KOOPFM: BASE-ANCHORED FLOW MATCHING FOR LONG-TERM EMULATION OF FORCED DYNAMICAL SYSTEMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Long-term rollouts of chaotic dynamical systems are critical across computational science, from turbulence to climate projection. Under time-varying external forcing, the system transitions from an initial-value problem to a boundary condition problem, in which long-run statistics are set by the forcing trajectory. Neural operators are increasingly being used in this setting, but deterministic models are spectrally biased and may drift, whereas generative ones fix the spectra issue but transport mass from isotropic Gaussian noise. Thus, the vector field must rebuild global structure, forcing response and fine-scale turbulence at once, at many network evaluations per step. Motivated by scale separation and the Mori–Zwanzig formalism, we propose **KoopFM**, which couples a Koopman-autoencoder base model with *base-anchored flow matching*, where we replace the standard Gaussian source of the probability path with a source centered on the base prediction, so the residual travels a much shorter path. We identify the source variance σ as the central mechanism, as it prevents degeneracy and determines that path length, as well as the correction range of the flow. Across three chaotic dynamical systems, we show that the anchoring construction has two main advantages. First, whenever the one-step conditional is narrow, inference needs as few as two NFEs without any distillation. Second, the damped, forcing-aware Koopman anchor improves the out-of-distribution forcing response on the chaotic Lorenz–96 system, and on forced Rayleigh–Bénard convection the anchored model outperforms vanilla flow matching beyond the training range of the forcing, up to a metric-dependent limit.

1 INTRODUCTION

Simulating the long-term evolution of chaotic, multiscale dynamical systems is a foundational challenge across computational physics, engineering and especially climate science, where policy-relevant scenario studies need large ensembles of forced, multi-decadal projections that are too costly to run at high resolution (Eyring et al., 2024). Machine-learned surrogates of dynamical systems (Li et al., 2021; Lu et al., 2021; Li et al., 2022; Lam et al., 2023; Bi et al., 2023; Price et al., 2025) show remarkable short-term predictive skill, but sustained long-horizon emulation remains difficult, because small errors such as inaccurate spectra can accumulate into structural smearing or runaway instability in autoregressive settings (Nathaniel et al., 2024; Bonavita, 2023; Lehmann et al., 2026). The difficulty compounds when the system is driven by a time-dependent forcing, such as a transient boundary heat flux or atmospheric greenhouse-gas concentrations (Watt-Meyer et al., 2024; Kochkov et al., 2024). The focus then shifts from a classical initial-value problem to a boundary condition problem, in which the long-run statistical equilibrium is governed by the forcing trajectory rather than by the initial condition. As in climate projection, an emulator must then model invariant measures, spectra and forcing response over complete trajectories instead of minimizing pointwise errors, and evaluation should target *forcing response*, *multi-thousand-step rollout stability*, and *multiscale expressiveness*.

Deterministic operators trained with a pointwise Mean-Squared Error (MSE) loss regress toward the conditional mean, suppressing high-wavenumber variability and yielding overly smooth, implausible, or divergent long-term rollouts (Rahaman et al., 2019; Lam et al., 2023; Bonavita, 2023). Diffusion and flow matching models instead learn a distributional denoising or transport objective,

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

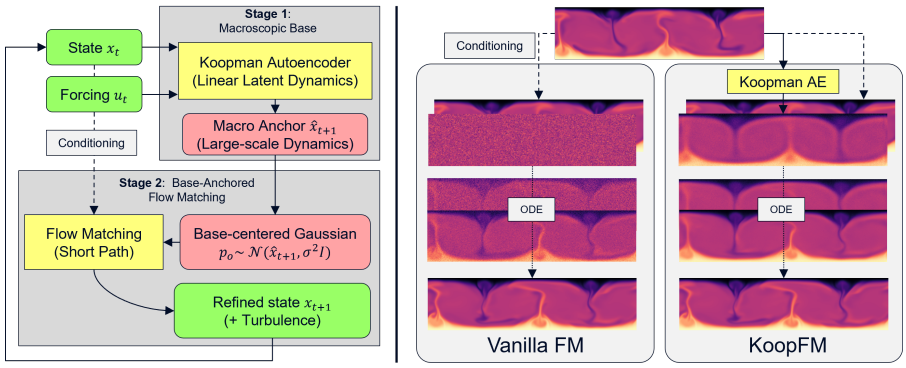


Figure 1: **Base-anchored flow matching.** *Left:* Stage 1 provides a bounded and forcing-aware anchor $\hat{x}_{t+1}$; Stage 2 then transports from a Gaussian centered on that anchor to the (full) next state, which is re-encoded at the next step to close the autoregressive loop. *Right:* a visualization of the sampling procedure. A standard Gaussian source has to rebuild the field; the anchored source starts from an already physical, forcing-aligned state, so the transport only has to correct what the anchor is missing.

avoiding this conditional-mean collapse (Kohl et al., 2024; Lippe et al., 2023; Price et al., 2025). However, standard formulations transport isotropic Gaussian noise to the data distribution, requiring the learned dynamics to reconstruct global constraints, forced responses, and fine-scale eddies along a long, costly path that can drift from the physical attractor.

We argue instead that a generative emulator for chaotic dynamics is better *anchored* to damped base dynamics. Motivated by the Mori–Zwanzig formalism (Section 2), we split the forced dynamics $F(x, u)$ into a deterministic macroscopic Markovian evolution $D(x, u)$ and an unresolved chaotic fluctuation $S(x)$. **KoopFM** (Figure 1) is structured around this decomposition. Stage 1 is a Koopman autoencoder that maps the state into a latent space where the dynamics and the parametric forcing are approximately linearized, giving a bounded macroscopic prediction $\hat{x}_{t+1}$ that acts as an anchor on a damped resolved manifold. Stage 2 then defines a flow-matching probability path initialized at $p_0 = \mathcal{N}(\hat{x}_{t+1}, \sigma^2 I)$ rather than at unstructured noise. The residual path is then roughly $6\times$ shorter than the Gaussian one, and the unresolved fluctuations ξ are injected stochastically along it.

Contributions. We propose KoopFM as an instantiation of base-anchored flow matching for forced chaotic systems, which shows two main advantages: whenever the one-step conditional is narrow, it requires as few as two function evaluations (NFEs), effectively minimizing the inference overhead of generative models (Table 2b). Additionally, the anchored model improves held-out forcing response and shows improved extrapolation performance beyond the training range up to a metric-dependent boundary. We also identify the source noise σ as central: it prevents a training degeneracy (Section 4.3) and sets the scope of the flow to correct the base prediction.

2 Background

Forced chaotic systems. The systems considered here are parameterized and non-autonomous, defined on a spatial domain $\Omega \subset \mathbb{R}^d$ with state $x(t) \in \mathcal{X} \subseteq \mathbb{R}^{d_x}$ and driven by an exogenous, time-varying forcing $u(t) \in \mathcal{U}$:

$$\frac{\mathrm{d}x}{\mathrm{d}t} = F(x(t), u(t)), \qquad x(0) = x_0. \tag{1}$$

Memory of x_0 decays over chaotic mixing timescales, so long-horizon statistics are governed by the forcing trajectory $u(t)$ rather than by the initial condition. A surrogate must therefore reproduce the *forcing response*, and not merely a single trajectory.

Koopman operator with exogenous inputs. Koopman operator theory (Koopman, 1931; Mezić, 2005; Brunton et al., 2021) lifts non-linear state dynamics to an infinite-dimensional Hilbert space

of observables where temporal evolution is linear. In data-driven approximations, an encoder $\phi_E : \mathcal{X} \to \mathcal{Z} \subseteq \mathbb{R}^{d_z}$ ($d_z \ll d_x$) embeds the state into a truncated latent space, while an embedding network $\psi : \mathcal{U} \to \mathbb{R}^{d'_u}$ encodes the forcing:

$$z_{t+1} = Az_t + G\,\psi(u_t), \qquad \hat{x}_{t+1} = \phi_D(z_{t+1}), \tag{2}$$

with decoder ϕ_D, discrete Koopman operator $A \in \mathbb{R}^{d_z \times d_z}$, and $G \in \mathbb{R}^{d_z \times d'_u}$ a latent Green's response matrix capturing actuation by the external forcing (Proctor et al., 2014; Gin et al., 2021).

Mori–Zwanzig formalism and scale separation. For a projection of a high-dimensional system onto a truncated resolved subspace $\mathcal{Z}$, the exact evolution of the resolved observables obeys the Mori–Zwanzig identity (Mori, 1965; Zwanzig, 2001; Lin & Lu, 2021):

$$\frac{\mathrm{d}\hat{x}}{\mathrm{d}t} = \underbrace{\Omega(\hat{x})}_{\text{Markovian (resolved)}} + \underbrace{\int_0^t K(t-s, \hat{x}(s))\,\mathrm{d}s}_{\text{memory kernel}} + \underbrace{\xi(t)}_{\text{orthogonal noise (subgrid)}}. \tag{3}$$

A finite-dimensional Koopman model parameterizes the resolved Markovian term $\Omega(\hat{x}) \approx D(x, u)$. We do not parameterize the memory kernel K, representing the delayed interaction of the resolved state with the marginalized unresolved scales. Because Stage 1 is re-encoded from the full generated state at every step, the emulator is Markov in the full state, and whatever the one-step Koopman map misses, memory and noise alike, is left to Stage 2. In Section 3 we contrast our treatment of K with the published Koopman–Mori–Zwanzig closures, which instead parameterize it explicitly and drop ξ.

Conditional flow matching. Flow matching (Lipman et al., 2023; Tong et al., 2024) trains a time-dependent vector field $v_\theta(x_\tau, \tau)$ along an internal generative trajectory $\tau \in [0, 1]$ connecting a source p_0 to a target p_1. Under the affine path $x_\tau = (1-\tau)x_0 + \tau x_1$ the conditional velocity is $\dot{x}_\tau = x_1 - x_0$, and

$$\mathcal{L}_{\mathrm{FM}}(\theta) = \mathbb{E}_{\tau, x_0, x_1} \left\| v_\theta(x_\tau, \tau, c) - (x_1 - x_0) \right\|_2^2, \tag{4}$$

with c an optional conditioning input. Sampling integrates $\mathrm{d}x/\mathrm{d}\tau = v_\theta$ from $\tau = 0$ to 1. The number of function evaluations (NFE) required is set by how curved and how long the path is.

3 Related Work

Neural PDE surrogates, deterministic and generative. Operator learning architectures (Li et al., 2021; 2022; Lu et al., 2021) are accurate over short horizons, but minimize a conditional mean and are therefore spectrally biased (Rahaman et al., 2019), with errors that compound autoregressively (Brandstetter et al., 2023). Deterministic remedies have been architectural or corrective (Zhang et al., 2024; Buitrago Ruiz et al., 2025; Cao et al., 2025; Kochkov et al., 2021; Wang et al., 2024). Generative surrogates attack the same bias from the sampling side (Kohl et al., 2024; Wang et al., 2025; Kollovieh et al., 2025; Hu et al., 2025; Kassaï Koupaï et al., 2025; Shysheya et al., 2024; Pedersen et al., 2025; Lienen et al., 2024). The learned correction of Kochkov et al. (2021) is the closest deterministic framing to our work. Ours is corrective in the same sense, but the correction is generative and drawn from a source centered on the state on the resolved manifold.

Combining deterministic predictors with generative models. Given a deterministic predictor and a generative model, the literature evolved around three couplings: conditioning the generative model on the deterministic prediction while still starting from noise (Oommen et al., 2025; Shysheya et al., 2024), most closely Cohesion (Nathaniel & Gentine, 2026), which guides a diffusion model with a deep-Koopman prior of the coherent flow; refining along a noise ladder trained on perturbations of the ground truth, as in PDE-Refiner (Lippe et al., 2023) and FlowRefiner (Dai et al., 2026); and, as here, starting the transport at or near the deterministic prediction. The latter is well established: Grad-TTS decodes from a Gaussian centered on an encoder output (Popov et al., 2021), ResShift shifts the residual from a degraded to a clean image (Yue et al., 2023), and stochastic interpolants, Schrödinger bridges and inversion by direct iteration transport between arbitrary or data-dependent pairs (Albergo et al., 2025; 2024; Liu et al., 2023; Delbracio & Milanfar,

2024). Residual generative models do the same in residual coordinates: CorrDiff diffuses a regression model's residual (Mardani et al., 2025), and ArchesWeatherGen applies flow matching to the unit-variance residual of a deterministic forecast (Couairon et al., 2026), carried to decadal climate emulation by ArchesClimate (Clyne et al., 2025); in state space this is a Gaussian source centered on the forecast with the residual's standard deviation. Adaptive flow matching (Fotiadis et al., 2025) likewise sets the noise from an encoder's error, for super-resolution without autoregression. We add a forcing-aware Koopman anchor re-encoded at every step, an analysis of the source width, and an evaluation of forcing response and extrapolation. Near-deterministic sources have known failure modes, ill-conditioning as the noise vanishes (Zeng & Yan, 2025) and endpoint underfitting in diffusion bridges (Gao et al., 2026); we identify a further one specific to anchored sources (Section 4.3), and PDE-Refiner likewise finds the noise level to be its critical hyperparameter.

Koopman operators. Modern Koopman autoencoders (Lusch et al., 2018; Morton et al., 2018; Grozavescu et al., 2026) *learn* the lifting coordinates of Koopman operator theory (Koopman, 1931; Brunton et al., 2021), and with control inputs (Proctor et al., 2014) model parametric response. Closest to Stage 1 are Gupta et al. (2025) and Lu et al. (2026), who add an explicit Mori–Zwanzig memory term and a recurrent-memory closure respectively; both instead parameterize the memory convolution of Eq. 3 deterministically and drop ξ, which we model with Stage 2.

4 METHOD

4.1 STAGE 1: A CHEAP, STABLE BASE PREDICTOR

Anchoring needs a base predictor that is fast, stable under repeated application, and responsive to the forcing. We use a Koopman-style autoencoder (Lusch et al., 2018; Morton et al., 2018): the encoders ϕ_E, ψ lift the state and the forcing into a latent space and propagate them as in Eq. 2, with an added bias b. We parameterize A as block-diagonal, with independent 2×2 blocks, such that each block acts as a damped rotation with modulus $\gamma_k \in (0, 1)$, preventing runaway under repeated latent application. We note, however, that we do not unroll in latent space, as is typically done for Koopman operators, but the autoregressive loop always includes Stage 2 to model the unresolved effects (Figure 1), as in the autoregressive mode of Nathaniel & Gentine (2026), so the damping regularizes the anchor but gives no stability guarantee for the emulator. The encoder is convolutional, and its output defines two variants: ***KoopFM-vec*** flattens it into a single latent vector, used for 1D systems such as Lorenz–96 (Section 5.1), while ***KoopFM-field*** skips the final flattening layer and keeps the latent spatial axes, so the latent is a coarse grid of feature vectors. KoopFM-field is the default for Kolmogorov flow and Rayleigh–Bénard (Sections 5.2, 5.3), where we also report KoopFM-vec for comparison (Section 5.3). In KoopFM-field, A acts independently at each latent site, and its block parameters are tied along directions in which the system is homogeneous and left free along directions in which it is not. Each site can thus carry its own timescale while the operator stays equivariant under the symmetry the system actually has. Which axes are which is a property of the system, not of the method (detailed per-system shapes and parameter counts are in Appendix E.3). Stage 1 is pre-trained and then frozen, which exposes the encoder to a train–test gap at inference time, where it receives generated samples rather than ground truth. To make the encoder more robust, we corrupt the encoder's input with Gaussian noise on a cosine ramp (Algorithm 1) during training.

We note that nothing in Section 4.2 requires latent linearity, and the anchor is deliberately low-capacity and spectrally deficient: rolled out alone, its performance is much worse than any full model (Table 16). Stage 2 is then meant to supply the missing high-frequency content.

4.2 STAGE 2: ANCHORED FLOW MATCHING

We define the source at $\tau = 0$ as a Gaussian centered on the base prediction,

$$p_0(x_0 \mid \hat{x}_{t+1}) = \mathcal{N}\big(\hat{x}_{t+1}, \mathrm{diag}(\sigma)^2\big), \qquad p_1 = \delta(x_1 - x_{t+1}), \tag{5}$$

with σ an online estimate of the anchor's residual RMS per channel, which KoopFM-field further modulates per grid point (Algorithm 2). We train v_θ with the objective in Eq. 4. The network is conditioned on the previous state x_t and on the base prediction $\hat{x}_{t+1}$ by channel concatenation, with

τ and u_t injected through adaptive group normalization (Dhariwal & Nichol, 2021). Therefore, the base prediction enters in two ways: as the starting point of the flow and as a conditioning signal. Conditioning the network on the (unnoised) base prediction matters because the minimizer of Eq. 4 is the conditional expectation of the regression target given everything the network is shown (Lipman et al., 2023; Albergo et al., 2025; 2024),

$$v_\theta^\star(x_\tau, \tau, c) = \mathbb{E}[\, x_1 - x_0 \mid x_\tau, \tau, c\,], \tag{6}$$

and reading this at the end of the path, $\tau \to 1$, is what separates the two source choices.

For a standard Gaussian source with $x_0 \sim \mathcal{N}(0, I)$ the source is independent of the data, so as $x_\tau \to x_1$ we get $\mathbb{E}[x_0 \mid x_\tau] \to 0$ and $v_\theta^\star(x_1) = x_1$. The optimal field at the end of the path points away from the origin, and the network is never required to represent the source at all. With the anchored source, in contrast, p_0 is centered on $\hat{x}_{t+1}$, so x_0 is strongly correlated with x_1, and the same limit reads $v_\theta^\star(x_1) = x_1 - \mathbb{E}[x_0 \mid x_1, x_t, u_t]$. As in standard conditional flow matching, training pairs x_0 and x_1 under an independent coupling, so $x_0 \perp x_1 \mid x_t, u_t$ and $\mathbb{E}[x_0 \mid x_1, x_t, u_t] = \mathbb{E}[x_0 \mid x_t, u_t]$. The base predictor is a deterministic function of that conditioning, so this equals $\hat{x}_{t+1}$ and the optimal field is the residual $x_{t+1} - \hat{x}_{t+1}$ (for a non-degenerate conditional; Appendix A.2). A network given (x_t, u_t) could in principle recompute $\hat{x}_{t+1}$; supplying it spares the network from re-learning Stage 1. This is the same construction as residual generative models that start from a degraded observation or a deterministic forecast (Delbracio & Milanfar, 2024; Yue et al., 2023; Couairon et al., 2026). Concretely, the anchor's own offset is $\|x_{t+1} - \hat{x}_{t+1}\|/\|x_{t+1}\| \approx 0.16$, against $\approx \sqrt{2}$ for a standard Gaussian source, making the anchored transport roughly $6\times$ shorter (see Appendix A.2 for details), which directly translates to solver efficiency (Table 2b).

Stage 2 models the full one-step conditional. Stage 1 supplies a point estimate of the one-step conditional, $\hat{x}_{t+1} = \mathbb{E}_\phi[x_{t+1} \mid x_t, u_t]$. Stage 2 models that conditional in full. Sampling the ODE from $x_0 \sim p_0(\cdot \mid \hat{x}_{t+1})$ to $\tau = 1$ draws from $p_\theta(x_{t+1} \mid x_t, u_t)$, with the anchor entering only as the location of the source and as conditioning. A rollout composes these one-step conditionals, re-encoding the refined state at each step so that generated small-scale structure feeds back into the base trajectory (Figure 2); pseudocode is in Appendix A.1.

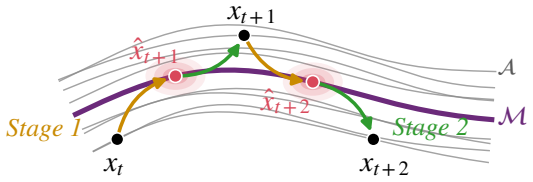


Figure 2: **Rollout.** Stage 1 re-encodes onto the resolved manifold $\mathcal{M}$ and steps to $\hat{x}_{t+1}$; Stage 2 flows from p_0 (shaded) into the attractor $\mathcal{A}$.

4.3 Why source noise is necessary

The simplest anchored source dispenses with σ entirely, $p_0 = \delta(x_0 - \hat{x}_{t+1})$, which gives a fully deterministic path. In principle, this is a legitimate construction, since a low-capacity anchor acts as a corruption operator and Bansal et al. (2023) show that diffusion-style models can be built on arbitrary deterministic corruptions. The path then becomes $x_\tau = \hat{x}_{t+1} + \tau\,(x_{t+1} - \hat{x}_{t+1})$, so

$$x_\tau - \hat{x}_{t+1} = \tau\,(x_1 - x_0) \quad \Longrightarrow \quad \dot{x}_\tau = \frac{x_\tau - \hat{x}_{t+1}}{\tau}. \tag{7}$$

However, since the network now receives $\hat{x}_{t+1}$ as an input channel and τ as conditioning, the regression target is recoverable by arithmetic for every τ bounded away from zero. Only $\tau \approx 0$ is genuinely hard, and that is precisely where the sampling ODE begins. We observe this problem in practice when training with $\sigma = 0$, which gives the lowest training loss in this study and the worst model (Table 3).

How to choose σ. We track σ as an exponential moving average of the anchor's own one-step residual RMS, following the adaptive noise scaling of Fotiadis et al. (2025). It measures how wrong the anchor already is, not a physical spread of outcomes, and thereby sets the flow's correction range, i.e. how far from the anchor it can move samples. A more accurate anchor can therefore give a worse emulator, so σ is less a hyperparameter to tune than a guide to how expressive the anchor should be (Appendices B.1, B.2).

5 EXPERIMENTS AND RESULTS

We evaluate our algorithm on three systems of increasing complexity, with the aim of showing when anchoring the source on a reduced-order base prediction helps. The first is the two-tier Lorenz–96, where forcing response is measured at constant forcing. The second is the Kolmogorov flow of StocBench (Pfister et al., 2026) in two variants that differ only in whether the forcing is stochastic, which lets us vary the width of the one-step conditional directly and locate where the anchor's advantage holds and where it does not. The third is forced 2D Rayleigh–Bénard convection under transient forcing, the primary benchmark and the most complex of the three, where both the efficiency and the extrapolation advantages converge. Here, the forcing response is measured under transient rather than constant forcing, complementing the Lorenz–96 setup.

Baselines. We compare our model to deterministic operators: a U-Net (Ronneberger et al., 2015) and an FNO (Li et al., 2021), trained either as 1-step models or with 4-step backpropagation through time (BPTT) (List et al., 2023; Brandstetter et al., 2023), whenever the training budget allows. As generative baselines we use vanilla conditional flow matching from a Gaussian source, in a setup akin to Kohl et al. (2024); and PDE-Refiner (Lippe et al., 2023) with the default of three refinement steps and $\sigma_{\min}$ set to their published values on 1D (2×10^{-3}) and 2D (3.2×10^{-2}) systems. On Rayleigh–Bénard we add Koop-Res, a deterministic additive-residual model sharing KoopFM's frozen base, which isolates the flow's contribution over a plain residual regression. Parameters and training compute are matched across models where possible (Appendix E.3).

5.1 TWO-TIER LORENZ–96

The two-tier Lorenz–96 (Lorenz, 2006) tests the long-term response at constant forcing, separating the forcing level from the elapsed time. A slow tier X_k ($K = 36$) is coupled to $J = 10$ fast variables Y per site, with forcing applied to the slow variables. Training uses four forcing schedules within $F \in [6, 12]$; evaluation holds F constant throughout each rollout, over 17 forcing levels (five initial conditions each, 85 trajectories total), and we estimate the long-run energy response from the final 80% of each rollout, with $F > 12$ defining strict high-forcing extrapolation. The FNO and U-Net baselines are trained with BPTT-4, whereas Vanilla CFM, PDE-Refiner ($\sigma_{\min} = 2 \times 10^{-3}$), and KoopFM-vec are trained in single-step fashion. The primary metrics are MAEs of the long-run slow- and fast-tier mean-energy responses, reported over all forcing levels and over $F > 12$, alongside trajectory RMSE and joint energy distance (ED) on the same trajectories; all are lower-is-better. Full protocol, metric definitions, uncertainty estimates, and parameter counts are in Appendix C.

Long-run forcing response. Figure 3 presents the results on Lorenz–96. PDE-Refiner achieves the lowest trajectory RMSE and therefore tracks trajectories most accurately, while KoopFM shows the lowest joint ED, as well as the strongest response (lowest MAE) over both slow X and fast Y variables. For strict high-forcing extrapolation with $F > 12$, that winning margin over baselines increases further: every other model shows substantially increased response MAE, in particular for slow variables. KoopFM only marginally degrades here, showing strong extrapolation to unseen forcing regimes.

5.2 KOLMOGOROV FLOW, WITH AND WITHOUT STOCHASTIC FORCING

Our second system is the two-dimensional Kolmogorov flow of StocBench (Pfister et al., 2026), which provides two settings that differ *only* in whether the forcing is observed. The deterministic and stochastic versions directly translate to narrow and broad one-step conditionals and let us test when the short-path claim of Section 4.2 holds.

Benchmark and metrics. StocBench simulates the vorticity on a periodic square, driven by a random forcing on four fixed wavenumbers, under two prediction tasks that differ in whether that forcing is observed: in the *stochastic* task, the next state is random; in the *deterministic* task the forcing is held fixed over each prediction interval and given as input, so the next state is fully determined. We report the relative L_2 error of the conditional mean and spread against a reference ensemble from the solver, and, over 50 autoregressive steps, the error of the ensemble-averaged

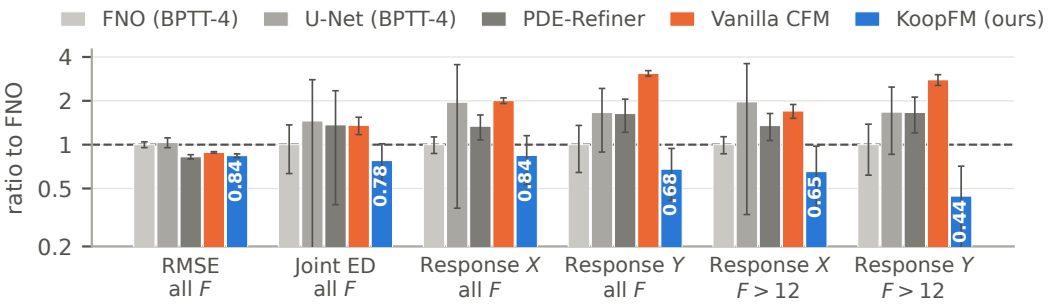


Figure 3: **Long-run response under constant forcing in Lorenz–96**, relative to FNO (BPTT-4), the strongest baseline on the response metrics (dashed; lower is better); $F > 12$ is strict extrapolation. Three-training-seed means $\pm$ std at 50 NFE; numbers are KoopFM's ratio. Absolute values: Table 8.

enstrophy spectrum. On the deterministic task we additionally report RMSE against the solver trajectory, tabulated alongside the mean error in Table 2a below. Full curves for both metrics, against NFE and over the rollout, are in Appendix D (Figures 5, 6). We compare KoopFM with Vanilla CFM at inference budgets from 1 to 50 NFE on both tasks, and with the U-Net, FNO and PDE-Refiner ($\sigma_{\min} = 3.2 \times 10^{-2}$) baselines on the deterministic one, using KoopFM-field.

Strong stochasticity limits the anchor. With unobserved forcing in the stochastic version of StocBench, anchoring brings no benefit. KoopFM trails Vanilla CFM slightly at every inference NFE budget (Figure 5a,b,d), and the efficiency advantage disappears. Both underestimate the conditional spread and consequently lose enstrophy during rollouts (Figure 5c,e,f). The reason is the source variance: the one-step distribution is broad, and a deterministic anchor can at best predict its conditional mean, so σ scales up with that spread. The source noise in our method is isotropic while the target spread is physically structured, so the transport is no longer a small correction.

Weak stochasticity favors the anchor. With the forcing observed, the aleatoric spread vanishes and the anchor's residual is small again. From 2 NFE on, KoopFM comes within 5% of the best baseline's one-step mean error (PDE-Refiner; 0.0191 vs. 0.0181), whereas Vanilla CFM stays above U-Net and PDE-Refiner at any budget (Table 2a, Figure 6a). On rollout RMSE and spurious predictive spread it is again ahead of Vanilla CFM and close to the deterministic baselines (Figure 6b,c). The RMSE saturates for every model by step $\approx$40 (Figure 6d), and on the enstrophy spectrum generative models trail the deterministic ones, with the anchored model again better than Vanilla CFM (Figure 6e,f). Since the two variants differ only in the forcing, this confirms the predicted boundary: the low-NFE property is a property of narrow one-step conditionals rather than of anchoring as such, which we test again on the next system.

5.3 Forced Rayleigh–Bénard convection

Benchmark and metrics. We simulate non-stationary forced Rayleigh–Bénard convection with the pseudo-spectral solver Dedalus (Burns et al., 2020), with the resolution $(N_x, N_z) = (512, 128)$, domain and discretization of The Well (Ohana et al., 2025), at Prandtl number $Pr = 1$, driven by a time-varying Rayleigh number $Ra(t) \in [10^6, 10^8]$. Training covers monotone ramps, plateaus and peak-and-decline schedules; results are reported on two held-out schedules, $\mathrm{RBC}_{\text{inc}}$ and $\mathrm{RBC}_{\text{inc-dec}}$, simulated from ten and nine independent random initial conditions (ICs), respectively (Appendix E.1). We observe significant differences between the trajectories of individual ICs, so standard deviations over ICs are not informative. Instead, we perform a one-sided exact paired sign-flip test, which differences models *per IC*, canceling the shared difficulty of that condition: for each metric, we bold the best entry and other entries the test cannot separate from it ($p > 0.05$). All models are trained 1-step, except FNO, which is cheap enough per step to be additionally trained with 4-step BPTT at matched compute (Appendix E.3). PDE-Refiner is run at two validation-selected $\sigma_{\min}$ settings (Appendix E.5). Our default is KoopFM-field (Section 4.1); we additionally report KoopFM-vec, since the two latents trade off differently across metrics below. We report State L_2 as the single-draw relative L_2 over the first 50 autoregressive steps, W_1 as the Wasserstein distance

Table 1: **Whole-trajectory statistics, autoregressive.** Full trajectory ($T = 4790$), mean across initial conditions; lower is better. **Bold** marks the best entry and every entry an exact paired sign-flip test cannot separate from it (per-IC test; details in Appendix E.7). Generated with 25 NFE for Vanilla CFM, 10 for KoopFM.

Model	$\mathrm{RBC}_{\mathrm{inc}}$					$\mathrm{RBC}_{\mathrm{inc\text{-}dec}}$				
	W1(Nu)	W1(Re)	W1(p)	LSD(E)	LSD(p)	W1(Nu)	W1(Re)	W1(p)	LSD(E)	LSD(p)
FNO (BPTT-4)	0.6161	6.73	2.45	**0.3352**	1.1565	0.2001	5.88	**2.07**	0.3085	1.0186
FNO (1-step)	4.7820	236.46	45.36	0.8190	1.3266	3.2877	186.48	23.24	0.5865	1.1946
U-Net (1-step)	**0.2137**	**4.88**	2.38	**0.3269**	0.8484	0.2460	4.85	2.24	0.3056	0.7050
Vanilla CFM	**0.2463**	**4.86**	2.70	**0.3287**	0.6852	**0.1800**	**3.33**	2.50	**0.3020**	0.5164
PDE-Refiner ($2 \cdot 10^{-3}$)	**0.2524**	5.83	2.95	**0.3343**	**0.2762**	0.2589	6.02	2.85	0.3081	**0.2361**
PDE-Refiner ($3.2 \cdot 10^{-2}$)	0.2768	**4.52**	2.37	0.3341	0.3839	0.3095	4.06	2.22	0.3072	0.2995
Koop-Res	0.2605	5.40	2.29	**0.3295**	0.7839	0.2819	**3.71**	2.19	0.3014	0.6326
KoopFM-vec (ours)	**0.2343**	7.17	2.47	**0.3298**	0.5025	**0.1949**	4.90	2.72	0.3078	0.3942
KoopFM-field (ours)	0.3052	5.28	**2.16**	**0.3283**	0.4312	0.2768	**3.19**	**2.06**	**0.2974**	0.3409

Table 2: **Performance over solver budget.** KoopFM at NFE 2 matches or beats Vanilla CFM's largest budget. **(a)** Deterministic Kolmogorov flow; best baseline (PDE-Refiner): mean error 0.0181, RMSE 0.716. **(b)** Rayleigh–Bénard, autoregressive, mean over both scenarios; $W_1(p)$ is flat in NFE. Buoyancy ladder to 200 NFE: Appendix E.3.

(a) Kolmogorov flow.

NFE	**2**	5	10	50
Vanilla CFM				
mean error ↓	0.0237	0.0225	0.0222	0.0221
rollout RMSE ↓	0.776	0.755	0.757	0.794
KoopFM-field (ours)				
mean error ↓	**0.0196**	0.0191	0.0191	0.0191
rollout RMSE ↓	**0.731**	0.718	0.724	0.754

(b) Rayleigh–Bénard.

NFE	**2**	5	10	25
Vanilla CFM				
$W_1(p) \downarrow$	2.604	2.576	2.596	2.604
LSD$(p) \downarrow$	1.013	0.810	0.704	0.605
LSD$(E) \downarrow$	0.322	0.312	0.315	0.316
KoopFM-field (ours)				
$W_1(p) \downarrow$	**2.054**	2.081	2.074	2.094
LSD$(p) \downarrow$	**0.588**	0.451	0.385	0.372
LSD$(E) \downarrow$	**0.311**	0.310	0.310	0.308

between predicted and true distributions of the Nusselt (Nu) and RMS Reynolds (Re) number series and of pointwise pressure, and LSD as the log-spectral distance of enstrophy and pressure spectra (Appendix E.2).

Whole-trajectory statistics. Table 1 reports results of whole-trajectory statistics under autoregressive rollout using the paired sign-flip test. By roughly 500 steps every model has decorrelated from the reference trajectory and saturates near a common, high relative-error ceiling, so these metrics are *climate* measurements, asking whether a model lands on the right attractor rather than tracking a particular trajectory. They separate models that short-horizon error does not: FNO (1-step)'s 50-step error is statistically indistinguishable from the best model's (Table 15), yet its $W_1(Re)$ shows it tracks the wrong attractor.

KoopFM is not uniformly best apart from $W_1(p)$, where it wins almost every paired comparison against every baseline and the same split holds spectrally (Figure 7, Appendix E.9, E.7); Vanilla CFM is still best on velocity, and PDE-Refiner dominates pressure in $\mathrm{RBC}_{\mathrm{inc\text{-}dec}}$ specifically. The claim is specific to the climate measurement: the ordering changes under teacher-forced one-step prediction, where U-Net is best on $W_1(p)$ (Table 18), so what the anchor buys is the pressure structure of the attractor a model settles onto, not its one-step pressure error. This tracks a broader tradeoff in KoopFM between $W_1(Nu)$ and $W_1(p)$: KoopFM-vec performs best on $W_1(Nu)$, while KoopFM-field dominates $W_1(p)$. We suspect Nu is sensitive to spatially coherent bias, which a non-spatial latent rarely produces, whereas pressure is set by a global elliptic constraint, not local advection, which a whole-field latent captures naturally.

Extrapolation: forcing beyond the training range. To test what happens once the forcing leaves the trained range, we retrain both Vanilla CFM and KoopFM from scratch with the $\mathrm{RBC}_{\mathrm{inc\text{-}high}}$

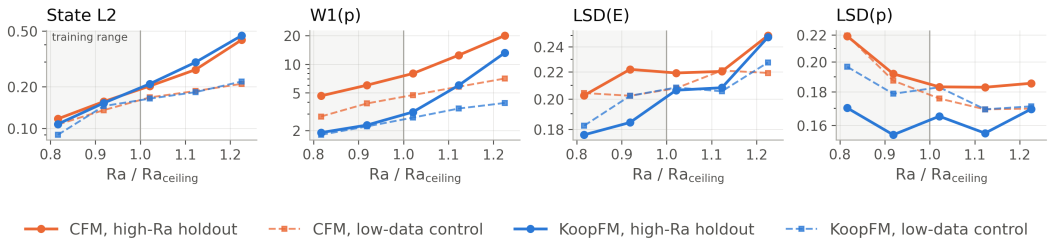


Figure 4: **Extrapolation past the training ceiling.** Metrics in 50-step autoregressive windows, from 0.80–1.20× the training ceiling ($Ra_{\text{ceil}} = 1.80 \times 10^7$); lower is better. Solid: the high-Ra holdout, the only extrapolating models. Dashed: the low-data control. The vertical line marks Ra_{ceil}. The full-data models are flat across the ladder and omitted. KoopFM in blue, Vanilla CFM in orange.

family removed, lowering the ceiling to $Ra_{\text{ceil}} = 1.80 \times 10^7$ and discarding 25% of the data (the *high-Ra holdout*), and separately train a *low-data control* that discards a different 25% while keeping the high-Ra data, so anything the control shows is a data-volume artifact rather than a range effect (Appendix E.1). Already inside the training range, the holdout costs Vanilla CFM 1.6–1.8× in $W_1(p)$ relative to its control and KoopFM at most 15% (one training run per arm), so part of the gap below is inherited from the training range.

Beyond the ceiling (Figure 4), the head-to-head ratio degrades smoothly and crosses unity at a different point for each metric (Table 22): KoopFM loses Nu first, at 1.14×, whereas it still holds LSD(p) and $W_1(p)$ significantly better at 1.2×; LSD(E) is tied from the ceiling to 1.2×. Relative to its own control, however, KoopFM degrades more slowly only up to about 1.1×; from 1.2× on its range effect on $W_1(p)$ and the fluxes is larger. On State L_2 both models degrade alike. $W_1(Nu)$ and $W_1(Re)$ need longer samples, so we measure them over each full test trajectory, whose forcing ramps past the ceiling (Table 23). On $\text{RBC}_{\text{inc-dec}}$ (peak 1.11×), KoopFM beats Vanilla CFM on four of five distributional metrics (marginally losing LSD(E)), but on $W_1(Nu)$ and $W_1(Re)$ the Vanilla CFM control degrades even more than its holdout, so these two wins are not range effects. On RBC_{inc} (up to 1.94×) it keeps its advantage on pressure, ties on LSD(E) and loses the two conserved fluxes. At $Ra \approx 3 \times 10^7$ (1.67×), both holdout models reach 3.8× (KoopFM) and 5.4× (Vanilla CFM) the ensemble-mean error of their full-data counterparts, so neither is still usable.

6 Limitations and Conclusion

Limitations and future work. Our method shows strong extrapolation capabilities on Lorenz–96; on Rayleigh–Bénard, however, the advantage holds only up to 14–40% above the training ceiling and is partly carried over from the training range rather than slower degradation. We also do not uniformly improve short-horizon accuracy, nor do we outperform baselines on $W_1(Nu)$. Beyond the training range, that is partly due to a violated incompressibility constraint (Appendix E.12), which is likely fixable by a training penalty on the buoyancy flux. Other anchor types are also worth exploring, especially for strongly stochastic settings, although we see that a good one-step predictor does not guarantee a good anchor (Appendix B.2). The method would likely also benefit from explicitly modeling the MZ memory kernel of Section 2 (Gupta et al., 2025). Finally, training the two stages jointly would let the anchor learn a flow-optimal source p_0, and with NFE$\approx$ 2 unrolled training becomes affordable, letting the anchor adapt to the exposure bias of inference rollouts.

Conclusion. KoopFM replaces the standard Gaussian source of flow matching with one centered on a bounded anchor. Its short path cuts the solver budget to two steps on Rayleigh–Bénard and deterministic Kolmogorov flow, and the anchor improves forcing-response extrapolation on Lorenz–96 and pressure structure on Rayleigh–Bénard. For ML-based climate modeling, this suggests emulators in which the anchor carries the scenario-forced response and a few-step flow samples the internal variability, cheap enough for the large ensembles needed for climate change mitigation and adaptation (Eyring et al., 2024). Anchoring alone does not give a robust emulator for boundary condition problems, but its remaining failure modes are clear targets, starting with a jointly trained closure.

AI USE STATEMENT

In this work, we used generative AI tools to provide feedback on research methodology or experiments, implement methods, propose or refine hypotheses, clean and reformat datasets, support qualitative and thematic data analysis, and interpret results. We have not used generative AI tools to help develop theoretical models or conceptual frameworks, or to generate synthetic data sets; "assist with translation", "assist in the writing of proofs", "formulate mathematical claims" and "provide critical ingredients for proving mathematical claims" are not applicable to this work. Additionally, we used generative AI tools to create or modify scientific figures or images, suggest experimental parameters, create or edit software code, draft parts of a research paper to improve readability, identify relevant literature and suggest a structure for a research paper. We have reviewed all AI-assisted work for potential plagiarism of ideas and for correct interpretation of results, and have manually verified the code. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

Training and sampling for both stages are given as pseudocode in Appendix A.1. Model architectures, parameter counts and training compute, including the compute-parity argument for baselines, are in Appendix E.3; baseline-specific configuration (U-Net, FNO, and PDE-Refiner's $\sigma_{\min}$ selection) is in Appendix E.5. Data generation follows published benchmarks: Rayleigh–Bénard convection is generated with Dedalus (Burns et al., 2020) on the domain and grid of The Well (Ohana et al., 2025), with the equations, forcing schedules, splits and preprocessing in Appendix E.1 and the metric definitions in Appendix E.2; Kolmogorov flow follows StocBench (Pfister et al., 2026) unmodified (Section 5.2); and two-tier Lorenz–96's full protocol, forcing levels, metric definitions and training-seed uncertainty are given in Appendix C. The two diagnostics for the degeneracy of Section 4.3, namely generation error rather than flow matching loss for model selection, and the τ-resolved loss profile, are cheap to reproduce and are recommended for any anchored-source generative surrogate. We will publish the code needed to reproduce all experiments on GitHub upon acceptance.

REFERENCES

Michael Albergo, Nicholas M. Boffi, and Eric Vanden-Eijnden. Stochastic Interpolants: A Unifying Framework for Flows and Diffusions. *Journal of Machine Learning Research*, 26(209):1–80, 2025. ISSN 1533-7928. URL http://jmlr.org/papers/v26/23-1605.html.

Michael S. Albergo, Mark Goldstein, Nicholas M. Boffi, Rajesh Ranganath, and Eric Vanden-Eijnden. Stochastic interpolants with data-dependent couplings, September 2024. URL http://arxiv.org/abs/2310.03725. arXiv:2310.03725 [cs.LG].

Arpit Bansal, Eitan Borgnia, Hong-Min Chu, Jie Li, Hamid Kazemi, Furong Huang, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Cold Diffusion: Inverting Arbitrary Image Transforms Without Noise. In *Advances in Neural Information Processing Systems*, volume 36, pp. 41259–41282. Curran Associates, Inc., 2023. doi: 10.52202/075280-1789. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/80fe51a7d8d0c73ff7439c2a2554ed53-Abstract-Conference.html.

Kaifeng Bi, Lingxi Xie, Hengheng Zhang, Xin Chen, Xiaotao Gu, and Qi Tian. Accurate medium-range global weather forecasting with 3D neural networks. *Nature*, 619(7970):533–538, July 2023. ISSN 1476-4687. doi: 10.1038/s41586-023-06185-3. URL https://www.nature.com/articles/s41586-023-06185-3. Publisher: Nature Publishing Group.

Massimo Bonavita. On some limitations of data-driven weather forecasting models, November 2023. URL http://arxiv.org/abs/2309.08473. arXiv:2309.08473 [stat.ML].

Johannes Brandstetter, Daniel Worrall, and Max Welling. Message Passing Neural PDE Solvers, March 2023. URL http://arxiv.org/abs/2202.03376. arXiv:2202.03376 [cs.LG].

Steven L. Brunton, Marko Budišić, Eurika Kaiser, and J. Nathan Kutz. Modern Koopman Theory for Dynamical Systems, October 2021. URL http://arxiv.org/abs/2102.12086. arXiv:2102.12086 [math].

Ricardo Buitrago Ruiz, Tanya Marwah, Albert Gu, and Andrej Risteski. On the Benefits of Memory for Modeling Time-Dependent PDEs. *International Conference on Learning Representations*, 2025:54972–55002, May 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/89379d5fc6eb34ff98488202fb52b9d0-Abstract-Conference.html.

Keaton J. Burns, Geoffrey M. Vasil, Jeffrey S. Oishi, Daniel Lecoanet, and Benjamin P. Brown. Dedalus: A flexible framework for numerical simulations with spectral methods. *Physical Review Research*, 2(2):023068, April 2020. ISSN 2643-1564. doi: 10.1103/PhysRevResearch.2.023068. URL https://link.aps.org/doi/10.1103/PhysRevResearch.2.023068.

Shuhao Cao, Francesco Brarda, Ruipeng Li, and Yuanzhe Xi. Spectral-Refiner: Accurate Fine-Tuning of Spatiotemporal Fourier Neural Operator for Turbulent Flows. *International Conference on Learning Representations*, 2025:60348–60384, May 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/972eea97af7fe9a7588f5af136edef0c-Abstract-Conference.html.

Graham Clyne, Guillaume Couairon, Guillaume Gastineau, Claire Monteleoni, and Anastase Charantonis. ArchesClimate: Probabilistic Decadal Ensemble Generation With Flow Matching, September 2025. URL https://arxiv.org/abs/2509.15942v3.

Guillaume Couairon, Renu Singh, Anastase Charantonis, Christian Lessig, and Claire Monteleoni. ArchesWeatherGen: Skillful and compute-efficient probabilistic weather forecasting with machine learning. *Science Advances*, 12(17):eadx2372, 2026. doi: 10.1126/sciadv.adx2372. URL https://www.science.org/doi/abs/10.1126/sciadv.adx2372.

Yilong Dai, Yiming Sun, Yiheng Chen, Shengyu Chen, Xiaowei Jia, and Runlong Yu. FlowRefiner: Flow Matching-Based Iterative Refinement for 3D Turbulent Flow Simulation, April 2026. URL http://arxiv.org/abs/2604.17149. arXiv:2604.17149 [physics.flu-dyn].

Mauricio Delbracio and Peyman Milanfar. Inversion by Direct Iteration: An Alternative to Denoising Diffusion for Image Restoration, February 2024. URL http://arxiv.org/abs/2303.11435. arXiv:2303.11435 [eess.IV].

Prafulla Dhariwal and Alex Nichol. Diffusion Models Beat GANs on Image Synthesis, June 2021. URL http://arxiv.org/abs/2105.05233. arXiv:2105.05233 [cs].

Veronika Eyring, Pierre Gentine, Gustau Camps-Valls, David M. Lawrence, and Markus Reichstein. AI-empowered next-generation multiscale climate modelling for mitigation and adaptation. *Nature Geoscience*, 17(10):963–971, October 2024. ISSN 1752-0908. doi: 10.1038/s41561-024-01527-w. URL https://www.nature.com/articles/s41561-024-01527-w. Publisher: Nature Publishing Group.

Stathi Fotiadis, Noah D Brenowitz, Tomas Geffner, Yair Cohen, Michael Pritchard, Arash Vahdat, and Morteza Mardani. Adaptive flow matching for resolving small-scale physics. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=YJ1My9ttEN.

Yurong Gao, Zicheng Zhang, Congying Han, Tiande Guo, and Xinmin Qiu. Resolving Endpoint Underfitting in Diffusion Bridges via Noise Alignment, May 2026. URL http://arxiv.org/abs/2605.28962. arXiv:2605.28962 [cs.CV].

Craig R. Gin, Daniel E. Shea, Steven L. Brunton, and J. Nathan Kutz. DeepGreen: deep learning of Green's functions for nonlinear boundary value problems. *Scientific Reports*, 11(1):21614, November 2021. ISSN 2045-2322. doi: 10.1038/s41598-021-00773-x. URL https://www.nature.com/articles/s41598-021-00773-x. Publisher: Nature Publishing Group.

Rares Grozavescu, Pengyu Zhang, Etienne Meunier, and Mark Girolami. Koopman Autoencoders with Continuous-Time Latent Dynamics for Fluid Dynamics Forecasting, March 2026. URL http://arxiv.org/abs/2602.02832. arXiv:2602.02832 [cs].

Priyam Gupta, Peter J. Schmid, Denis Sipp, Taraneh Sayadi, and Georgios Rigas. Mori-Zwanzig latent space Koopman closure for nonlinear autoencoder, May 2025. URL http://arxiv.org/abs/2310.10745. arXiv:2310.10745 [cs].

Peiyan Hu, Rui Wang, Xiang Zheng, Tao Zhang, Haodong Feng, Ruiqi Feng, Long Wei, Yue Wang, Zhi-Ming Ma, and Tailin Wu. Wavelet Diffusion Neural Operator. *International Conference on Learning Representations*, 2025:12291–12333, May 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/20eadb6338a53114a6371f102c5ded95-Abstract-Conference.html.

Armand Kassaï Koupaï, Lise Le Boudec, Louis Serrano, and Patrick Gallinari. ENMA: Tokenwise Autoregression for Generative Neural PDE Operators, June 2025. URL https://ui.adsabs.harvard.edu/abs/2025arXiv250606158K. ADS Bibcode: 2025arXiv250606158K.

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, May 2021. doi: 10.1073/pnas.2101784118. URL https://www.pnas.org/doi/full/10.1073/pnas.2101784118. Publisher: Proceedings of the National Academy of Sciences.

Dmitrii Kochkov, Janni Yuval, Ian Langmore, Peter Norgaard, Jamie Smith, Griffin Mooers, Milan Klöwer, James Lottes, Stephan Rasp, Peter Düben, Sam Hatfield, Peter Battaglia, Alvaro Sanchez-Gonzalez, Matthew Willson, Michael P. Brenner, and Stephan Hoyer. Neural General Circulation Models for Weather and Climate. *Nature*, 632(8027):1060–1066, August 2024. ISSN 0028-0836, 1476-4687. doi: 10.1038/s41586-024-07744-y. URL http://arxiv.org/abs/2311.07222. arXiv:2311.07222 [physics].

Georg Kohl, Li-Wei Chen, and Nils Thuerey. Benchmarking Autoregressive Conditional Diffusion Models for Turbulent Flow Simulation, December 2024. URL http://arxiv.org/abs/2309.01745. arXiv:2309.01745 [cs].

Marcel Kollovieh, Marten Lienen, David Lüdke, Leo Schwinn, and Stephan Günnemann. Flow Matching with Gaussian Process Priors for Probabilistic Time Series Forecasting, May 2025. URL http://arxiv.org/abs/2410.03024. arXiv:2410.03024 [cs].

B. O. Koopman. Hamiltonian Systems and Transformation in Hilbert Space. *Proceedings of the National Academy of Sciences*, 17(5):315–318, May 1931. doi: 10.1073/pnas.17.5.315. URL https://www.pnas.org/doi/abs/10.1073/pnas.17.5.315. Publisher: Proceedings of the National Academy of Sciences.

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

Fanny Lehmann, Firat Ozdemir, Yun Cheng, Torsten Hoefler, Sebastian Schemm, Benedikt Soja, and Siddhartha Mishra. Can AI Weather Models Predict Beyond Two Weeks? A Quantitative Benchmark and Analysis of Long Rollouts, June 2026. URL http://arxiv.org/abs/2605.30184. arXiv:2605.30184 [cs.LG].

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier Neural Operator for Parametric Partial Differential Equations, May 2021. URL http://arxiv.org/abs/2010.08895. arXiv:2010.08895 [cs].

Zongyi Li, Miguel Liu-Schiaffini, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Learning Chaotic Dynamics in Dissipative Systems. In *Advances in Neural Information Processing Systems*,

volume 35, pp. 16768–16781. Curran Associates, Inc., 2022. doi: 10.52202/068431-1220. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/6ad68277e27b42c60ac228c9859fc1a2-Abstract-Conference.html.

Marten Lienen, David Lüdke, Jan Hansen-Palmus, and Stephan Günnemann. From Zero to Turbulence: Generative Modeling for 3D Flow Simulation. *International Conference on Learning Representations*, 2024:5203–5220, May 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/15972a9575e0f03bf82f00aebeb40774-Abstract-Conference.html.

Kevin K. Lin and Fei Lu. Data-driven model reduction, Wiener projections, and the Koopman-Mori-Zwanzig formalism. *Journal of Computational Physics*, 424:109864, January 2021. ISSN 0021-9991. doi: 10.1016/j.jcp.2020.109864. URL https://www.sciencedirect.com/science/article/pii/S0021999120306380.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow Matching for Generative Modeling, February 2023. URL http://arxiv.org/abs/2210.02747. arXiv:2210.02747 [cs].

Phillip Lippe, Bas Veeling, Paris Perdikaris, Richard Turner, and Johannes Brandstetter. PDE-Refiner: Achieving Accurate Long Rollouts with Neural PDE Solvers. *Advances in Neural Information Processing Systems*, 36:67398–67433, December 2023. doi: 10.52202/075280-2946. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/d529b943af3dba734f8a7d49efcb6d09-Abstract-Conference.html.

Björn Malte List, Liwei Chen, Kartik Bali, and Nils Thuerey. How Temporal Unrolling Supports Neural Physics Simulators. October 2023. URL https://openreview.net/forum?id=JkMAlN3YcI.

Guan-Horng Liu, Arash Vahdat, De-An Huang, Evangelos A. Theodorou, Weili Nie, and Anima Anandkumar. I^{2}SB: Image-to-Image Schrödinger Bridge, May 2023. URL http://arxiv.org/abs/2302.05872. arXiv:2302.05872 [cs.CV].

Edward N. Lorenz. Predictability – a problem partly solved. In Tim Palmer and RenateEditors Hagedorn (eds.), *Predictability of Weather and Climate*, pp. 40–58. Cambridge University Press, 2006.

Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3(3):218–229, March 2021. ISSN 2522-5839. doi: 10.1038/s42256-021-00302-5. URL https://www.nature.com/articles/s42256-021-00302-5. Publisher: Nature Publishing Group.

Wanfeng Lu, He Ma, Wei Lin, and Qunxi Zhu. Interpretable Functional Koopman Learning with Non-Markovian Closure for Spatiotemporal Systems. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=lquDiBCgNJ.

Bethany Lusch, J. Nathan Kutz, and Steven L. Brunton. Deep learning for universal linear embeddings of nonlinear dynamics. *Nature Communications*, 9(1):4950, November 2018. ISSN 2041-1723. doi: 10.1038/s41467-018-07210-0. URL https://www.nature.com/articles/s41467-018-07210-0. Publisher: Nature Publishing Group.

Morteza Mardani, Noah Brenowitz, Yair Cohen, Jaideep Pathak, Chieh-Yu Chen, Cheng-Chin Liu, Arash Vahdat, Mohammad Amin Nabian, Tao Ge, Akshay Subramaniam, Karthik Kashinath, Jan Kautz, and Mike Pritchard. Residual corrective diffusion modeling for km-scale atmospheric downscaling. *Communications Earth & Environment*, 6(1):124, February 2025. ISSN 2662-4435. doi: 10.1038/s43247-025-02042-5. URL https://www.nature.com/articles/s43247-025-02042-5. Publisher: Nature Publishing Group.

Igor Mezić. Spectral Properties of Dynamical Systems, Model Reduction and Decompositions. *Nonlinear Dynamics*, 41(1):309–325, August 2005. ISSN 1573-269X. doi: 10.1007/s11071-005-2824-x. URL https://doi.org/10.1007/s11071-005-2824-x.

Hazime Mori. Transport, Collective Motion, and Brownian Motion. *Progress of Theoretical Physics*, 33(3):423–455, March 1965. ISSN 0033-068X. doi: 10.1143/PTP.33.423. URL https://doi.org/10.1143/PTP.33.423.

Jeremy Morton, Antony Jameson, Mykel J Kochenderfer, and Freddie Witherden. Deep Dynamical Modeling and Control of Unsteady Fluid Flows. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper/2018/hash/2b0aa0d9e30ea3a55fc271ced8364536-Abstract.html.

Juan Nathaniel and Pierre Gentine. Generative emulation of chaotic dynamics with coherent prior. *Computer Methods in Applied Mechanics and Engineering*, 448:118410, 2026. Publisher: Elsevier.

Juan Nathaniel, Yongquan Qu, Tung Nguyen, Sungduk Yu, Julius Busecke, Aditya Grover, and Pierre Gentine. ChaosBench: A Multi-Channel, Physics-Based Benchmark for Subseasonal-to-Seasonal Climate Prediction, November 2024. URL http://arxiv.org/abs/2402.00712. arXiv:2402.00712 [cs].

Ruben Ohana, Michael McCabe, Lucas Meyer, Rudy Morel, Fruzsina J. Agocs, Miguel Beneitez, Marsha Berger, Blakesley Burkhart, Keaton Burns, Stuart B. Dalziel, Drummond B. Fielding, Daniel Fortunato, Jared A. Goldberg, Keiya Hirashima, Yan-Fei Jiang, Rich R. Kerswell, Suryanarayana Maddu, Jonah Miller, Payel Mukhopadhyay, Stefan S. Nixon, Jeff Shen, Romain Watteaux, Bruno Régaldo-Saint Blancard, François Rozet, Liam H. Parker, Miles Cranmer, and Shirley Ho. The Well: a Large-Scale Collection of Diverse Physics Simulations for Machine Learning, February 2025. URL http://arxiv.org/abs/2412.00568. arXiv:2412.00568 [cs].

Vivek Oommen, Aniruddha Bora, Zhen Zhang, and George Em Karniadakis. Integrating Neural Operators with Diffusion Models Improves Spectral Representation in Turbulence Modeling, February 2025. URL http://arxiv.org/abs/2409.08477. arXiv:2409.08477 [cs.LG].

Chris Pedersen, Laure Zanna, and Joan Bruna. Thermalizer: Stable autoregressive neural emulation of spatiotemporal chaos, July 2025. URL http://arxiv.org/abs/2503.18731. arXiv:2503.18731 [cs.LG].

Sebastian Pfister, Benjamin Holzschuh, and Nils Thuerey. StocBench: A Benchmark for Generative Modeling of Stochastic Dynamics, August 2026. URL http://arxiv.org/abs/2608.22309. arXiv:2608.22309 [cs.LG].

Vadim Popov, Ivan Vovk, Vladimir Gogoryan, Tasnima Sadekova, and Mikhail Kudinov. Grad-TTS: A Diffusion Probabilistic Model for Text-to-Speech. In *Proceedings of the 38th International Conference on Machine Learning*, pp. 8599–8608. PMLR, July 2021. URL https://proceedings.mlr.press/v139/popov21a.html. ISSN: 2640-3498.

Ilan Price, Alvaro Sanchez-Gonzalez, Ferran Alet, Tom R. Andersson, Andrew El-Kadi, Dominic Masters, Timo Ewalds, Jacklynn Stott, Shakir Mohamed, Peter Battaglia, Remi Lam, and Matthew Willson. Probabilistic weather forecasting with machine learning. *Nature*, 637 (8044):84–90, January 2025. ISSN 1476-4687. doi: 10.1038/s41586-024-08252-9. URL https://www.nature.com/articles/s41586-024-08252-9. Publisher: Nature Publishing Group.

Joshua L. Proctor, Steven L. Brunton, and J. Nathan Kutz. Dynamic mode decomposition with control, September 2014. URL http://arxiv.org/abs/1409.6358. arXiv:1409.6358 [math.OC].

Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred Hamprecht, Yoshua Bengio, and Aaron Courville. On the Spectral Bias of Neural Networks. In *Proceedings of the 36th International Conference on Machine Learning*, pp. 5301–5310. PMLR, May 2019. URL https://proceedings.mlr.press/v97/rahaman19a.html. ISSN: 2640-3498.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation, May 2015. URL http://arxiv.org/abs/1505.04597. arXiv:1505.04597 [cs.CV].

Aliaksandra Shysheya, Cristiana Diaconu, Federico Bergamin, Paris Perdikaris, José Miguel Hernández-Lobato, Richard E. Turner, and Emile Mathieu. On conditional diffusion models for PDE simulations. In *Advances in Neural Information Processing Systems*, volume 37, pp. 23246–23300. Curran Associates, Inc., 2024. doi: 10.52202/079017-0732. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/2974844555dc383ea16c5f35833c7a57-Abstract-Conference.html.

Alexander Tong, Kilian Fatras, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Guy Wolf, and Yoshua Bengio. Improving and generalizing flow-based generative models with minibatch optimal transport, March 2024. URL http://arxiv.org/abs/2302.00482. arXiv:2302.00482 [cs].

Haixin Wang, Jiashu Pan, Hao Wu, Fan Zhang, and Tailin Wu. FourierFlow: Frequency-aware Flow Matching for Generative Turbulence Modeling, June 2025. URL http://arxiv.org/abs/2506.00862. arXiv:2506.00862 [cs.LG].

Qi Wang, Pu Ren, Hao Zhou, Xin-Yang Liu, Zhiwen Deng, Yi Zhang, Ruizhi Chengze, Hongsheng Liu, Zidong Wang, Jian-Xun Wang, Ji-Rong Wen, Hao Sun, and Yang Liu. Pˆ2Cˆ2Net: PDE-Preserved Coarse Correction Network for efficient prediction of spatiotemporal dynamics. In *Advances in Neural Information Processing Systems*, volume 37, pp. 68897–68925. Curran Associates, Inc., 2024. doi: 10.52202/079017-2201. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/7f605d59a0dbde101518b552cb616ddf-Abstract-Conference.html.

Oliver Watt-Meyer, Brian Henn, Jeremy McGibbon, Spencer K. Clark, Anna Kwa, W. Andre Perkins, Elynn Wu, Lucas Harris, and Christopher S. Bretherton. ACE2: Accurately learning subseasonal to decadal atmospheric variability and forced responses, November 2024. URL http://arxiv.org/abs/2411.11268. arXiv:2411.11268 [physics] version: 1.

Zongsheng Yue, Jianyi Wang, and Chen Change Loy. ResShift: Efficient Diffusion Model for Image Super-resolution by Residual Shifting. In *Advances in Neural Information Processing Systems*, volume 36, pp. 13294–13307. Curran Associates, Inc., 2023. doi: 10.52202/075280-0583. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/2ac2eac5098dba08208807b65c5851cc-Abstract-Conference.html.

Weili Zeng and Yichao Yan. Flow Matching in the Low-Noise Regime: Pathologies and a Contrastive Remedy, September 2025. URL http://arxiv.org/abs/2509.20952. arXiv:2509.20952 [cs.LG].

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. *International Conference on Learning Representations*, 2024:11612–11653, May 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/312237ba5de457df7bc8f88d4de21c4c-Abstract-Conference.html.

Robert Zwanzig. *Nonequilibrium Statistical Mechanics*. Oxford University Press, April 2001. ISBN 978-0-19-514018-7. doi: 10.1093/oso/9780195140187.001.0001. URL https://doi.org/10.1093/oso/9780195140187.001.0001.

A METHOD DETAILS

A.1 TRAINING AND SAMPLING, IN PSEUDOCODE

Stage 1 (the base predictor) and Stage 2 (the anchored flow) are trained in sequence: Stage 1 is optimized first and then frozen, and Stage 2 is trained against it, as in Algorithms 1–2. Algorithm 3 is the resulting autoregressive sampler. Notation follows Sections 2–4.2: ϕ_E, ϕ_D are the Koopman encoder/decoder, ψ embeds the forcing, (A, G, b) is the latent transition of Eq. 2, $\hat{x}_{t+1}$ is the Stage-1 anchor, σ is the source scale of Eq. 5, and v_θ is the Stage-2 velocity field. The algorithms describe the Rayleigh–Bénard runs exactly; the joint variant used for Lorenz–96 is described at the end of this section.

The Stage-1 objective builds on the loss design of Lusch et al. (2018). A reconstruction term makes $\phi_D \circ \phi_E$ an autoencoder. A prediction term compares the decoded latent rollout $\hat{x}_{t+m+1}$ with the true state over M unrolled steps, so the operator is trained for what it is used for, forecasting in state space. A linearity (alignment) term ties the propagated latent z_{t+m+1} to the encoding $\tilde{z}_{t+m+1}$ of the true state, so that the encoder produces coordinates in which the linear transition actually holds; we implement it as an MSE plus a cosine term, each with weight λ, and drop the L_∞ penalty of Lusch et al. (2018). For the KoopFM-field anchor we add a cycle-consistency term with weight μ that re-encodes the decoded prediction, $\phi_E(\hat{x}_{t+m+1}) \approx z_{t+m+1}$, so that the anchor is stable under the decode–re-encode cycle it undergoes at every step of Algorithm 3. The forcing enters additively through $G\,\psi(u)$, as in the controlled Koopman model of Morton et al. (2018). In Algorithm 1, $\|\cdot\|^2$ denotes the mean squared entry (MSE), each term is averaged over the frames of the window, cos is taken over the full flattened latent, and no stop-gradient is applied to any target: $\tilde{z}$ and $\phi_E(\hat{x})$ receive gradients too. Hyperparameter values are in Appendix E.4.

Algorithm 1 Stage 1: base predictor training

Require: windows $(x_t, \ldots, x_{t+M})$ with forcing u, unroll length M, relative input-noise scale η, loss weights λ, μ, total optimizer steps K

initialize $\phi_E, \phi_D, \psi, A, G, b$; per-channel scale $s \leftarrow \mathbf{1}$
for $k = 1, \ldots, K$, minibatch of windows **do**
  $s \leftarrow 0.99\, s + 0.01\, \mathrm{std}_{\text{channel}}(x_{t:t+M})$ ▷ EMA of per-channel std
  $\eta_k \leftarrow \eta \cdot \frac{1}{2}\big(1 - \cos(\pi k/K)\big)$ ▷ cosine ramp from 0 to η
  for $m = 0, \ldots, M$ **do**
    $\tilde{z}_{t+m} \leftarrow \phi_E\big(x_{t+m} + \eta_k\, s \odot \epsilon_m\big), \quad \epsilon_m \sim \mathcal{N}(0, I)$ ▷ noisy encoding of every frame
  end for
  $\mathcal{L}_{\text{rec}} \leftarrow \frac{1}{M+1} \sum_{m=0}^{M} \|\phi_D(\tilde{z}_{t+m}) - x_{t+m}\|^2$ ▷ target is the clean frame
  $z_t \leftarrow \tilde{z}_t$
  for $m = 0, \ldots, M-1$ **do** ▷ latent-only unroll
    $z_{t+m+1} \leftarrow A\, z_{t+m} + G\, \psi(u) + b$ ▷ u held fixed over the window
    $\hat{x}_{t+m+1} \leftarrow \phi_D(z_{t+m+1})$
  end for
  $\mathcal{L}_{\text{pred}} \leftarrow \frac{1}{M} \sum_{m=1}^{M} \|\hat{x}_{t+m} - x_{t+m}\|^2$
  $\mathcal{L}_{\text{lin}} \leftarrow \frac{1}{M} \sum_{m=1}^{M} \Big[\|z_{t+m} - \tilde{z}_{t+m}\|^2 + \big(1 - \cos(z_{t+m}, \tilde{z}_{t+m})\big)\Big]$
  $\mathcal{L}_{\text{cyc}} \leftarrow \frac{1}{M} \sum_{m=1}^{M} \|z_{t+m} - \phi_E(\hat{x}_{t+m})\|^2$
  $\mathcal{L} \leftarrow \mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{pred}} + \lambda\, \mathcal{L}_{\text{lin}} + \mu\, \mathcal{L}_{\text{cyc}}$
  update $\phi_E, \phi_D, \psi, A, G, b$ by AdamW on $\mathcal{L}$ ▷ grad. clip 1.0
end for
return $\theta_1 = (\phi_E, \phi_D, \psi, A, G, b)$, then **freeze** θ_1

A.2 THE ANCHORED TRANSPORT LENGTH

The anchor's own offset is $\|x_{t+1} - \hat{x}_{t+1}\|/\|x_{t+1}\| \approx 0.16$, and a Gaussian source gives $\approx \sqrt{2}$. Comparing these two numbers directly gives a naive $9\times$ reduction, but that compares a noise-free bias against a full, noise-inclusive Gaussian-source distance, so it overstates the gain. The actual anchored source also carries the injected noise $\sigma\epsilon$ needed to avoid the degeneracy of Section 4.3, with σ calibrated to that same residual, so the noise roughly doubles the expected squared transport

Algorithm 2 Stage 2: anchored flow-matching training

Require: frozen θ_1, pairs (x_t, u_t, x_{t+1}), EMA weight β, conditioning-noise scale ζ
initialize v_θ; $s \leftarrow 0.1 \cdot \mathbf{1} \in \mathbb{R}^C$; $\rho \leftarrow \mathbf{1} \in \mathbb{R}^{C\times H\times W}$
for each minibatch (x_t, u_t, x_{t+1}) **do**
  $z_t \leftarrow \phi_E(x_t)$; $\hat{z} \leftarrow Az_t + G\,\psi(u_t) + b$; $\hat{x}_{t+1} \leftarrow \phi_D(\hat{z})$ ▷ Stage-1 forward pass, no gradient
  $r \leftarrow x_{t+1} - \hat{x}_{t+1}$ ▷ anchor residual
  $s \leftarrow (1-\beta)\,s + \beta \cdot \mathrm{RMS}_{\text{batch, space}}(r)$ ▷ per-channel scale
  $\rho \leftarrow (1-\beta)\,\rho + \beta \cdot \mathrm{RMS}_{\text{batch}}(r) \,/\, \overline{\mathrm{RMS}_{\text{batch}}(r)}^{\text{space}}$ ▷ KoopFM-field only; KoopFM-vec keeps $\rho = \mathbf{1}$
  $\sigma \leftarrow s \odot \rho$ ▷ no gradient; frozen for inference
  sample $\tau \sim \mathcal{U}(0,1)$ per sample, $\epsilon, \epsilon' \sim \mathcal{N}(0, I)$
  $x_0 \leftarrow \hat{x}_{t+1} + \sigma \odot \epsilon$ ▷ source, Eq. 5
  $x_\tau \leftarrow (1-\tau)\,x_0 + \tau\,x_{t+1}$
  $\tilde{x}_t \leftarrow x_t + \zeta\,\epsilon'$ ▷ noisy history for the network only
  $\mathcal{L}_{\mathrm{FM}} \leftarrow \left\| v_\theta(x_\tau, \tau, \tilde{x}_t, \hat{x}_{t+1}, u_t) - (x_{t+1} - x_0) \right\|^2$
  update v_θ by AdamW on $\mathcal{L}_{\mathrm{FM}}$ ▷ grad. clip 1.0; θ_1 stays frozen
end for
return v_θ, final σ

Algorithm 3 Sampling: autoregressive rollout

Require: initial state x_0, forcing $u_0, \ldots, u_{T-1}$, solver steps NFE, trained θ_1, v_θ, σ
for $t = 0, \ldots, T-1$ **do**
  $z_t \leftarrow \phi_E(x_t)$; $\hat{z} \leftarrow Az_t + G\,\psi(u_t) + b$; $\hat{x}_{t+1} \leftarrow \phi_D(\hat{z})$ ▷ Stage-1 anchor, re-encoded each step
  $x \leftarrow \hat{x}_{t+1} + \sigma \odot \epsilon, \quad \epsilon \sim \mathcal{N}(0, I)$ ▷ sample the source
  $\Delta\tau \leftarrow 1/\mathrm{NFE}$
  for $k = 0, \ldots, \mathrm{NFE} - 1$ **do**
    $\tau_k \leftarrow k\,\Delta\tau$
    $x \leftarrow x + \Delta\tau \cdot v_\theta(x, \tau_k, x_t, \hat{x}_{t+1}, u_t)$ ▷ Euler step
  end for
  $x_{t+1} \leftarrow x$
end for
return $x_1, \ldots, x_T$

length on the anchored side too. The $\sqrt{2}$ factor common to both sides then cancels, and the fair, noise-inclusive comparison is $1/0.16 \approx 6\times$ shorter.

Degenerate conditionals. The end-of-path limit in Section 4.2 assumes that the one-step conditional has non-zero spread. If x_{t+1} is a deterministic function of (x_t, u_t), then $x_\tau = (1-\tau)x_0 + \tau x_1$ reveals x_0 for every $\tau < 1$, so that $\mathbb{E}[x_0 \mid x_\tau, x_t, u_t] = x_0$ and the optimal velocity is $x_1 - x_0$ along each path. The marginal paths are then straight for any source that is independent of x_1 given (x_t, u_t), Gaussian or anchored, and a single Euler step is exact for the optimal field.

B Source noise scale and anchor choice

All experiments in this section use forced Rayleigh–Bénard convection (Section 5.3; setup in Appendix E), with the Stage 2 recipe held fixed and only σ or the anchor changed. Table 3 reports the losses behind the $\sigma = 0$ degeneracy of Section 4.3: the lowest flow-matching loss in this study, and the worst samples.

B.1 Training-time scaling of σ

Every KoopFM run in the main text ties σ to the anchor's residual (Section 4.3). To test whether the flow benefits from a wider source, we retrain KoopFM-field from scratch at $2\times$ and $3\times$ the anchor-

Table 3: **The $\sigma = 0$ model has the lowest training loss and the worst samples.** Validation generation error is measured by integrating the sampling ODE (NFE=10). The flow matching losses are not comparable to it and, taken on their own, are misleading.

	train $\mathcal{L}_{\mathrm{FM}}$	val $\mathcal{L}_{\mathrm{FM}}$	val generation MSE
KoopFM, $\sigma \approx 0.11$	0.00035	0.00038	**0.00001**
KoopFM, $\sigma = 0$	**0.00004**	**0.00009**	0.00018

Table 4: **The anchor-matched source width is close to optimal.** Training-time σ multiplier for KoopFM-field at a fixed, frozen anchor; all three runs stopped at epoch 483. Whole trajectory, 25 NFE, mean over both scenarios. Percentages are against the $1\times$ reference; * significantly better and † significantly worse under a one-sided exact paired sign-flip test over the shared initial conditions, plain a tie. Spread/Skill is better nearer 1; every other row is lower-is-better.

	$1\times$ (reference)	$2\times$	$3\times$
autoregressive			
$W_1(Nu)$	0.2821	0.2437 (-13.6%)	0.2126* (-24.6%)
$W_1(Re)$	4.3226	4.4494 (+2.9%)	4.1952 (-2.9%)
$W_1(p)$	2.0943	2.1425 (+2.3%)	2.0941 (≈0%)
LSD(E)	0.3084	0.3180† (+3.1%)	0.3129 (+1.4%)
LSD(p)	0.3719	0.4103† (+10.3%)	0.4340† (+16.7%)
State L_2	0.0718	0.0650* (-9.5%)	0.0726 (+1.1%)
Spread/Skill	0.415	0.445	0.451
CRPS	0.05563	0.05265	0.05277
teacher-forced			
$W_1(Nu)$	0.0848	0.0603* (-29.0%)	0.0634* (-25.3%)
$W_1(Re)$	0.8025	0.7891 (-1.7%)	0.6311* (-21.3%)
$W_1(p)$	0.6362	0.5636* (-11.4%)	0.6056* (-4.8%)
LSD(E)	0.0406	0.0350* (-14.0%)	0.0335* (-17.6%)
LSD(p)	0.3222	0.3463† (+7.5%)	0.3772† (+17.1%)
State L_2	0.0153	0.0142* (-7.5%)	0.0150* (-2.1%)
Spread/Skill	0.753	0.839	0.853
CRPS	0.00916	0.00755	0.00747

matched σ (0.119 averaged over channels), changing nothing else: same frozen anchor, backbone, optimizer, epoch budget and seeds. Every rung is still far narrower than the unit-variance source of Vanilla CFM, which would correspond to roughly 7–14×.

Table 4 shows that $1\times$ is close to the optimum. At $2\times$ the wider source improves the Nusselt distribution, short-horizon error and calibration. From $2\times$ to $3\times$, however, the calibration gain stalls, State L_2 loses its entire $2\times$ gain, and the teacher-forced $W_1(Nu)$ and $W_1(p)$ gains shrink; under autoregression only $W_1(Nu)$ keeps improving. LSD(p) is the only metric that degrades monotonically, significantly at both rungs and in both modes. Resolved by Ra, this cost is confined to the lowest-Ra windows, where pressure fluctuations are weakest: at $2\times$ only the $Ra \approx 1.1 \times 10^6$ windows are hurt (by about 20%), at $3\times$ the 5×10^6 window is hurt as well (+12%), and the windows at 10^7 and above improve at both rungs. This is consistent with the source noise acting as a floor that masks weak pressure structure. A $1.5\times$ rung, which we did not run, would locate the optimum between $1\times$ and $2\times$ more precisely.

B.2 Choice of anchor

Nothing in Section 4.2 requires a particular Stage 1, so we ran the identical flow-matching recipe (same U-Net backbone, objective, optimizer and 500 epochs) on five anchors: the Koopman anchors of KoopFM-field and KoopFM-vec, a Koopman anchor with an Ra-parameterized forcing operator, and two FNOs of different capacity (Table 5). σ is not tuned but read from each model's EMA of its anchor's one-step residual, so the σ column measures anchor accuracy.

Table 5: **Five Stage-1 anchors under an identical Stage-2 recipe.** σ is each model's EMA of the anchor's one-step residual RMS in normalized units, per channel and averaged. Emulator columns are autoregressive over the full trajectory, pooled over both test schedules ($n = 19$ ICs); State L_2 is the 50-step window at $Ra \approx 1.1 \times 10^6$. **Bold**: best in each column. †Driven by 2 of the 19 ICs diverging; on the other 17 this anchor beats the KoopFM-vec anchor on $W_1(Re)$.

Stage-1 anchor	anchor params	σ (source std.) (b, p, u, w)	σ mean	$W_1(Nu)$	$W_1(Re)$	$W_1(p)$	LSD(p)	State L_2
Koopman (KoopFM-field)	0.164M	$0.128, 0.072, 0.141, 0.135$	0.119	0.292	**4.29**	**2.11**	**0.388**	**0.070**
Koopman (KoopFM-vec)	1.266M	$0.108, 0.059, 0.112, 0.107$	0.096	**0.216**	6.10	2.59	0.451	0.107
Koopman, Ra-param. forcing	1.274M	$0.061, 0.031, 0.076, 0.073$	0.060	1.212	40.8†	11.43†	0.558	0.080
FNO, 1.24M	1.239M	$0.040, 0.022, 0.044, 0.045$	0.038	15.34	216.9	31.85	1.523	0.111
FNO, 8.61M	8.613M	$0.023, 0.011, 0.019, 0.022$	0.019	3.151	150.4	22.45	0.633	0.077

Two findings stand out. First, **the anchor barely affects short-horizon accuracy but decides the long run**: across the five anchors, 50-step State L_2 varies by a factor of 1.6 (0.070–0.111), whereas $W_1(Re)$ varies by a factor of 50 (4.3–217). Selection on trajectory error would not distinguish these models, while selection on the invariant measure would reject three of them. Second, **a more accurate anchor does not give a better emulator**: σ falls by $6\times$ from the least to the most accurate anchor, the two most accurate anchors (the FNOs) produce the two worst climates, and the least accurate one (KoopFM-field) the best.

The 8.61M FNO anchor makes the mechanism measurable, because it *is* the FNO (1-step) baseline. The FNO-anchored model attains the best teacher-forced one-step error of all variants tested (0.0140 against 0.0164), yet its long-run statistics are an order of magnitude worse than those of KoopFM-field and in the range of the FNO baseline itself. With a Koopman anchor at $\sigma \approx 0.13$, the flow corrects 98% of the anchor's Nusselt bias; with the FNO anchor at $\sigma \approx 0.02$, it corrects 7% and inherits the anchor's attractor error. Since σ at once prevents the degeneracy of Section 4.3, sets the transport length and sets the flow's correction range, improving the anchor shrinks all three, so Stage 1 should not be trained for accuracy alone.

Overtraining the anchor. Training the KoopFM-vec anchor $10\times$ longer (58K against 5.7K optimizer steps) improves the anchor but degrades the emulator (State L_2 0.107 $\rightarrow$ 0.116, $W_1(Nu)$ $2.2\times$ worse). Encoding the *same* generated rollout with each anchor's encoder shows why (Table 6): the overtrained encoder is half as sensitive to input error in general, but twice as sensitive to the flow's own artifacts relative to Gaussian noise of the same RMS, so the generated states become out-of-distribution for it. Since the anchor is re-encoded at every step, this governs the rollout. The KoopFM-field encoder stays near excess $= 1$ at any training budget and can be trained to convergence.

Table 6: **Encoder sensitivity to the flow's own output.** The same generated rollout is encoded with each anchor's encoder and compared with the encoded truth. Gain: latent drift divided by the input relative error. Excess: latent drift divided by the drift caused by Gaussian noise of identical RMS; values above 1 mean the flow's artifacts are specifically out-of-distribution for the encoder.

anchor	optimizer steps	gain	excess
KoopFM-vec	5,681	0.37	3.13
KoopFM-vec, overtrained	58,110	0.17	**6.21**
KoopFM-field	42,763	0.14	**1.07**
KoopFM-field, longer training	74,500	0.11	1.39

C LORENZ–96

C.1 SYSTEM AND DATA

We integrate the two-tier Lorenz–96 system (Lorenz, 2006) with $K = 36$ slow variables X_k and $J = 10$ fast variables $Y_{j,k}$ per slow site ($d = 396$), coupling constants $h = 1$, $c = 10$ and $b = 10$, and the forcing F acting on the slow tier, using fourth-order Runge–Kutta at $\Delta t = 10^{-3}$. After a spin-up

of 20 time units at the initial forcing, each trajectory runs for 200 time units with states saved every 0.04; the first 200 saved states are discarded, leaving 4,801. Forcing schedules reuse the shapes of the Rayleigh–Bénard protocol under the map $F = 6 + 3(\log_{10} Ra - 6)$, which sends the Rayleigh–Bénard training range $Ra \in [10^6, 10^8]$ to $F \in [6, 12]$. Training uses the four Rayleigh–Bénard training schedules with four replicates each (16 trajectories). Validation uses one further replicate of each training schedule (in distribution) and five replicates of the held-out `decline_high` schedule (out of distribution). Inputs are normalized with training-set statistics, and each transition takes the forcing at its left endpoint.

C.2 Training and evaluation protocol

Prior and flow training. We use three training seeds: 17031, 65643, and 94117. Koopman priors are trained for 350 epochs with eight latent transitions per training window. Their checkpoints minimize the Stage 1 OOD validation loss. Stage 2 freezes the prior weights. All FM runs use 300 epochs, batch size 512, AdamW with learning rate 2×10^{-4}, cosine-annealed to zero, and weight decay 10^{-5}, an exponential moving average of the weights (decay 0.9999), and conditioning-state noise of standard deviation 0.05 during CFM regression. KoopFM uses an adaptive anchor-noise coefficient, initialized at 0.05, updated by an EMA at rate 0.001 toward the anchor's relative one-step RMSE, and capped at 0.25. Its effective source standard deviation is this coefficient times the within-example standard deviation of the normalized anchor. The coefficient is frozen at inference.

One-step CFM uses one velocity-network regression evaluation and no training ODE solve. BPTT-4 differentiates through four generated physical transitions, using either two or eight Euler evaluations per transition. The objective is the mean CFM loss plus 0.1 times the mean normalized-state rollout MSE over those transitions. Generated states remain connected to the gradient graph. The prior weights stay frozen, but derivatives through the prior with respect to its input state are retained. Vanilla CFM uses the same rollout objective and Euler budgets, with a standard Gaussian source and no anchor input.

The BPTT forward pass uses $4(N_{\mathrm{tr}} + 1)$ velocity-network calls per four-transition window: 12 for training NFE 2 and 36 for training NFE 8. These counts include the CFM regression calls but exclude backward recomputation, validation, and prior pretraining. Training NFE and inference NFE are separate choices.

Checkpoint selection and inference. For each FM run, seven EMA checkpoint candidates are evaluated: epochs 49, 99, 149, 199, 249, and 299, and the checkpoint with the best one-step validation loss. Selection uses full rollouts on four IID and five OOD validation forcing schedules at inference NFE 50. The selection score is the geometric mean of ratios to a fixed Gaussian validation reference across trajectory RMSE, temporal log-spectral distance, marginal Wasserstein distance, and absolute energy bias. The IID group and each OOD scenario enter this score. Constant-forcing evaluation results do not select checkpoints.

All reported FM results use 50 inference NFEs, eight ensemble members, and 4,801-state autoregressive rollouts (4,800 transitions), without teacher forcing after initialization. Evaluation holds forcing fixed within each trajectory, using five initial conditions at each of 17 constant-forcing levels, $\{4, 6, 8, 10, 12\} \cup \{12.5, 13, 13.5, \ldots, 18\}$, for 85 trajectories in total (Section 5.1). Each model and training setting is evaluated over three training seeds.

Scope of the comparison. Vanilla CFM and KoopFM share the evaluation trajectories, forcing information, training-set normalization, ensemble size, Euler budget, and metric implementation. Their corresponding Stage 2 settings also share the training schedule and, for BPTT, the unroll horizon and training Euler budget. The clean anchor is computed from the current state and forcing; it supplies no future-state observation. KoopFM additionally requires prior pretraining and prior evaluation, and its anchor input increases the flow-network parameter count. The comparison therefore aligns evaluation conditions and Stage 2 training settings, while parameter count and total compute differ.

FNO, U-Net, and PDE-Refiner are evaluated over three training seeds on the same trajectories and metrics, with their own training and checkpoint-selection protocols. FNO and U-Net are determin-

istic; PDE-Refiner uses eight ensemble members. These are architecture references, whereas the Vanilla CFM settings provide the direct flow-matching comparison.

C.3 Evaluation metrics

Let $x_{r,t} \in \mathbb{R}^d$ be the simulator state and $\widehat{x}_{r,t}^{(m)}$ ensemble member m for trajectory r, where $d = 396$, $t = 0, \ldots, N-1$, and $N = 4{,}801$. There are $M = 8$ members for stochastic models and $M = 1$ for deterministic models. Except for the normalized joint energy distance defined below, metrics use physical-space states. Scores are computed separately for each trajectory and then averaged with equal trajectory weights over the constant-forcing evaluation set.

Trajectory RMSE. With $\overline{x}_{r,t} = M^{-1} \sum_m \widehat{x}_{r,t}^{(m)}$, the reported score on R trajectories is

$$\mathrm{RMSE} = \frac{1}{R} \sum_{r=1}^{R} \left[\frac{1}{Nd} \sum_{t=0}^{N-1} \|\overline{x}_{r,t} - x_{r,t}\|_2^2 \right]^{1/2}. \tag{8}$$

This evaluates the ensemble-mean trajectory, not the average member-wise RMSE. We average over all 85 constant-forcing trajectories and all recorded states, including the initial transient.

Long-run energy-response MAE. The slow- and fast-tier energies are $E_X(x) = (2K)^{-1} \sum_k X_k^2$ and $E_Y(x) = (2KJ)^{-1} \sum_{k,j} Y_{j,k}^2$. Let $\mathcal{T}$ contain the final 80% of recorded times, implemented by discarding the first $\lfloor 0.2N \rfloor$ states. For $q \in \{X, Y\}$, define

$$\overline{E}_{q,r} = \frac{1}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} E_q(x_{r,t}), \tag{9}$$

$$\widehat{\overline{E}}_{q,r} = \frac{1}{M|\mathcal{T}|} \sum_{m=1}^{M} \sum_{t \in \mathcal{T}} E_q(\widehat{x}_{r,t}^{(m)}). \tag{10}$$

The response error over a trajectory set $\mathcal{R}$ is

$$\mathrm{MAE}_q(\mathcal{R}) = \frac{1}{|\mathcal{R}|} \sum_{r \in \mathcal{R}} \left| \widehat{\overline{E}}_{q,r} - \overline{E}_{q,r} \right|. \tag{11}$$

We average member energies rather than taking the energy of the ensemble-mean state. The all-F score uses all 85 constant-forcing trajectories; the $F > 12$ score uses the 60 high-forcing trajectories. The absolute error is taken before averaging over initial conditions.

Joint energy distance. Joint energy distance uses the scaled state distance

$$\rho(a, b) = \left[\frac{1}{d} \sum_{j=1}^{d} \left(\frac{a_j - b_j}{s_j} \right)^2 \right]^{1/2}, \tag{12}$$

where s_j is the training-set standard deviation of coordinate j. For each trajectory, we select $n = 256$ evenly spaced times from the final 80%. At each selected time we take the simulator state a_i and one predicted state b_i, cycling through ensemble members. Predicted and simulator samples use the same recorded times and constant forcing level. We compute

$$\mathrm{ED} = \frac{2}{n^2} \sum_{i,j=1}^{n} \rho(a_i, b_j) - \frac{1}{n^2} \sum_{i,j=1}^{n} \rho(a_i, a_j) - \frac{1}{n^2} \sum_{i,j=1}^{n} \rho(b_i, b_j). \tag{13}$$

All pair sums include the diagonal; negative numerical roundoff is clipped to zero. This score compares empirical multivariate state distributions after the burn-in, including cross-variable dependence. We compute it separately for each trajectory before averaging over all 85 trajectories; states from different forcing levels are not pooled. ED is a distributional diagnostic, distinct from the physical energy-response MAE in Eq. 11.

Table 7: **L96 parameter counts.** The prior is frozen during Stage 2. The flow column counts Stage 2 trainable parameters; the total includes the prior.

Model	Latent coordinates	Prior	Flow	Total
Vanilla CFM	—	—	3,130,540	3,130,540
KoopFM-vec	16	289,260	3,333,292	3,622,552
KoopFM-vec ($d_z = 144$)	144	326,892	3,333,292	3,660,184

Training-seed uncertainty. Each fitted model first yields one trajectory-averaged value per metric. Tables then report the mean and sample standard deviation across three training seeds. These standard deviations are not confidence intervals, and bold entries identify the lowest means without asserting statistical significance.

C.4 Model and parameter counts

On Lorenz–96 we use KoopFM-vec throughout, with a latent vector of dimension $d_z = 16$; a $d_z = 144$ version is reported as a latent-size ablation. For Lorenz–96, the latent transition of Section 4.1 additionally contains a bilinear forcing term, $z_{t+1} = Az_t + F_t\, Bz_t + GF_t + b$, so the effective operator $A + F_tB$ varies with the forcing level. It remains linear in z_t at fixed forcing, but with B unconstrained, the damping of A alone no longer guarantees contraction.

All KoopFM runs use the same width-512, four-block flow MLP. Its input concatenates the current state, clean anchor, and flow state, each with 396 coordinates; flow time and forcing enter through FiLM conditioning. Explicit anchor conditioning adds 202,752 flow-network parameters relative to Vanilla CFM (Table 7). Each inference transition requires the stated number of velocity-network evaluations plus one prior evaluation.

The $d_z = 16$ prior for seed 65643 was trained at learning rate 2×10^{-4}, all other priors at 10^{-3}. Within each latent size and seed, all Stage 2 training settings reuse the same prior.

C.5 Complete constant-forcing results

Table 8 reports all constant-forcing results at 50 inference NFEs. Vanilla CFM and KoopFM-vec are evaluated under one-step training and BPTT-4 with training NFE 2 and 8; the $d_z = 144$ ablation only under BPTT-4 with training NFE 8.

For KoopFM-vec, BPTT-4 with training NFE 8 reduces trajectory RMSE from 1.505 to 1.477 relative to one-step training, but increases high-forcing response MAE from $2.738/0.00395$ to $5.381/0.00607$ ($5.980/0.00595$ at training NFE 2). Vanilla CFM also benefits from BPTT in some settings: training at two NFEs yields high-forcing response MAE $3.626/0.01512$, against $5.925/0.02107$ at eight. These results do not establish a uniform response benefit from Stage 2 unrolling, and they concern the training integration budget only: every row uses 50 inference NFEs.

Joint ED provides a complementary check on the long-run state distribution. KoopFM-vec has lower ED than Vanilla CFM under one-step training (0.0761 versus 0.1329) and BPTT-4 with training NFE 8 (0.0566 versus 0.0685), but slightly higher ED at training NFE 2 (0.0602 versus 0.0578). The lowest displayed ED is obtained by the $d_z = 144$ ablation (0.0563). These are comparisons on the same constant-forcing trajectories.

D Kolmogorov flow

This appendix details the two variants of Section 5.2, which differ only in whether the forcing on the vorticity equation is observed by the models.

Data. Both tasks use the StocBench datasets (Pfister et al., 2026): the vorticity form of the two-dimensional incompressible Navier–Stokes equations on a periodic square of side 2π, integrated

Table 8: **Complete constant-forcing Lorenz–96 comparison.** Three-training-seed means, with ± sample standard deviations beneath, at 50 FM inference NFEs; lower is better. Bold marks the lowest mean in each column. B4/2 and B4/8 denote BPTT-4 with training NFE 2 and 8.

	All F diagnostics		Response MAE, all F		Response MAE, $F > 12$	
Model / training	RMSE	Joint ED	X	Y	X	Y
FNO (BPTT-4)	1.791 ±0.082	0.0979 ±0.0359	3.072 ±0.399	0.00672 ±0.00239	4.194 ±0.563	0.00891 ±0.00341
U-Net (BPTT-4)	1.851 ±0.143	0.1423 ±0.1311	6.007 ±4.882	0.01116 ±0.00518	8.249 ±6.860	0.01490 ±0.00725
PDE-Refiner	1.478 ±0.050	0.1338 ±0.0959	4.111 ±0.802	0.01099 ±0.00281	5.670 ±1.194	0.01482 ±0.00408
Vanilla CFM, one-step	1.585 ±0.019	0.1329 ±0.0181	6.162 ±0.276	0.02074 ±0.00092	7.139 ±0.778	0.02484 ±0.00210
Vanilla CFM, B4/2	1.508 ±0.014	0.0578 ±0.0071	3.025 ±0.280	0.01256 ±0.00077	3.626 ±0.389	0.01512 ±0.00108
Vanilla CFM, B4/8	1.510 ±0.010	0.0685 ±0.0014	4.651 ±0.474	0.01696 ±0.00046	5.925 ±0.648	0.02107 ±0.00062
KoopFM-vec, one-step	1.505 ±0.042	0.0761 ±0.0232	2.588 ±0.953	**0.00456** ±0.00177	2.738 ±1.355	**0.00395** ±0.00239
KoopFM-vec, B4/2	1.496 ±0.031	0.0602 ±0.0113	4.763 ±1.264	0.00529 ±0.00418	5.980 ±1.740	0.00595 ±0.00583
KoopFM-vec, B4/8	**1.477** ±0.025	0.0566 ±0.0093	4.301 ±0.156	0.00545 ±0.00414	5.381 ±0.181	0.00607 ±0.00550
KoopFM-vec ($d_z = 144$), B4/8	1.513 ±0.011	**0.0563** ±0.0043	**1.678** ±0.114	0.01043 ±0.00195	**1.851** ±0.102	0.01286 ±0.00270

pseudo-spectrally at 256^2 with $\Delta t = 10^{-4}$, viscosity $\nu = 10^{-3}$ and linear drag 0.1, driven by a random forcing on the wavenumbers $(6, 0)$, $(7, 0)$, $(5, 5)$ and $(8, 8)$ with amplitude $\varepsilon = 1$. Snapshots are saved every 0.5 time units, block-mean downsampled to 64^2, and normalized by a vorticity scale of 3. Training data come from three generation seeds, each with 500 trajectories of 200 snapshots after 100 warm-up snapshots. On the deterministic task the forcing is held fixed over each snapshot interval and appended to the model input as an extra channel; on the stochastic task it is resampled at every solver step and not observed. One-step statistics are scored on 48 held-out initial conditions, against $5{,}000$ solver draws each on the stochastic task and against the true next frame on the deterministic one.

Models. All flow-based models and the U-Net regressor share the StocBench U-Net, with 64 base channels, channel multipliers $(1, 2, 2, 2)$, two residual blocks per level, self-attention at the second level, dropout 0.1 and circular padding. The FNO uses 16 Fourier modes per axis, width 64 and four layers, and is trained with 4-step backpropagation through time; PDE-Refiner uses $K = 3$ refinement steps with $\sigma_{\min} = 3.16 \times 10^{-2}$. The KoopFM anchor is a periodic convolutional autoencoder with two stride-2 stages, giving a field latent of 64 channels on a 16×16 grid, and a single damped-rotation operator A shared over the whole periodic domain. Unlike Section 4.1, the anchor has no forcing path, $z_{t+1} = Az_t + b$: on the deterministic task the forcing reaches it only through the encoder input, and on the stochastic task not at all. It is trained with the reconstruction, prediction and latent (MSE plus cosine, weight 0.1) losses of Algorithm 1 over a 4-step unroll, without input noise or the cycle-consistency term, using AdamW (learning rate 10^{-4}, weight decay 10^{-4}) with cosine annealing to 10^{-6}. Stage 2 feeds the U-Net $[x_t, \hat{x}_{t+1}, x_\tau]$, with σ an EMA (weight 10^{-3}, initial value 0.1) of the anchor's per-channel RMSE. KoopFM and Vanilla CFM both use $\sigma_{\min} = 10^{-3}$ and Euler integration.

Training and evaluation. Every model is trained for $1{,}000$ epochs from a single training seed with batch size 256 and an exponential moving average of the weights (decay 0.999); apart from the anchor, all use Adam at a constant learning rate of 10^{-4} without weight decay. Evaluation uses $3{,}000$-member ensembles, inference budgets of 1, 2, 3, 4, 5, 10, 20, 30, 40 and 50 NFE, and 50-step autoregressive rollouts for the RMSE and enstrophy-spectrum metrics.

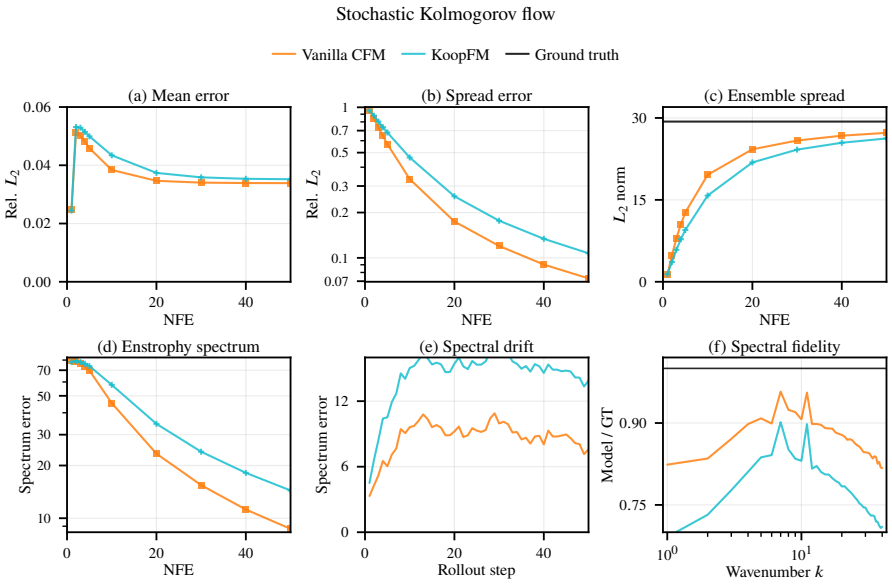


Figure 5: **Stochastic Kolmogorov flow.** Top row, against NFE: one-step errors of the conditional mean (a) and spread (b), and the predicted spread against the solver reference (c). Bottom row: enstrophy spectrum error against NFE (d) and over a 50-step rollout (e), with the resulting spectra relative to the solver (f). KoopFM trails Vanilla CFM slightly on every metric, and both underestimate the spread and the small-scale enstrophy.

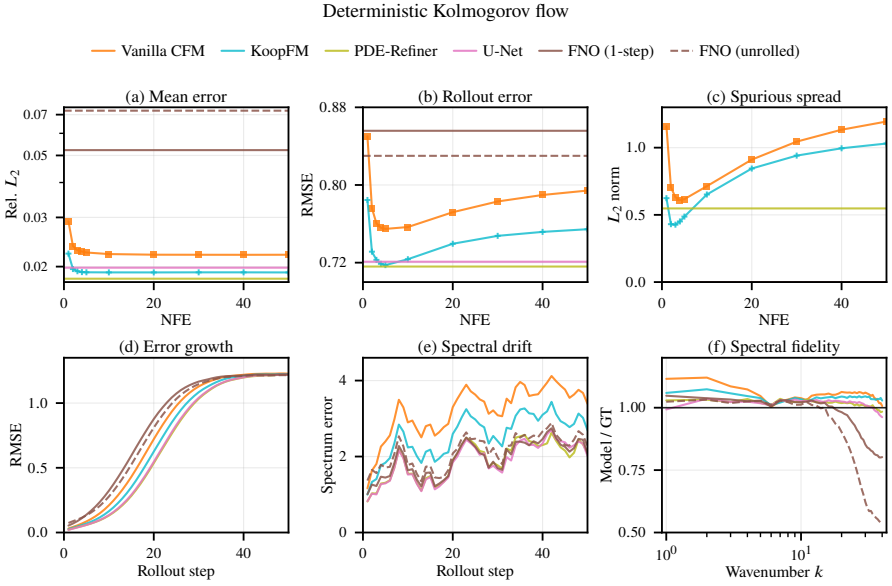


Figure 6: **Deterministic Kolmogorov flow.** Top row, against NFE: one-step mean error (a), rollout RMSE (b) and spurious predictive spread (c), which is epistemic on this task. Bottom row, over a 50-step rollout: RMSE (d) and enstrophy spectrum error (e), with the resulting spectra relative to the solver (f). KoopFM's one-step error is below the U-Net's and within about 5% of PDE-Refiner's, and it has the least spurious spread of the two flow models.

E RAYLEIGH–BÉNARD CONVECTION

E.1 DATA GENERATION

Equations and solver. We integrate the two-dimensional Boussinesq equations in diffusive units (length H, time H^2/κ),

$$\nabla\cdot\mathbf{u}=0, \qquad \partial_t b-\nabla^2 b=-\mathbf{u}\cdot\nabla b, \qquad \partial_t\mathbf{u}-Pr\,\nabla^2\mathbf{u}+\nabla p=-\mathbf{u}\cdot\nabla\mathbf{u}+Ra(t)\,Pr\,b\,\mathbf{e}_z, \quad (14)$$

with $\mathbf{u}=(u,w)$, on $x\in[0,4)$ (periodic) and $z\in[0,1]$, with $Pr=1$, no-slip walls, $b=1$ at $z=0$, $b=0$ at $z=1$ and the pressure gauge $\int p\,\mathrm{d}x\,\mathrm{d}z=0$. Dedalus v3 (Burns et al., 2020) discretizes x with 512 real Fourier modes and z with 128 Chebyshev polynomials, with $3/2$ dealiasing and the tau method for the boundary conditions, following the Rayleigh–Bénard setup of The Well (Ohana et al., 2025). Time integration uses the second-order implicit–explicit Runge–Kutta scheme RK222 with an adaptive CFL step (safety factor 0.4, $\Delta t\le 10^{-3}$). The forcing enters only through the buoyancy term of Eq. 14, whose Rayleigh number is reset before every solver step from the schedule $Ra(t)=f(t/t_{\mathrm{end}})$, $t_{\mathrm{end}}=2.5$. Each run starts from the conductive profile $b=1-z$ plus Gaussian noise of amplitude 10^{-3} tapered by $z(1-z)$; the random seed of that noise is the only difference between the initial conditions (ICs) of a schedule.

Schedules and splits. Table 9 lists the seven forcing schedules. Snapshots of (b,p,u,w), Ra and Nu are written every $\Delta t_s=5\times10^{-4}$ (5,000 per run); the first 200 ($t\le0.1$) are discarded as start-up transient, leaving 4,791–4,794 frames per trajectory, and one emulator step is one snapshot interval. Training uses four schedules with four ICs each (16 trajectories, 76,688 frames), in-distribution validation the fifth IC of each training schedule, out-of-distribution validation $\mathrm{RBC}_{\text{dec-high}}$ with ten ICs, and the test set $\mathrm{RBC}_{\text{inc}}$ and $\mathrm{RBC}_{\text{inc-dec}}$ with ten and nine ICs (one $\mathrm{RBC}_{\text{inc-dec}}$ run is missing). $\mathrm{RBC}_{\text{dec-high}}$ peaks at 3.8×10^6, so no validation-based selection on this system, neither the anchor checkpoint (Appendix E.4) nor PDE-Refiner's $\sigma_{\min}$ (Appendix E.5), sees Ra above the 1.8×10^7 ceiling of the high-Ra holdout (Section 5.3). That holdout removes the four $\mathrm{RBC}_{\text{inc-high}}$ trajectories, the only training data above the $\mathrm{RBC}_{\text{plateau-high}}$ plateau at 1.8×10^7, and the low-data control removes the four $\mathrm{RBC}_{\text{plateau-low}}$ trajectories instead. Both reuse the normalization statistics and forcing scaling of the full training set.

Preprocessing and model inputs. For half-precision storage, each pressure snapshot has its domain mean removed and is divided by 10^4 (only ∇p enters Eq. 14). The hydrostatic part that balances $Ra\,Pr\,b\,\mathbf{e}_z$ is retained, so the pressure amplitude grows roughly linearly with Ra. All models receive the four channels standardized with the per-channel training mean and standard deviation, and the forcing input u_t is $\log_{10}Ra$ of the target frame, min–max scaled to the training range $[10^6,10^8]$. Metrics are computed after undoing the standardization, i.e. in the stored units above.

Table 9: **Rayleigh–Bénard forcing schedules**, in units of 10^6, with $\tau=t/t_{\mathrm{end}}\in[0,1]$, $s(y)=1/(1+e^{-y})$, and $\ell_{k,\tau_0}(\tau)$ the logistic $s(k(\tau-\tau_0))$ rescaled affinely to 0 at $\tau=0$ and 1 at $\tau=1$. The range is that of the stored frames, after the start-up transient. ICs are per split; each training schedule has a fifth IC for in-distribution validation.

Schedule	Split	$Ra(\tau)/10^6$	Range (10^6)	ICs
$\mathrm{RBC}_{\text{inc-high}}$	train	$100^{\tau^{1.5}}$	$1.0\to100$	4
$\mathrm{RBC}_{\text{plateau-high}}$	train	$1+17\,\ell_{11,0.5}(\tau)$	$1.0\to18$	4
$\mathrm{RBC}_{\text{plateau-low}}$	train	$1+9\,\ell_{11,0.42}(\tau)\,\big(1-0.3\,s(8(\tau-0.7))\big)$	$1.1\to8.4\to7.5$	4
$\mathrm{RBC}_{\text{dec-low}}$	train	$1+\big(70^{\tau^{1.2}}-1\big)\big(1-s(18(\tau-0.45))\big)$	$1.1\to3.2\to1.0$	4
$\mathrm{RBC}_{\text{dec-high}}$	val. OOD	as $\mathrm{RBC}_{\text{dec-low}}$ but $s(16(\tau-0.42))$, $+1.5\,\ell_{12,0.35}(\tau)$	$1.1\to3.8\to2.5$	10
$\mathrm{RBC}_{\text{inc}}$	test	$35^{\tau^{1.15}}$	$1.1\to35$	10
$\mathrm{RBC}_{\text{inc-dec}}$	test	Gaussian bell, peak at $\tau=0.5$, widths 0.15/0.25	$1.1\to20\to2.0$	9

E.2 EVALUATION METRICS

All metrics are computed per IC, on a single sampled rollout unless an ensemble is stated, and then averaged over ICs. $\langle\cdot\rangle$ denotes the unweighted mean over the 128×512 grid points.

State L_2. For each snapshot and channel, $\|\hat{x}_c - x_c\|_2 / \|x_c\|_2$ over the grid, averaged over the four channels and over the first 50 steps of the window.

$W_1(Nu)$ and $W_1(Re)$. From predicted and true snapshots we form $Nu(t) = 1 + \langle bw \rangle$ and $Re(t) = \langle u^2 + w^2 \rangle^{1/2} L_z / Pr$. W_1 is the one-dimensional Wasserstein distance between the empirical distributions of the predicted and the true values over all steps of the evaluated period, the full trajectory ($T = 4790$) unless stated.

$W_1(p)$. For each snapshot, the one-dimensional Wasserstein distance between the empirical distributions of the 65,536 grid-point pressure values of prediction and truth, which for equal sample sizes is the mean absolute difference of the sorted values; the reported score averages this over all steps. Since only the domain mean is removed, these one-point distributions include the hydrostatic component, and the metric does not depend on where in the domain a value occurs.

LSD. Each snapshot is multiplied by an energy-compensated two-dimensional Hann window and Fourier transformed (forward normalization), and the power $\frac{1}{2}|\hat{f}|^2$ is summed over unit-width shells of the index wavenumber $|n| = (n_x^2 + n_z^2)^{1/2}$, keeping $|n| \leq 42$ (2/3 of the vertical Nyquist index). With spectra P_t and $\hat{P}_t$, $\mathrm{LSD}_t = \left[\mathrm{mean}_n \left(\log_{10} P_t(n) - \log_{10} \hat{P}_t(n)\right)^2\right]^{1/2}$, with a floor of 10^{-10} inside the logarithm, and the reported score averages LSD_t over all steps. LSD(p) uses the pressure. LSD(E) uses the vorticity $\omega = \partial_x w - \partial_z u$ from second-order central differences. The Chebyshev vertical grid is treated as uniform in the Fourier transform and the finite differences, identically for every model.

Windows and tests. Ra-resolved results use 50-step windows starting at frames 0, 3219 and 4603 of $\mathrm{RBC}_{\mathrm{inc}}$ ($Ra \approx 1.1 \times 10^6$, 10^7, 3×10^7) and 0, 975 and 2046 of $\mathrm{RBC}_{\mathrm{inc\text{-}dec}}$ ($Ra \approx 1.1 \times 10^6$, 5×10^6, 1.9×10^7), within which Ra drifts by 2–10%. Paired comparisons use the one-sided exact sign-flip test of Appendix E.7, $p = (\#\{\text{sign assignments with mean} \geq \text{observed}\} + 1)/(2^n + 1)$, whose smallest attainable value is 0.002 for $n = 10$ and 0.004 for $n = 9$; confidence intervals are percentile-bootstrap intervals (20,000 resamples) on the mean paired difference.

E.3 Experimental details and hyperparameters

All Rayleigh–Bénard models are trained at $N_x = 512$, $N_z = 128$ with AdamW, batch size 32 per GPU on 16 GH200 GPUs, cosine learning-rate annealing, gradient-norm clipping at 1.0 and an exponential moving average of the weights (decay 0.9999), which is the model evaluated. The deterministic U-Net, KoopFM, Vanilla CFM and PDE-Refiner share one time-conditioned U-Net backbone (base width 32, multipliers $[1, 2, 4, 8]$, adaptive group normalization for τ and forcing); flow models are sampled with Euler steps at 10 NFE unless stated. KoopFM's Stage 2 (Algorithm 2) uses learning rate 2×10^{-4} annealed to 0, weight decay 10^{-5}, σ-EMA weight $\beta = 10^{-3}$, conditioning noise $\zeta = 0.05$ and $\sigma_{\min} = 0$; the anchor is described in Appendix E.4. Models trained with unrolling (FNO BPTT-4 and the anchor) use windows of 5 frames, one-step models windows of 2.

Compute and parameter parity. All models are scheduled for 500 epochs at 100–120 GPU-hours each, and the last checkpoint is evaluated. The KoopFM anchor is trained beforehand for 300 epochs on 4 GPUs (20 GPU-hours). Parameter counts range from 0.90× to 1.13× PDE-Refiner's; KoopFM-field adds a 164K-parameter anchor, 1.7% of the model. The FNO trains more cheaply per step and is therefore given 4-step backpropagation through time, an advantage rather than a parity requirement: every other model would cost at least 4× its training compute under BPTT-4, and the generative models 4× that again in network evaluations per step (Table 10).

Inference and training cost. The anchor is effectively free at inference: KoopFM runs at 21.9 ms against 24.3 ms for Vanilla CFM at 2 NFE, whereas substituting the U-Net or the FNO as anchor would roughly double the model (1.94× and 1.90×). PDE-Refiner is fixed at $K+1 = 4$ network evaluations, while KoopFM's NFE is a free choice: State L_2 is flat down to 2 NFE (0.105 against 0.108 at 10), whereas LSD(p) rises from 0.504 to 0.737. Training is where KoopFM is behind, at 20 GPU-hours for the anchor plus 112 for the flow against 108 for PDE-Refiner, about 1.22×.

Table 10: **Trainable parameters of the Rayleigh–Bénard models.** †FNO's spectral weights are complex: 8.61M by parameter count, 17.0M real degrees of freedom.

model	trainable	of which anchor	vs. PDE-Refiner
PDE-Refiner / Vanilla CFM	9.58M	—	$1.000\times$
U-Net (1-step)	9.03M	—	$0.943\times$
FNO (BPTT-4)	8.61M†	—	$0.899\times$
KoopFM-vec	10.84M	1.27M	$1.132\times$
KoopFM-field	**9.74M**	**0.164M**	$\mathbf{1.017}\times$

Table 11: **One-step buoyancy RMSE against number of function evaluations.** The anchored model at 2 NFE is better than Vanilla CFM at any budget. In autoregressive rollout the anchored model's State L_2 is flat across $5/10/50$ NFE (0.1069 / 0.1077 / 0.1073).

NFE	1	**2**	10	50	200
KoopFM-field (ours)	0.00152	**0.00112**	0.00118	0.00122	0.00124
Vanilla CFM	0.00226	0.00151	0.00126	0.00127	0.00129

E.4 Base predictor: architecture and training

Encoder and latent geometry. Four stride-2 convolutions map the $4 \times 128 \times 512$ state to $z_t \in \mathbb{R}^{64\times8\times32}$, a $16\times$ downsampled grid carrying 64 channels, with widths $[16, 32, 64, 64]$, GroupNorm, and a final 1×1 convolution. The decoder mirrors it.

The transition operator. A is parameterized in real modal canonical form as 32 rotation-dilation blocks per latent site, each with modulus $\gamma_k \in (0, 1)$. Block parameters are *shared* along the periodic x axis and *free* along the wall-bounded z axis, so every latent row carries its own spectrum while the operator remains translation equivariant in the homogeneous direction. That amounts to 512 free parameters. Written on the flattened latent, A is block diagonal with 8192 blocks generated by those 512 values.

The forcing path. $e_t = \psi(u_t) \in \mathbb{R}^{64}$ contracts against $G \in \mathbb{R}^{64\times8\times1\times64}$ to give a per-row latent increment that broadcasts along x, matching the z-structure of buoyancy forcing, plus a per-row bias b. The forcing path is 33K parameters against 130K for the encoder–decoder and 512 for A, 164K in total.

The KoopFM-vec anchor. The ablation baseline flattens the encoder output through a $\mathrm{Linear}(4096 \to 128)$ bottleneck and applies a flat operator on $z_t \in \mathbb{R}^{128}$, at 1.27M parameters.

Training. Algorithm 1 with $M = 4$ (windows of 5 frames), $\lambda = 0.1$, and encoder-input noise of relative scale $\eta = 0.1$ on the cosine ramp. The forcing input u is $\log_{10} Ra$ of the last frame of the window, min–max scaled to the training range. We use AdamW (weight decay 10^{-4}) with cosine annealing to 10^{-6} over all steps, gradient-norm clipping at 1.0, an exponential moving average of the weights (decay 0.9999), and batch size 128 per GPU on 4 GPUs for 300 epochs, keeping the checkpoint with the lowest OOD validation loss. The KoopFM-field anchor uses $\mu = 1$ and learning rate 10^{-4}; the KoopFM-vec anchor uses $\mu = 0$ (no cycle-consistency term) and learning rate 2×10^{-4}. Stage 2 freezes the selected checkpoint.

E.5 Baselines

Deterministic baselines. The U-Net baseline uses an identical backbone without flow conditioning. FNO uses 16 Fourier modes per dimension, width 64, 4 layers.

PDE-Refiner: tuning $\sigma_{\min}$. Lippe et al. (2023) identify the minimum noise level as PDE-Refiner's critical hyperparameter and tune it per system. Their one-dimensional value had been used here initially. They also state how to choose it. The value "is chosen based on the frequency spectrum of the given data" (their Section 3), the selection itself is made by a search on validation data (their

Appendix D.1), and their Appendix E.7 reports a broad plateau with performance falling only once the noise exceeds the lowest-amplitude content. We follow that procedure.

One conversion matters. The refinement schedule is $\sigma_k = \sigma_{\min}^{k/K}$ with $\sigma_{\min}$ a standard deviation, but the values they quote are variances. Thus $\sigma_{\min}^2 = 2 \times 10^{-7}$ for Kuramoto–Sivashinsky is $\sigma_{\min} = 4.5 \times 10^{-4}$, and $\sigma_{\min}^2 = 10^{-3}$ for their two-dimensional Kolmogorov flow is $\sigma_{\min} = 3.2 \times 10^{-2}$. The search carried out here spans both of the authors' own settings.

Their selection rule is applied to the data used here as follows. Over 384 training frames in normalized units, the RMS amplitude of content above 0.7 of Nyquist is 0.018 for buoyancy and 0.0054 for pressure, while both velocity components are already down at 8×10^{-5}; above 0.8 of Nyquist every channel is at or below 10^{-4} and the DNS spectrum terminates. The published one-dimensional $\sigma_{\min} = 4.5 \times 10^{-4}$ therefore sits roughly $40\times$ below the finest buoyancy structure, and the two-dimensional 3.2×10^{-2} sits above it.

We therefore trained PDE-Refiner at three values of $\sigma_{\min}$, namely the authors' 1D setting, the authors' 2D setting, and one derived from the present spectrum by their rule. Only that value was changed, leaving K, the number of function evaluations, the objective and the architecture untouched, and the selection among them was made on validation as they do. Both tuned values improve the baseline by a wide margin. On $\mathrm{RBC}_{\text{dec-high}}$, autoregressive $W_1(Nu)$ falls from 0.344 to 0.083 and 0.070, and CRPS from 0.0110 to 0.0052 and 0.0031. The two are not separable on every criterion, since 2×10^{-3} wins the pressure-spectral scores and 3.162×10^{-2} the probabilistic ones. The authors' own 2D value, $\sigma_{\min} = 3.162 \times 10^{-2}$, is therefore reported. It needs no justification from the present work and is the stronger of the two on CRPS, by -40%, and on calibration, at spread-skill $0.50 \to 0.93$. We keep 2×10^{-3} as a second row throughout so that the choice remains visible.

Table 12: $\sigma_{\min}$ **selection on** $\mathrm{RBC}_{\text{dec-high}}$ (10 seeds), which is disjoint from both reported test scenarios. Lower is better except Spread/Skill, where 1.0 is calibrated. Both tuned values beat the published 1D default by a wide margin. Between them the decision splits by metric class, and the authors' own 2D value is the one reported.

	autoregressive			teacher-forced, $N = 8$		
$\sigma_{\min}$	$W_1(Nu)$	$W_1(Re)$	LSD(p)	CRPS	EnsMean L_2	Spread/Skill
4.5×10^{-4} *(their 1D)*	0.3439	9.444	0.3973	0.0110	0.0175	0.283
2×10^{-3} *(this spectrum)*	0.0830	2.403	**0.3182**	0.0052	0.0101	0.497
3.162×10^{-2} *(their 2D)*	**0.0698**	**2.162**	0.5013	**0.0031**	**0.0063**	**0.933**

The tuning is consequential, and not in favor of the anchored model. At 3.162×10^{-2} the baseline is better calibrated in the two laminar teacher-forced windows, at spread-skill 0.92 against 0.67 on $\mathrm{RBC}_{\text{inc}}$, and attains a lower CRPS there. Under autoregressive rollout, the mode that matters for an emulator, the anchored model remains ahead of it on CRPS and on calibration at all six Ra windows. The $W_1(p)$ result is unaffected. The two tuned settings are tied on it (validation 1.237 against 1.272, $p = 0.15$), and both remain behind the anchored model on test.

Driving each checkpoint's own sampler and recording the intermediate states locates the gain. The deterministic k=0 branch improves from 0.0424 to 0.0358 per-channel relative L_2, the refinement chain contributes $\sim 15\%$ in *both* models, and the final refinement step changes nothing in either. The tuned baseline's advantage is therefore in its deterministic regressor rather than in its generative refinement, which is consistent with its remaining under-dispersion and its over-smoothed spectrum (Table 16).

Paired sign-flip over the ten validation initial conditions separates the two tuned settings largely by metric class. 3.162×10^{-2} is significantly better on teacher-forced $W_1(Re)$ and LSD(E) ($p = 0.002$), on CRPS ($d = -0.0021$) and on calibration ($d = -0.378$ in $|\mathrm{S}/\mathrm{S} - 1|$); 2×10^{-3} is significantly better on LSD(p) in both modes ($p = 0.002$) and on teacher-forced $W_1(p)$ ($p = 0.002$). The five remaining comparisons are ties. The ordering replicates on the test scenarios, so the selection is not an artifact of the split.

E.6 MEASUREMENT DETAILS

Why the extrapolation windows are short. Ra ramps *inside* a window, so each point is really an interval. Over 50 steps the drift is 4.1% in Ra against a rung spacing of 10%, whereas 200 steps would drift 17.5% and smear each point across two rungs, by a model-dependent amount since error grows within a window. The computed ladder runs from 0.70 to $1.60\ Ra_{\mathrm{ceil}}$; Figure 8 and Table 22 show it up to 1.40 for legibility. It stops at 1.60 because the $\mathrm{RBC}_{\mathrm{inc}}$ trajectory ends at $1.94\ Ra_{\mathrm{ceil}}$ and both models are already collapsed beyond it, and it does *not* stop at 1.50, the untabulated rung where the advantage on $W_1(p)$ expires.

Relative error is not comparable across Ra. Two caveats matter for how the Ra-resolved tables should be read. First, relative error normalizes by the target's own RMS, and that denominator grows by $27.7\times$ for pressure, $6.1\times$ for u and $5.9\times$ for w between $Ra = 10^6$ and 3×10^7. Relative errors are therefore *not* comparable across Ra. The scale-free statement is the forecast horizon. Every model beats persistence past 50 steps at $Ra = 10^6$ and 10^7, but only for 11–21 steps at 3×10^7. Second, training coverage collapses over the same range. While 16.2% of training frames lie within ± 0.05 dex of $Ra = 1.1 \times 10^6$, only 2.4% do at 10^7 and 0.96% at 3×10^7, where a single forcing scenario contributes all of them. High-Ra degradation is therefore partly a data statement, and not only a model statement.

E.7 PAIRED STATISTICS

Every model is evaluated on the same test trajectories, so model differences should be assessed per initial condition rather than between marginal means. The shared component is large. Across initial conditions the standard deviation of short-horizon error is ≈ 0.03, an order of magnitude above the differences between models, and a single hard initial condition raises the CRPS of *every* model in the study by a factor of 2.5–5.

We use a one-sided exact paired sign-flip permutation test. Under the null that two models are equally good, the sign of each per-IC difference is exchangeable, so for $n \leq 20$ all 2^n assignments are enumerated and the observed mean is located in that null distribution (with the $+1$ correction of Appendix E.2). We first used a percentile bootstrap and rejected it. On synthetic equal-model pairs at $n = 10$ it reported a significant difference 10% of the time at a nominal 5%, which is twice the intended rate. The sign-flip test measures 6% at nominal 5%, with 67% power against a one-standard-deviation gap and 99% against two.

Pairing changes conclusions rather than only tightening intervals. On short-horizon error at $Ra \approx 10^6$, KoopFM against Vanilla CFM gives $p = 0.018$ paired and $p = 0.25$ by an unpaired Welch test on the same numbers. The gain from pairing is 1.2–$3.9\times$ in the standard deviation of the difference, and scales with how strongly two models agree about which initial conditions are hard ($r = 0.57$–0.99). It is largest between architecturally similar models and smallest against FNO (BPTT-4), which fails on a visibly different set.

The paired differences themselves. Tables 13–15 report the quantity the test actually uses, for the three metrics on which claims are made. These are the mean per-IC difference against the best model in each column, a 95% bootstrap confidence interval on that difference, and the number of initial conditions on which the model beats the best. The intervals show how large each gap is; the bold entries in the body tables are set by the one-sided sign-flip test above, not by these intervals, so the two can disagree in borderline cells. The $W_1(p)$ table gives the pressure result in its strongest form. On $\mathrm{RBC}_{\mathrm{inc}}$ under autoregressive rollout, every rival except the diverging FNO (1-step) is between $+0.13$ and $+0.79$ with a confidence interval excluding zero, and none wins more than 1 of 10 initial conditions; on $\mathrm{RBC}_{\mathrm{inc\text{-}dec}}$ the one exception is FNO (BPTT-4), which is tied (3 of 9 initial conditions won).

Since the full evaluation grid contains several hundred pairwise comparisons, significance is not marked with stars. The requirement instead is that a result **replicate across both forcing scenarios**, and p-values are reported. This is the criterion that separates the $W_1(p)$ result (15 wins and 1 tie out of 16) from the apparent high-Ra reversal of Appendix E.10, which does not replicate.

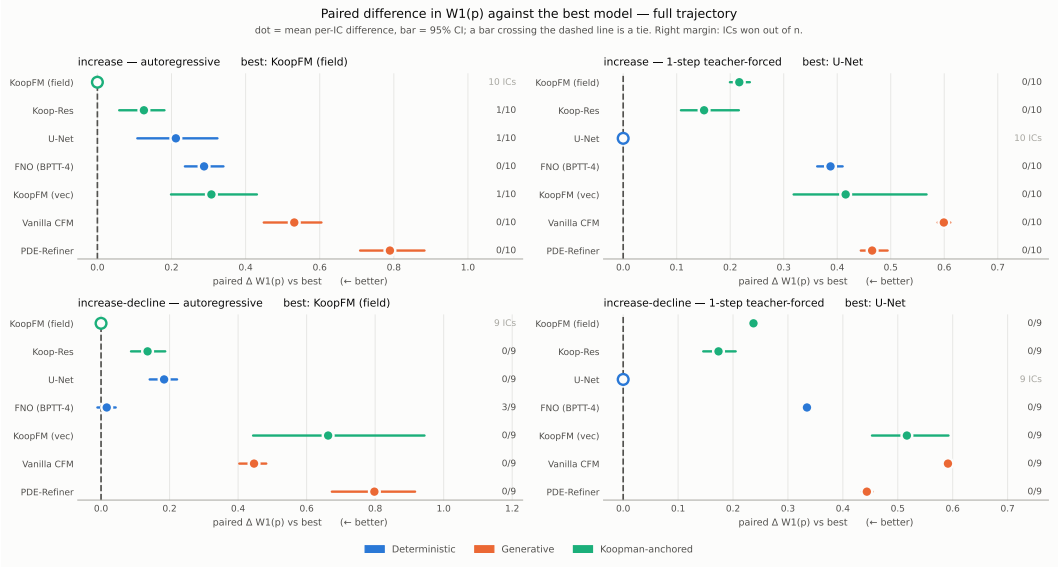


Figure 7: $W_1(p)$ **paired differences, both rollout modes.** $W_1(p)$ against the best model in each scenario, autoregressive and teacher-forced one-step. Dot: mean per-initial-condition difference; bar: 95% bootstrap confidence interval; a bar crossing the dashed line is a tie. Right margin: initial conditions won out of n. The reference model differs between modes. KoopFM is best under autoregressive rollout, and U-Net under teacher forcing.

Table 13: **Paired head-to-head, W1(p).** W1(p), full trajectory. The best model carries its absolute score (mean±sd); every other row is the mean paired difference against it, a 95% bootstrap CI on that difference, and initial conditions won out of n. Bold entries in the body tables come from the one-sided sign-flip test, not from these intervals.

Model	$\mathrm{RBC}_{\mathrm{inc}}$		$\mathrm{RBC}_{\mathrm{inc\text{-}dec}}$	
	AR	1-step TF	AR	1-step TF
FNO (BPTT-4)	+0.2879 [+0.2361, +0.3403] 0/10	+0.3875 [+0.3625, +0.4102] 0/10	+0.0166 [−0.0107, +0.0428] 3/9	+0.3346 [+0.3293, +0.3402] 0/9
FNO (1-step)	+43.1945 [+33.5327, +48.1392] 0/10	+0.1565 [+0.1457, +0.1687] 0/10	+21.1851 [+8.6988, +33.7206] 1/9	+0.1167 [+0.1126, +0.1207] 0/9
U-Net	+0.2118 [+0.1077, +0.3241] 1/10	**0.4246±0.0402** ***(ref)***	+0.1840 [+0.1418, +0.2219] 0/9	**0.3736±0.0058** ***(ref)***
Vanilla CFM	+0.5313 [+0.4489, +0.6048] 0/10	+0.5992 [+0.5877, +0.6113] 0/10	+0.4465 [+0.4034, +0.4823] 0/9	+0.5914 [+0.5870, +0.5957] 0/9
PDE-Refiner (2e-3)	+0.7895 [+0.7089, +0.8828] 0/10	+0.4652 [+0.4440, +0.4943] 0/10	+0.7978 [+0.6736, +0.9167] 0/9	+0.4437 [+0.4343, +0.4539] 0/9
PDE-Refiner (3.2e-2)	+0.2077 [+0.1456, +0.2705] 0/10	+0.3659 [+0.3522, +0.3845] 0/10	+0.1610 [+0.1241, +0.1938] 0/9	+0.3323 [+0.3235, +0.3411] 0/9
Koop-Res	+0.1255 [+0.0588, +0.1811] 1/10	+0.1512 [+0.1079, +0.2163] 0/10	+0.1358 [+0.0874, +0.1875] 0/9	+0.1736 [+0.1457, +0.2049] 0/9
KoopFM-vec	+0.3077 [+0.1985, +0.4306] 1/10	+0.4158 [+0.3185, +0.5670] 0/10	+0.6628 [+0.4437, +0.9441] 0/9	+0.5166 [+0.4530, +0.5924] 0/9
KoopFM-field	**2.1648±0.1095** ***(ref)***	+0.2170 [+0.1994, +0.2368] 0/10	**2.0554±0.0512** ***(ref)***	+0.2372 [+0.2300, +0.2454] 0/9

Across-IC spread. The body tables report means and mark significance with the paired test; full mean±sd grids for every table in the paper, giving the across-initial-condition standard deviation as a descriptive measure of how much each metric varies between initial conditions, are released with the code rather than reproduced here. Nothing in those grids is bolded, since these numbers do not carry the comparison, and reading significance off overlapping error bars is precisely the mistake the separation is meant to prevent (Section 5.3); the aggregate magnitude of that spread is instead reported once, in Appendix E.7.

E.8 Ensemble scoring and calibration

We score ensembles of $N = 8$ draws per initial condition (deterministic models: $N = 1$). CRPS uses the fair (Ferro) estimator, $\widehat{\mathrm{CRPS}} = \frac{1}{N}\sum_i |x_i - y| - \frac{1}{2N(N-1)}\sum_{i,j} |x_i - x_j|$, whose $(N-1)$ denominator removes the small-ensemble bias of the naive $1/(2N^2)$ form; at $N = 1$ it reduces exactly to the mean absolute error, so deterministic and generative models are ranked by the same proper score. CRPS is normalized per channel by the RMS of the truth before averaging, since the four fields differ by roughly three orders of magnitude. Spread is the root-mean-square ensemble spread $\sqrt{\overline{\mathrm{Var}}}$, not the spatial mean of the pointwise standard deviation, which under-reports it by up

Table 14: **Paired head-to-head, LSD(p).** LSD(p), full trajectory. The best model carries its absolute score (mean±sd); every other row is the mean paired difference against it, a 95% bootstrap CI on that difference, and initial conditions won out of n. Bold entries in the body tables come from the one-sided sign-flip test, not from these intervals.

Model	$\mathrm{RBC}_{\mathrm{inc}}$		$\mathrm{RBC}_{\mathrm{inc\text{-}dec}}$	
	AR	1-step TF	AR	1-step TF
FNO (BPTT-4)	+0.8803 [+0.8741, +0.8853] 0/10	+1.0504 [+1.0444, +1.0555] 0/10	+0.7826 [+0.7779, +0.7878] 0/9	+0.9370 [+0.9334, +0.9403] 0/9
FNO (1-step)	+1.0504 [+0.9676, +1.0961] 0/10	+0.8510 [+0.8439, +0.8570] 0/10	+0.9585 [+0.7394, +1.1851] 0/9	+0.7476 [+0.7438, +0.7511] 0/9
U-Net	+0.5722 [+0.5684, +0.5759] 0/10	+0.5240 [+0.5208, +0.5273] 0/10	+0.4690 [+0.4644, +0.4732] 0/9	+0.4339 [+0.4317, +0.4359] 0/9
Vanilla CFM	+0.4090 [+0.4039, +0.4131] 0/10	+0.5066 [+0.5040, +0.5094] 0/10	+0.2803 [+0.2765, +0.2842] 0/9	+0.3669 [+0.3652, +0.3687] 0/9
PDE-Refiner (2e-3)	**0.2762±0.0074** ***(ref)***	**0.1326±0.0049** ***(ref)***	**0.2361±0.0051** ***(ref)***	**0.0958±0.0021** ***(ref)***
PDE-Refiner (3.2e-2)	+0.1077 [+0.1045, +0.1107] 0/10	+0.1538 [+0.1501, +0.1571] 0/10	+0.0634 [+0.0592, +0.0673] 0/9	+0.0955 [+0.0940, +0.0972] 0/9
Koop-Res	+0.5077 [+0.4967, +0.5189] 0/10	+0.6208 [+0.6035, +0.6456] 0/10	+0.3965 [+0.3879, +0.4053] 0/9	+0.4984 [+0.4862, +0.5133] 0/9
KoopFM-vec	+0.2263 [+0.2159, +0.2404] 0/10	+0.3251 [+0.3165, +0.3359] 0/10	+0.1581 [+0.1505, +0.1687] 0/9	+0.2500 [+0.2441, +0.2568] 0/9
KoopFM-field	+0.1550 [+0.1494, +0.1602] 0/10	+0.2487 [+0.2460, +0.2509] 0/10	+0.1048 [+0.1005, +0.1091] 0/9	+0.1879 [+0.1866, +0.1892] 0/9

Table 15: **Paired head-to-head, State L2.** State L2 on 50-step windows, autoregressive. The best model carries its absolute score (mean±sd); every other row is the mean paired difference against it, a 95% bootstrap CI on that difference, and initial conditions won out of n. Bold entries in the body tables come from the one-sided sign-flip test, not from these intervals.

Model	$\mathrm{RBC}_{\mathrm{inc}}$			$\mathrm{RBC}_{\mathrm{inc\text{-}dec}}$		
	Ra 1.1e6	Ra 1e7	Ra 3e7	Ra 1.1e6	Ra 5e6	Ra 2e7
FNO (BPTT-4)	+0.0006 [−0.0150, +0.0143] 3/10	**0.0519±0.0242** ***(ref)***	+0.0041 [−0.0193, +0.0273] 4/10	+0.0005 [−0.0173, +0.0164] 3/9	+0.0010 [−0.0045, +0.0064] 4/9	+0.0036 [−0.0213, +0.0257] 3/9
FNO (1-step)	+0.0081 [−0.0087, +0.0284] 4/10	+0.0210 [−0.0002, +0.0461] 3/10	+0.0225 [−0.0053, +0.0528] 4/10	+0.0001 [−0.0125, +0.0127] 4/9	**0.0548±0.0082** ***(ref)***	**0.1554±0.0352** ***(ref)***
U-Net	+0.0076 [−0.0001, +0.0145] 3/10	+0.0196 [+0.0025, +0.0374] 2/10	+0.0266 [−0.0023, +0.0583] 3/10	+0.0091 [−0.0003, +0.0173] 2/9	+0.0156 [+0.0105, +0.0214] 0/9	+0.0188 [−0.0006, +0.0392] 3/9
Vanilla CFM	+0.0087 [+0.0024, +0.0148] 1/10	+0.0170 [+0.0001, +0.0345] 3/10	+0.0205 [−0.0064, +0.0458] 4/10	+0.0102 [+0.0045, +0.0162] 1/9	+0.0144 [+0.0086, +0.0199] 1/9	+0.0165 [+0.0040, +0.0279] 2/9
PDE-Refiner (2e-3)	+0.0258 [+0.0093, +0.0413] 3/10	+0.0321 [+0.0118, +0.0550] 2/10	+0.0258 [+0.0046, +0.0462] 3/10	+0.0325 [+0.0165, +0.0482] 2/9	+0.0224 [+0.0143, +0.0324] 0/9	+0.0337 [+0.0144, +0.0532] 2/9
PDE-Refiner (3.2e-2)	+0.0120 [−0.0015, +0.0255] 3/10	+0.0288 [+0.0077, +0.0517] 2/10	+0.0251 [−0.0056, +0.0572] 3/10	+0.0143 [−0.0000, +0.0281] 3/9	+0.0193 [+0.0095, +0.0318] 1/9	+0.0160 [−0.0015, +0.0302] 1/9
Koop-Res	+0.0017 [−0.0073, +0.0104] 5/10	+0.0124 [−0.0024, +0.0285] 4/10	+0.0390 [+0.0046, +0.0791] 4/10	+0.0052 [−0.0031, +0.0131] 4/9	+0.0051 [+0.0011, +0.0100] 3/9	+0.0002 [−0.0183, +0.0160] 3/9
KoopFM-vec	+0.0338 [+0.0077, +0.0626] 2/10	+0.0389 [+0.0184, +0.0610] 2/10	**0.1606±0.0418** ***(ref)***	+0.0405 [+0.0153, +0.0690] 2/9	+0.0399 [+0.0257, +0.0555] 0/9	+0.0186 [−0.0065, +0.0420] 2/9
KoopFM-field	**0.0730±0.0318** ***(ref)***	+0.0280 [+0.0091, +0.0489] 3/10	+0.0170 [−0.0074, +0.0402] 3/10	**0.0658±0.0207** ***(ref)***	+0.0119 [+0.0073, +0.0157] 1/9	+0.0255 [+0.0085, +0.0446] 2/9

to 21% on a heterogeneous field; on a synthetic perfectly calibrated ensemble this estimator returns 1.000. Spread-skill is formed per channel and averaged unweighted.

Results. No model in this study is calibrated under autoregressive rollout, the mode that matters for an emulator: without re-synchronizing on the truth every ensemble is under-dispersed, its spread saturating while the error keeps growing. KoopFM sits at spread-skill 0.24–0.57 across the six Ra windows of the two test scenarios, and Vanilla CFM is the best calibrated model at five of the six. Under teacher forcing, a 500-step average of KoopFM's ratio lands near 1 (0.96), but only because the ratio sweeps through 1 inside the window, from 0.69 to 1.18 across lead-time bands; resolved in Ra it ranges from 0.54 to 0.95. CRPS shows the same window effect: in the conventional window KoopFM is tied for best with the U-Net and Vanilla CFM, but at every other combination of scenario and Ra Vanilla CFM wins ($p \leq 0.004$). We therefore claim neither a calibration nor a CRPS advantage. More generally, a spread-skill ratio averaged over a window that contains its own crossing point is not evidence of calibration, and teacher-forced ensemble scores say little about autoregressive ones.

E.9 Spectral bias

Table 16 is the full spectral-bias grid behind the spectral statements of Sections 4.1 and 5.3. Rolled out alone, the base predictor's velocity-band error is 7–48× that of any full model. Among the full models, the split of Table 1 holds spectrally: KoopFM-field has the smallest mean pressure-band bias and Vanilla CFM the smallest velocity-band bias.

E.10 Ra-resolved and teacher-forced tables

Table 17 gives the autoregressive $\mathrm{RBC}_{\mathrm{inc}}$. The remaining three combinations follow, together with the teacher-forced whole-trajectory statistics. Ra columns are labeled by actual Rayleigh number. The two scenarios reach different Ra, so the two halves are not directly comparable.

E.11 The extrapolation ladder

Figure 8 extends Figure 4 to the 0.70–1.40× ladder, and Table 22 is the rung-by-rung form of its above-ceiling range, 1.00 to 1.40 Ra_{ceil}, and the source of the crossover distances quoted in

Table 16: **Spectral bias**, $\log_{10}(\langle P_{\text{pred}}\rangle_{t,s}/\langle P_{\text{true}}\rangle_{t,s})$ over the autoregressive rollout; 0 is exact, positive is spurious small-scale energy, negative is over-smoothing. PSDs are time-averaged *before* the logarithm. Left: pressure, per wavenumber. Right: band mean and maximum $|D|$ over $k \in [20, 42]$, for pressure and for the velocity components u, w. All columns are RBC_{inc}.

	pressure, per wavenumber			pressure band $\lvert D\rvert$		velocity band $\lvert D\rvert$	
	$p@k$=20	$p@k$=30	$p@k$=42	mean	max	mean	max
FNO (BPTT-4)	+0.25	+0.68	+1.09	0.685	1.090	0.167	0.321
U-Net (1-step)	+0.11	+0.38	+0.77	0.406	0.769	0.033	0.065
Koop-Res	+0.06	+0.20	+0.62	0.257	0.618	0.030	0.061
PDE-Refiner (2×10^{-3})	−0.05	−0.10	−0.18	0.113	0.187	0.103	0.275
PDE-Refiner (3.2×10^{-2})	−0.05	−0.06	−0.06	0.060	**0.074**	0.070	0.165
Vanilla CFM	−0.01	+0.03	+0.34	0.090	0.343	**0.026**	**0.047**
KoopFM-vec	+0.02	+0.06	+0.18	0.072	0.178	0.039	0.077
KoopFM-field (ours)	+0.01	+0.04	**+0.13**	**0.056**	0.135	0.043	0.086
base predictor alone	+0.65	+0.87	+0.92	0.840	1.046	1.251	2.567

Table 17: **Ra-resolved, RBC_{inc}, autoregressive.** 50-step windows from three Ra, RBC_{inc}, autoregressive. W1(Nu)/W1(Re) are omitted: they are distributional and not estimable from 50 snapshots. Lower is better. Mean across initial conditions; **bold** is the best model and every model tied with it under the paired test (Section 5.3). Across-IC spread: Appendix E.7.

Model	Ra 1.1e6				Ra 1e7				Ra 3e7			
	State L2	W1(p)	LSD(E)	LSD(p)	State L2	W1(p)	LSD(E)	LSD(p)	State L2	W1(p)	LSD(E)	LSD(p)
FNO (BPTT-4)	**0.0735**	**0.2960**	0.1918	2.1048	**0.0519**	**0.5913**	**0.1058**	0.8181	**0.1647**	6.73	0.3086	0.4103
FNO (1-step)	**0.0810**	**0.2508**	0.1980	1.9464	**0.0729**	**0.6380**	**0.1098**	0.6326	**0.1831**	**5.92**	0.2429	0.3454
U-Net	**0.0805**	0.5036	**0.1772**	1.8275	0.0715	0.9162	**0.1120**	0.4830	**0.1872**	6.66	**0.2254**	0.3062
Vanilla CFM	0.0817	1.3070	**0.1826**	1.9405	**0.0689**	1.5300	0.1289	0.3495	**0.1811**	**6.00**	**0.2116**	0.1601
PDE-Refiner (2e-3)	0.0988	0.5971	0.2463	**0.5282**	0.0840	2.3701	0.1565	**0.1108**	0.1863	7.78	0.2444	0.1857
PDE-Refiner (3.2e-2)	**0.0850**	0.6024	0.2118	1.0222	0.0807	1.2297	0.1397	**0.1151**	**0.1857**	**6.05**	0.2296	0.1660
Koop-Res	**0.0746**	0.4527	**0.1765**	1.7539	**0.0643**	0.6833	**0.1026**	0.3748	0.1995	6.43	**0.2253**	0.2153
KoopFM-vec	0.1068	0.7854	0.2304	1.3269	0.0908	1.3425	0.1754	0.2270	**0.1606**	7.59	0.2213	**0.1544**
KoopFM-field	**0.0730**	**0.3435**	**0.1676**	1.1281	0.0799	0.9532	0.1435	0.1916	**0.1776**	**5.46**	**0.2194**	**0.1573**

Section 5.3. Table 23 gives the whole-trajectory comparison behind the $W_1(Nu)$ and $W_1(Re)$ statements there. In the $Ra \approx 3 \times 10^7$ window of RBC_{inc}, at $1.67\,Ra_{\text{ceil}}$, the ensemble-mean error ($N = 8$) of the holdout models is $3.80\times$ (KoopFM) and $5.39\times$ (Vanilla CFM) that of the corresponding full-data models on the same window.

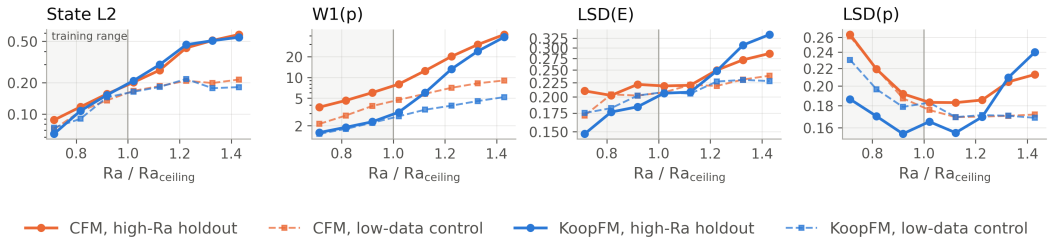


Figure 8: **Extrapolation ladder, full range.** As Figure 4, over 0.70–1.40× the training ceiling.

Pushed all the way to $\text{RBC}_{\text{inc-high}}$ itself, at $5.6\,Ra_{\text{ceil}}$, only the pressure spectrum is left: KoopFM is worse on all three magnitude statistics and on LSD(E) on 4 of 4 initial conditions and better on LSD(p) on 4 of 4. At $n = 4$ the sign-flip floor is $1/16$, so that is a direction and not a test.

E.12 The Nusselt runaway decomposed

The obvious reading of a +54% heat flux is that the Koopman operator, driven past its fitted range, grows in amplitude. That reading is measurably wrong. Every channel *shrinks*, the vertical velocity to 0.94 and buoyancy to 0.93 of their true standard deviations at 1.60×. Decomposing $Nu - 1 =$

Table 18: **Whole-trajectory statistics, teacher-forced one-step.** Full trajectory (T=4790), teacher-forced one-step. Lower is better. Mean across initial conditions; **bold** is the best model and every model tied with it under the paired test (Section 5.3). Across-IC spread: Appendix E.7.

Model	$\mathrm{RBC}_{\mathrm{inc}}$					$\mathrm{RBC}_{\mathrm{inc\text{-}dec}}$				
	W1(Nu)	W1(Re)	W1(p)	LSD(E)	LSD(p)	W1(Nu)	W1(Re)	W1(p)	LSD(E)	LSD(p)
FNO (BPTT-4)	0.1302	2.0266	0.8121	0.1058	1.1830	0.0755	2.4418	0.7082	0.0986	1.0329
FNO (1-step)	0.0738	0.8888	0.5811	0.0679	0.9836	0.0483	0.9756	0.4902	0.0643	0.8434
U-Net	**0.0280**	**0.7021**	**0.4246**	0.0354	0.6566	**0.0351**	0.5826	**0.3736**	**0.0319**	0.5297
Vanilla CFM	0.1273	0.8272	1.0239	**0.0348**	0.6392	0.1128	**0.5353**	0.9650	0.0339	0.4628
PDE-Refiner (2e-3)	0.1473	1.9015	0.8899	0.0519	**0.1326**	0.1493	1.5316	0.8173	0.0526	**0.0958**
PDE-Refiner (3.2e-2)	0.1433	1.1733	0.7905	0.0377	0.2865	0.1630	0.7184	0.7059	0.0410	0.1913
Koop-Res	0.1083	**0.6815**	0.5758	0.0374	0.7534	0.0963	**0.5453**	0.5471	0.0333	0.5942
KoopFM-vec	0.1113	1.0660	0.8404	0.0431	0.4577	0.0812	0.8540	0.8902	0.0446	0.3458
KoopFM-field	0.0847	0.9565	0.6416	0.0390	0.3813	0.0813	0.6336	0.6108	0.0402	0.2837

Table 19: **Ra-resolved, $\mathrm{RBC}_{\mathrm{inc}}$, teacher-forced one-step.** 50-step windows from three Ra, $\mathrm{RBC}_{\mathrm{inc}}$, teacher-forced one-step. W1(Nu)/W1(Re) are omitted: they are distributional and not estimable from 50 snapshots. Lower is better. Mean across initial conditions; **bold** is the best model and every model tied with it under the paired test (Section 5.3). Across-IC spread: Appendix E.7.

Model	Ra 1.1e6				Ra 1e7				Ra 3e7			
	State L2	W1(p)	LSD(E)	LSD(p)	State L2	W1(p)	LSD(E)	LSD(p)	State L2	W1(p)	LSD(E)	LSD(p)
FNO (BPTT-4)	0.0234	0.2584	0.1015	2.1146	0.0153	0.4325	0.0716	0.8209	0.0662	3.82	0.2510	0.5429
FNO (1-step)	0.0173	0.1553	0.0809	1.9018	0.0100	0.3074	0.0437	0.6384	0.0597	3.00	0.1543	0.4078
U-Net	**0.0097**	**0.1104**	0.0370	1.4276	**0.0065**	**0.2426**	**0.0210**	0.3550	0.0399	**1.98**	**0.0878**	0.2039
Vanilla CFM	0.0238	0.8338	0.0708	1.9557	0.0089	0.8817	**0.0226**	0.3427	**0.0381**	**2.05**	**0.0886**	**0.0928**
PDE-Refiner (2e-3)	0.0184	0.3646	0.0347	**0.4779**	0.0118	0.8064	0.0395	**0.0386**	0.0478	2.75	0.1454	0.1349
PDE-Refiner (3.2e-2)	0.0141	0.4600	**0.0214**	0.9849	0.0106	0.6164	0.0340	0.0608	0.0437	2.41	0.1114	0.1018
Koop-Res	0.0162	0.3458	0.0456	1.7307	0.0067	0.3300	**0.0218**	0.3589	0.0397	2.25	0.0922	0.1772
KoopFM-vec	0.0241	0.5867	0.0396	1.3134	0.0117	0.5336	0.0333	0.1951	0.0457	3.03	0.1105	0.1055
KoopFM-field	0.0164	0.2769	0.0307	1.1229	0.0111	0.4423	0.0303	0.1705	0.0479	2.64	0.1116	0.1101

$\langle bw \rangle = \mathrm{corr}(b,w)\,\sigma_b\sigma_w + \langle b\rangle\langle w\rangle$ attributes the excess to two terms of comparable size. The first is a genuine loss of structure. Here $\mathrm{corr}(b,w)$ inflates to $1.48\times$ truth as small scales decay and spurious large-scale coherence takes over, contributing $+29.5\%$ once the shrinking amplitudes are accounted for. The second is a constraint violation. Incompressibility with impermeable lids forces $\langle w\rangle = 0$ exactly, and the simulation satisfies this to $|\langle w\rangle|/\sigma_w \approx 7\times10^{-7}$. The holdout anchor instead develops a net vertical velocity that grows $13\times$ along the ladder, to 8.9×10^{-3}, and *drifts within each rollout*, from 2.2 to 9.7 over 50 steps at $1.60\times$. Since $\langle b\rangle \approx 0.5$ does not vanish, that sub-1% mean velocity is multiplied by 0.5 and lands directly in the flux, which is worth a further $+29.2\%$. The two terms sum to $+58.7\%$ against a measured flux excess of $+61.5\%$, so they account for the whole effect.

Table 20: **Ra-resolved, $\mathrm{RBC}_{\text{inc-dec}}$, autoregressive.** 50-step windows from three Ra, $\mathrm{RBC}_{\text{inc-dec}}$, autoregressive. W1(Nu)/W1(Re) are omitted: they are distributional and not estimable from 50 snapshots. Lower is better. Mean across initial conditions; **bold** is the best model and every model tied with it under the paired test (Section 5.3). Across-IC spread: Appendix E.7.

Model	Ra 1.1e6				Ra 5e6				Ra 2e7			
	State L2	W1(p)	LSD(E)	LSD(p)	State L2	W1(p)	LSD(E)	LSD(p)	State L2	W1(p)	LSD(E)	LSD(p)
FNO (BPTT-4)	**0.0663**	0.2858	0.1781	2.0752	**0.0557**	0.6374	**0.1780**	1.1761	**0.1590**	**2.53**	0.2311	0.6618
FNO (1-step)	**0.0660**	**0.1902**	**0.1836**	1.8967	**0.0548**	**0.5515**	**0.1706**	0.9615	**0.1554**	**2.61**	**0.2068**	0.5220
U-Net	**0.0749**	0.4450	**0.1646**	1.7901	0.0704	0.8495	0.1853	0.7583	**0.1741**	**2.79**	**0.1979**	0.3895
Vanilla CFM	0.0760	1.0897	**0.1681**	1.9245	0.0691	1.4078	0.1871	0.6364	0.1719	3.24	**0.1982**	0.1905
PDE-Refiner (2e-3)	0.0984	0.5934	0.2320	**0.5122**	0.0772	1.3399	0.2010	**0.1535**	0.1891	4.22	0.2260	0.1596
PDE-Refiner (3.2e-2)	**0.0801**	0.5845	0.1910	1.0098	0.0740	1.2027	0.1966	0.2069	**0.1713**	3.18	**0.2019**	**0.1457**
Koop-Res	**0.0710**	0.4513	**0.1625**	1.7367	0.0598	0.7787	**0.1699**	0.6763	**0.1555**	**2.71**	**0.1929**	0.2744
KoopFM-vec	0.1064	0.7946	0.2162	1.3210	0.0947	1.2713	0.2310	0.3425	**0.1739**	3.40	**0.2098**	0.1705
KoopFM-field	**0.0658**	0.3346	**0.1535**	1.1099	0.0667	0.7931	**0.1846**	0.2770	0.1808	2.95	0.2109	0.1713

Table 21: **Ra-resolved, $\mathrm{RBC}_{\text{inc-dec}}$, teacher-forced one-step.** 50-step windows from three Ra, $\mathrm{RBC}_{\text{inc-dec}}$, teacher-forced one-step. W1(Nu)/W1(Re) are omitted: they are distributional and not estimable from 50 snapshots. Lower is better. Mean across initial conditions; **bold** is the best model and every model tied with it under the paired test (Section 5.3). Across-IC spread: Appendix E.7.

Model	Ra 1.1e6				Ra 5e6				Ra 2e7			
	State L2	W1(p)	LSD(E)	LSD(p)	State L2	W1(p)	LSD(E)	LSD(p)	State L2	W1(p)	LSD(E)	LSD(p)
FNO (BPTT-4)	0.0191	0.2388	0.0895	2.0807	0.0209	0.3890	0.0955	1.2011	0.0363	1.4190	0.1500	0.6955
FNO (1-step)	0.0134	0.1291	0.0689	1.8601	0.0153	0.2723	0.0680	0.9923	0.0270	0.9831	0.0883	0.5288
U-Net	**0.0083**	**0.0912**	0.0317	1.4049	**0.0096**	**0.2215**	0.0339	0.6074	**0.0164**	**0.6694**	**0.0432**	0.2689
Vanilla CFM	0.0221	0.8292	0.0627	1.9233	0.0119	0.8715	**0.0255**	0.6269	0.0191	1.1707	0.0502	0.1417
PDE-Refiner (2e-3)	0.0166	0.3441	0.0244	**0.4625**	0.0161	0.6347	0.0508	**0.0620**	0.0257	1.4379	0.0893	0.0698
PDE-Refiner (3.2e-2)	0.0129	0.4798	**0.0157**	0.9669	0.0134	0.5798	0.0378	0.1552	0.0226	1.0853	0.0715	**0.0590**
Koop-Res	0.0147	0.3236	0.0389	1.7030	0.0109	0.4248	0.0339	0.6849	**0.0163**	0.7955	**0.0439**	0.2533
KoopFM-vec	0.0220	0.5456	0.0349	1.2889	0.0154	0.6410	0.0386	0.3292	0.0255	1.2839	0.0643	0.1222
KoopFM-field	0.0138	0.2398	0.0221	1.1054	0.0136	0.3618	0.0361	0.2620	0.0260	1.0450	0.0652	0.1201

Table 22: **The anchor's advantage has a measurable range, and it expires metric by metric.** Head-to-head ratio, KoopFM-field over Vanilla CFM, both trained without $\mathrm{RBC}_{\text{inc-high}}$; below 1 the anchored model is ahead. One-sided exact paired sign-flip over $n = 10$ shared initial conditions, with p the p-value for "KoopFM is worse": **bold** significantly better ($p \geq 0.95$), † significantly worse ($p \leq 0.05$), plain a tie. *Crossover* is where the ratio reaches 1 by linear interpolation between rungs; rows are ordered by it. Signed metrics are compared as |bias|. ‡Below $1.20\times$ both models hold $|Re\ \text{bias}| < 1.1\%$, so the ratio there is a quotient of two negligible numbers and carries no information; the Re crossover is read from the monotone regime above it.

	Ra/Ra_{ceil}					
Metric	1.00	1.10	1.20	1.30	1.40	Crossover
Nu bias	**0.19**	**0.54**	1.65†	3.72†	5.31†	1.14
LSD(E)	0.94	0.94	0.99	1.13†	1.17†	1.20
LSD(p)	**0.90**	**0.85**	**0.91**	1.02	1.13†	1.28
Re bias	0.67	1.50	**0.55**	**0.95**	1.17†	1.32‡
$W_1(p)$	**0.39**	**0.48**	**0.66**	**0.80**	**0.92**	> 1.40
State L_2	1.04	1.13†	1.07†	1.00	**0.94**	never

Table 23: **Whole-trajectory extrapolation.** Full autoregressive test trajectories ($T = 4790$, 25 NFE) of the high-Ra holdout models, which never saw $Ra > 1.8 \times 10^7$; $\mathrm{RBC}_{\text{inc-dec}}$ peaks at $1.11\times$ and $\mathrm{RBC}_{\text{inc}}$ ramps to $1.94\times$ this ceiling. *Left:* holdout Vanilla CFM against holdout KoopFM, and their ratio, with **bold** significantly better and † significantly worse for KoopFM (one-sided exact paired sign-flip, $n = 10$ and 9). *Right:* each holdout and low-data control as a ratio to the same model's full-data run; * marks significantly worse than that run. Degradation that the control reproduces is a data-volume effect, and only the remainder is attributable to the range.

	holdout, absolute			ratio to own full-data run			
Metric	CFM	KoopFM	ratio	CFM hold.	CFM ctrl.	KoopFM hold.	KoopFM ctrl.
$\mathrm{RBC}_{\text{inc-dec}}$ (peak $1.11\times$ the ceiling)							
$W_1(Nu)$	0.387	0.252	**0.65**	2.15*	3.24*	0.92	0.88
$W_1(Re)$	8.606	4.817	**0.56**	2.59*	3.24*	1.54*	2.40*
$W_1(p)$	4.324	2.332	**0.54**	1.73*	1.21*	1.13*	1.05*
LSD(E)	0.299	0.304	1.02†	0.99	0.99	1.03*	1.04*
LSD(p)	0.548	0.411	**0.75**	1.06*	1.08*	1.24*	1.37*
$\mathrm{RBC}_{\text{inc}}$ (ramp to $1.94\times$ the ceiling)							
$W_1(Nu)$	0.337	0.874	2.59†	1.37*	2.50*	3.00*	1.19
$W_1(Re)$	12.60	14.43	1.15†	2.60*	2.14*	2.67*	1.24*
$W_1(p)$	10.22	9.389	**0.92**	3.79*	1.28*	4.41*	1.05*
LSD(E)	0.333	0.333	1.00	1.01	0.98	1.04*	1.07*
LSD(p)	0.729	0.546	**0.75**	1.06*	1.07*	1.34*	1.44*