.

Autoregressive evaluation. Final evaluation was performed independently on all four held-out segments. Each rollout began from the first state of a held-out segment and was advanced autoregressively without access to later reference states. The standard rollout contained 256 predicted states, corresponding to

$$T_{\mathrm{rollout}} = 256\Delta t_{\mathrm{data}} = 128 \tag{53}$$

normalized time units. For direct multi-step prediction with local horizon n_t, one call returns n_t consecutive future states; every state is retained for evaluation and the final state initializes the next call. A 256-state rollout therefore requires $256/n_t$ complete calls when n_t divides 256.

Because the Hasegawa–Wakatani dynamics are chaotic, sufficiently long generated trajectories are not expected to retain pointwise correspondence with a particular reference realization. Here, one prediction step at $n_t = 1$ corresponds to 0.5 normalized time units, whereas the dimensionless Lyapunov time is shorter than one normalized time unit (Pedersen et al., 1996). We therefore complement finite-horizon pointwise rollout errors with statistical comparisons. Spatial Fourier spectra, temporal Fourier spectra, and temporal autocorrelations are computed independently for each held-out rollout and then averaged over the four held-out segments. This per-segment calculation avoids artificial temporal correlations across boundaries between distinct portions of the segmented sequence. A representative field rollout and its model-level spatial spectra are shown in Appendix H.4.1.

BOUT++-matched physics residual. For physics-informed experiments, the discrete residual was designed to reproduce both the modified Hasegawa–Wakatani configuration and the numerical spatial operators of the BOUT++ data-generating solver to a high degree. The Hasegawa–Wakatani residual uses $\Delta t_{\mathrm{data}} = 0.5$, $\alpha = 0.1$, $\kappa = 0.5$, $D_n = D_\omega = 10^{-4}$, and the same modified coupling used in the simulation. Zonal averages are removed along the final spatial dimension before the resistive coupling is evaluated.

The residual uses a direct PyTorch translation of the BOUT++ Arakawa bracket, including its sign and array-orientation conventions. A numerical comparison of the translated stencil with the corresponding BOUT++ expression agreed to floating-point roundoff on test fields. The residual also reproduces the BOUT++ centered derivative for the density-gradient term and the hybrid perpendicular Laplacian in Eq. (46). The biharmonic terms are evaluated by applying this hybrid operator twice. Vorticity and its time derivative are reconstructed as

$$\omega = L_B\phi, \qquad \frac{\partial\omega}{\partial t} = L_B\left(\frac{\partial\phi}{\partial t}\right). \tag{54}$$

The residual between two generated states separated by δt uses the temporal midpoint state

$$u_{1/2} = \frac{u_{\mathrm{prev}} + u_{\mathrm{next}}}{2} \tag{55}$$

and the finite-difference derivative

$$\left.\frac{\partial u}{\partial t}\right|_{1/2} \approx \frac{u_{\mathrm{next}} - u_{\mathrm{prev}}}{\delta t}. \tag{56}$$

When the model generates multiple internal states within one stored-data interval, δt is reduced according to the number of internal substeps. This construction closely matches the generating

equations and their spatial discretization, but it is not the internal algebraic residual of adaptive PVODE. The saved fields do not expose PVODE's internal stages, so a nonzero residual can remain because of temporal sampling and midpoint truncation error. The small gauge-row discrepancy from the periodic potential inversion is also deliberately omitted because it is numerical rather than physical.

B Expanded Related Work and Novelty Positioning

This appendix expands the literature positioning summarized in Section 2. We distinguish the individual ingredients of our method from their specific placement and combination in the proposed model. Learned warping, hierarchical processing, temporal querying, physics-informed training, and super-resolution are established ideas. Our claims concern decoder-conditioned alignment of input-time skip features, efficient temporal resolution within an outer forecast interval, and equation-supervised operator learning when both future targets and complete high-resolution training states are withheld.

B.1 Multiscale and hierarchical PDE architectures

Convolutional encoder–decoders. U-Net architectures construct multiresolution representations through a contracting encoder and an expanding decoder, with equal-resolution skip connections transferring localized information that may be degraded during downsampling (Ronneberger et al., 2015). This organization combines short computational paths for fine spatial detail with coarse representations whose receptive fields cover larger portions of the domain. Convolutional U-Nets have consequently been applied directly to field prediction, including velocity and pressure solutions for incompressible flows around parameterized geometries (Chen et al., 2019). For fixed kernel sizes, channel widths, and hierarchy depth, their computational cost scales linearly with the number of grid points. In dynamical forecasting, however, the encoder and decoder no longer necessarily represent the same effective time. The hierarchy and its skip connections preserve and transfer spatial detail, but do not by themselves ensure that equal-resolution features remain spatially correspondent across a forecast interval.

Attention-based hierarchies. Global self-attention in transformer models provides direct nonlocal interaction, but its computational and memory costs are quadratic in the number of spatial tokens (Vaswani et al., 2017). This scaling is restrictive for high-resolution physical fields. Swin Transformers instead organize tokens hierarchically and restrict self-attention to local windows (Liu et al., 2021). Like convolutions, an individual windowed-attention block has limited spatial reach. Alternating shifted windows allow information to cross previous window boundaries, while patch merging and expansion enable communication over progressively larger regions. For a fixed window size and channel width, the resulting attention cost is linear in the number of spatial tokens. Swin U-Net places these blocks within a U-shaped encoder–decoder and uses equal-resolution skips to transfer fine spatial information (Cao et al., 2023). PDE-specific transformer architectures similarly combine local attention, multiresolution token processing, and hierarchical fusion (Holzschuh et al., 2025). Our convolutional and Swin backbones therefore share the same broad encoder–decoder hierarchy while using different local within-level mixers. Applying the same alignment mechanism to both tests whether its benefit arises from restoring skip correspondence rather than from one particular mixing operation.

Hierarchical neural operators. U-shaped and multigrid constructions also appear in neural operator architectures. U-NO uses a contracting and expanding operator architecture to support deeper and more memory-efficient solution operators, including spatiotemporal Navier–Stokes prediction (Rahman et al., 2023). U-FNO appends a compact U-Net path to later Fourier layers, combining global spectral mixing with local multiscale processing for multiphase-flow prediction (Wen et al., 2022). MgNO parameterizes learned operator maps through a multigrid structure in which restriction, coarse-grid processing, and prolongation are components of the neural operator (He et al., 2024). These works demonstrate the value of hierarchical processing in operator learning, but they do not specifically transport input-time skip features before fusion with a temporally advanced decoder. Convolutional Neural Operators also use multiresolution convolutional processing, with an emphasis on alias-controlled and discretization-consistent operator learning (Raonić et al., 2023). CNO is

therefore a relevant convolutional-operator baseline while leaving open the narrower question of whether future-oriented decoder features should receive conventionally indexed or explicitly aligned encoder skips.

Fourier operators and fine-scale content. Fourier Neural Operators communicate globally through learned transformations of retained Fourier modes (Li et al., 2021a). The Fourier transforms used for this spectral mixing have $\mathcal{O}(N \log N)$ complexity for N grid points, providing efficient global communication without a deep sequence of local operations. Practical implementations nevertheless retain only a finite collection of modes. This truncation limits the high-frequency information communicated through the global spectral path, although local channel mixing and nonlinear layers may still represent or generate fine-scale content. The limitation is particularly relevant to turbulent forecasting, where localized gradients, filaments, and small-scale fluctuations influence both instantaneous field error and autoregressive stability. Fourier and hierarchical architectures therefore provide complementary mechanisms for spatial communication. FNOs mix retained modes globally, whereas encoder–decoder hierarchies combine local processing, resolution changes, and skip transfer to preserve fine detail while communicating across scales. U-FNO explicitly combines these mechanisms (Wen et al., 2022). Our contribution addresses a crucial issue within hierarchical forecasting models that use equal-resolution encoder–decoder skips: whether the input-time features passed through those skips remain spatially correspondent with the future-oriented representations that receive them.

B.2 Learned feature alignment and transport

Temporal mismatch in hierarchical PDE predictors. For static reconstruction or segmentation, equal-resolution encoder and decoder features describe the same underlying input. In time-dependent prediction, the encoder is constructed from the current state while the decoder represents a future state. Features associated with transported or deformed structures may therefore occupy different grid locations even when their tensors have the same spatial resolution.

SineNet provides the most direct prior analysis of this issue (Zhang et al., 2024). It identifies temporal misalignment in U-Net skip connections as a limitation of PDE forecasting and replaces one deep U-shaped transition with several sequential U-shaped waves. Each wave advances the features over a smaller portion of the total transition, reducing the temporal change that must occur within an individual wave. Increasing the number of waves at a comparable parameter budget improves performance in the reported experiments.

HEAP avoids a conventional encoder–decoder temporal mismatch by separately encoding a hierarchy of scale-specific latent fields and advancing all levels synchronously with a hierarchical predictor (Khrabry et al., 2026b). DW-Net uses both intra-wave and cross-wave skip connections so that multiscale representations persist and are progressively refined across stacked U-shaped waves (Khrabry et al., 2026a). These architectures reorganize the temporal evolution of the hierarchy. Our method instead retains an end-to-end U-Net or Swin U-Net and explicitly samples the input-time skip at locations predicted by the future-oriented decoder before each fusion.

Learned transformations and deformable sampling. Related learned transformations and sampling offsets have been explored primarily in computer vision. Spatial Transformer Networks introduced differentiable input-conditioned transformations through a predicted transformation, a sampling grid, and interpolation (Jaderberg et al., 2015). Deformable convolutions predict offsets for individual kernel samples, adapting the local receptive field to the input geometry (Dai et al., 2017). In dense prediction, AlignSeg, SFNet, and FaPN align cross-resolution representations before aggregation using learned interpolation, semantic-flow fields, or deformable sampling (Huang et al., 2021b; Li et al., 2020; Huang et al., 2021a). These methods address geometric or cross-resolution mismatch within static-output vision problems. They provide methodological precedents for learned feature correspondence but do not address the effective-time difference between encoder and decoder features when forecasting an unobserved physical state.

Other U-shaped models use related operations for different purposes. Shift-Net searches and rearranges encoder features to restore image content in missing regions (Yan et al., 2018). Deformable U-Net variants adapt sampling within feature-extraction blocks (Duan et al., 2025). Registration models such as VoxelMorph predict a dense displacement that warps an input or output image (Balakrishnan et al., 2019), while U-Net–STN applies a learned spatial transformation to a predicted

physical field in output space (Yin et al., 2023). These operations differ from aligning the internal input-time skip immediately before it is fused with a future-state decoder representation.

Alignment across observed times. Video models align information across frames to exploit temporal correspondence. NetWarp uses optical flow between adjacent observed frames to warp internal CNN representations before video segmentation (Gadde et al., 2017). PWC-Net estimates correspondence coarse-to-fine using feature pyramids, warping, and cost volumes (Sun et al., 2018), while EDVR uses pyramidal and deformable feature alignment for video restoration under large motion (Wang et al., 2019). Deep Voxel Flow predicts sampling fields that transport observed content to interpolated or extrapolated frames (Liu et al., 2017).

These methods demonstrate that cross-time feature warping can improve correspondence. Their alignment is generally estimated from two or more observed frames, or is used to synthesize output pixels directly. In our forecasting setting, the target physical field is unavailable. A future-oriented decoder representation must instead infer where useful input-time encoder information should be sampled.

Transport-aware PDE architectures. Learned transport and warping remain comparatively limited as architectural components for PDE forecasting, with Advection-Augmented CNNs and Flowers providing the closest comparisons considered here. Advection-Augmented CNNs incorporate learned semi-Lagrangian transport into repeated reaction–diffusion–advection updates (Zakariaei et al., 2024). A state-dependent displacement transports the evolving feature field, while learned reaction and diffusion operations model local channel interaction and smoothing. Its displacement advances the current feature stream rather than reconciling a stored input-time skip with the future-oriented decoder feature that receives it.

Flowers makes learned warping the principal spatial-mixing operation inside a multilevel U-Net-like architecture (Muser et al., 2026). Each block predicts multiple displacements, samples projected feature values at the displaced coordinates, and combines the resulting heads. Warping is therefore used throughout the encoder, bottleneck, and decoder as an alternative to convolutional or attention-based spatial mixing, while conventional same-resolution skip concatenations provide the multiscale scaffold.

Our distinction is the location and conditioning of the operation: each future-oriented decoder level predicts how its corresponding input-time encoder feature should be sampled immediately before skip fusion. Standard convolution or shifted-window attention remains the within-level mixer, and alignment is applied to restore correspondence between the two sides of a temporally mismatched skip. The operation therefore asks where information useful to a particular future decoder position is located in the input-time encoder tensor.

Geometry deformation versus temporal correspondence. Learned deformation also appears in neural operators. Geo-FNO maps an irregular physical domain to a regular latent grid so that Fourier layers can operate efficiently (Li et al., 2023b). Its deformation addresses domain geometry and computational representation rather than correspondence between input-time and forecast-time representations.

The offsets predicted by our model are interpreted as latent correspondence fields. They may correlate with physical transport, particularly in strongly advective systems, but they are not constrained to equal a material flow map. A latent channel may encode an interface, orientation, phase, gradient, or other learned attribute whose useful source location differs from the trajectory of a physical fluid parcel. We therefore claim future-decoder-conditioned skip alignment, not recovery of the physical velocity or flow map from latent displacements.

B.3 Multi-state, queried-time, and continuous-time prediction

Fixed transitions and temporal bundling. Many neural PDE surrogates learn a fixed-step state transition and extend it autoregressively. This creates a mismatch between training on reference inputs and deployment on model-generated states. Solver-in-the-loop training exposes learned components to recurrent solver states (Um et al., 2020). Message-passing neural PDE solvers use pushforward training and temporal bundling, in which one model invocation returns several future

states (Brandstetter et al., 2022). PDE-Refiner repeatedly corrects individual forecasts and emphasizes weak spatial-frequency components (Lippe et al., 2023). These methods improve rollout stability or reduce the frequency of complete autoregressive calls, while retaining a prescribed set of output times.

Direct multi-state prediction in our experiments belongs to this broad temporal-bundling family. A final projection jointly emits several discrete future states. We do not claim novelty for this output parameterization. Instead, we use it to test whether explicit skip alignment becomes especially important when one hierarchy must represent several future states.

Queried-time operators. Queried-time models condition a predicted state on an explicit temporal coordinate. DeepONet provides a natural construction because its trunk evaluates the output function at requested coordinates (Lu et al., 2021). Physics-informed DeepONets combine coordinate-query operator prediction with governing-equation residuals and can learn families of time-dependent PDE solutions with limited or absent paired trajectories (Wang et al., 2021). Temporal neural operators and GraphDeepONet also use explicit temporal conditioning for time-dependent operator learning and temporal extrapolation (Diab & Al-Kobaisi, 2025; Cho et al., 2024).

Continuous-in-time Fourier operators map inputs to requested times without iterating over a prescribed temporal grid (Park et al., 2023). Poseidon uses time-conditioned normalization within a multiscale operator transformer and exploits semigroup structure for continuous-time evaluation and scalable pretraining (Herde et al., 2024). These methods establish time conditioning and arbitrary-time querying as prior art.

Our queried-time architecture is designed around computational reuse within a U-shaped field-to-field model. The encoder, bottleneck, and lower decoder are evaluated once for an outer interval. Their features are shared across requested offsets, while only the highest-resolution decoder branch is reevaluated for each query. The temporal query modulates this branch, including the displacement used to align its skips. This design does not assume that lower-resolution dynamics are intrinsically independent of time; it is an efficiency-oriented restriction whose adequacy is evaluated empirically.

Derivative and whole-response alternatives. A different family learns an instantaneous vector field, $\partial_t u = f_\theta(u, t)$, and delegates time advancement to a numerical integrator. Neural Ordinary Differential Equations established this continuous-time paradigm for learned hidden dynamics (Chen et al., 2018), and PDE-specific variants learn state-space or operator-valued derivatives for numerical integration (Zhou & Barati Farimani, 2024). Such models offer adaptive temporal evaluation but require repeated derivative calls through an integration procedure.

Whole-response operators instead predict a prescribed temporal object. PINO commonly uses a three-dimensional FNO that mixes spatial and temporal modes over a finite-horizon space–time field (Li et al., 2021b). The Laplace Neural Operator represents transient responses through learned pole–residue structure in the Laplace domain (Cao et al., 2024). These approaches differ from repeated moderate outer blocks: they represent a larger temporal response directly rather than returning selected states from an autoregressively extensible transition.

Our model accepts a continuous-valued offset but is trained and evaluated on sampled fields. Claims of temporal interpolation should therefore be supported by evaluation at offsets excluded from training. The architectural contribution is limited to shared-trunk evaluation, late temporal conditioning, and its integration with decoder-predicted skip alignment.

Intermediate predictions for equation supervision. Intermediate queried or bundled states also provide a discretization of the evolution inside one outer forecast interval. Adjacent predictions can define shorter finite-difference residual intervals without shortening the interval advanced by a complete shared hierarchy. This separates three quantities that are otherwise coupled in a conventional fixed-step model: the outer autoregressive interval, the temporal spacing of returned outputs, and the interval used to evaluate equation consistency.

Physics-informed operator and coordinate-query methods already evaluate equation residuals at temporal collocation points (Raissi et al., 2019; Wang et al., 2021; Li et al., 2021b). Our distinction is not temporal collocation itself. It is the use of intermediate states produced by one aligned hierarchical transition, with most spatial computation shared across queries, to define the adjacent

residual intervals. Random subsampling of candidate intervals is an implementation strategy for reducing repeated finest-branch evaluation and is described in the method and experimental protocol rather than claimed as a separate conceptual contribution.

B.4 PHYSICS-INFORMED LEARNING AND TRAINING-INFORMATION REGIMES

Equation supervision without future-state targets. Physics-informed learning incorporates governing equations into the training objective. PINNs represent a solution as a differentiable coordinate function and impose equation, initial, and boundary residuals at collocation points (Raissi et al., 2019). Their basic formulation learns an individual solution instance rather than a reusable field-to-field transition across varying initial conditions.

Grid-based physics-informed learning has a substantial convolutional and recurrent lineage. Physics-constrained encoder–decoder surrogates have been trained without labeled solution outputs by incorporating governing equations into deterministic or probabilistic objectives (Zhu et al., 2019). Autoregressive dense encoder–decoders have learned nonlinear transient dynamics from discrete equation constraints, including chaotic and shock-forming systems (Geneva & Zabaras, 2020). PhyCRNet combines an encoder–decoder ConvLSTM with discrete PDE residuals and autoregressive updates (Ren et al., 2022), while related models enforce physical consistency in latent and output spaces (Mavi et al., 2023). Physics may also be embedded directly into recurrent convolutional updates or multiresolution residual blocks (Rao et al., 2023; Liu et al., 2024). These works establish label-free grid-based prediction, recurrent equation supervision, and architectural encoding of physical structure.

Our equation-supervised model belongs to this broad family. Its distinction is that a hierarchical transition produces several intermediate states inside one outer interval, and adjacent states define shorter residual intervals. Skip features used to construct those states are aligned to their future-oriented decoder representations.

Hybrid and physics-only operator learning. PINO extends residual-based training to neural operators (Li et al., 2021b). Its hybrid setting combines coarse-resolution solution examples with equation residuals evaluated at a finer resolution. For evolutionary PDEs, the coarse solution examples supply supervised input-to-solution information: an input or initial condition is associated with its subsequent coarse solution response. Fine-grid residual evaluation regularizes the operator and supports evaluation beyond the supervised discretization, but the temporal evolution remains represented in the training solution examples.

PINO also reports physics-only operator learning without solver-generated solution trajectories. In its Kolmogorov-flow physics branch, complete Gaussian-random-field initial conditions are sampled directly on the high-resolution PDE grid and enforced through an initial-condition loss, while a space–time FNO predicts the ensuing solution block. Thus, "no data" in this setting means no solver-generated future trajectory targets. It does not mean that the initial state is observed only coarsely or that its fine-scale realization is absent.

This distinction separates two information reductions. Future-state targets can be removed while retaining complete high-resolution initial conditions. Alternatively, high-resolution labels can be removed from a reconstruction problem while sparse or coarse observations of the evolving trajectory remain available. Our strongest regime combines the absence of future targets with the absence of a complete high-resolution training state.

Supervised turbulent super-resolution. Supervised super-resolution methods learn a mapping from reduced fields to their corresponding high-resolution realizations. Convolutional and multiscale skip-connected networks reconstruct cylinder-wake and homogeneous turbulence fields from severely under-resolved snapshots (Fukami et al., 2019). Spatiotemporal extensions combine spatial super-resolution with temporal inbetweening to recover high-resolution fields from inputs that are coarse in both space and time (Fukami et al., 2021). These methods demonstrate that learned spatial priors can reconstruct turbulent structure more effectively than ordinary interpolation, but require paired low- and high-resolution realizations during training.

Physics-informed super-resolution from temporal observations. Physics-informed reconstruction can reduce or eliminate corresponding high-resolution training labels. Kelshaw et al. map sparse observations to high-resolution fields and constrain the reconstructed temporal sequence using observation consistency and a high-resolution PDE residual (Kelshaw et al., 2022). The method demonstrates reconstruction of chaotic Kolmogorov flow without ordinary high-resolution target fields. Its inputs are nevertheless temporally indexed sparse observations from the trajectory being reconstructed. Observations at known times constrain the reconstructed states at those times, while the governing equation couples the states across the observation window.

PC-CNN extends physics-constrained reconstruction to nonlinear spatiotemporal inverse problems using a time-windowing formulation (Kelshaw & Magri, 2024). It reconstructs turbulent high-resolution solutions from sparse spatiotemporal information and demonstrates physically plausible spectral content. The problem remains reconstruction of a particular spatiotemporal realization from measurements distributed over that realization. Later observations in the window provide information about the evolution following its initial time.

PhySR instead begins from an available coarse spatiotemporal solution (Ren et al., 2023). It separates temporal refinement from spatial reconstruction using convolutional-recurrent and residual upsampling modules, with governing equations and hard boundary conditions regularizing the refined sequence. The future coarse states are therefore part of the supplied input sequence rather than quantities inferred autonomously from one current observation.

Page uses a differentiable flow solver to constrain a reconstructed state by requiring its advancement to reproduce later coarse observations (Page, 2025). This removes direct high-resolution reconstruction targets but still uses future coarse information to identify a dynamically consistent fine-scale state. Such future-observation consistency is powerful for inverse problems, but it is unavailable in an autonomous forecast from a single current state.

These works establish physics-informed high-resolution reconstruction under substantially reduced spatial information. They do not, however, establish the same operator-learning problem considered here. A reconstruction method infers one trajectory using observations distributed over that trajectory. Our training procedure learns model parameters intended to advance previously unseen states without receiving observations from their subsequent evolution.

Training from reduced current states. In our reduced-observation experiments, each training example supplies a spatially reduced current state. The corresponding full-resolution current realization is withheld from the model input and is not enforced through a high-resolution initial-condition loss. No future coarse observation or future high-resolution state from that example is used as a target. The loss is constructed from the governing equation evaluated along states generated by the model.

This information regime creates a partial-observability problem. Many high-resolution states share the same block averages, and one coarse current observation generally does not uniquely identify the unresolved phases and structures. Equation consistency alone can restrict admissible evolution but need not select the particular hidden realization associated with the original pooled field.

We therefore introduce a persistent prediction-derived unresolved component. For each training example, the component removed by pooling and restoration is extracted from a model-generated state and retained for later presentations of that example. The stored component is detached from the computation graph and updated as optimization proceeds. It is not obtained from the withheld full-resolution current state, a future target, or a prescribed spectrum-matched completion. The model's evolving predictions instead construct the missing spatial content used in subsequent training inputs.

The cache changes the effective training-state distribution over optimization. Initially, inputs may contain only their resolved coarse component and, in the spatially uniform case, a small white-noise symmetry-breaking perturbation. Model predictions then generate unresolved structure, which is carried into later training iterations and refined together with the transition model. This mechanism is related in motivation to temporal memory and latent-state estimation, but its role is specifically to construct unresolved training-state content that was absent from the supplied observations.

Scope of the reduced-observation claim. Our reported experiment concerns the information required to train a full-resolution forecasting model. At evaluation, autoregressive rollouts begin

from full-resolution held-out states. The experiment therefore asks whether accurate full-resolution dynamics can be learned when future targets and spatially resolved training inputs are unavailable. It does not ask the model to infer the exact high-resolution realization associated with a new coarse test observation.

This scope differs from physics-informed trajectory reconstruction. Reconstruction methods receive temporally distributed observations and infer the missing entries or scales of that observed realization. Our model receives no observations of the subsequent evolution but is not required, in the reported evaluation, to identify a hidden test-time realization from one coarse measurement. These are complementary problems rather than a strict ordering of difficulty.

A future deployment study could combine both requirements: initialize from a coarse held-out observation, infer a conditional high-resolution state, and forecast autonomously without later observations. Such an experiment would test both state identification and transition learning. The present claims are restricted to transition learning under reduced training-state information.

Terminology. We use the following distinctions throughout the paper:

- *Supervised operator learning* uses initial-to-future solution examples as training targets.
- *Hybrid data–physics training* combines supervised solution examples with equation residuals, potentially evaluated at a finer resolution.
- *Physics-only or target-free operator learning* omits future solution targets but may still supply complete high-resolution initial conditions.
- *High-resolution-target-free reconstruction* omits corresponding high-resolution labels but may use coarse or sparse observations at multiple times from the trajectory being reconstructed.
- *Reduced-current-state operator learning*, as studied in our reported experiments, omits future targets, future coarse observations, and complete high-resolution training inputs, but uses spatially reduced current states and the governing equations.
- *Trajectory-free coarse-only training* is reserved for settings in which coarse initial fields are generated procedurally and no solver-generated solution trajectories or complete high-resolution initial realizations are supplied.

These terms describe the information supplied to training and should not be conflated with the resolution at which the network is evaluated. In particular, physics-only training is not necessarily coarse-only training, and fine-grid residual evaluation is not equivalent to reconstructing information that was removed from the supplied initial observation.

C Architecture and spatial-alignment implementation details

C.1 Convolutional backbone

All principal aligned and unaligned convolutional U-Net models use one input state and the same system-dependent physical channels for input and prediction. Kolmogorov flow has one channel, Hasegawa–Wakatani turbulence has two, and smoke has three. Each predicted state has the same number of physical channels as its input state. Endpoint models return one state, queried-time models return one state per query, and direct multi-state models use a time-major projection with KC channels, which is reshaped into K states with C channels each.

The convolutional hierarchy uses residual encoder, decoder, and bottleneck feature blocks. Each encoder feature block contains one 3×3 convolution, each decoder feature block contains two 3×3 convolutions, and the bottleneck contains two 3×3 convolutions. Every feature convolution is followed by one-group Group Normalization and GELU activation. When input and output channel counts differ, the residual branch uses a 1×1 projection; otherwise, it uses the identity. GELU is applied after residual addition.

Downsampling uses a 2×2, stride-2 convolution followed by one-group Group Normalization and GELU. Upsampling uses a 2×2, stride-2 transpose convolution. At each decoder level, equal-resolution encoder and decoder representations are concatenated before the decoder feature block.

Channel width doubles after every resolution reduction. For base width w and L factor-two resolution transitions, the channel sequence is

$$(w, 2w, \ldots, 2^L w). \tag{57}$$

Kolmogorov models use $L = 5$ transitions and hence six represented resolutions, with channel sequence $(w, 2w, 4w, 8w, 16w, 32w)$. Hasegawa–Wakatani and smoke models use $L = 4$ transitions and hence five represented resolutions, $(w, 2w, 4w, 8w, 16w)$. Increasing the Hasegawa–Wakatani hierarchy from four to five transitions did not improve validation performance, indicating that four transitions were sufficient to span the characteristic spatial scales resolved in this dataset.

Periodic convolutional models use circular padding. Smoke uses nonperiodic padding consistent with its boundary treatment. The aligned and unaligned convolutional U-Nets are otherwise matched. The aligned U-Net augments decoder upsamplers with displacement-producing channels, whereas the unaligned U-Net disables displacement prediction and fuses unwarped skip features directly.

C.2 SWIN BACKBONE

The Swin backbone uses convolutional patch projection, hierarchical factor-two downsampling, transpose-convolution upsampling, alternating regular and shifted windows, Layer Normalization, windowed multi-head self-attention, and MLP sub-blocks. Periodic models allow shifted-window interaction across the physical seam. The aligned and unaligned versions retain the same hierarchy and within-level processing and differ only in skip alignment.

Decoder-predicted displacement. In the principal aligned configuration, the transpose convolution at decoder level ℓ emits both feature channels and raw displacement channels. Only the displacement-producing weights and biases are initialized to zero, so the initial warp is the identity while the feature-producing channels retain their ordinary initialization. For raw displacement a_ℓ,

$$\delta_\ell = \delta_{\max} \tanh(a_\ell), \tag{58}$$

with default $\delta_{\max} = 0.25$ in normalized grid coordinates. The implementation uses backward sampling: output position $\mathbf{x}$ reads from source position $\mathbf{x} + \delta_\ell(\mathbf{x})$. In direct-output models, the displacement warps the encoder skip before fusion. In the queried-time model, the finest query-dependent stage predicts separate displacement fields for the encoder and upsampled decoder representations before their fusion.

Encoder-predicted displacement. For the encoder-predicted ablation, a local convolutional shift module acts directly on E_ℓ and emits the raw displacement field. Its weights and biases are zero initialized and the same bounded sampling operation is applied. Thus decoder- and encoder-predicted variants differ primarily in the representation from which correspondence is inferred, not in the interpolation operator.

Shared and per-channel displacement. A spatially shared predictor emits one d-dimensional field for all skip channels. In the per-channel formulation used by the principal models, the decoder emits one field per latent feature channel. This additional flexibility reflects the semantic diversity of latent channels rather than an assumption that channels correspond to separate physical fluids. The reported ablations compare shared and per-channel fields.

Independent and accumulated fields. The principal models predict displacement independently at each resolution. The implementation also supports accumulated shifts, in which a coarser normalized displacement is interpolated to the next finer level and added to its predicted displacement. Because per-channel warping and accumulated shifts are distinct design choices in the current implementation, accumulated shifts are not used with per-channel displacement. Reported configurations state explicitly when accumulated shifts are enabled.

Sampling and boundaries. Sampling is bilinear in two dimensions and trilinear in three dimensions with corner-aligned normalized coordinates. On an axis with N feature cells, normalized shift δ corresponds to $\delta(N-1)/2$ cell spacings. For periodic domains, source coordinates are wrapped modulo the spatial extent and sampling is performed from a threefold tiled feature tensor so that interpolation is continuous across the seam. Non-periodic sampling uses zero padding outside the domain. These conventions match the boundary treatment used by the corresponding model blocks.

C.3 BASELINE ARCHITECTURE CONFIGURATIONS

All baseline architectures receive the same physical input state and predict the same system-dependent output variables as the corresponding aligned and unaligned convolutional U-Nets.

Swin U-Nets. The Kolmogorov and Hasegawa–Wakatani experiments use matched unaligned and aligned Swin U-Nets. They operate on 256×256 fields using a factor-four patch projection, producing a 64×64 patch grid, followed by one factor-two hierarchy transition to 32×32. The hierarchy therefore contains two represented patch-grid resolutions. These models use window size 16, head counts $(8, 16)$, channel doubling, two Swin blocks per encoder stage, two bottleneck blocks, and two decoder blocks. They use rotary positional encoding and no additional absolute or window-local positional embedding.

The smoke experiments use the corresponding matched unaligned and aligned Swin U-Nets. They apply the same broad design to 128×128 fields, giving patch-grid resolutions 32×32 and 16×16. The aligned Swin models use decoder-conditioned skip displacement, while their unaligned counterparts retain the same hierarchy and feature processing but omit skip warping.

Vision Transformers. The Kolmogorov and Hasegawa–Wakatani baseline uses 8×8 nonoverlapping patches, six global-attention blocks, 16 attention heads, an MLP expansion ratio of four, two-dimensional rotary positional encoding, and no additive positional embedding. The smoke baseline uses 4×4 patches, six global-attention blocks, eight attention heads, an MLP expansion ratio of four, and fixed two-dimensional sinusoidal positional embeddings without rotary encoding. Neither ViT uses a class token or convolutional decoder. A linear token-wise regression head predicts one output patch per token and rearranges the patches into the physical field. The reported ViTs predict the physical state directly rather than adding a predicted residual to the input. Their embedding widths vary across the reported accuracy–cost operating points.

Fourier Neural Operators. For Kolmogorov flow and Hasegawa–Wakatani turbulence, we evaluate two four-layer FNO configurations. The smaller FNO uses 32 hidden channels and retains 64 Fourier modes along each spatial dimension. The larger FNO uses 64 hidden channels and retains 128 modes along each spatial dimension. Both are implemented using `neuraloperator` version 2.0.0.

D TEMPORAL OUTPUT AND TIME-CONDITIONING DETAILS

Direct multi-state output. For K prescribed future states and C physical channels, a non-time-conditioned direct model uses KC final output channels ordered by time and channel, $[t_1 c_1, \ldots, t_1 c_C, t_2 c_1, \ldots, t_K c_C]$, and reshapes them to $B \times K \times C \times H \times W$ (with the analogous three-dimensional shape). All K states are produced by one complete hierarchy evaluation.

Queried-time tensorization. A time-conditioned model accepts a scalar query, a query vector shared across a batch, or per-example queries in $[0, 1]$. In the late-branch configuration, the query-independent trunk is evaluated once. At decoder level 1, the upsampling and skip merge are shared, after which the merged feature is expanded along the query axis. The batch and query dimensions are flattened for the conditioned decoder-level-1 feature block and the finest decoder level, then restored after final projection. Thus the expensive encoder, bottleneck, and lower decoder are not recomputed for each query.

Conditioning. The scalar query modulates normalized activations through feature-wise affine conditioning (Huang & Belongie, 2017; Perez et al., 2018), with related time-conditioning precedents in diffusion backbones (Nichol & Dhariwal, 2021). Convolutional blocks apply modulation after Group Normalization and Swin attention and MLP blocks after Layer Normalization (Wu & He, 2018; Ba et al., 2016). The default scalar-time projection is linear and predicts channel-wise scale and shift vectors applied as

$$h \leftarrow h \odot \big(1 + \gamma(\tau)\big) + \beta(\tau). \tag{59}$$

The final output projection is not conditioned directly; it acts on the query-conditioned finest-resolution representation. An MLP scalar projection, additive conditioning, residual scaling, and

broader conditioning scopes are secondary variants. Late conditioning is an efficiency-oriented restriction rather than an assumption that coarse representations are time independent.

Time-conditioned encoder–decoder warping. At the finest late-conditioned upsampling stage, the auxiliary time-conditioned transpose convolution emits feature channels together with separate raw displacements for the encoder skip and upsampled decoder feature. Each is independently bounded with the same $\tanh$ parameterization before sampling. The two tensors can be warped in a single batched grid-sampling call when their shapes agree. The implementation also supports an encoder-only queried warp and full-hierarchy time conditioning; the principal queried-time results use late conditioning with encoder–decoder warping at the finest level.

Random temporal sampling. Evaluating all queried offsets during every optimizer update repeats the conditioned branch K times. To reduce this cost, queried-time training can sample candidate offsets without replacement. Sampling uses random permutations of the candidate grid, so offsets are unique within each complete or partial draw before a new permutation is used. In supervised queried-time training, a sampled query is paired with its corresponding target for that batch element. In physics-informed training, a sampled residual interval requests the previous and next endpoints needed to evaluate that interval; the final endpoint is additionally requested when required by another active training component. Full output grids are used during evaluation and trajectory generation.

Autoregressive generation. Fixed-horizon models retain only the endpoint and advance by N_{out} stored intervals per complete call. Direct multi-state and queried-time models retain every requested state inside the current block; those states are recorded for evaluation, while only the final state is appended to the history used for the next block. When a history longer than one state is configured, the oldest state is removed and the predicted endpoint is appended. All reported experiments use a one-state history.

E Equation-supervised training details

For physics-informed training, the model produces $K = N_{\mathrm{out}}M$ internal states for positive N_{out}, with M generated intervals per stored-data interval. Both direct block output and queried-time output are supported. The implementation can represent predicted states directly or as increments relative to the supplied current state; the latter avoids reconstructing temporal differences by subtracting two nearly equal absolute single-precision states inside residual implementations. Absolute predicted states are still used for spatial PDE terms and evaluation.

The current state is prepended to the generated sequence, and the residual is evaluated on adjacent intervals. For Kolmogorov and Hasegawa–Wakatani dynamics, the time derivative is formed by a centered difference between adjacent generated states, while nonlinear and spatial terms are evaluated at their arithmetic midpoint. For linear terms this has the Crank–Nicolson form (Crank & Nicolson, 1947); for nonlinear terms it is midpoint evaluation and is not generally identical to trapezoidal Crank–Nicolson. Smoke instead uses a generator-inspired residual that follows the principal semi-Lagrangian advection, buoyancy, diffusion, boundary, and projection operations. The system-specific residual operators, including their spatial discretizations and boundary treatment, are described with the corresponding experimental systems.

The reported equation-supervised experiments use no future-state data loss; future reference fields are used only for diagnostics and evaluation.

F Temporal derivative and residual evaluation

Queried-time prediction makes $\widehat{u}(\tau)$ differentiable with respect to its temporal query wherever the constituent network operations support the required derivative. Direct automatic differentiation with respect to τ, however, is not uniformly available for the aligned architecture because its decoder-conditioned correspondence uses differentiable grid sampling, for which the required forward-mode derivative was unavailable in the PyTorch implementation used in this work.

We therefore evaluate temporal derivatives from adjacent generated states using finite differences. This uses the same intermediate states required for trajectory output, applies consistently to both direct

multi-state and queried-time models, and preserves the discrete-time convention of the system-specific residuals. Finite differences are used throughout the reported equation-supervised experiments.

G SPATIALLY REDUCED OBSERVATIONS AND PERSISTENT UNRESOLVED STATE

Reduction and restoration. For the reported Kolmogorov experiments, $\mathcal{D}_r$ applies nonoverlapping average pooling by factor r independently in each spatial direction. The pooled values are treated as block-center samples and interpolated bicubically back to the 256×256 model grid using non-corner-aligned coordinates. Circular padding and subsequent cropping preserve periodicity during interpolation. Reduction factors $r = 4$, 32, 128, and 256 correspond to effective observed grids of 64×64, 8×8, 2×2, and 1×1, respectively.

Example-indexed persistent state. Each dataset sample carries the pair (q, t_0) consisting of its source trajectory index and starting-time index. The persistent unresolved state is stored in a dictionary using this pair as its key and is initialized to zero unless an experiment explicitly enables deterministic initial cache noise. When the example is encountered, its cached field is transferred from CPU memory to the model device and added only to the newest history state. The clean current state is first reduced and restored, optional fresh training noise is then added, and the cached unresolved field is added last. After the forward pass, loss evaluation, backpropagation, and optimizer update, the cache is updated from a selected absolute internal prediction produced by the preceding forward pass; no second forward pass with the updated parameters is performed. The selected prediction is reduced and restored with the same observation operator, and its pooling complement is detached, transferred to CPU memory, and stored for the next presentation of that example. No gradients propagate through cache history, and unresolved fields are not exchanged among examples.

For selected internal index k_{copy}, the persistent unresolved state is updated as

$$h_i \leftarrow \widehat{u}_i^{(k_{\mathrm{copy}})} - \mathcal{U}_r \mathcal{D}_r \left(\widehat{u}_i^{(k_{\mathrm{copy}})} \right). \tag{60}$$

Thus, whenever an example is encountered, its stored field is replaced by the unresolved component of the selected prediction.

Cache-update time. The reported reduced-input Kolmogorov experiments use $N_{\mathrm{out}} = 1$ and $M = 8$. For $r = 4$, the cache is updated from the first internal prediction, corresponding to one eighth of a stored-data interval. The coarse observation already contains substantial spatial organization in this regime, so the early update limits displacement between that component and the retained unresolved structures. For $r = 32$, $r = 128$, and $r = 256$, the cache is updated from the eighth and final internal prediction, allowing a longer interval over which initially absent structure can develop. These are reduction-dependent design choices rather than conclusions of an isolated update-time ablation.

Physics-time curriculum. For severe reductions, the finite-difference interval used in the training residual is temporarily enlarged early in optimization,

$$\delta t_{\mathrm{phys}}^{(e)} = m^{(e)} \frac{\Delta t_{\mathrm{data}}}{M}, \tag{61}$$

with $m^{(e)}$ decreased linearly to one. For $r = 128$, it decreases from 10 to 1 over epochs 1–20; for $r = 256$, over epochs 1–30; $r = 4$ uses $m^{(e)} = 1$ throughout. Validation and final autoregressive evaluation always reset the multiplier to one.

Symmetry breaking. When $r = 256$, average pooling leaves each training state spatially constant. The reported run initializes the persistent cache to zero and adds independent zero-mean Gaussian white noise during training. At epoch e, its standard deviation relative to the population standard deviation of the corresponding channel in the clean training split is $0.1\, 2^{-(e-1)/5}$. The noise is sampled freshly and is used only to break spatial symmetry; it does not encode the reference spatial spectrum or a withheld fine-scale realization.

Scope and additional controls. The reduced-observation experiment concerns the information required to train a full-resolution transition model. Full-resolution held-out states initialize evaluation rollouts, and no subsequent reference state is supplied to the model. Fresh input noise is used only where explicitly identified, including the spatially uniform observation experiment and the fresh-noise controls reported below. It is not part of the other persistent-unresolved-state configurations.

H Supplementary Experimental Results

H.1 Trajectory splits and comparison protocol

Each run seed determines model initialization and a trajectory or temporal-segment partition constructed before temporal windows are enumerated. Samples from one trajectory or segment never cross the train/test boundary within a run. Learning-rate selection uses validation seeds 32 and 42 under a common search budget, whereas final reporting uses seeds 2, 12, and 22. Metrics from the final runs are not used for learning-rate selection. Partitions are generated independently across seeds and may therefore overlap, but directly compared configurations use the same partition for each seed.

Relative L_2 error. For predicted and reference fields $\widehat{u}, u \in \mathbb{R}^{B\times C\times H\times W}$, we compute the relative field error separately for every sample and physical channel,

$$\mathcal{E}_{\mathrm{rel}}(\widehat{u},u)=\frac{1}{BC}\sum_{b=1}^{B}\sum_{c=1}^{C}\left[\frac{\sum_{i=1}^{H}\sum_{j=1}^{W}(\widehat{u}_{bcij}-u_{bcij})^2}{\max\left(\sum_{i=1}^{H}\sum_{j=1}^{W}u_{bcij}^2,\epsilon\right)}\right]^{1/2},\qquad \epsilon=10^{-5}. \tag{62}$$

The spatial sums extend over both spatial dimensions. Errors are normalized separately for each physical channel and then averaged with equal weight over channels and samples, so larger-amplitude channels do not dominate multichannel Hasegawa–Wakatani or smoke results. For multi-state outputs, the same reduction additionally averages over all evaluated output times. Complete evaluation scripts accompany the individual run records.

Supervised training loss. Endpoint-only, direct multi-state, and queried-time data-supervised models are trained using the same relative-error reduction reported for evaluation, given in Eq. (62). For direct multi-state models, the loss is additionally averaged over all predicted output times. For queried-time models trained with random temporal sampling, it is evaluated between each sampled prediction and its corresponding target. The reported target-free equation-supervised models do not use this data loss during training; they are optimized using the system-specific equation residuals described above.

Autoregressive and statistical evaluation. Endpoint-only models return one state per invocation. Direct multi-state and queried-time models retain all returned stored-data checkpoints, while the last state of each outer block initializes the next invocation. Spatial and temporal spectra and autocorrelations are computed independently for each held-out trajectory or Hasegawa–Wakatani temporal segment before averaging; segments are never concatenated across their boundaries. Accelerator timing uses synchronized measurements after untimed warm-up calls.

Matched architectural comparisons. The principal comparisons are controlled architectural interventions. The aligned and unaligned convolutional U-Nets share the same residual backbone, hidden-channel width, number of resolution transitions, optimizer, learning rate and schedule, epoch count, trajectory split, and random seeds; they differ only in the skip-alignment mechanism. The same matching is used for aligned and unaligned Swin models. Kolmogorov convolutional hierarchies use five factor-two resolution transitions, giving six represented resolutions. Hasegawa–Wakatani and smoke use four transitions, giving five represented resolutions. Increasing the Hasegawa–Wakatani hierarchy from four to five transitions did not improve validation performance, indicating that four transitions were sufficient to span the characteristic spatial scales resolved in this dataset. These depths are fixed rather than optimized separately for aligned and unaligned variants.

Optimization and computational setup. All principal supervised architecture comparisons are trained for 48 epochs. Within each system, supervised aligned and unaligned convolutional U-Nets,

Table 1: Principal system and training settings. Learning rates vary across model families and are selected using validation data.

Setting	Kolmogorov	Hasegawa–Wakatani	Smoke
Physical channels	1	2	3
Input history states	1	1	1
Convolutional transitions	5	4	4
Supervised epochs	48	48	48
Supervised batch size	2	4	8
Equation-supervised batch size	2	2	2
Domain	periodic	periodic	nonperiodic

Swin U-Nets, FNOs, and ViTs use the same batch size: 2 for Kolmogorov flow, 4 for Hasegawa–Wakatani turbulence, and 8 for smoke. Principal equation-supervised experiments use batch size 2. Unless explicitly identified as an AdamW optimization, models are trained with Adam. Learning rates are selected using the validation seeds, and final-test trajectories are not used for learning-rate, architecture, or checkpoint selection.

All reported results use the same custom epoch-wise learning-rate schedule. Let E denote the total number of training epochs, $H = \max(1, \lfloor E/2 \rfloor)$, and $F = \max(1, \lfloor E/5 \rfloor)$. At zero-based scheduler epoch e, the learning rate is

$$\eta_e = \eta_0 \left[0.8 - 0.5 \cos\left(\frac{2\pi e}{H}\right)\right] \begin{cases} 1, & e < H, \\ \exp[-(e-H)/F], & e \geq H, \end{cases} \tag{63}$$

where η_0 is the configured base learning rate. The oscillatory factor is active throughout training, while its envelope decays exponentially during the second half. The scheduler is implemented with PyTorch `LambdaLR` and advanced once after each completed training epoch. Learning-rate searches vary η_0 while retaining this schedule. The schedule does not impose a minimum learning-rate floor.

The reduced-observation Kolmogorov models use Adam with base learning rate 5×10^{-4}, 144 epochs, and batch size 2. The supplemental 160-epoch Hasegawa–Wakatani optimization uses AdamW with weight decay 0.1 and base learning rate 10^{-4}. All reported models are trained and evaluated on a single NVIDIA A100 80 GB GPU. Reported wall-clock training and generation times were measured on this hardware. The FNO environment uses `neuraloperator` version 2.0.0.

Width and learning-rate selection. We evaluate predefined hidden-channel widths and retain the operating points to characterize accuracy as a function of training time, inference time, memory, and parameter count. Width is therefore a scaling variable rather than a single selected hyperparameter. Learning rate is selected from the same predefined search space and with the same two-seed budget for convolutional, Swin, Fourier, and vision-transformer models. Training schedules, epoch budgets, and input information are otherwise identical within each system and prediction regime. The aligned and unaligned members of a matched pair use the same selected learning rate. Final means and sample standard deviations use seeds 2, 12, and 22; best-seed values are shown only as descriptive operating points.

H.2 Complete supervised results and seed robustness

The main text summarizes the principal 32-channel comparisons and the $n_t = 4$ accuracy–cost view. This appendix expands the analysis across available widths, outer prediction intervals, output parameterizations, and evaluation horizons. Filled markers denote seed means, open stars denote best completed seeds, and error bars denote sample standard deviations. All convolutional comparisons and most ViT and FNO comparisons contain three completed seeds. A missing or zero-length error bar should be interpreted together with the reported run count rather than as evidence of negligible initialization sensitivity.

Kolmogorov dependence on evaluation horizon. Figures 4 and 5 report the same model families and outer intervals at rollout steps 16 and 64. At step 16, aligned and unaligned convolutional models are already separated at most matched operating points, but the differences are generally modest

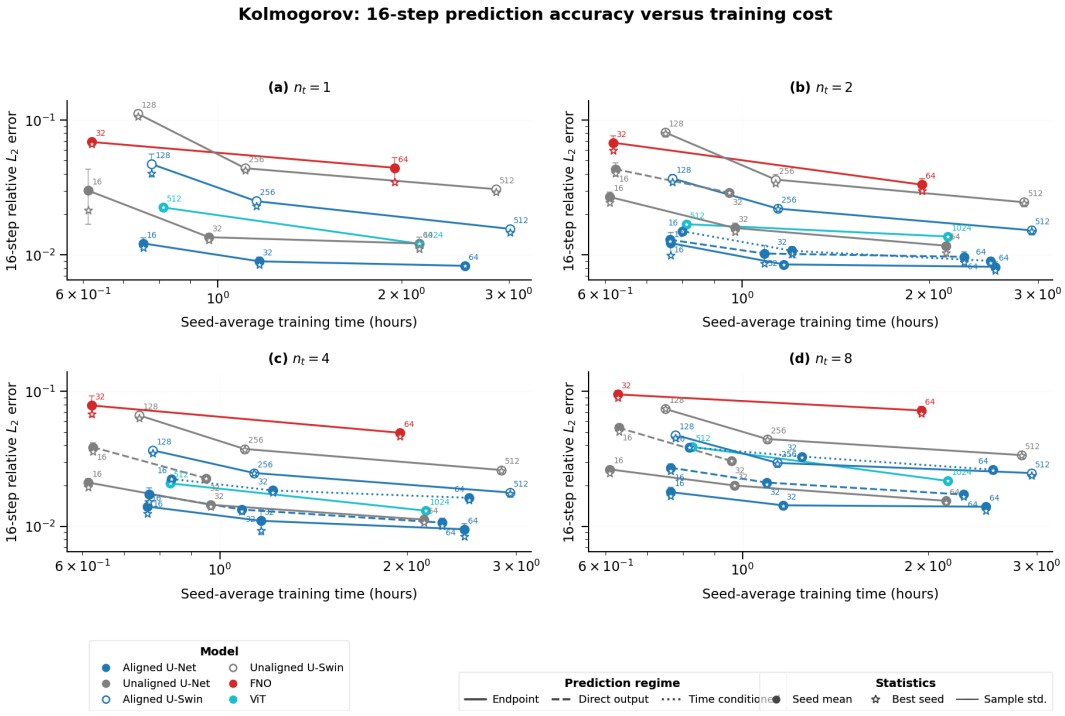

Figure 4: **Kolmogorov short-horizon accuracy versus measured training cost.** Relative L_2 error after 16 stored-data forecast steps for $n_t \in \{1, 2, 4, 8\}$. Curves connect widths within the same model family and prediction regime. Filled markers show seed means, open stars show best completed seeds, and error bars show sample standard deviations. Numeric labels give hidden-channel counts. Comparison with Fig. 5 shows that the aligned–unaligned separation generally increases over longer autoregressive rollout.

compared with step 64. The separation grows after repeated autoregressive composition, especially for $n_t = 1$ and 2 and for direct multi-state output. For example, the 32-channel endpoint models at $n_t = 1$ differ by only about a factor of 1.5 at step 16 but by about a factor of 3 at step 64. The persistence of the ordering across both horizons, together with its amplification at step 64, shows that alignment improves rollout stability rather than merely the accuracy of the first few transitions.

Robustness across width and outer interval. The complete step-64 sweep supports the conclusions drawn from the 32-channel curves in the main figure. Aligned U-Nets improve over matched unaligned U-Nets across all displayed widths at $n_t = 1$ and remain favorable throughout the interval sweep. At $n_t = 1$, the seed-mean reduction is approximately $4.0\times$, $3.0\times$, and $2.7\times$ at widths 16, 32, and 64, respectively. A 16-channel aligned model also outperforms the 64-channel unaligned model, demonstrating that the gain is not explained by additional capacity. Moderate intervals preserve the aligned endpoint accuracy while reducing complete hierarchy evaluations, whereas $n_t = 8$ increasingly exposes the difficulty of learning a longer transition.

Interpretation of best and average performance. Best-seed markers identify the best completed realization but are not used as the sole robustness statistic. The principal conclusions are supported by seed means wherever replicated runs are available. Some exploratory model families and several 32-channel unaligned multi-state configurations contain only one completed run; these points are retained to document the available operating points but are not used alone to establish robustness. The fully replicated 16-channel multi-state comparison discussed in Section 4.3 provides the cleaner aligned–unaligned test for that regime.

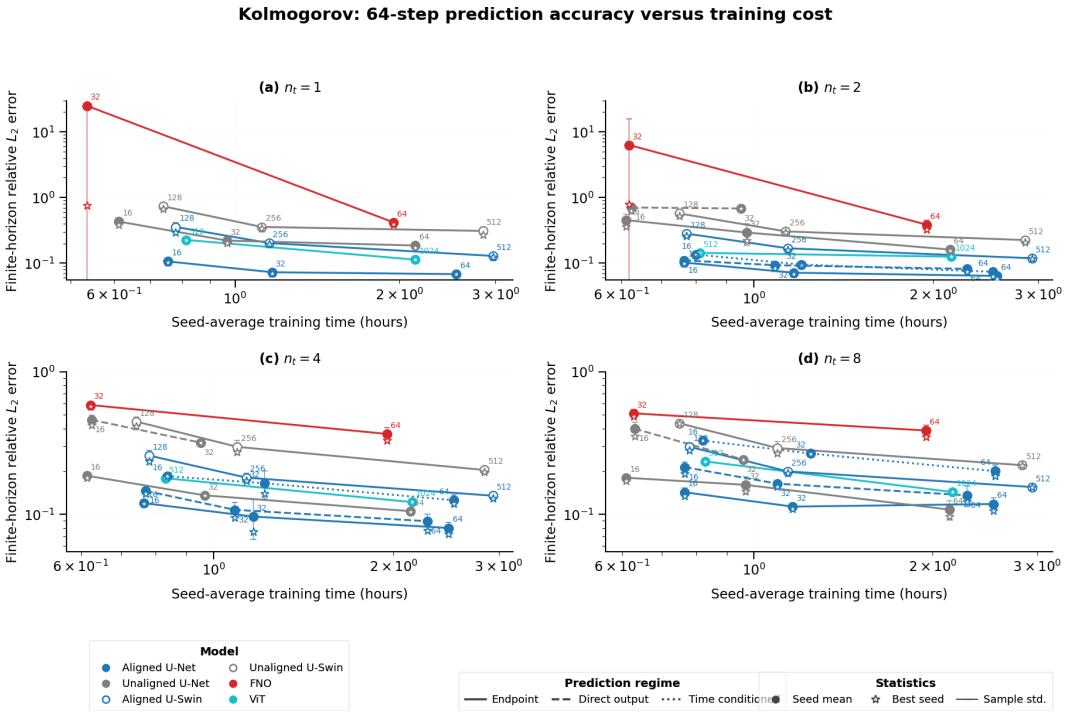

Figure 5: **Kolmogorov longer-horizon accuracy versus measured training cost.** Relative L_2 error after 64 stored-data forecast steps for $n_t \in \{1, 2, 4, 8\}$. Plotting conventions match Fig. 4. Alignment improves the convolutional accuracy–cost region throughout the interval sweep, with the largest separations at shorter outer intervals and for multi-state prediction. Several FNO operating points that remain finite at short horizon become highly inaccurate or unstable by step 64.

H.3 Alignment-design ablations

We compare the same four skip-alignment variants for Kolmogorov flow and Hasegawa–Wakatani turbulence. The ablation varies the representation used to predict displacement, whether one displacement is shared across latent channels or independently predicted for each channel, and whether alignment is applied only at the finest decoder merge or at every decoder resolution. These experiments compare aligned convolutional architectures only; aligned–unaligned comparisons are reported separately in the principal architecture results.

Variants and protocol. Let E_ℓ denote the input-time encoder feature and $D_{\ell+1}$ the coarser, future-oriented decoder representation, with $\ell = 0$ the finest decoder merge. In *encoder-predicted alignment*, a local convolution acting on E_ℓ predicts the displacement used to sample that skip feature. In *decoder-predicted alignment*, decoder upsampling jointly emits the upsampled decoder feature and displacement field, so correspondence is inferred from the future-oriented decoder path. A *shared* displacement applies one two-component field to every latent channel at a resolution, whereas a *per-channel* displacement predicts an independent field for each latent skip channel. *Finest-level* alignment warps only the highest-resolution skip, whereas *all-level* alignment independently warps every decoder skip.

The outer prediction interval n_t is the number of consecutive stored-data time steps advanced by one complete model invocation. Thus, a model with $n_t = 4$ advances four saved-data intervals per invocation before its final prediction initializes the next autoregressive block. We evaluate $n_t \in \{1, 2, 4, 8\}$. All Kolmogorov runs use 16 hidden channels and learning rate 10^{-3}; all Hasegawa–Wakatani runs use 32 hidden channels and learning rate 10^{-3}. Kolmogorov performance is measured by relative L_2 error after 16 stored-data forecast steps, and Hasegawa–Wakatani performance after eight stored-data forecast steps. Figure 6 and Table 2 report only the four parameter combinations present for both systems.

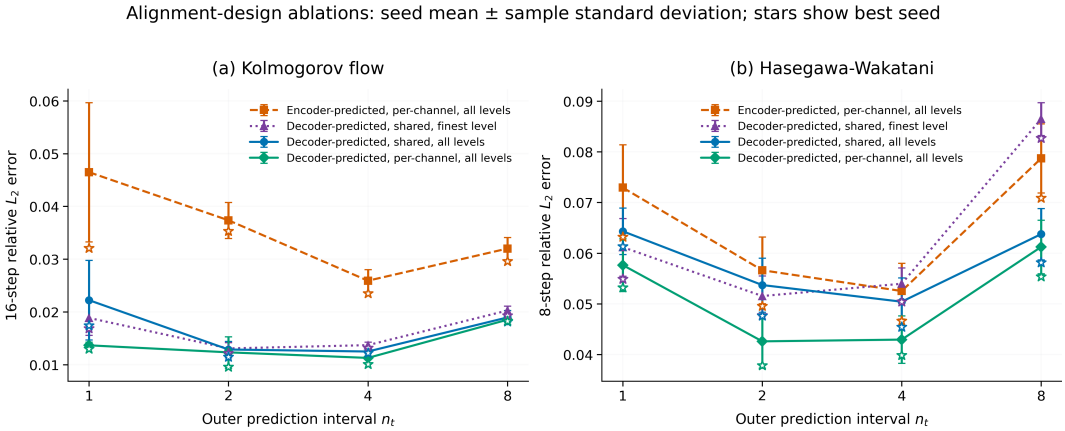

Figure 6: **Alignment-design ablations.** These plots compare the performance of aligned convolutional models under four matched choices for skip alignment between encoder and decoder representations. Lines and filled markers show seed means, error bars show sample standard deviations, and open stars show the best completed seed. (a) Kolmogorov-flow relative L_2 error after 16 stored-data forecast steps. (b) Hasegawa–Wakatani relative L_2 error after eight stored-data forecast steps. The logarithmic horizontal axis gives the outer prediction interval n_t, the number of stored-data time steps advanced by one complete model invocation. Labels identify the representation used to predict displacement, whether displacement is shared or channel specific, and whether only the finest skip or all skip resolutions are aligned.

Predictor source has the dominant effect. The clean matched comparison holds per-channel, all-level alignment fixed and changes only the representation used to predict displacement. Decoder prediction reduces the Kolmogorov seed mean by 70.6%, 67.0%, 56.4%, and 42.1% for $n_t = 1, 2, 4,$ and 8, respectively; the corresponding Hasegawa–Wakatani reductions are 21.0%, 24.8%, 18.3%, and 22.2%. The improvement persists across systems and outer intervals, supporting the use of a future-oriented decoder representation to infer correspondence.

Shared versus per-channel correspondence. With decoder prediction and all-level alignment fixed, per-channel displacement improves the seed mean at every tested interval in both systems. The reductions are 2–38% for Kolmogorov flow and 4–21% for Hasegawa–Wakatani. The effect is secondary to predictor source but is consistent in direction, indicating that different latent channels benefit from independently selected source locations.

Finest-level versus all-level alignment. With decoder prediction and shared displacement fixed, extending alignment from the finest skip to all decoder resolutions has the smallest and least uniform effect. For Kolmogorov flow, all-level alignment is worse at $n_t = 1$ but modestly better for $n_t = 2, 4,$ and 8. For Hasegawa–Wakatani, it is slightly worse at $n_t = 1$ and 2 but better at $n_t = 4$ and substantially better at $n_t = 8$. Coarser-level alignment therefore becomes most useful for longer outer transitions, while the finest skip can be sufficient at short intervals.

Overall interpretation. Across the common parameter matrix, decoder prediction is the principal design choice, per-channel displacement provides a smaller consistent refinement, and all-level coverage provides a more interval-dependent refinement. The decoder-predicted per-channel all-level variant has the lowest seed mean at every tested n_t for both systems except that the Kolmogorov shared and per-channel all-level means are nearly tied at $n_t = 2$. Small differences among secondary variants should therefore not be overinterpreted, whereas the cross-system advantage of decoder-predicted correspondence is robust.

H.4 Baseline and accuracy–cost comparisons

This section expands the main-text $n_t = 4$ trade-off to all evaluated outer intervals and systems. Training time is measured wall-clock time for the reported implementation and hardware, so the

Table 2: **Numerical results for the matched alignment-design ablations in Figure 6.** Best is the lowest error among completed seeds. Mean and sample standard deviation are computed across the reported runs, and Seeds gives the number of unique completed seeds. Kolmogorov uses relative L_2 error after 16 stored-data forecast steps, and Hasegawa–Wakatani after eight stored-data forecast steps.

System	Alignment variant	n_t	Best	Mean $\pm$ std.	Seeds
Kolmogorov	Encoder-predicted, per-channel, all levels	1	0.0321	0.0465 ± 0.0132	3
	Encoder-predicted, per-channel, all levels	2	0.0353	0.0374 ± 0.0034	3
	Encoder-predicted, per-channel, all levels	4	0.0235	0.0259 ± 0.0021	3
	Encoder-predicted, per-channel, all levels	8	0.0296	0.0320 ± 0.0021	3
	Decoder-predicted, shared, finest level	1	0.0169	0.0189 ± 0.0033	3
	Decoder-predicted, shared, finest level	2	0.0122	0.0131 ± 0.0011	3
	Decoder-predicted, shared, finest level	4	0.0132	0.0137 ± 0.0006	3
	Decoder-predicted, shared, finest level	8	0.0193	0.0203 ± 0.0008	3
	Decoder-predicted, shared, all levels	1	0.0174	0.0222 ± 0.0075	3
	Decoder-predicted, shared, all levels	2	0.0115	0.0129 ± 0.0016	3
	Decoder-predicted, shared, all levels	4	0.0122	0.0125 ± 0.0004	3
	Decoder-predicted, shared, all levels	8	0.0182	0.0189 ± 0.0013	3
	Decoder-predicted, per-channel, all levels	1	0.0130	0.0137 ± 0.0006	3
	Decoder-predicted, per-channel, all levels	2	0.0096	0.0123 ± 0.0030	3
	Decoder-predicted, per-channel, all levels	4	0.0101	0.0113 ± 0.0011	3
	Decoder-predicted, per-channel, all levels	8	0.0182	0.0185 ± 0.0004	3
Hasegawa–Wakatani	Encoder-predicted, per-channel, all levels	1	0.0632	0.0729 ± 0.0085	3
	Encoder-predicted, per-channel, all levels	2	0.0496	0.0566 ± 0.0066	3
	Encoder-predicted, per-channel, all levels	4	0.0466	0.0525 ± 0.0055	3
	Encoder-predicted, per-channel, all levels	8	0.0709	0.0787 ± 0.0068	3
	Decoder-predicted, shared, finest level	1	0.0549	0.0612 ± 0.0057	3
	Decoder-predicted, shared, finest level	2	0.0477	0.0515 ± 0.0040	3
	Decoder-predicted, shared, finest level	4	0.0504	0.0540 ± 0.0032	3
	Decoder-predicted, shared, finest level	8	0.0827	0.0865 ± 0.0033	3
	Decoder-predicted, shared, all levels	1	0.0613	0.0643 ± 0.0046	3
	Decoder-predicted, shared, all levels	2	0.0476	0.0537 ± 0.0053	3
	Decoder-predicted, shared, all levels	4	0.0454	0.0504 ± 0.0047	3
	Decoder-predicted, shared, all levels	8	0.0581	0.0638 ± 0.0051	3
	Decoder-predicted, per-channel, all levels	1	0.0532	0.0576 ± 0.0052	3
	Decoder-predicted, per-channel, all levels	2	0.0379	0.0426 ± 0.0053	3
	Decoder-predicted, per-channel, all levels	4	0.0398	0.0429 ± 0.0047	3
	Decoder-predicted, per-channel, all levels	8	0.0554	0.0612 ± 0.0053	3

horizontal axis should be interpreted as an empirical operating cost rather than an architecture-independent complexity measure.

Hasegawa–Wakatani accuracy–cost results. Figure 7 reports step-16 error for all evaluated HW outer intervals. The aligned U-Net improves over its matched unaligned counterpart across the principal width and interval comparisons, while direct and queried-time multi-state prediction remain similar through moderate block lengths. The aligned–unaligned gap is smaller than for long-horizon Kolmogorov prediction but is consistent across the displayed 32-channel endpoint sweep. The shallow Swin comparisons provide a qualitative test with a different within-level mixer: the tested models use a factor-four patch projection followed by one factor-two hierarchy transition, giving two represented patch-grid resolutions. Alignment improves most matched Swin operating points, supporting transfer of the mechanism beyond convolution, but the experiment is not a general ranking of convolutional and attention-based hierarchies.

H.4.1 Representative Hasegawa–Wakatani rollout and spatial spectra

Figure 8 illustrates the transition from trajectory-level to statistical agreement during a long Hasegawa–Wakatani rollout. The displayed model is the completed 32-channel aligned U-Net with the lowest 16-step relative L_2 error at $n_t = 4$ under endpoint-only prediction. Through step 40, the generated density and potential closely track the reference fields, reproducing the same evolving structures at approximately the same locations. By step 80, differences in the positions and shapes of individual structures become substantial, although the generated fields retain the characteristic multiscale density fluctuations and large-scale zonal organization of the potential. At step 160, the generated trajectory no longer corresponds pointwise to the particular reference realization. Exact frame-by-frame agreement is not expected over such a long rollout: because one prediction step corresponds to 0.5 normalized time units, the displayed steps 20, 40, 80, and 160 correspond to normalized times 10, 20,

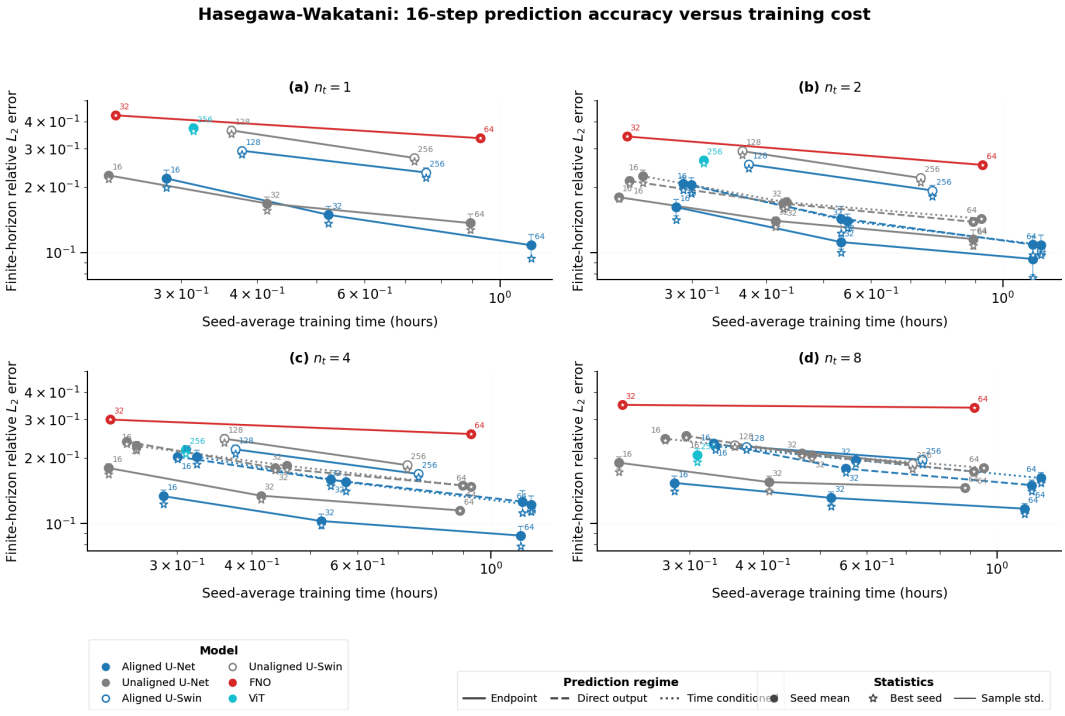

Figure 7: **Hasegawa–Wakatani finite-horizon accuracy versus measured training cost.** Relative L_2 error after 16 stored-data forecast steps for $n_t \in \{1, 2, 4, 8\}$. Filled markers show seed means, open stars show best completed seeds, error bars show sample standard deviations, and numeric labels give hidden-channel counts. The aligned convolutional models provide the strongest completed accuracy–cost region, while the shallow Swin results qualitatively demonstrate that alignment can also benefit windowed-attention hierarchies. Exact completed-run counts accompany the released numerical results.

40, and 80, respectively, all many times longer than the dimensionless Lyapunov time of less than one normalized time unit (Pedersen et al., 1996). Agreement at the later horizons should therefore be assessed through spatial and temporal statistics rather than pointwise field comparisons.

The spatial spectra in Figure 9 remain closely matched after pointwise correspondence with the particular reference trajectory is lost. For both fields, generated and reference spectra agree over the wavenumber range carrying nearly all spectral magnitude. The visible high-wavenumber difference for ϕ is confined to a very low-magnitude tail, approximately three orders below the dominant harmonic, and would be visually negligible on a linear scale. The autonomous rollout therefore preserves the principal distribution of spatial scales even when comparison with the particular reference realization is no longer meaningful.

The field panels show one held-out temporal segment and the model realization selected by its lowest 16-step error. They are therefore illustrative rather than an aggregate robustness result. In contrast, the spectra are computed separately over all four held-out temporal segments and then averaged. The seed-level finite-horizon and spectral comparisons elsewhere in this appendix remain the basis for comparisons across model configurations.

Buoyant-smoke accuracy–cost results. Figure 10 separates the solver-step equation-supervised and stored-data-step data-supervised settings. At the solver-step interval, the 32-channel aligned model reduces the seed-mean error from 0.489 to 0.229, and the 64-channel aligned model improves the best completed result further. At the stored-data interval, aligned and unaligned convolutional models are much closer, and the effect of increasing width is comparable to or larger than the remaining alignment difference. Because interval and objective change together, the two panels do not isolate equation supervision from temporal resolution. They show instead that the finer transition

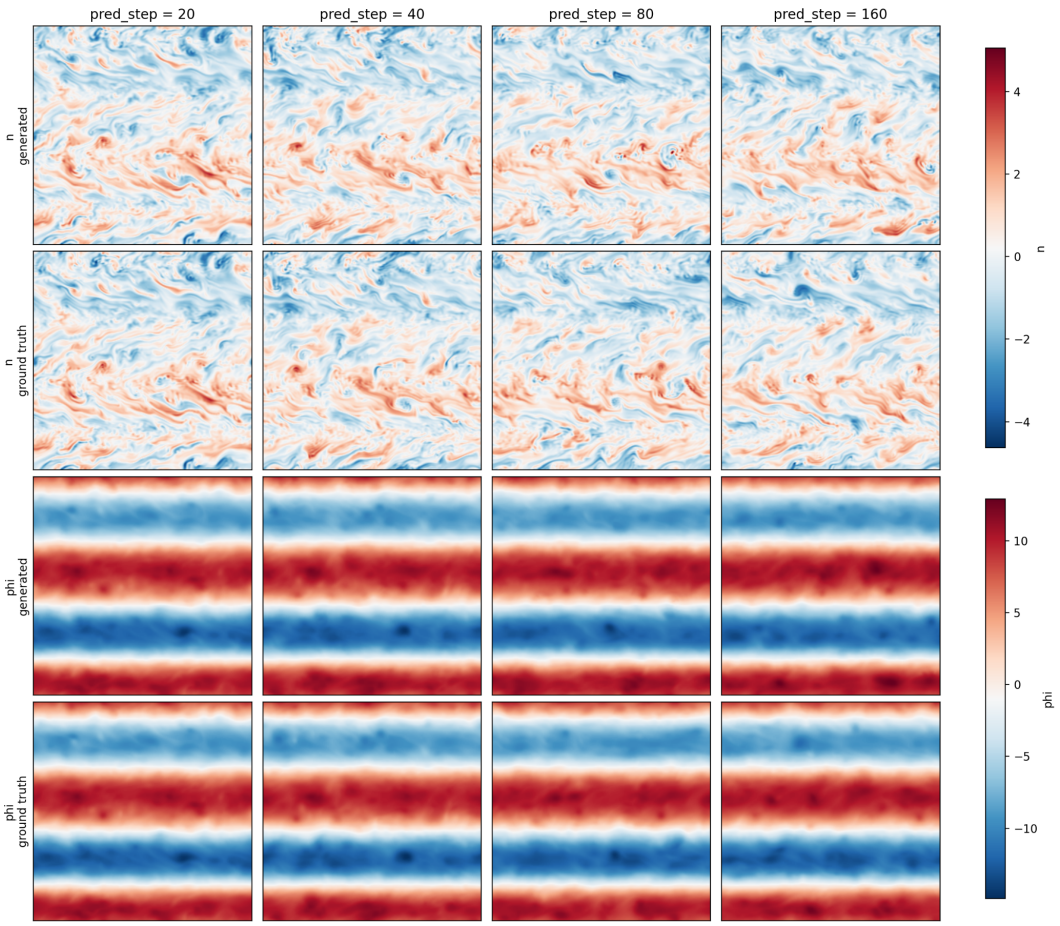

Figure 8: **Transition from trajectory-level to statistical agreement in a Hasegawa–Wakatani rollout.** Generated and reference density n and electrostatic potential ϕ at forecast steps 20, 40, 80, and 160. The rollout is produced by the completed 32-channel aligned U-Net with the lowest 16-step relative L_2 error for $n_t = 4$ endpoint-only prediction. Generated fields are shown above their corresponding references, with a common color scale within each physical channel. Through step 40, the prediction reproduces the same evolving structures at approximately the same locations as the reference. Pointwise differences become substantial by step 80, and by step 160 the generated trajectory no longer corresponds to the particular reference realization. Because these checkpoints extend many Lyapunov times beyond the initial condition, the later fields should be interpreted together with the long-rollout statistical comparisons rather than as exact pointwise forecasts.

accessible through the equation residual is both more accurate in the present experiment and more sensitive to skip correspondence.

H.4.2 Representative equation-supervised buoyant-smoke rollout

Figure 11 shows a representative rollout from the best completed 32-channel aligned U-Net trained with equation supervision at the solver time step. The training objective uses the discrete smoke-system residual and no future-state targets; stored trajectory data supply only current states from which generated evolution begins. Although the model advances at the solver interval, the figure displays stored-data checkpoints so that every generated field can be compared directly with an available reference field. With $\Delta t_{\mathrm{data}} = 1.5$, forecast steps 1–5 correspond to times 1.5, 3.0, 4.5, 6.0, and 7.5 after the supplied initial state.

Across the displayed horizon, the model closely reproduces transport and deformation of the smoke concentration together with the associated horizontal and vertical velocity fields. The dominant

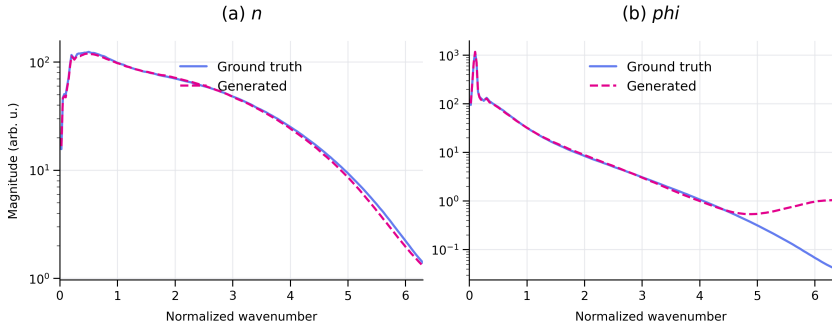

Figure 9: **Long-rollout spatial spectra for the Hasegawa–Wakatani trajectory model.** Time-averaged spatial spectra of (a) density n and (b) electrostatic potential ϕ for generated and reference rollouts from the same model configuration as Figure 8. Statistics are computed separately for each of the four held-out temporal segments and then averaged. The spectra agree closely across the dynamically significant wavenumber range. The visible separation in the high-wavenumber tail of ϕ occurs at magnitudes approximately three orders below the dominant harmonic and is emphasized by the logarithmic vertical scale.

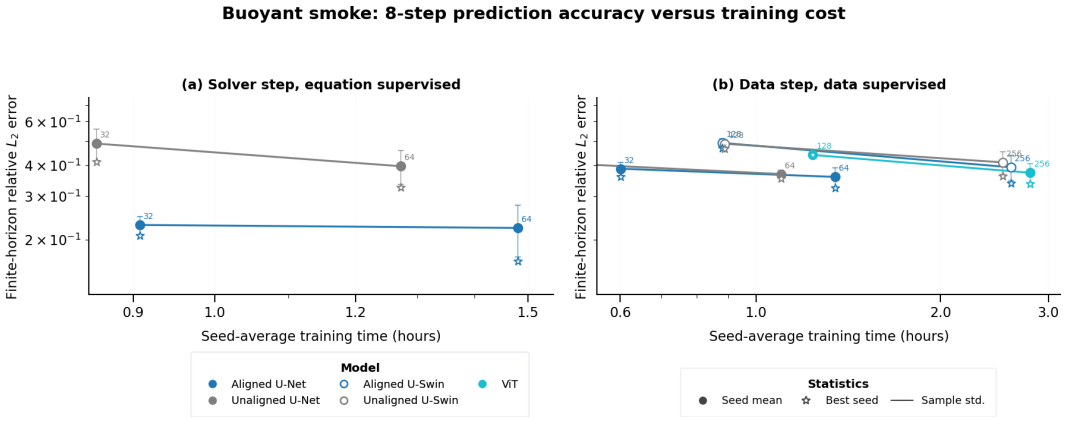

Figure 10: **Buoyant-smoke accuracy versus measured training cost.** Eight-step relative L_2 error for (a) one solver-step transitions trained with equation supervision and (b) stored-data-step transitions trained with data supervision. Filled markers show seed means, open stars show best completed seeds, error bars show sample standard deviations, and numeric labels give hidden-channel counts. The comparison across panels is complementary rather than controlled because both the transition interval and training objective differ. Alignment has a substantially larger effect for the finer solver-step transition.

interfaces, vortical structures, and velocity patterns remain at approximately the same locations as in the reference fields. Differences are concentrated primarily in fine-scale interfaces and localized extrema and become somewhat more visible at later steps, but no substantial loss of the overall flow organization is apparent.

Long-rollout spatial-spectral fidelity. Figure 12 compares the time-averaged spatial Fourier-spectrum error at $n_t = 4$. The statistic is computed over the long autoregressive rollouts described in Appendix A, separately for each held-out trajectory or segment before averaging. For Kolmogorov flow, the aligned convolutional models generally combine low finite-horizon error with favorable spectral fidelity. Several unaligned and FNO operating points have extremely large seed means because one or more runs become unstable; these values are marked as off-scale rather than compressing the stable region of the plot. Thus short-horizon accuracy alone can hide substantial differences in long-rollout behavior.

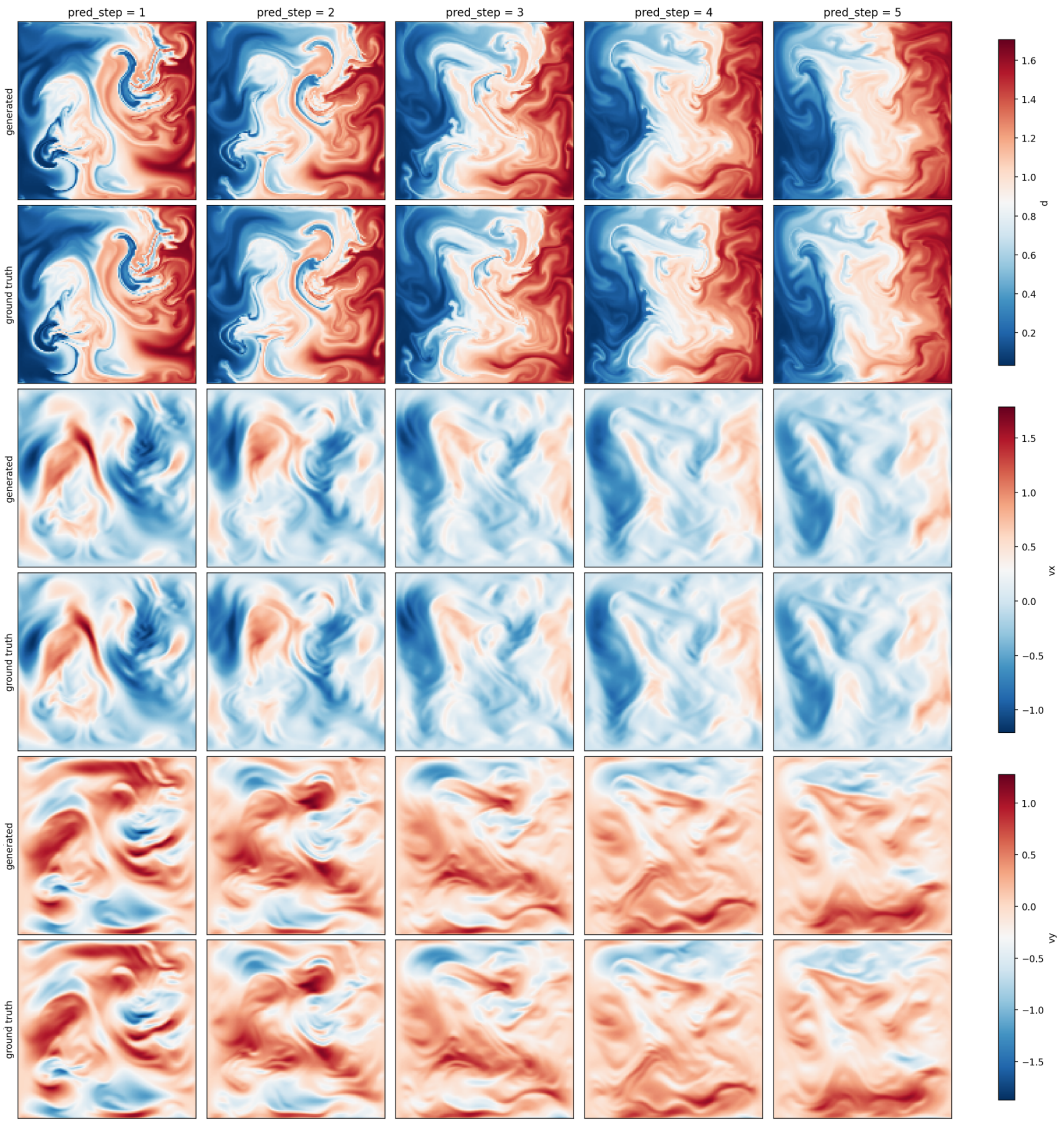

Figure 11: **Representative equation-supervised buoyant-smoke rollout.** Generated and reference smoke concentration d, horizontal velocity v_x, and vertical velocity v_y at stored-data forecast steps 1–5, corresponding to times 1.5, 3.0, 4.5, 6.0, and 7.5 after the supplied initial state. The rollout is produced by the best completed 32-channel aligned U-Net trained using the solver-inspired equation loss at the solver time step, without future-state targets; stored trajectory data supply only current states from which generated evolution begins. The model advances at the solver interval, while the displayed checkpoints coincide with available stored reference fields. Generated fields are shown above their corresponding references, with a common color scale within each physical channel. The model closely reproduces the evolving plume interfaces, dominant vortical structures, and associated velocity patterns throughout the displayed rollout. Remaining differences are most visible in fine-scale interfaces and localized extrema.

For Hasegawa–Wakatani turbulence, the relationship is less uniform. Several models with competitive step-16 error have broad seed variation in the spatial-Fourier metric, and the ordering can change across width and architecture. This behavior is expected to be more visible in statistical diagnostics because a long generated trajectory need not retain pointwise correspondence with a particular reference realization. The FFT comparison is therefore not used as a substitute for finite-horizon field error. Instead, the two metrics answer complementary questions: finite-horizon error evaluates

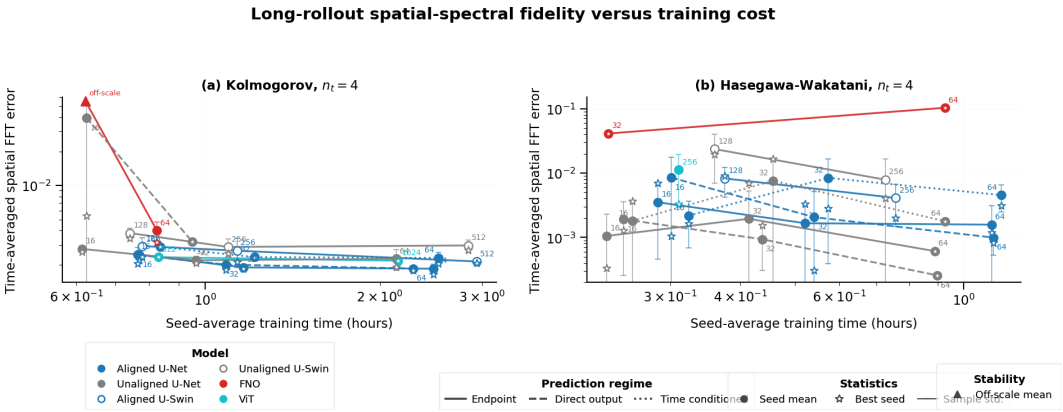

Figure 12: **Long-rollout spatial-spectral fidelity versus measured training cost at $n_t = 4$.** Time-averaged spatial FFT error for (a) Kolmogorov flow and (b) Hasegawa–Wakatani turbulence. Filled markers show seed means, open stars show best completed seeds, error bars show sample standard deviations, and numeric labels give hidden-channel counts. Upward triangles at the plot boundary denote off-scale seed means caused by unstable or extremely inaccurate long rollouts. Kolmogorov results show a clearer correspondence between alignment, finite-horizon accuracy, and spectral fidelity; HW exhibits greater model- and seed-dependent variation, motivating joint reporting of field and statistical metrics.

pointwise trajectory prediction, while the Fourier statistic tests whether a long autonomous rollout retains the reference distribution of spatial scales.

H.5 Multi-state prediction ablations

The main text compares endpoint-only, direct multi-state, and queried-time prediction using the outer interval n_t. Here we expand the comparison across widths and report the interpretation of intermediate-state accuracy and cost.

Endpoint retention under direct multi-state output. Direct-output models emit all prescribed offsets through one final projection. At $n_t = 4$, the 64-channel aligned Kolmogorov model obtains a best step-64 endpoint error of 0.077, compared with 0.073 for the matched endpoint-only model. The corresponding seed means remain close on the measured accuracy–training-cost plot, showing that the complete internal checkpoint sequence can be obtained without a large endpoint penalty. Because the projection width grows with the number of returned states while the encoder and decoder hierarchy are evaluated once, measured trajectory-generation time changes much less with n_t than it would if the complete model were called independently at every checkpoint.

Interaction between alignment and block length. The strongest controlled evidence comes from the replicated 16-channel Kolmogorov comparison. The aligned and unaligned best step-64 errors are 0.103 and 0.623 at $n_t = 2$, 0.140 and 0.422 at $n_t = 4$, and 0.193 and 0.354 at $n_t = 8$. Alignment therefore provides its largest relative benefit when the unaligned model must construct several future states from one input-time hierarchy. The advantage is also visible at step 16, but it grows after further autoregressive composition, supporting the interpretation that spatially aligned skips reduce error amplification as well as local block error.

Several 32-channel unaligned Kolmogorov direct-output configurations have only one completed run. They are included in the numerical release and the complete trade-off figures but are not used as the principal robustness comparison. By contrast, the aligned direct-output curves contain replicated runs and remain competitive with endpoint-only aligned models across widths.

Direct versus queried-time output. Direct and queried-time output solve related but distinct problems. Direct output assigns dedicated channels to a fixed discrete set of offsets and is computationally attractive when that grid is known in advance. Queried-time output shares a temporal representation

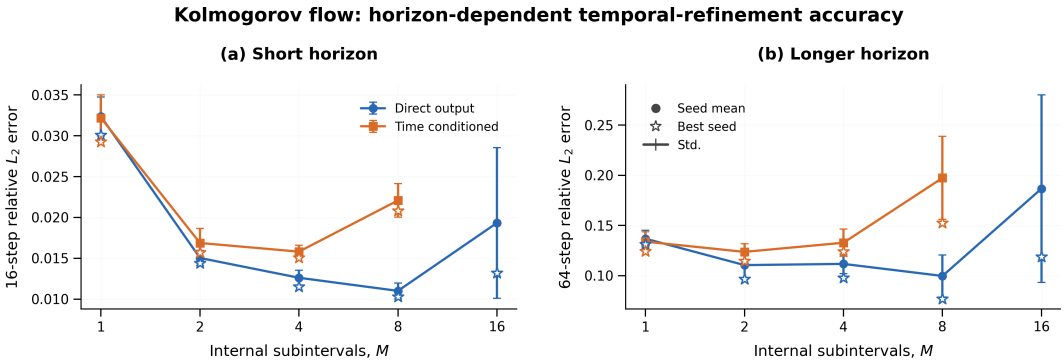

Figure 13: **Kolmogorov horizon-dependent temporal-refinement accuracy.** Relative L_2 error after (a) 16 and (b) 64 stored-data forecast steps for 96-epoch direct-output and time-conditioned models. Filled markers show seed means, error bars show sample standard deviations, and open stars show best completed seeds.

and can accept a continuous-valued offset, but it reevaluates the conditioned finest branch for each requested time. For HW, the two parameterizations have similar finite-horizon accuracy through moderate n_t. For Kolmogorov flow, direct output becomes more accurate as the block length grows. These results establish system-dependent performance on training-represented offsets; they do not establish interpolation to offsets excluded from training.

Intermediate states and equation supervision. All intermediate states are retained for evaluation even though only the endpoint initializes the next autoregressive block. In the supervised experiments these states determine whether one hierarchy evaluation can reproduce the stored-data trajectory inside its outer interval. In the target-free experiments of Section 4.4, the same construction has an additional role: adjacent generated states define the shorter finite-difference intervals used by the governing-equation residual. Multi-state prediction therefore links the supervised architecture study to temporally refined equation supervision without requiring the outer autoregressive interval to be shortened.

H.6 Temporal-refinement accuracy, spectral fidelity, and cost

The main text reports the principal finite-horizon temporal-refinement results. This appendix compares short and longer rollout horizons, reports long-rollout spectral statistics, and documents representative training and generation costs. All principal curves use 32-channel aligned convolutional models. Kolmogorov models use 96 epochs and learning rate 5×10^{-4}. The matched Hasegawa–Wakatani curves use 96 epochs and learning rate 3×10^{-4}; the separately optimized direct-output curve uses 160 epochs and learning rate 10^{-4} and is visually distinguished throughout.

H.6.1 Short- and longer-horizon field accuracy

Figure 13 compares Kolmogorov errors at 16 and 64 rollout steps. Direct output improves through $M = 8$ at both horizons: the seed-mean error decreases from 0.0324 to 0.0110 at step 16 and from 0.1368 to 0.0996 at step 64. At $M = 16$, both errors and their seed dispersion increase. Time conditioning instead reaches its best region near $M = 2$–4 and deteriorates at $M = 8$. The short-horizon trend therefore largely persists through the longer rollout, while excessive refinement amplifies optimization sensitivity.

Figure 14 shows the corresponding Hasegawa–Wakatani comparison at 4 and 16 rollout steps. Under the matched 96-epoch budget, time conditioning improves through $M = 8$ and remains tightly clustered across seeds. Direct output improves through $M = 4$ but becomes strongly seed dependent at larger M. Longer direct-output training with a smaller learning rate recovers a competitive best run at $M = 8$, but not a competitive seed mean. The distinction between best and average performance grows substantially from the 4-step to the 16-step horizon, showing that time conditioning primarily

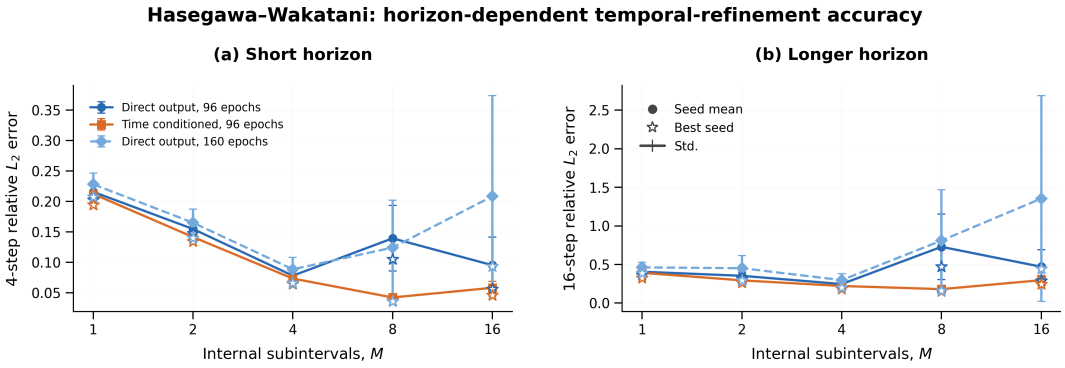

Figure 14: **Hasegawa–Wakatani horizon-dependent temporal-refinement accuracy.** Relative L_2 error after (a) 4 and (b) 16 stored-data forecast steps. Solid curves show the matched 96-epoch results. The dashed light-blue curve shows separately optimized direct-output models trained for 160 epochs with learning rate 10^{-4}. Filled markers show seed means, error bars show sample standard deviations, and open stars show best completed seeds.

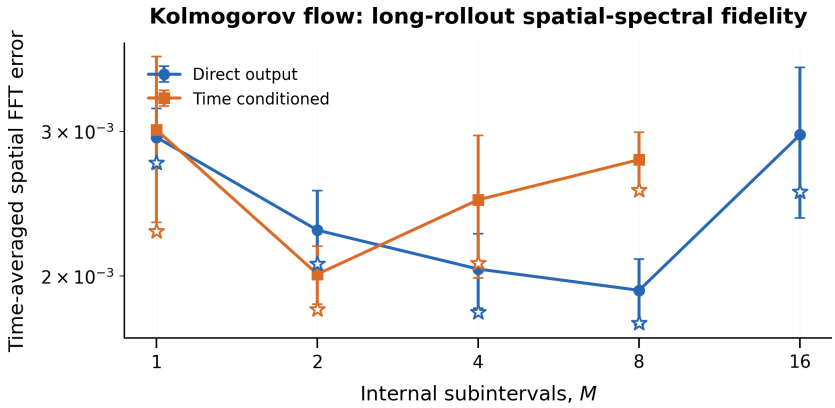

Figure 15: **Kolmogorov long-rollout spatial-spectral fidelity.** Time-averaged spatial FFT error versus the number of internal subintervals for 96-epoch models. Filled markers show seed means, error bars show sample standard deviations, and open stars show best completed seeds.

improves optimization reliability and suppresses autoregressive amplification of poorly optimized runs.

H.6.2 Long-rollout spatial-spectral fidelity

For Kolmogorov flow, improved finite-horizon accuracy is accompanied by improved long-rollout spatial statistics. For direct output, increasing refinement from $M = 1$ to $M = 8$ decreases the seed-average time-averaged spatial FFT error from 2.95×10^{-3} to 1.92×10^{-3} and the best-seed error from 2.75×10^{-3} to 1.75×10^{-3}. Both statistics deteriorate at $M = 16$, consistently with the intermediate optimum in field error. Figure 15 reports the complete 96-epoch comparison.

The Hasegawa–Wakatani state contains density n and electrostatic potential ϕ. Because averaging their FFT errors can conceal field-specific behavior, Figure 16 reports them separately. For the potential, the time-conditioned seed-average FFT error decreases from 0.706 at $M = 1$ to 0.294 at $M = 8$ and 0.278 at $M = 16$. At the two largest refinement levels, time conditioning also lowers the seed-average density-spectrum error relative to direct output: 7.00 versus 22.75 at $M = 8$ and 1.79 versus 6.15 at $M = 16$. The intermediate density results remain strongly seed dependent, so temporal refinement does not yield a uniformly monotonic spectral improvement across both fields. Nevertheless, the finite-horizon benefit of time conditioning at large M is accompanied by improved long-rollout spectral fidelity, particularly for the electrostatic potential.

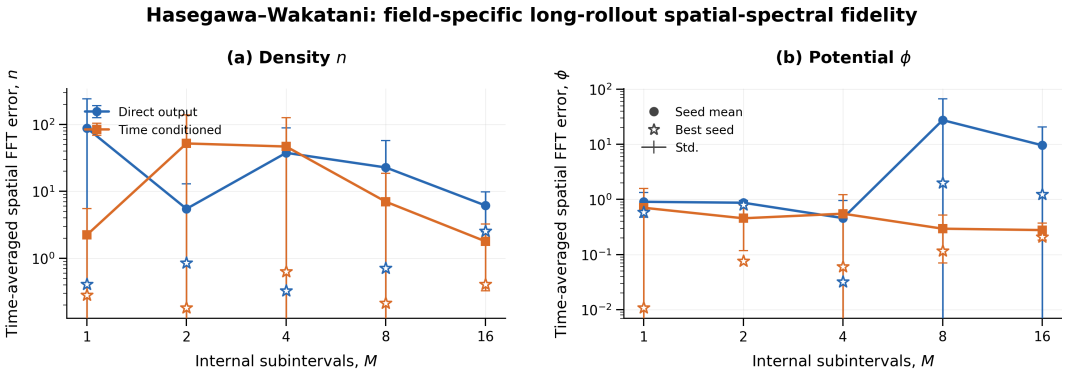

Figure 16: **Hasegawa–Wakatani field-specific long-rollout spatial-spectral fidelity.** Time-averaged spatial FFT error for (a) density n and (b) electrostatic potential ϕ using the matched 96-epoch protocol. Blue denotes direct output and orange denotes time conditioning. Filled markers show seed means, error bars show sample standard deviations, and open stars show best completed seeds. Logarithmic vertical axes accommodate the large seed variability of several configurations.

Table 3: **Representative accuracy and cost for equation-supervised temporal refinement.** Kolmogorov reports 64-step relative L_2 error and 64-step trajectory-generation time; Hasegawa–Wakatani reports 16-step quantities. Times are seed averages. The 160-epoch Hasegawa–Wakatani row is a supplemental optimization setting rather than part of the matched 96-epoch comparison.

System	Output	M	Epochs	Learning rate	Best	Mean $\pm$ std.	Train / generation time
Kolmogorov	Direct	1	96	5×10^{-4}	0.1311	0.1368 ± 0.0085	2.35 h / 2.58 s
Kolmogorov	Direct	8	96	5×10^{-4}	0.0769	0.0996 ± 0.0210	2.32 h / 2.58 s
Kolmogorov	Time conditioned	8	96	5×10^{-4}	0.1524	0.1973 ± 0.0413	3.07 h / 8.68 s
Hasegawa–Wakatani	Direct	1	96	3×10^{-4}	0.3646	0.4062 ± 0.0398	1.81 h / 0.392 s
Hasegawa–Wakatani	Direct	8	96	3×10^{-4}	0.4709	0.7288 ± 0.4253	1.86 h / 0.391 s
Hasegawa–Wakatani	Time conditioned	8	96	3×10^{-4}	0.1517	0.1805 ± 0.0309	2.34 h / 1.075 s
Hasegawa–Wakatani	Direct	8	160	10^{-4}	0.1597	0.8102 ± 0.6582	3.08 h / 0.391 s

H.6.3 Computational cost

Table 3 reports representative operating points. Direct output emits all internal states jointly, so its measured trajectory-generation time is essentially independent of M. Time conditioning reevaluates the conditioned branch for each requested offset and therefore incurs query-dependent generation cost. At $M = 8$, time-conditioned generation is approximately 3.4 times slower than direct output for both systems in the present implementation. For Hasegawa–Wakatani, the additional 160-epoch direct-output optimization recovers a competitive best run but not a competitive seed mean.

H.7 Alignment under equation-supervised temporal refinement

The supervised experiments establish that spatial alignment improves autoregressive forecasting, particularly when one hierarchical model invocation returns several future states. We next test whether alignment remains important when those internal states are trained without future-state targets and are constrained only through the governing equations.

Experimental setting. We consider target-free equation-supervised training on Kolmogorov flow. All models receive full-resolution current states, and no spatial reduction, persistent unresolved-state augmentation, or super-resolution procedure is used in this ablation. One complete model invocation advances the solution over one stored-data interval, Δt_{data}. The model produces M internal future states, which divide this outer interval into M generated subintervals of duration

$$\delta t = \frac{\Delta t_{\mathrm{data}}}{M}. \tag{64}$$

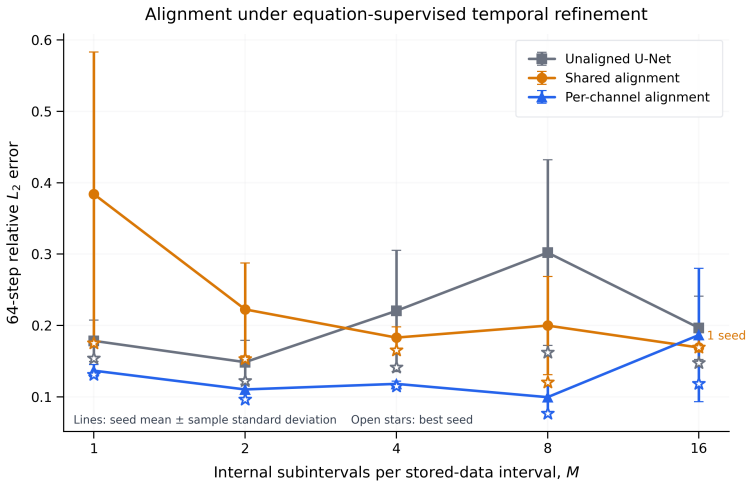

Figure 17: **Alignment under equation-supervised temporal refinement.** Kolmogorov-flow relative L_2 error after 64 stored-data forecast steps as a function of the number M of internally generated subintervals in one stored-data interval. All models use full-resolution current states and are trained without future-state targets. Lines and filled markers show seed means, error bars show sample standard deviations, and open stars show the best completed seed. Per-channel alignment provides the lowest seed-mean error for $M \leq 8$ and allows the model to benefit from temporal refinement through $M = 8$. Shared alignment is less consistent, while the unaligned U-Net deteriorates when the internal output block is extended beyond $M = 2$. The shared-alignment result at $M = 16$ contains one completed seed and therefore has no error bar.

The current state is prepended to the predicted sequence, and the Kolmogorov-equation residual is evaluated between each pair of consecutive generated states. Increasing M therefore refines the temporal discretization used for equation supervision without changing the outer autoregressive interval or the number of complete hierarchy evaluations per stored-data interval.

We evaluate $M \in \{1, 2, 4, 8, 16\}$. All models use 32 hidden channels and a learning rate of 5×10^{-4}. Performance is measured by relative L_2 error after 64 stored-data rollout steps. Except for the shared-alignment configuration at $M = 16$, which currently has one completed run, each point contains three completed seeds. Figure 17 shows seed means, sample standard deviations, and best completed seeds; Table 4 reports the corresponding numerical values.

Alignment variants. We compare three convolutional encoder–decoder models with otherwise matched training settings:

- *Unaligned U-Net*: equal-resolution encoder and decoder representations are fused at matching grid indices without spatial warping.
- *Shared alignment*: at each aligned skip, one decoder-predicted displacement field is applied to all latent encoder channels.
- *Per-channel alignment*: each latent encoder channel receives an independently predicted displacement field before fusion with the decoder representation.

The aligned models use decoder-predicted correspondence at all skip resolutions. Thus, displacement is inferred from the future-oriented decoder path rather than from the input-time encoder feature. The comparison isolates the absence or presence of skip alignment and, within the aligned architecture, whether spatial correspondence is shared across latent channels or resolved independently by channel.

Per-channel alignment enables effective temporal refinement. Per-channel alignment produces the lowest seed-mean error at every refinement level through $M = 8$. Relative to the unaligned U-Net, it reduces the seed-mean 64-step error by 23.4%, 25.7%, 46.4%, and 67.0% for $M = 1, 2, 4$, and 8, respectively. The growing separation through $M = 8$ shows that alignment becomes particularly

Table 4: **Numerical results for the equation-supervised alignment ablation in Figure 17.** Best is the lowest 64-step relative L_2 error among completed seeds. Mean and sample standard deviation are calculated across the reported unique seeds. A standard deviation is not reported for shared alignment at $M = 16$ because only one seed is available.

Model variant	M	Best	Mean $\pm$ std.	Seeds
Unaligned U-Net	1	0.1538	0.1785 ± 0.0290	3
Unaligned U-Net	2	0.1227	0.1487 ± 0.0307	3
Unaligned U-Net	4	0.1413	0.2205 ± 0.0849	3
Unaligned U-Net	8	0.1625	0.3022 ± 0.1301	3
Unaligned U-Net	16	0.1477	0.1967 ± 0.0444	3
Shared alignment	1	0.1754	0.3840 ± 0.1990	3
Shared alignment	2	0.1531	0.2226 ± 0.0650	3
Shared alignment	4	0.1656	0.1831 ± 0.0152	3
Shared alignment	8	0.1206	0.1999 ± 0.0688	3
Shared alignment	16	0.1692	0.1692	1
Per-channel alignment	1	0.1311	0.1368 ± 0.0085	3
Per-channel alignment	2	0.0964	0.1104 ± 0.0143	3
Per-channel alignment	4	0.1150	0.1183 ± 0.0039	3
Per-channel alignment	8	0.0769	0.0996 ± 0.0210	3
Per-channel alignment	16	0.1186	0.1867 ± 0.0934	3

important when one model invocation must construct a longer sequence of physically constrained internal states.

The strongest result occurs at $M = 8$, the best tested refinement level for the per-channel aligned model. Its seed-mean error is 0.0996 ± 0.0210, compared with 0.1999 ± 0.0688 for shared alignment and 0.3022 ± 0.1301 for the unaligned U-Net. Per-channel alignment therefore reduces the mean error by 50.2% relative to shared alignment and 67.0% relative to the unaligned model. The corresponding best-seed errors are 0.0769, 0.1206, and 0.1625, respectively. The per-channel model also has substantially lower seed dispersion than the unaligned model at this refinement level.

Shared alignment is insufficient for consistent improvement. Using one displacement field for every latent channel provides a weaker and more refinement-dependent benefit. Shared alignment is less accurate than the unaligned U-Net at $M = 1$ and $M = 2$, but becomes more accurate at $M = 4$ and $M = 8$. This behavior indicates that the presence of a spatial warp is not by itself sufficient. Latent channels generally encode different features, and constraining all channels to retrieve information from the same source locations can limit the usefulness of alignment. Allowing channel-specific correspondence produces a consistent improvement over both shared alignment and no alignment for $M \leq 8$.

The shared-alignment point at $M = 16$ has only one completed seed and should not be used to rank the variants at that refinement level. The absence of seed replication is indicated explicitly in Figure 17 and Table 4.

Temporal refinement and spatial alignment interact. The three architectures respond differently as the residual interval is refined. The unaligned U-Net improves from $M = 1$ to $M = 2$, but its mean error then rises from 0.1487 at $M = 2$ to 0.2205 at $M = 4$ and 0.3022 at $M = 8$. Temporal refinement alone is therefore insufficient to improve forecasting. Although shorter residual intervals provide a more accurate temporal discretization, the model must jointly construct a longer sequence of internal states. Conventional equal-index skip fusion becomes increasingly restrictive as those states depart from the input configuration.

In contrast, the per-channel aligned model retains low error throughout $M = 2, 4$, and 8 and obtains its best performance at $M = 8$. The result supports an interaction between the two components of the method: temporally resolved equation supervision is most effective when the hierarchical representation also preserves spatial correspondence. Alignment is therefore not only a supervised-forecasting modification. It contributes directly to the model's ability to construct the internally resolved trajectory used by the physics objective.

Table 5: **Kolmogorov training from spatially reduced current-state inputs.** All models use target-free equation supervision, direct output, $M = 8$, 32 hidden channels, learning rate 5×10^{-4}, and 144 epochs. "First" and "final" refer to the first and eighth internally predicted states used to update the persistent prediction-derived unresolved state. The physics-time multiplier is applied only during training and is reset to one for evaluation. Noise levels are relative to the population standard deviation of the clean training field.

r	Grid	Seeds	Best	Mean $\pm$ std.	State update	Time multiplier	Curriculum	Fresh noise
4	64×64	3	0.0935	0.1021 ± 0.0123	First	1	None	None
32	8×8	3	0.0972	0.1148 ± 0.0299	Final	1	None	None
128	2×2	3	0.1097	0.1345 ± 0.0388	Final	$10 \to 1$	20 epochs	None
256	1×1	3	0.1517	0.1947 ± 0.0383	Final	$10 \to 1$	30 epochs	0.1, half-life 5

Excessive refinement remains difficult. Increasing the refinement from $M = 8$ to $M = 16$ does not produce an additional improvement for the per-channel aligned model. Its seed mean rises from 0.0996 to 0.1867, and its sample standard deviation increases from 0.0210 to 0.0934. The large difference between its best-seed error, 0.1186, and seed mean indicates increased optimization sensitivity. Consequently, alignment does not remove the intermediate optimum in temporal refinement. Rather, it enables the model to benefit from refinement over a substantially wider and more useful range. At still larger M, the increased output dimension, number of jointly constrained internal states, and optimization burden can outweigh the benefit of shorter residual intervals.

Conclusion. This ablation confirms that spatial alignment remains important when future states are excluded from the training objective. Per-channel, decoder-predicted alignment consistently provides the strongest results and is particularly important at the temporal refinement level that gives the best equation-supervised accuracy. The experiment therefore links the paper's spatial and temporal contributions: accurate equation supervision over short internal intervals depends on an architecture that can construct those internal states without repeatedly fusing future-oriented decoder representations with spatially stale, conventionally indexed encoder features.

H.8 Reduced-observation results and controls

H.8.1 Dependence on observed spatial resolution

All reduced-observation experiments use target-free equation supervision, direct output, an outer prediction interval of one stored-data interval, eight internal temporal intervals, 32 hidden channels, learning rate 5×10^{-4}, and 144 training epochs. Each training input is block averaged in both spatial directions and restored to the 256×256 model grid by bicubic interpolation with periodic boundary handling. The persistent prediction-derived unresolved state is retained separately for each training example. Final rollouts begin from full-resolution held-out states and use the standard physical-time convention. A representative 1×1 rollout and corresponding spatial spectra are shown in Appendix H.8.2.

Table 5 reports the accuracy and reduction-adapted training settings. Reducing the effective observed grid from 64×64 to 8×8 changes the best 64-step error from 0.0935 to 0.0972, despite a 64-fold decrease in observed spatial values. The error rises gradually to 0.1097 for 2×2 observations and 0.1517 for a spatially uniform 1×1 observation.

Persistent-state update time. For $r = 4$, the persistent state is updated from the first of eight internal predictions. The reduced input retains substantial spatial organization, and the early update limits displacement between that resolved structure and the prediction-derived unresolved state. For $r \in \{32, 128, 256\}$, the state is updated from the eighth and final internal prediction. At these stronger reductions, the observation contains less spatial organization, and the longer evolution interval allows initially missing structure to develop before it is retained.

Physics-time curriculum. For $r = 128$ and $r = 256$, the temporal interval used in the equation residual is enlarged early in training through a multiplicative curriculum. The multiplier decreases linearly from 10 to 1 over the first 20 epochs for $r = 128$ and over the first 30 epochs for $r = 256$. It

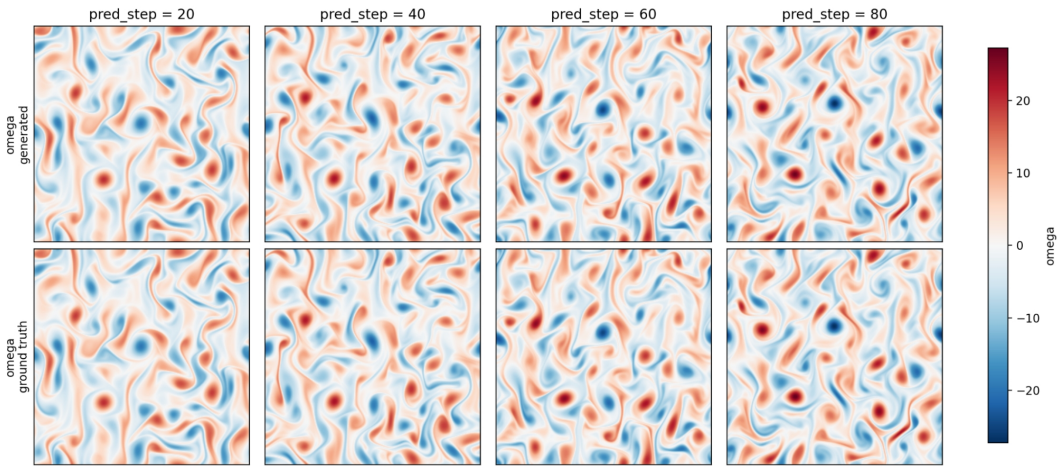

Figure 18: **Kolmogorov trajectory generated after training from spatially uniform current-state observations.** Top: autoregressively generated vorticity fields from the best completed 1×1 reduced-observation run. Bottom: corresponding reference fields from one held-out trajectory. Columns show forecast steps 20, 40, 60, and 80, and all panels use the same vorticity color scale. During training, each supplied current-state observation retains only its spatial mean; decaying Gaussian white noise initially breaks spatial symmetry, and equation supervision together with the persistent prediction-derived unresolved state develops and retains spatial structure. No future-state targets or spatially resolved current-state inputs are used for training. Evaluation is initialized from a full-resolution held-out state and is thereafter fully autoregressive. The displayed example is qualitative; aggregate results over three seeds are reported in Table 5.

remains equal to one for $r = 4$ and $r = 32$. Validation and final autoregressive evaluation always use the standard multiplier of one.

Symmetry breaking at 1×1. For $r = 256$, the reduced and restored observation is spatially uniform. Independent zero-mean Gaussian white noise is added during training to break spatial symmetry. At epoch e, its standard deviation relative to the population standard deviation of the clean training field is

$$\sigma_{\text{noise}}^{(e)} = 0.1\, 2^{-(e-1)/5}. \tag{65}$$

The noise is sampled freshly and is not retained for a particular example. Persistent spatial structure is instead supplied by the prediction-derived unresolved state as it develops during optimization.

The gradual deterioration through the 2×2 observation shows that the governing equations and evolving prediction-derived state supply much of the information required to learn the transition. The larger error at 1×1 indicates that retaining coarse spatial organization remains beneficial, while the useful forecast accuracy obtained from a spatially uniform observation demonstrates the potential of the reduction-adapted procedure.

H.8.2 Qualitative trajectory and spatial spectra from spatially uniform observations

Figure 18 shows an autoregressive rollout from the best completed 1×1 reduced-observation run. During training, each supplied current-state observation contains only its spatial mean and therefore no spatially varying information from the corresponding full-resolution field. Decaying Gaussian white noise breaks spatial symmetry early in training, while equation supervision and the persistent prediction-derived unresolved state allow spatial structure to develop and be retained across optimization. No future-state targets or spatially resolved current-state inputs are supplied during training. As in all reduced-observation experiments, evaluation begins from a full-resolution held-out state and then proceeds autoregressively without later reference fields.

At the plotted scale, the generated vorticity fields are visually almost indistinguishable from the corresponding reference fields through step 80, the final displayed checkpoint. The rollout preserves

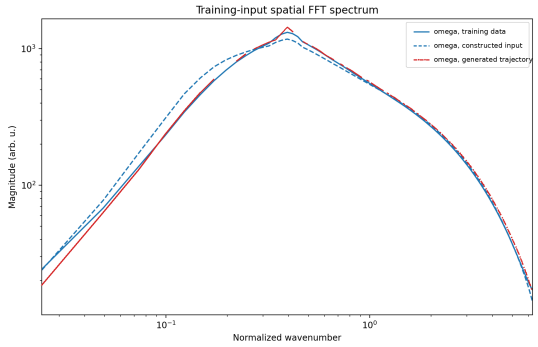

Figure 19: **Spatial spectra of constructed training inputs and generated trajectories in the spatially uniform observation regime.** The solid blue curve shows the ground-truth training-state spectrum, the dashed blue curve shows the spectrum of the effective model inputs at the end of training, and the red dash-dotted curve shows the time-averaged spectrum of the generated 304-step autoregressive rollouts. For the 1×1 observation, the supplied current state retains only its spatial mean, so the spatially varying part of the effective training input is the persistent prediction-derived unresolved state. The generated-rollout spectrum is computed separately for each of the ten held-out trajectories and then averaged.

the locations, signs, amplitudes, and shapes of the dominant vortical structures, together with much of the surrounding filamentary organization. Small differences can be found upon close inspection of individual fine-scale filaments, particularly at the later checkpoints, but no clear loss of spatial organization is apparent in the displayed rollout.

Figure 19 provides a complementary view of the learned spatial structure. By the end of training, the constructed inputs reproduce the principal distribution of vorticity magnitude across spatial scales, despite beginning from observations with no spatial variation. The generated autoregressive trajectories match the reference spectrum even more closely across most represented wavenumbers. Thus, the persistent prediction-derived state supplies training inputs with a realistic multiscale distribution, while the learned transition maintains that distribution during autonomous rollout.

These visualizations are illustrative rather than additional aggregate comparisons: they use the best completed 1×1 model realization, and the field panels show one held-out trajectory. The three-seed finite-horizon errors in Table 5 remain the primary quantitative comparison, while the generated-trajectory spectrum is averaged over all ten held-out rollouts.

H.8.3 Controls without persistent prediction-derived state

We disable the persistent prediction-derived unresolved state and test whether fresh Gaussian perturbations can instead supply useful unresolved variation. The controls use spatial-reduction factors $r = 4$ and $r = 32$, corresponding to effective observed grids of 64×64 and 8×8. All configurations use direct output, eight internal residual intervals, 32 hidden channels, learning rate 5×10^{-4}, batch size two, and 144 epochs. The reduced fields are restored by bicubic interpolation with periodic boundary handling; evaluation begins from full-resolution held-out states and uses neither reduction nor noise.

For noise level λ, independent zero-mean Gaussian noise with standard deviation

$$\sigma_{\mathrm{noise}} = \lambda \, \sigma_{\mathrm{clean,train}} \tag{66}$$

is added to the reduced and restored input, where $\sigma_{\mathrm{clean,train}}$ is the population standard deviation of the clean training field. The perturbation is freshly sampled and has constant amplitude throughout training. We test $\lambda \in \{0, 0.0125, 0.025, 0.05, 0.1, 0.2\}$ for both reductions and additionally $\lambda = 1.0$ for $r = 4$. Each configuration uses three seeds.

At $r = 4$, performance varies nonmonotonically with noise amplitude. The seed-mean error decreases from 1.1140 without noise to 0.2443 ± 0.0262 at $\lambda = 0.025$, a 78.1% reduction, then increases for larger amplitudes. Moderate fresh variation therefore facilitates equation-supervised learning, but

Table 6: **Fresh-noise controls without persistent prediction-derived state.** Noise level is relative to the clean training-field standard deviation. Best is the lowest 64-step relative L_2 error among three seeds; mean and sample standard deviation are across seeds.

Reduction r	Noise level	Best	Mean $\pm$ std.
4	0	1.0644	1.1140 ± 0.0645
4	0.0125	0.2721	0.3461 ± 0.0900
4	0.025	**0.2271**	$\mathbf{0.2443 \pm 0.0262}$
4	0.05	0.2821	0.3029 ± 0.0235
4	0.1	0.5460	0.5622 ± 0.0153
4	0.2	0.8598	1.0153 ± 0.2217
4	1.0	1.1546	1.3619 ± 0.3069
32	0	82.0755	$(1.02 \pm 1.76) \times 10^{14}$
32	0.0125	12.6456	$(2.13 \pm 2.23) \times 10^{4}$
32	0.025	2.3955	13.3879 ± 15.8807
32	0.05	1.1657	$(3.92 \pm 6.78) \times 10^{3}$
32	0.1	1.1073	58.6468 ± 99.6438
32	0.2	**1.0450**	538.7817 ± 931.2583

only over a finite amplitude range. The best fresh-noise control remains substantially worse than the matched persistent-state result, 0.1021 ± 0.0123.

At $r = 32$, increasing noise improves the best individual runs, but at least one seed fails severely at every tested amplitude. The best-seed error reaches 1.0450 at $\lambda = 0.2$, while the corresponding mean is 538.8 ± 931.3; at $\lambda = 0.025$, the best seed is 2.3955 and the mean is 13.39 ± 15.88. Thus, fresh perturbations can occasionally yield a finite rollout from 8×8 observations but do not provide robust learning. The matched persistent-state model instead obtains 0.1148 ± 0.0299.

Fresh noise supplies spatial variation but is resampled independently, is not specific to a training example, and does not adapt to the model's generated dynamics. The comparison therefore supports persistence and prediction-derived adaptation as important under substantial spatial information loss, rather than attributing the principal result merely to high-frequency input variation.