EchoKoop: KOOPMAN FORECASTING WITH LEARNED LIFTING AND FROZEN MULTISCALE DYNAMICS

Anonymous authors
Paper under double-blind review

ABSTRACT

Neural Koopman forecasters typically learn both nonlinear observables and a finite-dimensional latent transition operator. While expressive, the learned operator can be parameter-intensive and difficult to stabilize under repeated long-horizon rollout. We introduce EchoKoop, a forecasting framework based on the principle of *learning the lift while freezing the flow*. Instead of optimizing Koopman transition matrices, EchoKoop learns nonlinear observables that are propagated by fixed multiscale normal operators. These operators generate flows with different memory ranges, while the learned encoder, modal projections, selectors, and decoders adapt observed histories to this dynamical basis. Removing trainable transition matrices eliminates their parameter cost and spectral drift during training. Across ten long-term forecasting benchmarks, EchoKoop attains the lowest dataset-averaged MSE on six; controlled ablations show no consistent gain from training its transitions.

1 INTRODUCTION

Long-horizon time-series forecasting requires models that can represent nonlinear temporal structure while controlling error propagation. Koopman operator theory offers a useful perspective: nonlinear dynamics can be represented as linear evolution in a lifted space of observables (Koopman, 1931; Mezić, 2005; Rowley et al., 2009; Williams et al., 2015; Brunton et al., 2022). Neural Koopman models typically realize this idea using a nonlinear lifting map, a finite-dimensional linear transition, and a decoder (Li et al., 2017; Takeishi et al., 2017; Lusch et al., 2018; Otto & Rowley, 2019). At the finite-dimensional level, the key question is how accurately a linear transition approximates the lifted dynamics: $\Phi(\mathcal{F}(\mathbf{x})) = \mathbf{K}\Phi(\mathbf{x}) + \mathbf{r}(\mathbf{x})$, where $\mathbf{r}(\mathbf{x})$ is the one-step residual (Brunton et al., 2016). The representation, transition, and residual play distinct roles in long-horizon prediction. Appendix A discusses related work in greater detail, while Appendix B.1 formalizes this distinction.

Most existing methods learn the observables and the transition jointly. These components are tightly coupled: under an invertible change of latent coordinates, the lifting map, transition, and decoder transform accordingly, leaving the resulting predictions unchanged. This coordinate non-uniqueness complicates optimization and interpretation. Long-horizon forecasting introduces a further challenge: error amplification under repeated application of the transition is bounded by $\|\mathbf{K}^h\|_2$. During training, $\mathbf{K}$ can become highly non-normal or ill-conditioned, so that $\|\mathbf{K}^h\|_2$ exhibits transient growth even when the spectral radius of $\mathbf{K}$ remains below one (Trefethen, 1997). Moreover, a dense $d \times d$ transition matrix requires $O(d^2)$ trainable parameters (Gu et al., 2022; Gupta et al., 2022). Existing methods mitigate these problems using norm-constrained, low-rank, or stable parameterizations (Khosravi, 2021; Kostic et al., 2022; Pan & Duraisamy, 2020), but they still optimize the transition that is recursively applied during rollout. This raises a more basic question: *must the latent linear transition be learned as a free parameter, or can it be determined from the learned observables?*

We answer this question with EchoKoop (Figure 1), which follows the principle of *learning the lift while freezing the flow*. To prevent amplification under repeated latent transitions, we use a fixed bank of normal, contractive transition operators, realized through damped rotations with prescribed frequencies and decay half-lives. Learned projections map the observed history into modal coordinates, and fixed phase and decay factors align the resulting initial states with the forecast origin.

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.

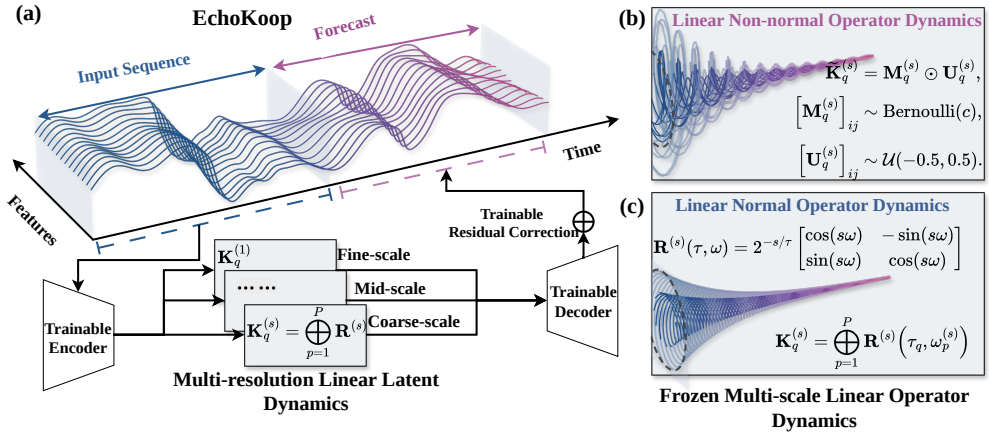

Figure 1: **EchoKoop architecture and frozen multiscale linear dynamics.** **(a)** A trainable encoder maps the input sequence into branch-specific latent spaces at fine, intermediate, and coarse resolutions. Frozen linear operator banks propagate each branch, followed by a trainable decoder and cross-scale correction. Blue and magenta denote the input and forecast intervals. **(b)** Iterating a sparse non-normal operator may transiently amplify and deform trajectories before decay. **(c)** Iterating the normal damped-rotation operators produces smooth rotational flows with prescribed frequencies and monotonic geometric contraction.

A causal encoder first lifts the observed window. At each temporal scale, the aligned initial states evolve under fixed operators, and their trajectories are combined by lightweight selectors, decoders, and a cross-scale correction module. Coarser grids use fewer propagation steps to represent slowly varying components, while the finest grid retains short-scale variation. Model capacity thus shifts from estimating transition matrices to learning predictive coordinates, a requirement also emphasized by information-theoretic Koopman learning (Cheng et al., 2026).

Unlike structured Koopman methods that retain trainable diagonal, low-rank, or frequency-domain transitions (Khosravi, 2021; Kostic et al., 2022; Zhang et al., 2025), EchoKoop removes the trainable latent transition altogether. Unlike input-driven echo-state networks and structured reservoirs (Jaeger, 2001; Lukoševičius & Jaeger, 2009; Tino, 2020; Li et al., 2024), EchoKoop applies autonomous linear rollouts after a learned lifting. The model learns how observations enter and leave the fixed operator bank, while keeping latent propagation prescribed.

This separation has a finite-horizon rationale. When the sampled modal trajectories span the forecast horizon, suitable input-dependent coefficients can represent any sequence over that horizon. Related universality results show that linear recurrences combined with nonlinear projections can support expressive sequence models (Orvieto et al., 2024). For a practical finite dictionary, forecast quality depends on both the approximation capacity of the selected modes and the accuracy of the inferred initial state. In the latent space, the effect of an initial-state error at horizon h is bounded by $\|\mathbf{K}^h\|_2$ times the size of that error. EchoKoop broadens the temporal basis through multiple frequencies, decay scales, and temporal resolutions, while its normal, contractive transitions limit this propagation by construction.

Our contributions are:

- We formulate the principle of *learning the lift while freezing the flow* and provide a finite-horizon reparameterization result together with an error bound that separates approximation error from rollout amplification.
- We construct a multiscale forecasting basis from frozen normal operators, realized by damped-rotation blocks with prescribed frequency and decay scales, and initialize their states by aligning learned historical features with the forecast origin.
- Through forecasting benchmarks, controlled ablations, and cross-architecture interventions, we show that learning the latent transition provides no consistent advantage in the studied settings, while multiresolution rollout and phase-consistent initialization have larger effects.

2 Background and Positioning

2.1 Neural Koopman Forecasting

Consider a discrete-time dynamical system

$$\mathbf{x}_{t+1} = \mathcal{F}(\mathbf{x}_t),$$

where $\mathbf{x}_t \in \mathcal{X}$ denotes a state that may include an observation history. The Koopman operator describes this nonlinear evolution through its action on an observable $g : \mathcal{X} \to \mathbb{R}$:

$$(\mathcal{K}g)(\mathbf{x}_t) = g(\mathcal{F}(\mathbf{x}_t)) = g(\mathbf{x}_{t+1}).$$

Although $\mathcal{F}$ may be nonlinear, $\mathcal{K}$ is linear but generally infinite-dimensional (Koopman, 1931; Brunton et al., 2022).

Neural Koopman models approximate this evolution in a finite-dimensional learned coordinate system. An encoder $\mathcal{E}_\theta : \mathcal{X} \to \mathbb{R}^d$ constructs latent observables, a matrix $\mathbf{K} \in \mathbb{R}^{d\times d}$ propagates them, and a decoder $\mathcal{D}_\theta$ maps the evolved coordinates back to the observation space:

$$\mathbf{z}_t = \mathcal{E}_\theta(\mathbf{x}_t), \qquad \widehat{\mathbf{x}}_{t+h} = \mathcal{D}_\theta(\mathbf{K}^h \mathbf{z}_t). \tag{1}$$

This encoder–transition–decoder construction is widely used in deep Koopman learning and forecasting (Takeishi et al., 2017; Lusch et al., 2018; Otto & Rowley, 2019). We use θ for the collection of trainable parameters, with each module using its own subset.

The coordinates and transition are not independently identifiable. For any invertible $\mathbf{S}$, the transformations

$$\widetilde{\mathcal{E}}_\theta = \mathbf{S}\mathcal{E}_\theta, \quad \widetilde{\mathbf{K}} = \mathbf{S}\mathbf{K}\mathbf{S}^{-1}, \quad \widetilde{\mathcal{D}}_\theta(\mathbf{z}) = \mathcal{D}_\theta(\mathbf{S}^{-1}\mathbf{z})$$

leave Equation (1) unchanged. Forecasting accuracy alone therefore does not identify a unique latent transition.

Recursive propagation also makes the geometry of $\mathbf{K}$ consequential. If the initial latent state contains an error $\boldsymbol{\epsilon}_0$, then its effect after h steps is

$$\Delta\mathbf{z}_h = \mathbf{K}^h\boldsymbol{\epsilon}_0, \qquad \|\Delta\mathbf{z}_h\|_2 \le \|\mathbf{K}^h\|_2\|\boldsymbol{\epsilon}_0\|_2.$$

Thus, finite-horizon sensitivity depends on the matrix powers $\|\mathbf{K}^h\|_2$, not only on the spectral radius $\rho(\mathbf{K})$. In particular, a non-normal matrix may satisfy $\rho(\mathbf{K}) < 1$ while exhibiting substantial transient amplification (Yildiz et al., 2012). Learning the transition can therefore alter both its asymptotic spectrum and its finite-horizon amplification during training.

2.2 Learned and Prescribed Operators

Most neural Koopman models learn or estimate the latent transition alongside the representation. Koopman autoencoders fit both components (Lusch et al., 2018; Otto & Rowley, 2019); forecasting models also use sample-dependent or structured operators (Wang et al., 2023; Liu et al., 2023; Zhang et al., 2025). Constraints on these operators can improve stability, but leave the recursively applied map trainable or data-estimated (Khosravi, 2021; Kostic et al., 2022; Pan & Duraisamy, 2020; Fan et al., 2026).

The proposed EchoKoop instead prescribes normal, contractive transitions and learns how the observed history enters and is read out from their flows. Appendix A.1 places this choice alongside other Koopman forecasters.

3 EchoKoop: Learn the Lift, Freeze the Flow

3.1 Overview

Given a historical window $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_L]^\top \in \mathbb{R}^{L\times C}$, we aim to predict $\mathbf{Y} = [\mathbf{y}_1, \dots, \mathbf{y}_H]^\top = [\mathbf{x}_{L+1}, \dots, \mathbf{x}_{L+H}]^\top \in \mathbb{R}^{H\times C}$. EchoKoop separates representation learning from latent propagation: a nonlinear encoder learns coordinates adapted to the data, whereas a prescribed family of stable linear flows advances these coordinates through time. The resulting architecture consists of

three components: (i) a learned multiscale lifting network, (ii) flows generated by frozen normal operators, and (iii) a learned decoder with a lightweight cross-scale correction module.

The fixed operators provide a controlled temporal basis; learned features and prescribed phase alignment determine how each observed history enters it. Below, we instantiate the operators as damped rotations with explicit frequency and decay scales.

3.2 Learned Multiscale Lifting

We first remove the last observed value,

$$\widetilde{\mathbf{x}}_t = \mathbf{x}_t - \mathbf{x}_L, \qquad t = 1, \dots, L, \tag{2}$$

and restore it after forecasting. A causal temporal encoder $\mathcal{E}_\theta$ maps the centered history to nonlinear features. The encoded sequence is then represented at a set of temporal resolutions $\mathcal{S}$. For scale $s \in \mathcal{S}$, non-overlapping groups of s time steps are aggregated into $N_s^{\text{in}} = \lfloor L/s \rfloor$ cells,

$$\mathbf{g}_j^{(s)} = \frac{1}{\sqrt{s}} \sum_{r=0}^{s-1} \mathcal{E}_\theta \left(\widetilde{\mathbf{X}} \right)_{js+r+1}, \qquad j = 0, \dots, N_s^{\text{in}} - 1, \tag{3}$$

where the normalization keeps the feature magnitude comparable across resolutions. Equation (3) uses $\mathcal{E}_\theta$ as shorthand for the effective lifted feature. In the implementation, the centered raw observation is concatenated with normalized convolutional features before aggregation; the complete construction is given in Appendix C.1. Fine resolutions preserve rapid variations, whereas coarse resolutions provide longer effective receptive fields with fewer rollout steps.

Each scale projects $\mathbf{g}_j^{(s)}$ into P two-dimensional latent pairs, so the modal width is $d_m = 2P$. Instead of initializing these pairs only from the final encoded token, EchoKoop aligns all historical cells with the forecast origin according to their ages and phases. The resulting initial states are denoted by $\mathbf{z}_{0,q,p}^{(s)} \in \mathbb{R}^2$, where $p = 1, \dots, P$ indexes frequency pairs and $q = 1, \dots, Q_a$ indexes candidate half-lives. The exact construction is provided in Appendix C.3.

3.3 Frozen Multiscale Flows

Here, an *operator* refers to an individual transition matrix, an *operator bank* to the collection indexed by half-life, and a *flow* to the temporal evolution generated by repeated application of these operators. Section 4.3 reuses the normal-operator construction in other host architectures under the name *Frozen Dynamics*.

At resolution s, EchoKoop uses a bank of Q_a fixed transition operators,

$$\mathbf{K}_q^{(s)} = \bigoplus_{p=1}^{P} \mathbf{R}^{(s)} \left(\tau_q, \omega_p^{(s)} \right), \qquad q = 1, \dots, Q_a, \tag{4}$$

where τ_q is a prescribed half-life and $\omega_p^{(s)}$ is the angular frequency assigned to the p-th latent pair. To obtain normal, contractive transitions with explicit oscillation and decay scales, we use damped-rotation blocks:

$$\mathbf{R}^{(s)}(\tau, \omega) = 2^{-s/\tau} \begin{bmatrix} \cos(s\omega) & -\sin(s\omega) \\ \sin(s\omega) & \cos(s\omega) \end{bmatrix}. \tag{5}$$

Thus, each latent pair has an explicit oscillation frequency and memory scale. Because these blocks are normal and have spectral radius $2^{-s/\tau} \leq 1$, their powers satisfy

$$\left\| \left(\mathbf{R}^{(s)}(\tau, \omega) \right)^n \right\|_2 = 2^{-ns/\tau},$$

preventing transient amplification inside the prescribed flow. Before selection, the Q_a bank members at scale s evolve in parallel. Their joint block-diagonal transition has n-step norm $\max_q 2^{-ns/\tau_q} \leq 1$. Thus, initialization error is attenuated, while one-step mismatch with a reference latent trajectory accumulates with decaying weights (Appendix B.3). This statement concerns the latent rollout before learned readout, not the complete predictor.

For each scale and modal pair, EchoKoop learns global selector weights $\alpha_{p,q}^{(s)}$ over the half-life dictionary. Their logits are trainable parameters, initialized either uniformly or from a specified positive profile, and shared across samples and forecast steps. Rather than estimating a new transition matrix, the model combines trajectories generated by the fixed operators:

$$\mathbf{z}_{n,p}^{(s)} = \sum_{q=1}^{Q_a} \alpha_{p,q}^{(s)} \left[\mathbf{R}^{(s)}\left(\tau_q, \omega_p^{(s)}\right)\right]^n \mathbf{z}_{0,q,p}^{(s)}, \qquad \sum_{q=1}^{Q_a} \alpha_{p,q}^{(s)} = 1. \tag{6}$$

The selector changes how fixed-bank trajectories are combined without changing their transition matrices; sample dependence enters through the history-dependent initial states. On a single native grid, a modal matrix with full row rank can represent any sequence over that grid's finite horizon through freely chosen coefficients and a linear readout, as established by Theorem 1 in Appendix B.2. This idealized result does not guarantee exact fit by the configured multiscale model.

Each branch performs $N_s^{\text{out}} = \lceil H/s \rceil$ native transitions and interpolates the resulting trajectory from native nodes $\{0, s, \ldots, N_s^{\text{out}} s\}$ to the original grid $\{0, 1, \ldots, H\}$. After fixed orthogonal mixing, let $\mathbf{Z}_{1:H}^{(s)} \in \mathbb{R}^{H \times 2P}$ denote the interpolated future latent trajectory. A branch-specific decoder $\mathcal{D}_\theta^{(s)}$ maps it back to the observation space. The base forecast is

$$\widehat{\mathbf{Y}}_{\text{base}} = \mathbf{1}_H \mathbf{x}_L^\top + \sum_{s \in \mathcal{S}} \mathcal{D}_\theta^{(s)}\left(\mathbf{Z}_{1:H}^{(s)}\right).$$

Finally, a lightweight attention module exchanges information across scales and produces a residual correction:

$$\widehat{\mathbf{Y}} = \widehat{\mathbf{Y}}_{\text{base}} + \mathcal{C}_\theta\left(\{\mathbf{Z}_{1:H}^{(s)}\}_{s \in \mathcal{S}}\right).$$

Its output projection is initialized to zero, so training begins from the additive multiscale forecast. All encoders, selectors, decoders, and the correction module are learned end-to-end, while every $\mathbf{K}_q^{(s)}$ remains frozen.

4 Experiments

Our experiments address four questions:

- **RQ1: Forecasting accuracy.** Can frozen latent flows remain competitive with learned-transition forecasters?
- **RQ2: Mechanism.** How do transition trainability, multiscale structure, and history initialization affect performance?
- **RQ3: Transferability.** Can the same frozen dynamical prior improve other Koopman-style forecasting architectures?
- **RQ4: Beyond standard forecasting benchmarks.** Does the approach extend to low-dimensional dynamical systems and high-dimensional physical fields?

Experimental setup. We evaluate EchoKoop on ten multivariate benchmarks covering four ETT datasets, Weather, Exchange, and four PEMS datasets, using chronological splits and four prediction horizons per dataset family. We compare against four general forecasters (PatchTST, iTransformer, TimesNet, and DLinear) (Nie et al., 2023; Liu et al., 2024; Wu et al., 2023; Zeng et al., 2023) and six Koopman-based methods (Koopa Liu et al. (2023), KUMA Xiong et al. (2026), K^2VAE Wu et al. (2025), SKOLR Zhang et al. (2025), FlowMixer Mehouachi & Jabari (2025), and AIKAE Frion et al. (2025)). MSE and MAE are computed in standardized space. Across methods, we use the same chronological partitions, training-set normalization, sliding-window targets, and metric calculation. For PEMS, the look-back length is $L = 12$ and the horizons are $H \in \{12, 24, 48, 96\}$. EchoKoop and all controlled comparisons are evaluated over three random seeds. Unless otherwise stated, EchoKoop uses three native resolutions $\mathcal{S} = \{1, 2, 4\}$ and $Q_a = 6$ frozen normal operators per resolution. Dataset splits, look-back lengths, horizons, flow dictionaries, model widths, and optimization settings are provided in Appendix D; complete per-horizon results are in Appendix E.

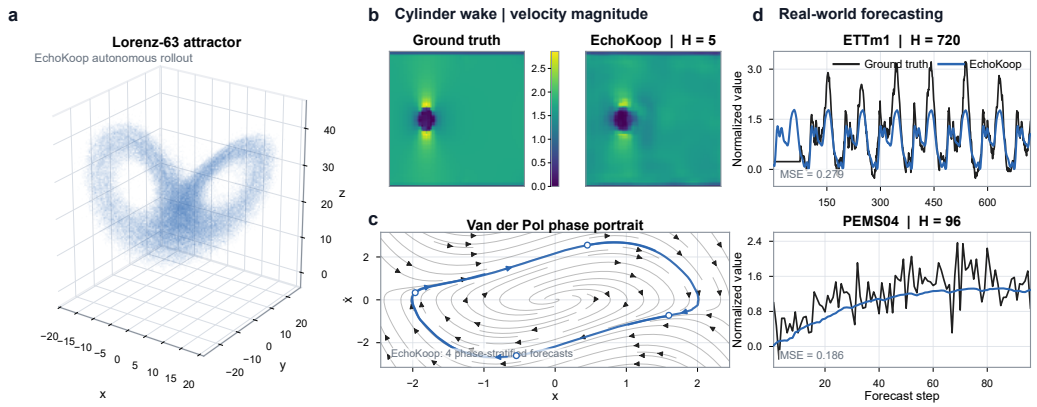

Figure 2: Qualitative visualization of the dynamics recovered and predicted by EchoKoop. (a) Autonomous rollout of the Lorenz-63 attractor. (b) Ground-truth and predicted cylinder-wake velocity fields at horizon $H = 5$. (c) Van der Pol phase portrait, with the analytical phase flow shown in black and EchoKoop trajectories shown in blue. (d) Representative forecasting results on ETTm1 and PEMS04.

Table 1: Dataset-level mean errors across four horizons (lower is better). The best baseline is selected by mean MSE among non-EchoKoop methods with complete results; its MAE is reported for the same method. Within each metric, Δ is baseline minus EchoKoop, so positive values favor EchoKoop. The lower paired value is bold. Complete results appear in Appendix E.

		MSE ↓			MAE ↓		
Dataset	Best baseline	EchoKoop	Baseline	Δ	EchoKoop	Baseline	Δ
ETTh1	SKOLR	**0.439**	0.449	+0.010	0.448	**0.446**	−0.002
ETTh2	SKOLR	0.391	**0.382**	−0.009	0.417	**0.407**	−0.010
ETTm1	PatchTST	**0.375**	0.389	+0.014	**0.394**	0.403	+0.009
ETTm2	Koopa	0.297	**0.282**	−0.015	0.342	**0.327**	−0.015
Weather	KUMA	0.271	**0.245**	−0.026	0.298	**0.274**	−0.024
Exchange	DLinear	0.118	**0.098**	−0.020	0.240	**0.220**	−0.020
PEMS03	KUMA	**0.135**	0.257	+0.122	**0.237**	0.299	+0.062
PEMS04	KUMA	**0.127**	0.190	+0.063	**0.237**	0.285	+0.048
PEMS07	KUMA	**0.111**	0.219	+0.108	**0.212**	0.262	+0.050
PEMS08	KUMA	**0.194**	0.275	+0.081	**0.272**	0.320	+0.048

4.1 Can Frozen Flows Remain Competitive?

EchoKoop obtains the lowest dataset-averaged MSE on six of the ten benchmarks: ETTh1, ETTm1, PEMS03, PEMS04, PEMS07, and PEMS08 (Table 1), with representative ETTm1 and PEMS04 forecasts shown in Figure 2(d). The largest margins occur on PEMS, mainly at longer horizons; at $H = 12$, EchoKoop and the strongest baselines are closely matched (Appendix E).

The pattern is not uniform: baselines remain better on ETTh2, ETTm2, Weather, and Exchange. The gap is small on ETTh2, grows with the forecast horizon on ETTm2 and Exchange, and persists across horizons on Weather. These differences may reflect how well the prescribed latent timescales and learned representations fit each dataset; On Weather and Exchange, making the transitions trainable does not close the gaps to the strongest baselines (Table 13). ETTh2 and ETTm2 were not included in these ablations. The available evidence therefore does not identify frozen transitions as the cause of the weaker results. Among the eight methods with complete results, Friedman tests find differences in dataset-level ranks for MSE and MAE ($p = 0.033$ and $p = 0.030$), with EchoKoop ranking second on both. No EchoKoop–baseline comparison survives Holm correction. The results therefore support competitive forecasting with frozen latent flows, but not pairwise superiority across datasets (Appendix H). Complete horizon-wise results are reported in Appendix E.

4.2 What Matters: Trainability or Structure?

We next assess transition trainability, multiresolution rollout, phase-aligned initialization, and cross-scale correction. The ablations are macro-averaged over the prespecified core ablation suite ETTh1, ETTm1, Exchange, Weather, PEMS04, and PEMS08, covering four horizons per dataset and three random seeds. This representative suite was used for all controlled ablations and is a subset of the ten benchmarks summarized in Table 1.

The frozen normal-operator reference obtains an average MSE/MAE of $0.2535/0.3142$. Figure 3 shows that retaining only the fine and middle resolutions raises MSE by 2.09%, and using only the fine resolution raises it by 8.90%. Removing phase-aware history alignment raises MSE by 5.39%; omitting cross-scale correction has a smaller effect (1.87%).

Making all six initially normal operators trainable changes average MSE from 0.2535 to 0.2532, an improvement of only 0.11%. The frozen and trainable variants each win 12 of the 24 matched dataset–horizon settings, and their macro-average difference is smaller than the observed seed variation. Training transition entries thus gives no consistent benefit under this architecture and protocol.

EchoKoop-R replaces the normal bank with six sparse random operators of matched spectral radii and learns temporal weights for their initial states. Its six-dataset average MSE is 6.30% higher than EchoKoop's. On three datasets, keeping EchoKoop's phase-based history construction with the same sparse operators yields still higher error (Table 14). The learned initializer helps the random bank, but the prescribed normal bank remains more accurate in this comparison. At the dataset level, none of the twelve ablation contrasts survives Holm correction. Exploratory tests across dataset–horizon settings support the multiscale and phase-alignment effects and the EchoKoop-R gap, although horizons from the same dataset are dependent (Appendix H). Complete dataset-wise ablations are given in Appendix G.

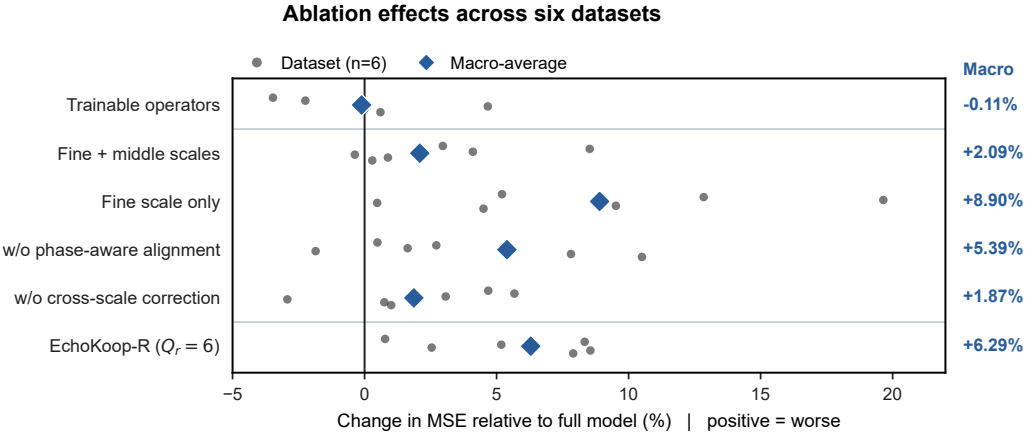

Figure 3: Six-dataset ablation effects relative to EchoKoop with frozen normal $Q_a = 6$ operators. Black points show dataset-level MSE changes after averaging four horizons and three seeds. Blue diamonds and right-hand labels show the change in macro-average MSE over 24 dataset–horizon settings, computed from unrounded values rather than by averaging the six percentages. Positive values indicate worse accuracy. EchoKoop-R uses learned temporal weights in place of phase-aligned initialization.

In the matched trainable-operator ablation, each initially normal operator is released as a full $d_m \times d_m$ matrix. Freezing therefore removes $N_{\mathrm{dyn}} = |\mathcal{S}|Q_a d_m^2$ trainable transition entries, where $|\mathcal{S}|$ is the number of temporal resolutions and Q_a is the number of operators per resolution. With three resolutions and six operators per resolution, this corresponds to 165,888 entries for $d_m = 96$ and 294,912 for $d_m = 128$. Fixed operators still require buffer storage and rollout computation, however, and our current implementation executes them densely. We therefore claim parameter-efficient dynamics modeling, not a universal wall-clock speedup.

Table 2: Cross-architecture Frozen Dynamics transfer. Δ_{MSE} is the six-dataset mean relative change from each native host after averaging MSE over three seeds (negative is better). "Improved" counts datasets with lower mean MSE (six per host; 18 overall). The $Q = 1$ and $Q = 6$ variants are evaluated independently.

		Koopa		SKOLR		K2VAE		Overall
Setting	Horizon	Δ_{MSE} (%)	Improved	Δ_{MSE} (%)	Improved	Δ_{MSE} (%)	Improved	Improved
Frozen Dynamics ($Q = 1$)	Short	-2.87	$6/6$	$\mathbf{-0.18}$	$2/6$	$\mathbf{-4.11}$	$4/6$	$\mathbf{12/18}$
	Long	-6.27	$6/6$	-0.35	$5/6$	$+3.10$	$0/6$	$11/18$
Frozen Dynamics ($Q = 6$)	Short	$\mathbf{-2.90}$	$6/6$	$+0.38$	$1/6$	-2.18	$3/6$	$10/18$
	Long	$\mathbf{-6.32}$	$6/6$	$\mathbf{-0.59}$	$6/6$	$+2.03$	$1/6$	$\mathbf{13/18}$

Table 3: Targeted mechanism tests for Frozen Dynamics. Entries are paired relative MSE changes (mean $\pm$ standard deviation; negative favors the first condition). Parentheses count wins: three seeds for the host-specific probes and nine host–seed pairs for frequency alignment.

Probe	Task and contrast	Δ_{MSE} (%; wins)
Decay-scale coverage	SKOLR–Exchange, $H = 48$; $Q = 8$ vs. $Q = 6$	$+0.59 \pm 1.69$ $(2/3)$
Frequency alignment	ETTm1, $H = 96$; aligned vs. shifted	$+0.30 \pm 2.28$ $(4/9)$
	ETTm1, $H = 720$; aligned vs. shifted	-1.15 ± 1.75 $(8/9)$
Low-rank adaptivity	Koopa–Exchange, $H = 192$; rank-4 vs. $Q = 6$	-0.84 ± 6.70 $(2/3)$

4.3 Do Frozen Normal Operators Transfer?

To test whether the result is specific to EchoKoop, we replace the native recurrent operators of Koopa, SKOLR, and K^2VAE with prescribed normal operators. We call this cross-architecture intervention *Frozen Dynamics*. We consider a single-operator variant ($Q = 1$) and a six-operator variant ($Q = 6$) with learned per-pair selection. Each variant is evaluated independently on six datasets, two forecasting regimes, and three random seeds, while the remaining host architecture and training protocol are held fixed.

Table 2 reports the cross-architecture transfer results. Frozen Dynamics improves 46 of the 72 host–dataset–horizon comparisons. Koopa shows the most consistent gains: both $Q = 1$ and $Q = 6$ improve all six datasets in both forecasting regimes, accounting for 24 of these improvements. Each variant reduces Koopa's six-dataset mean relative MSE by approximately 2.9% at short horizons and 6.3% at long horizons. Notably, the $Q = 1$ variant achieves its 12 improvements without adding trainable parameters (Table 10). SKOLR shows smaller long-horizon gains, whereas the results for K^2VAE are mixed and its mean long-horizon MSE increases after replacement.

Rollout diagnostics help explain Koopa's consistent gains. Its native sample-fitted DMD transitions have a median relative power gain of 7.93 (95th percentile 82.52), compared with unity for the frozen normal operators. Replacing the native map therefore removes a substantial source of transient amplification. The mixed K^2VAE results show that this constraint alone does not ensure better forecasts; its effect also depends on how the host constructs and uses its latent coordinates.

To examine where the fixed bank can be too restrictive, we test shorter decay scales, frequency alignment, and a low-rank correction on selected host–task pairs (Table 3).

Adding shorter decay scales does not improve mean MSE on SKOLR–Exchange, whereas frequency alignment helps at the longer ETTm1 horizon but not at the shorter one. On Koopa–Exchange, the rank-4 correction gives a small average gain with substantial seed variation, while raising mean relative power gain from 1.00 to 2.40 and mean spectral radius from 0.9993 to 1.072. These probes illustrate that a stable rollout is only part of the forecasting problem: the available modes must also suit the task. Full protocols and diagnostics appear in Appendix F.4.

4.4 Beyond Standard Benchmarks

We further evaluate frozen structured flows on three low-dimensional dynamical systems (Lorenz–63, Van der Pol, and the double pendulum) and three high-dimensional physical fields (cylinder

wake, dam flow, and ERA5). Representative forecasts appear in Figure 2. All learned-model results are averaged over three random seeds; protocols and full results are given in Appendices I and J.

Dynamical systems. Table 17 reports pointwise forecast accuracy. EchoKoop achieves the lowest mean NRMSE in eight of nine system–horizon settings. In the 1,000-step autonomous rollouts reported in Table 18, it leads on the Lorenz attractor-occupancy and variability measures, Lorenz valid prediction time, and double-pendulum energy drift. The exceptions include Van der Pol frequency error and valid rollout, where FlowMixer-C12 leads, and double-pendulum valid prediction time, where the VAE leads.

Figures 9 and 10 compare Lorenz–63 marginal, projected, and joint distributions under conditional prediction. Overall, the quantitative results support accurate forecasting and selected long-run statistics, but do not establish recovery of every invariant or the ground-truth Koopman operator.

Physical fields. Under the endpoint forecasting protocol, ConvLSTM (SHI et al., 2015) achieves the best dataset-averaged NRMSE and SSIM on all three fields (Table 19). EchoKoop performs best overall among the Koopman-oriented models: the multi-frame variant is stronger on Cylinder and Dam, whereas EchoKoop-1f is slightly stronger on ERA5. The shared-framework KAE-Info adaptation of InformationKoopman (Cheng et al., 2026) trails EchoKoop-1f on all three fields, with substantial across-seed variation on Cylinder and ERA5 (Table 19). Because the CFD training batch sizes also differ, this comparison does not isolate the effect of input-history length. The ERA5 examples in Figure 4 illustrate predictive similarity; NRMSE and SSIM alone do not establish physical consistency.

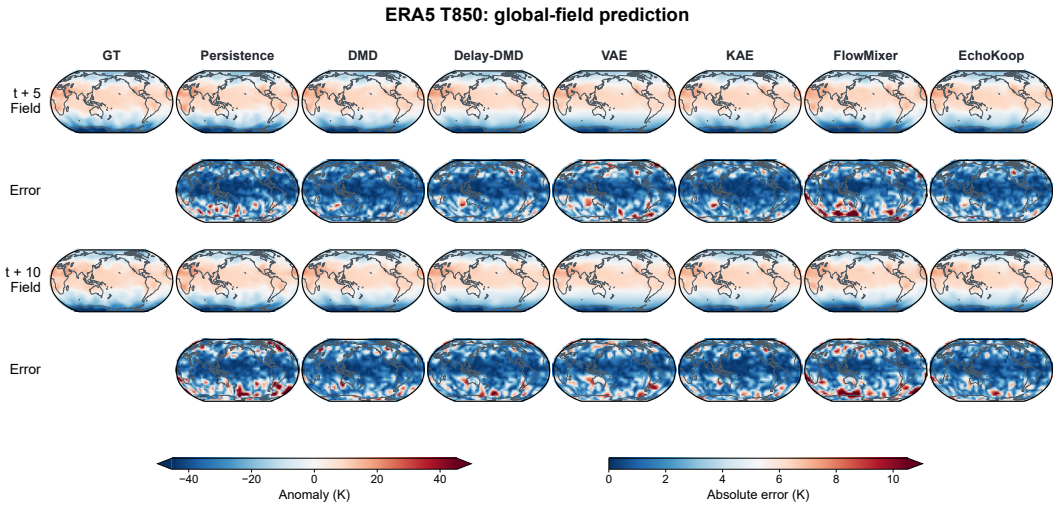

Figure 4: Long-horizon ERA5 T850 forecasting. Columns show ground truth (GT), Persistence, DMD, Delay-DMD, VAE, KAE and EchoKoop at prediction horizons $t + 5$ and $t + 10$. The field rows visualize T850 anomalies, while the lower rows show absolute physical-space errors under a common scale.

5 Conclusion

EchoKoop learns latent representations and readouts around frozen multiscale normal operators. It achieves the lowest dataset-averaged MSE on six of ten forecasting benchmarks without training transition matrices, although the dataset-level tests do not establish pairwise superiority over individual baselines. In matched ablations, training the transitions brings no consistent gain, while multiscale rollout and phase-aligned initialization have larger effects. Replacing operators in other architectures consistently improves Koopa but yields mixed results elsewhere. Together, these findings support frozen normal operators as a useful way to control latent rollout amplification, while showing that forecast accuracy still depends on the learned representation and the suitability of the prescribed modes.

AI USE STATEMENT

We used generative AI tools, including OpenAI Codex, to assist with theoretical framing and exposition, experimental design and diagnostics, code review and debugging, interpretation of recorded results, figure preparation, literature organization, and manuscript editing. The authors made the final research and methodological decisions, reviewed AI-assisted material against the relevant derivations, source code, and experimental records, and take responsibility for all claims, results, and artifacts in this paper.

REFERENCES

Omri Azencot, N. Benjamin Erichson, Vanessa Lin, and Michael Mahoney. Forecasting sequential data using consistent Koopman autoencoders. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 475–485. PMLR, 2020.

Hanru Bai and Weiyang Ding. KoNODE: Koopman-driven neural ordinary differential equations with evolving parameters for time series analysis. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 2484–2519. PMLR, 13–19 Jul 2025.

Steven L Brunton, Bingni W Brunton, Joshua L Proctor, and J Nathan Kutz. Koopman invariant subspaces and finite linear representations of nonlinear dynamical systems for control. *PLOS ONE*, 11(2):e0150171, 2016.

Steven L. Brunton, Marko Budišić, Eurika Kaiser, and J. Nathan Kutz. Modern Koopman theory for dynamical systems. *SIAM Review*, 64(2):229–340, 2022. doi: 10.1137/21M1401243.

Xiaoyuan Cheng, Wenxuan Yuan, Yiming Yang, Yuanzhao Zhang, Sibo Cheng, Yi He, and Zhuo Sun. Information shapes Koopman representation. In *The Fourteenth International Conference on Learning Representations*, 2026.

Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

Fletcher Fan, Bowen Yi, David Rye, Guodong Shi, and Ian R. Manchester. Learning stable Koopman embeddings for identification and control. *Automatica*, 189:113003, 2026. ISSN 0005-1098. doi: 10.1016/j.automatica.2026.113003.

Ali Forootani, Mohammad Khosravi, and Masoud Barati. DeepKoopFormer: A Koopman enhanced transformer based architecture for time series forecasting. *npj Artificial Intelligence*, 2:35, 2026. doi: 10.1038/s44387-026-00085-3.

Anthony Frion, Lucas Drumetz, Mauro Dalla Mura, Guillaume Tochon, and Abdeldjalil Aïssa El Bey. Augmented invertible Koopman autoencoder for long-term time series forecasting. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856.

Ben Gao, Jordan Patracone, Stephane Chretien, and Olivier Alata. Conformal online learning of deep Koopman linear embeddings. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 83470–83495. Curran Associates, Inc., 2025. doi: 10.52202/085713-2799.

Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. In *International Conference on Learning Representations*, 2022.

Ankit Gupta, Albert Gu, and Jonathan Berant. Diagonal state spaces are as effective as structured state spaces. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 22982–22994. Curran Associates, Inc., 2022. doi: 10.52202/068431-1670.

Hans Hersbach, Bill Bell, Paul Berrisford, Shoji Hirahara, András Horányi, Joaquín Muñoz-Sabater, Julien Nicolas, Carole Peubey, Raluca Radu, Dinand Schepers, et al. The ERA5 global reanalysis. *Quarterly Journal of the Royal Meteorological Society*, 146(730):1999–2049, 2020.

Sture Holm. A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, 6(2):65–70, 1979. ISSN 03036898, 14679469.

Herbert Jaeger. The "echo state" approach to analysing and training recurrent neural networks. GMD Report 148, German National Research Center for Information Technology, 2001.

Mason Kamb, Eurika Kaiser, Steven L. Brunton, and J. Nathan Kutz. Time-delay observables for Koopman: Theory and applications. *SIAM Journal on Applied Dynamical Systems*, 19(2):886–917, 2020. doi: 10.1137/18M1216572.

Mohammad Khosravi. Learning finite-dimensional representations for Koopman operators. In *Proceedings of the 3rd Conference on Learning for Dynamics and Control*, volume 144 of *Proceedings of Machine Learning Research*, pp. 1281–1281. PMLR, 2021.

Bernard O. Koopman. Hamiltonian systems and transformation in Hilbert space. *Proceedings of the National Academy of Sciences*, 17(5):315–318, 1931. doi: 10.1073/pnas.17.5.315.

Vladimir Kostic, Pietro Novelli, Andreas Maurer, Carlo Ciliberto, Lorenzo Rosasco, and Massimiliano Pontil. Learning dynamical systems via Koopman operator regression in reproducing kernel Hilbert spaces. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 4017–4031. Curran Associates, Inc., 2022. doi: 10.52202/068431-0290.

Henning Lange, Steven L. Brunton, and J. Nathan Kutz. From Fourier to Koopman: Spectral methods for long-term time series prediction. *Journal of Machine Learning Research*, 22(41):1–38, 2021.

Boyu Li, Robert Simon Fong, and Peter Tino. Simple cycle reservoirs are universal. *Journal of Machine Learning Research*, 25(158):1–28, 2024.

Qianxiao Li, Felix Dietrich, Erik M. Bollt, and Ioannis G. Kevrekidis. Extended dynamic mode decomposition with dictionary learning: A data-driven adaptive spectral decomposition of the Koopman operator. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 27(10):103111, 2017. doi: 10.1063/1.4993854.

Yong Liu, Chenyu Li, Jianmin Wang, and Mingsheng Long. Koopa: Learning non-stationary time series dynamics with Koopman predictors. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 12271–12290. Curran Associates, Inc., 2023. doi: 10.52202/075280-0538.

Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. iTransformer: Inverted transformers are effective for time series forecasting. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 11116–11140, 2024.

Mantas Lukoševičius and Herbert Jaeger. Reservoir computing approaches to recurrent neural network training. *Computer Science Review*, 3(3):127–149, 2009. doi: 10.1016/j.cosrev.2009.03.005.

Bethany Lusch, J Nathan Kutz, and Steven L Brunton. Deep learning for universal linear embeddings of nonlinear dynamics. *Nature Communications*, 9(1):4950, 2018. doi: 10.1038/s41467-018-07210-0.

Alex T. Mallen, Henning Lange, and J. Nathan Kutz. Deep probabilistic Koopman: Long-term time-series forecasting under periodic uncertainties. *International Journal of Forecasting*, 40(3):859–868, 2024. doi: 10.1016/j.ijforecast.2023.07.001.

Fares Mehouachi and Saif Eddin Jabari. FlowMixer: A depth-agnostic neural architecture for interpretable spatiotemporal forecasting. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 88811–88861. Curran Associates, Inc., 2025. doi: 10.52202/085713-2970.

Igor Mezić. Spectral properties of dynamical systems, model reduction and decompositions. *Nonlinear Dynamics*, 41(1–3):309–325, 2005. doi: 10.1007/s11071-005-2824-x.

Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *The Eleventh International Conference on Learning Representations*, 2023.

Antonio Orvieto, Soham De, Caglar Gulcehre, Razvan Pascanu, and Samuel L Smith. Universality of linear recurrences followed by non-linear projections: Finite-width guarantees and benefits of complex eigenvalues. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 38837–38863. PMLR, 21–27 Jul 2024.

Samuel E. Otto and Clarence W. Rowley. Linearly recurrent autoencoder networks for learning dynamics. *SIAM Journal on Applied Dynamical Systems*, 18(1):558–593, 2019. doi: 10.1137/18M1177846.

Shaowu Pan and Karthik Duraisamy. Physics-informed probabilistic learning of linear embeddings of nonlinear dynamics with guaranteed stability. *SIAM Journal on Applied Dynamical Systems*, 19(1):480–509, 2020. doi: 10.1137/19M1267246.

Clarence W. Rowley, Igor Mezić, Shervin Bagheri, Philipp Schlatter, and Dan S. Henningson. Spectral analysis of nonlinear flows. *Journal of Fluid Mechanics*, 641:115–127, 2009. doi: 10.1017/S0022112009992059.

Peter J. Schmid. Dynamic mode decomposition of numerical and experimental data. *Journal of Fluid Mechanics*, 656:5–28, 2010. doi: 10.1017/S0022112010001217.

Xingjian SHI, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-kin Wong, and Wang-chun WOO. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 28, pp. 802–810. Curran Associates, Inc., 2015.

Naoya Takeishi, Yoshinobu Kawahara, and Takehisa Yairi. Learning Koopman invariant subspaces for dynamic mode decomposition. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Peter Tino. Dynamical systems as temporal feature spaces. *Journal of Machine Learning Research*, 21(44):1–42, 2020.

Lloyd N. Trefethen. Pseudospectra of linear operators. *SIAM Review*, 39(3):383–406, 1997. doi: 10.1137/S0036144595295284.

Rui Wang, Yihe Dong, Sercan O Arik, and Rose Yu. Koopman neural operator forecaster for time-series with temporal distributional shifts. In *The Eleventh International Conference on Learning Representations*, 2023.

Matthew O. Williams, Ioannis G. Kevrekidis, and Clarence W. Rowley. A data-driven approximation of the Koopman operator: Extending dynamic mode decomposition. *Journal of Nonlinear Science*, 25(6):1307–1346, 2015. doi: 10.1007/s00332-015-9258-5.

Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. TimesNet: Temporal 2D-variation modeling for general time series analysis. In *The Eleventh International Conference on Learning Representations*, 2023.

Xingjian Wu, Xiangfei Qiu, Hongfan Gao, Jilin Hu, Bin Yang, and Chenjuan Guo. K^2VAE: A Koopman-Kalman enhanced variational AutoEncoder for probabilistic time series forecasting. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 67562–67583. PMLR, 2025.

Sijie Xiong, Cheng Tang, and Atsushi Shimada. KUMA: A novel framework with Koopman separation and efficient multilevel extraction in time series forecasting. In *Forty-third International Conference on Machine Learning*, 2026.

Izzet B. Yildiz, Herbert Jaeger, and Stefan J. Kiebel. Re-visiting the echo state property. *Neural Networks*, 35:1–9, 2012. ISSN 0893-6080. doi: 10.1016/j.neunet.2012.07.005.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(9):11121–11128, 2023. doi: 10.1609/aaai.v37i9.26317.

Yitian Zhang, Liheng Ma, Antonios Valkanas, Boris N. Oreshkin, and Mark Coates. SKOLR: Structured Koopman operator linear RNN for time-series forecasting. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 75734–75756. PMLR, 13–19 Jul 2025.

Ronghua Zheng, Hanru Bai, and Weiyang Ding. KooNPro: A variance-aware Koopman probabilistic model enhanced by neural process for time series forecasting. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 29457–29481, 2025.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting. In *AAAI Conference on Artificial Intelligence*, pp. 11106–11115, 2021. doi: 10.1609/AAAI.V35I12.17325.

A Extended Related Work and Method Positioning

A.1 Comparison with Koopman Forecasters

The finite-dimensional Koopman relation in Appendix B.1 separates the choice of observable coordinates, their proposed linear evolution, and the remaining mismatch. DMD uses measured states as fixed observables and fits the transition from snapshot pairs (Schmid, 2010); Delay-DMD first adds a prescribed history embedding (Kamb et al., 2020). A Koopman autoencoder instead learns both the coordinates and transition, using reconstruction, latent-linearity, and prediction objectives (Lusch et al., 2018; Azencot et al., 2020).

Koopa learns a representation with a globally learned operator for its time-invariant component and a locally fitted DMD operator for its time-varying component (Liu et al., 2023). It passes observation-space fitting errors between residual blocks. K^2VAE learns latent linear dynamics and feeds residual nonlinear information into a Kalman-based refinement that also models uncertainty (Wu et al., 2025). Both make use of what their linear components miss, but neither constructs the one-step latent residual in Equation (7).

Spectral and invariant-coordinate methods. Related approaches focus on learning invariant representations, invertible coordinates, probabilistic spectral responses, or online updates of Koopman embeddings (Lusch et al., 2018; Azencot et al., 2020; Mallen et al., 2024; Frion et al., 2025; Gao et al., 2025). InformationKoopman encourages temporally informative and diverse representations while estimating its operator from training trajectories (Cheng et al., 2026); it does not prescribe the operator as EchoKoop does.

Forecast-oriented Koopman propagation. KNF combines global and local operators with online feedback to handle temporal distribution shift (Wang et al., 2023). SKOLR connects a structured Koopman approximation to parallel linear-RNN updates and learns a spectral decomposition of the input (Zhang et al., 2025). KooNPro introduces a variance-aware continuous spectrum together with a neural-process component (Zheng et al., 2025). Their transitions are learned, inferred, or updated from data.

Koopman-guided hybrid architectures. FlowMixer uses constrained mixing operations and reversible transformations, from which Kronecker–Koopman modes and a semigroup interpretation can be derived (Mehouachi & Jabari, 2025). KoNODE uses Koopman evolution to model time-varying parameters of a neural ODE (Bai & Ding, 2025). KUMA combines input-dependent Koopman separation with a multilevel attention architecture (Xiong et al., 2026), while DeepKoopFormer integrates a spectrally constrained Koopman propagator with Transformer representations (Forootani et al., 2026). In these models, Koopman structure is one component of a broader forecasting architecture.

EchoKoop takes a different route: its frequency–decay operators remain fixed while the representation and readout adapt to the forecasting task. These modes serve as a forecasting basis, not as estimates of the true Koopman spectrum. Fourier-to-Koopman also uses oscillatory structure, but fits its modes from data (Lange et al., 2021).

The table 4 is an operational summary rather than an exhaustive taxonomy. Several methods could reasonably appear in more than one row. For example, FlowMixer combines constrained forecasting with modal interpretation, while K^2VAE combines learned latent propagation with probabilistic inference. The grouping records the primary role emphasized in each architecture and is used only to clarify the design space considered by EchoKoop.

A.2 Relation to Reservoir Computing

Reservoir computing separates a fixed recurrent system from a trained readout. In an echo-state network, observations drive the reservoir at each input step, and its state summarizes that input history (Jaeger, 2001; Lukoševičius & Jaeger, 2009). EchoKoop likewise fixes its transition, but learns a map from the complete history to the initial latent state and then rolls that state forward without future input injection. Its prescribed normal rotation blocks also differ from a generic random reservoir. A spectral-radius constraint alone does not guarantee the echo-state property for arbitrary

Table 4: Operational positioning of representative Koopman-based time-series models according to the dominant role of their Koopman components. The categories are not mutually exclusive and should not be interpreted as historical priority claims.

Dominant modeling role	Representative methods
Representation and spectral modeling	KAE (Lusch et al., 2018); Consistent KAE (Azencot et al., 2020); Deep Probabilistic Koopman (Mallen et al., 2024); AIKAE (Frion et al., 2025); COLoKe (Gao et al., 2025); InformationKoopman (Cheng et al., 2026)
Forecast-oriented learned or inferred propagation	KNF (Wang et al., 2023); Koopa (Liu et al., 2023); SKOLR (Zhang et al., 2025); K^2VAE (Wu et al., 2025); KooNPro (Zheng et al., 2025)
Koopman-guided hybrid architectures	FlowMixer (Mehouachi & Jabari, 2025); KoNODE (Bai & Ding, 2025); KUMA (Xiong et al., 2026); DeepKoopFormer (Forootani et al., 2026)
Prescribed-flow coordinate learning	**EchoKoop (ours):** learned lifting, selection, and decoding with a fixed multiresolution frequency–decay operator dictionary

recurrent matrices (Yildiz et al., 2012). EchoKoop-R isolates the effect of operator design by replacing the normal bank with sparse random operators of matched spectral radii while retaining the forecasting architecture.

A.3 Relation to State-Space Models and Linear RNNs

Modern state-space layers and linear RNNs also use linear recurrences; diagonal or complex-valued transitions support efficient long-sequence processing (Gu et al., 2022; Gupta et al., 2022). SKOLR connects Koopman forecasting to this family by combining a learned spectral decomposition with a structured Koopman operator implemented through parallel linear-RNN updates (Zhang et al., 2025). Each damped-rotation block in EchoKoop corresponds to a conjugate pair of complex eigenvalues, as shown in Appendix B.4. EchoKoop maps the observed window to latent initial states and rolls them out with prescribed frequency–decay operators; SKOLR learns its recurrent transition. Results on complex linear recurrences help explain the expressive value of oscillatory modes (Orvieto et al., 2024), but do not by themselves establish the accuracy or stability of EchoKoop's full decoder.

B Latent Evolution and Finite-Horizon Analysis

B.1 Latent Evolution and Residual Propagation

The Koopman operator acts linearly on observables even when the state map $\mathcal{F}$ is nonlinear. A finite-dimensional model seeks coordinates Φ whose evolution is approximately closed under a linear map. To make the approximation explicit, write

$$\Phi(\mathcal{F}(\mathbf{x})) = \mathbf{A}(\mathbf{x})\Phi(\mathbf{x}) + \mathbf{r}_{\Phi,\mathbf{A}}(\mathbf{x}), \qquad \mathbf{r}_{\Phi,\mathbf{A}}(\mathbf{x}) := \Phi(\mathcal{F}(\mathbf{x})) - \mathbf{A}(\mathbf{x})\Phi(\mathbf{x}). \tag{7}$$

The state $\mathbf{x}$ may include an observation history when the forecast model uses one. Along a trajectory, the successive matrices $\mathbf{A}(\mathcal{F}^{n-1}(\mathbf{x}))\cdots\mathbf{A}(\mathbf{x})$ form an operator cocycle. The usual fixed-matrix Koopman approximation is its constant-operator case, $\mathbf{A}(\mathbf{x}) = \mathbf{K}$. Equation (7) separates three choices: the coordinates, their proposed linear evolution, and the part that evolution does not capture. The residual is defined by the first two choices; it need not be a separately learned component. For a fixed $\mathbf{K}$, iteration gives

$$\Phi(\mathcal{F}^n(\mathbf{x})) = \mathbf{K}^n\Phi(\mathbf{x}) + \sum_{j=0}^{n-1} \mathbf{K}^{n-1-j}\mathbf{r}_{\Phi,\mathbf{K}}(\mathcal{F}^j(\mathbf{x})).$$

Thus a small one-step mismatch is useful only together with controlled propagation of that mismatch.

EchoKoop realizes the fixed-operator case of Equation (7) within each latent rollout. It is trained on forecast error, without a separate latent forcing term or a loss that directly minimizes $\|\mathbf{r}_{\Phi,\mathbf{K}}\|$ at each step. Its cross-scale correction adjusts the forecast after rollout, rather than supplying the one-step latent residual. Normality therefore controls how errors propagate under the fixed operator, while the learned coordinates must still make its modes useful for prediction.

The next two subsections examine finite-horizon capacity and error propagation separately.

B.2 Finite-Horizon Reparameterization

Fix a resolution s and consider its native temporal grid. A two-dimensional block in Equation (5) is the real representation of multiplication by the complex mode associated with (p, q),

$$\zeta_{s,p,q} = 2^{-s/\tau_q} \exp\Big(\mathrm{i}s\omega_p^{(s)}\Big).$$

On this grid, a linear readout with freely chosen initial modal coefficients produces combinations of the form $\sum_j c_j \zeta_j^n$, where j enumerates distinct modes from the fixed resolution.

Theorem 1 (Finite-horizon spanning)**.** *Let $\{\zeta_j\}_{j=1}^J$ be a fixed dictionary containing at least H distinct complex modes on a common temporal grid. Then, for any length-H complex scalar sequence $\{y_n \in \mathbb{C}\}_{n=0}^{H-1}$, there exist coefficients $\{c_j \in \mathbb{C}\}_{j=1}^J$ such that*

$$y_n = \sum_{j=1}^{J} c_j \zeta_j^n, \qquad n = 0, \dots, H-1. \tag{8}$$

For real-valued sequences, the same result holds in a real damped-rotation representation when the dictionary is closed under complex conjugation.

The result follows from the full row rank of a Vandermonde matrix. For multivariate forecasting, the coefficients can be selected independently for each output variable. It shows that input-dependent coefficients can express different finite-horizon forecasts without changing the modes themselves.

Theorem 1 describes what a sufficiently rich fixed basis can express through a linear readout on one grid. In EchoKoop, coarse-grid trajectories are interpolated and decoded by learned modules; the configured bank is also not required to meet the rank condition at every horizon. The result motivates fixed transitions without claiming recovery of the true Koopman spectrum.

B.3 Approximation and Rollout Amplification

At scale s, stack the Q_a member states before selection as $\mathbf{u}_n^{(s)} \in \mathbb{R}^{2PQ_a}$ and collect their operators into the augmented transition

$$\mathcal{K}^{(s)} = \bigoplus_{q=1}^{Q_a} \mathbf{K}_q^{(s)}.$$

Because every block is a scaled rotation, $\mathcal{K}^{(s)}$ is normal and

$$\left\| \left(\mathcal{K}^{(s)}\right)^n \right\|_2 = \bar{\rho}_s^n, \qquad \bar{\rho}_s = \max_q 2^{-s/\tau_q} \le 1.$$

Hereafter, s is fixed; we omit the superscript (s) on target and learned augmented states and on representation residuals.

Let $\mathbf{u}_n^\star \in \mathbb{R}^{2PQ_a}$ be any reference augmented latent trajectory and define its one-step representation residual under the fixed flow as

$$\mathbf{r}_n = \mathbf{u}_{n+1}^\star - \mathcal{K}^{(s)} \mathbf{u}_n^\star.$$

If the learned augmented initial state is $\widehat{\mathbf{u}}_0$, then the latent rollout error satisfies

$$\left\| \left(\mathcal{K}^{(s)}\right)^n \widehat{\mathbf{u}}_0 - \mathbf{u}_n^\star \right\|_2 \le \bar{\rho}_s^n \left\|\widehat{\mathbf{u}}_0 - \mathbf{u}_0^\star\right\|_2 + \sum_{j=0}^{n-1} \bar{\rho}_s^{n-1-j} \left\|\mathbf{r}_j\right\|_2. \tag{9}$$

The first term propagates the initial-state error, while the second measures the reference trajectory's mismatch with the fixed flow. The bound is informative when both errors are small; normal frozen operators ensure that neither is multiplied by a non-normal transient-growth factor. The architecture then applies a linear selector $\mathbf{A}_\alpha^{(s)} \in \mathbb{R}^{2P\times 2PQ_a}$, constructed from $\alpha_{p,q}^{(s)}$, so that $\mathbf{z}_n^{(s)} = \mathbf{A}_\alpha^{(s)}\mathbf{u}_n^{(s)}$ before orthogonal mixing and decoding.

The bound isolates amplification within the prescribed latent rollout. Its relation to forecast error also depends on the learned readout, considered below.

B.4 Connection Between Complex Modes and Real Rotation Blocks

Consider the complex modal multiplier

$$\zeta = \rho\exp(\mathrm{i}\theta), \qquad \rho = 2^{-s/\tau}, \qquad \theta = s\omega.$$

Writing a complex state as $z = a + \mathrm{i}b$, multiplication by ζ can be represented in real coordinates as

$$\begin{bmatrix}\mathrm{Re}(\zeta z)\\ \mathrm{Im}(\zeta z)\end{bmatrix} = \rho\begin{bmatrix}\cos\theta & -\sin\theta\\ \sin\theta & \cos\theta\end{bmatrix}\begin{bmatrix}a\\ b\end{bmatrix}. \tag{10}$$

Substituting $\rho = 2^{-s/\tau}$ and $\theta = s\omega$ recovers Equation (5). Hence, a pair of conjugate complex modes is represented by one real two-dimensional damped-rotation block.

After n native transitions,

$$\left[\mathbf{R}^{(s)}(\tau,\omega)\right]^n = 2^{-ns/\tau}\begin{bmatrix}\cos(ns\omega) & -\sin(ns\omega)\\ \sin(ns\omega) & \cos(ns\omega)\end{bmatrix}.$$

A linear readout from several such blocks therefore generates real parts of linear combinations of ζ_j^n.

B.5 Proof of the Finite-Horizon Spanning Result

Proof of Theorem 1. Select H distinct modes from the dictionary and, without loss of generality, denote them by $\zeta_1,\dots,\zeta_H$. Define the Vandermonde matrix

$$\mathbf{V} = \begin{bmatrix}1 & 1 & \cdots & 1\\ \zeta_1 & \zeta_2 & \cdots & \zeta_H\\ \vdots & \vdots & \ddots & \vdots\\ \zeta_1^{H-1} & \zeta_2^{H-1} & \cdots & \zeta_H^{H-1}\end{bmatrix}.$$

Its determinant is

$$\det(\mathbf{V}) = \prod_{1\le i<j\le H}(\zeta_j - \zeta_i).$$

Because the selected modes are distinct, $\det(\mathbf{V}) \neq 0$ and $\mathbf{V}$ is invertible.

For an arbitrary target vector

$$\mathbf{y} = [y_0,\dots,y_{H-1}]^\top,$$

choose

$$\mathbf{c} = \mathbf{V}^{-1}\mathbf{y}.$$

Then $\mathbf{V}\mathbf{c} = \mathbf{y}$, which is equivalent to Equation (8). If the complete dictionary has $J > H$ modes, the corresponding $H \times J$ matrix has full row rank, and a valid solution can instead be written as

$$\mathbf{c} = \mathbf{V}^\dagger\mathbf{y},$$

where $\mathbf{V}^\dagger$ is the Moore–Penrose pseudoinverse; because $\mathbf{V}$ has full row rank, it is a right inverse and satisfies $\mathbf{V}\mathbf{V}^\dagger = \mathbf{I}_H$.

For a real target sequence and a dictionary closed under complex conjugation, conjugating any valid expansion produces another valid expansion with conjugate-permuted coefficients. Averaging the two solutions gives conjugate-symmetric coefficients. Each conjugate pair can then be implemented by a real two-dimensional rotation block according to Equation (10). □

For C output variables, let $\mathbf{Y} \in \mathbb{R}^{H\times C}$ contain the target trajectory. Applying the same argument column-wise yields

$$\mathbf{Y} = \mathbf{V}\mathbf{C}, \qquad \mathbf{C} = \mathbf{V}^{\dagger}\mathbf{Y}.$$

Thus, increasing the number of variables changes the coefficient matrix but does not change the temporal spanning condition.

B.6 Conditional Forecasting with Fixed Modes

The preceding result concerns one trajectory. The next corollary explains how the coefficients can depend on an input history.

Corollary 1 (Conditional finite-horizon representation)**.** *Let Ω be a compact set of historical windows and let*

$$f^{\star} : \Omega \to \mathbb{R}^{H\times C}$$

be a continuous forecasting function. Suppose the fixed modal matrix on this grid, $\mathbf{V} \in \mathbb{C}^{H\times J}$, has full row rank and its modal dictionary is closed under complex conjugation. Then there exists a continuous coefficient map

$$c^{\star} : \Omega \to \mathbb{C}^{J\times C}$$

such that

$$f^{\star}(\mathbf{X}) = \mathbf{V}c^{\star}(\mathbf{X}) \qquad \textit{for all } \mathbf{X} \in \Omega.$$

Proof. Because $\mathbf{V}$ has full row rank, its Moore–Penrose pseudoinverse is a fixed right inverse. Define

$$c^{\star}(\mathbf{X}) = \mathbf{V}^{\dagger}f^{\star}(\mathbf{X}).$$

This map is continuous because it is the composition of the continuous function $f^{\star}$ and a fixed linear transformation. Moreover,

$$\mathbf{V}c^{\star}(\mathbf{X}) = \mathbf{V}\mathbf{V}^{\dagger}f^{\star}(\mathbf{X}) = f^{\star}(\mathbf{X}).$$

Because $f^{\star}(\mathbf{X})$ is real and the dictionary is conjugate closed, averaging this coefficient vector with its conjugate-permuted copy yields a continuous conjugate-symmetric coefficient map with the same reconstruction. Such coefficients are implemented by the real rotation blocks in Equation (10). □

If a neural lifting and readout network approximates $c^{\star}(\mathbf{X})$ uniformly by $\widehat{c}_{\theta}(\mathbf{X})$, then

$$\begin{aligned}\left\|\mathbf{V}\widehat{c}_{\theta}(\mathbf{X}) - f^{\star}(\mathbf{X})\right\|_{\mathrm{F}} &= \left\|\mathbf{V}\left(\widehat{c}_{\theta}(\mathbf{X}) - c^{\star}(\mathbf{X})\right)\right\|_{\mathrm{F}} \\ &\le \|\mathbf{V}\|_2 \left\|\widehat{c}_{\theta}(\mathbf{X}) - c^{\star}(\mathbf{X})\right\|_{\mathrm{F}}.\end{aligned}$$

This formalizes the "learn the lift, freeze the flow" perspective: input dependence can be carried by the coefficients even when the temporal basis is fixed.

EchoKoop does not compute $\mathbf{V}^{\dagger}f^{\star}(\mathbf{X})$. Its encoder and history-to-state initializer instead learn input-dependent initial states, which the decoder reads out after the fixed rollout. The corollary explains why fixed modes can coexist with input-dependent forecasts without prescribing the exact map learned by the full network.

B.7 Proof of the Approximation–Amplification Bound

For notational simplicity, fix one scale and write $\mathcal{K} = \mathcal{K}^{(s)}$. The reference augmented latent trajectory satisfies

$$\mathbf{u}^{\star}_{n+1} = \mathcal{K}\mathbf{u}^{\star}_{n} + \mathbf{r}_n, \tag{11}$$

where $\mathbf{r}_n$ measures the one-step mismatch between the target trajectory and the fixed flow. The model rollout is

$$\widehat{\mathbf{u}}_n = \mathcal{K}^n\widehat{\mathbf{u}}_0. \tag{12}$$

Define the latent error

$$\mathbf{e}_n = \widehat{\mathbf{u}}_n - \mathbf{u}^{\star}_n.$$

Using Equation (11) and Equation (12),

$$\mathbf{e}_{n+1} = \mathcal{K}\mathbf{e}_n - \mathbf{r}_n.$$

Unrolling this recursion gives

$$\mathbf{e}_n = \mathcal{K}^n \mathbf{e}_0 - \sum_{j=0}^{n-1} \mathcal{K}^{n-1-j} \mathbf{r}_j.$$

The triangle inequality then yields

$$\|\mathbf{e}_n\|_2 \le \|\mathcal{K}^n\|_2 \|\mathbf{e}_0\|_2 + \sum_{j=0}^{n-1} \|\mathcal{K}^{n-1-j}\|_2 \|\mathbf{r}_j\|_2. \tag{13}$$

The augmented EchoKoop transition is block diagonal and normal. Its largest block radius is

$$\bar{\rho}_s = \max_q 2^{-s/\tau_q}.$$

Therefore,

$$\|\mathcal{K}^n\|_2 = \bar{\rho}_s^n.$$

Substituting this equality into Equation (13) proves Equation (9).

If the representation residual is uniformly bounded by $\|\mathbf{r}_j\|_2 \le \epsilon_r$, then

$$\|\mathbf{e}_n\|_2 \le \bar{\rho}_s^n \|\mathbf{e}_0\|_2 + \epsilon_r \frac{1 - \bar{\rho}_s^n}{1 - \bar{\rho}_s}, \qquad \bar{\rho}_s < 1.$$

Thus, errors introduced at earlier steps are geometrically attenuated rather than transiently amplified.

B.8 Including the Decoder Error

To relate the latent bound to one forecast step, let $\mathbf{A}_\alpha^{(s)}$ be the selector defined in Section B.3 and let $\mathcal{D}_{\mathrm{pt}}^{(s)}$ be a pointwise readout. Write $\mathcal{G}^{(s)} = \mathcal{D}_{\mathrm{pt}}^{(s)} \circ \mathbf{O}^{(s)} \circ \mathbf{A}_\alpha^{(s)}$ for the resulting map from the augmented state to the observation space. Suppose it is $L_{\mathcal{G}}$-Lipschitz over the relevant latent domain:

$$\left\| \mathcal{G}^{(s)}(\mathbf{a}) - \mathcal{G}^{(s)}(\mathbf{b}) \right\|_2 \le L_{\mathcal{G}} \|\mathbf{a} - \mathbf{b}\|_2.$$

Define the effective-readout representation error as

$$\varepsilon_n^{\mathrm{dec}} = \left\| \mathcal{G}^{(s)}(\mathbf{u}_n^\star) - \mathbf{y}_n^\star \right\|_2.$$

Then

$$\begin{aligned} \left\| \mathcal{G}^{(s)}(\widehat{\mathbf{u}}_n) - \mathbf{y}_n^\star \right\|_2 \le \varepsilon_n^{\mathrm{dec}} &+ L_{\mathcal{G}} \bar{\rho}_s^n \, \|\widehat{\mathbf{u}}_0 - \mathbf{u}_0^\star\|_2 \\ &+ L_{\mathcal{G}} \sum_{j=0}^{n-1} \bar{\rho}_s^{n-1-j} \|\mathbf{r}_j\|_2. \end{aligned}$$

This separates readout error from initialization error and mismatch with the fixed flow. The implemented decoder acts on the full interpolated trajectory, rather than on individual steps. Applying the same triangle-inequality argument to that trajectory requires a Lipschitz bound for the complete branch readout. With multiple branches and cross-scale correction, the corresponding bound also depends on those learned modules; latent contraction alone does not make the complete predictor contractive.

B.9 Normality and Relative Transient Growth

A spectral-radius constraint alone does not prevent transient amplification for non-normal operators (Trefethen, 1997). For a transition matrix $\mathbf{A}$ with $\rho(\mathbf{A}) > 0$, define the finite-horizon relative growth factor

$$G_{\mathrm{rel}}^{(H)}(\mathbf{A}) = \max_{1 \le n \le H} \frac{\|\mathbf{A}^n\|_2}{\rho(\mathbf{A})^n}.$$

If $\mathbf{A}$ is normal, then

$$\|\mathbf{A}^n\|_2 = \rho(\mathbf{A})^n$$

and hence

$$G_{\mathrm{rel}}^{(H)}(\mathbf{A}) = 1.$$

For a non-normal learned transition, $G_{\mathrm{rel}}^{(H)}(\mathbf{A})$ can be substantially larger than one, even when $\rho(\mathbf{A}) \leq 1$. We use this quantity in Section 4 to compare learned transition matrices with the frozen normal flows.

B.10 Scope and Limitations of the Analysis

Theorem 1 establishes finite-horizon capacity when the dictionary contains sufficiently many distinct modes. Its practical value also depends on conditioning: closely spaced modes may require large coefficients even when an exact representation exists. Extrapolation beyond the fitted horizon depends on the frequency and half-life choices, the learned coordinates, and the training distribution.

Normality controls amplification within the frozen latent transition, while the complete predictor also contains learned nonlinear modules. In Equation (9), the decay factors limit the propagation of error, but the initial-state error $\|\widehat{\mathbf{u}}_0 - \mathbf{u}_0^\star\|_2$ and residuals $\|\mathbf{r}_j\|_2$ still depend on how well the learned representation uses the fixed dictionary. The analysis therefore concerns latent rollout stability, not contractivity of the complete predictor.

The quality of the learned coordinates remains a separate question: $\Phi \equiv 0$ makes the residual in Equation (7) vanish without encoding the observed dynamics. Information-theoretic Koopman learning addresses this issue through mutual-information and von Neumann-entropy objectives (Cheng et al., 2026). EchoKoop does not use these objectives; its forecast loss learns the coordinates, while the present bounds do not rule out latent mode collapse.

C Detailed Construction of EchoKoop

C.1 Causal Encoder and Multiscale Representation

Let $\widetilde{\mathbf{X}} \in \mathbb{R}^{L\times C}$ denote the history centered by its final observation, as defined in Equation (2). The lifting network begins with a pointwise projection, followed by residual causal temporal convolutions with dilation factors

$$1,\ 2,\ 4,\ 8,\ 16.$$

The causal padding ensures that the representation at time t depends only on observations up to t. At each time step, $\mathrm{LN}_{\mathrm{feat}}$ denotes LayerNorm over the d_ℓ learned feature coordinates. We concatenate these normalized features with the centered observations:

$$\mathbf{f}_t = \left[\widetilde{\mathbf{x}}_t\,;\,\mathrm{LN}_{\mathrm{feat}}\left(\mathcal{E}_\theta(\widetilde{\mathbf{X}})_t\right)\right].$$

For a temporal resolution s, the sequence is divided into non-overlapping cells. The j-th cell representation is

$$\mathbf{g}_j^{(s)} = \frac{1}{\sqrt{s}}\sum_{r=0}^{s-1}\mathbf{f}_{js+r+1}, \qquad j = 0,\dots,N_s^{\mathrm{in}}-1,$$

where $N_s^{\mathrm{in}} = \lfloor L/s \rfloor$. In practice, the input length is selected to be compatible with the resolutions used in the model.

A learned scale-specific projection maps each cell to P two-dimensional pairs:

$$\left[\mathbf{v}_{j,1}^{(s)};\dots;\mathbf{v}_{j,P}^{(s)}\right] = \mathbf{W}_{\mathrm{in}}^{(s)}\mathbf{g}_j^{(s)} + \mathbf{b}_{\mathrm{in}}^{(s)}, \qquad \mathbf{v}_{j,p}^{(s)} \in \mathbb{R}^2.$$

C.2 Frequency and Half-Life Dictionaries

For scale s, define

$$M_s = \max\left(1, \left\lfloor \frac{L}{2s} \right\rfloor\right).$$

The frequency index and angular frequency of pair p are

$$k_p^{(s)} = (p-1) \bmod M_s, \qquad \omega_p^{(s)} = \gamma_s \frac{2\pi k_p^{(s)}}{L},$$

where γ_s controls the frequency range assigned to scale s. Frequencies are repeated when $P > M_s$.

The decay dictionary contains Q_a prescribed half-lives,

$$\mathcal{T} = \{\tau_1, \ldots, \tau_{Q_a}\}, \qquad \mu_q = \frac{\log 2}{\tau_q}.$$

Consequently, advancing a state by s time steps multiplies its amplitude by

$$\rho_{s,q} = \exp(-s\mu_q) = 2^{-s/\tau_q}.$$

C.3 Phase-Consistent History-to-State Initialization

A downsampled cell represents an interval rather than a single time point. We assign the j-th cell the center

$$b_j^{(s)} = js + \frac{s-1}{2},$$

and measure its age relative to the last input observation:

$$a_j^{(s)} = L - 1 - b_j^{(s)}.$$

For half-life τ_q, the unnormalized temporal envelope is

$$e_{j,q}^{(s)} = \exp\left(-\mu_q a_j^{(s)}\right).$$

We normalize this envelope to reduce scale-dependent magnitude differences:

$$\eta_{j,q}^{(s)} = \frac{e_{j,q}^{(s)}}{\sqrt{L}\left(\sum_{\ell=0}^{N_s^{\text{in}}-1}\left(e_{\ell,q}^{(s)}\right)^2\right)^{1/2}}.$$

Each historical pair is rotated forward from its cell center to the forecast origin. The phase-consistent initial state is

$$\mathbf{z}_{0,q,p}^{(s)} = \sum_{j=0}^{N_s^{\text{in}}-1} \eta_{j,q}^{(s)} \begin{bmatrix} \cos\left(a_j^{(s)}\omega_p^{(s)}\right) & -\sin\left(a_j^{(s)}\omega_p^{(s)}\right) \\ \sin\left(a_j^{(s)}\omega_p^{(s)}\right) & \cos\left(a_j^{(s)}\omega_p^{(s)}\right) \end{bmatrix} \mathbf{v}_{j,p}^{(s)}.$$

This operation aggregates the complete encoded history while respecting the phase assigned to each latent pair. Removing the age-dependent rotation reduces the initialization to an amplitude-weighted aggregation and loses temporal alignment.

C.4 Fixed Transition Bank

The bank and its damped-rotation blocks are defined in Equation (4)–Equation (5). They are constructed once and stored as non-trainable model buffers.

Because a scaled rotation is normal,

$$\mathbf{R}^{(s)}(\tau,\omega)^\top \mathbf{R}^{(s)}(\tau,\omega) = 2^{-2s/\tau}\mathbf{I}_2.$$

It follows that

$$\left\| \left(\mathbf{K}_q^{(s)}\right)^n \right\|_2 = 2^{-ns/\tau_q}.$$

Hence, the fixed latent flow cannot exhibit the transient growth associated with non-normal transition matrices. This statement concerns the prescribed transition operator itself and does not imply that the learned encoder, decoder, or residual correction is contractive.

C.5 Globally Learned Operator Selection

EchoKoop learns one distribution over the Q_a fixed half-lives for each scale and frequency pair. Let $\ell_{p,q}^{(s)}$ denote its trainable logits. The selector weights are

$$\alpha_{p,q}^{(s)} = \frac{\exp\left(\ell_{p,q}^{(s)}\right)}{\sum_{r=1}^{Q_a} \exp\left(\ell_{p,r}^{(s)}\right)}.$$

The logits are initialized either uniformly (equal branch weights) or from a specified positive initialization profile, and they remain learnable in both cases. The resulting weights are global parameters shared across samples and forecast steps; sample dependence enters through $\mathbf{z}_{0,q,p}^{(s)}$, not through the selector.

The selected pair state follows Equation (6). Concatenating the pairs gives

$$\mathbf{z}_n^{(s)} = \left[\mathbf{z}_{n,1}^{(s)}; \ldots; \mathbf{z}_{n,P}^{(s)}\right] \in \mathbb{R}^{2P}.$$

We additionally apply a fixed orthogonal feature mixing matrix $\mathbf{O}^{(s)}$, generated by seeded QR decomposition:

$$\overline{\mathbf{z}}_n^{(s)} = \mathbf{O}^{(s)} \mathbf{z}_n^{(s)}, \qquad \left(\mathbf{O}^{(s)}\right)^\top \mathbf{O}^{(s)} = \mathbf{I}.$$

This mixes information across latent pairs without changing their Euclidean norms or introducing trainable transition parameters.

C.6 Native Rollout, Decoding, and Scale Fusion

The scale-s branch evolves on its native temporal grid. It produces

$$N_s^{\text{out}} = \left\lceil \frac{H}{s} \right\rceil$$

latent transitions and includes the initial state at $n = 0$. Linear interpolation maps source nodes $\{0, s, 2s, \ldots, N_s^{\text{out}} s\}$ to target nodes $\{0, 1, \ldots, H\}$:

$$\mathbf{Z}^{(s)} = \text{Interp}\left(\{\overline{\mathbf{z}}_n^{(s)}\}_{n=0}^{N_s^{\text{out}}}, \{0, 1, \ldots, H\}\right).$$

The initial point is removed before forecasting so that the decoder receives the H future latent states.

Each scale has a learned decoder composed of feature normalization, a linear projection, temporal residual blocks, and an output head:

$$\widehat{\mathbf{Y}}^{(s)} = \mathcal{D}_\theta^{(s)}\left(\mathbf{Z}_{1:H}^{(s)}\right).$$

The additive base forecast is

$$\widehat{\mathbf{Y}}_{\text{base}} = \mathbf{1}_H \mathbf{x}_L^\top + \sum_{s \in \mathcal{S}} \widehat{\mathbf{Y}}^{(s)}.$$

For cross-scale interaction, the interpolated latent trajectories are stacked along a branch dimension. Multi-head attention exchanges information among scales at each forecast step. Learned query pooling then summarizes the branch dimension:

$$\mathbf{G} = \text{QueryPool}\left[\text{MHA}\left(\{\mathbf{Z}_{1:H}^{(s)}\}_{s \in \mathcal{S}}\right)\right].$$

Temporal residual blocks transform $\mathbf{G}$ into a correction,

$$\Delta\widehat{\mathbf{Y}} = \mathcal{C}_\theta(\mathbf{G}), \qquad \widehat{\mathbf{Y}} = \widehat{\mathbf{Y}}_{\text{base}} + \Delta\widehat{\mathbf{Y}}.$$

The final projection of $\mathcal{C}_\theta$ is initialized to zero. Therefore, the initial model reduces to the sum of the independently decoded multiscale branches, and cross-scale corrections are introduced gradually during training.

C.7 Optimization Objective

The model is trained with the Huber loss:

$$\mathcal{L}_{\delta_{\mathrm{H}}} = \frac{1}{HC} \sum_{t=1}^{H} \sum_{c=1}^{C} \ell_{\delta_{\mathrm{H}}} \left(\widehat{y}_{t,c} - y_{t,c} \right),$$

where

$$\ell_{\delta_{\mathrm{H}}}(u) = \begin{cases} \frac{1}{2} u^2, & |u| \le \delta_{\mathrm{H}}, \\ \delta_{\mathrm{H}} \left(|u| - \frac{1}{2} \delta_{\mathrm{H}} \right), & |u| > \delta_{\mathrm{H}}, \end{cases} \qquad \delta_{\mathrm{H}} = 0.15.$$

Validation MSE is used for checkpoint selection.

C.8 Learned and Frozen Components

The learned components of EchoKoop are the causal lifting network, the scale-specific input projections, the half-life selectors, the branch decoders, and the cross-scale correction module. The frequency grids, half-life values, transition matrices, and orthogonal mixing matrices are fixed throughout training. In particular,

$$\frac{\partial \mathbf{K}_q^{(s)}}{\partial \Theta_{\mathrm{train}}} = \mathbf{0}, \qquad s \in \mathcal{S}, \quad q = 1, \ldots, Q_a.$$

Thus, learning is concentrated on finding coordinates and readout functions compatible with the prescribed dynamical basis.

C.9 Random-Reservoir Comparison

To assess a different frozen-flow design, we construct a sparse random bank whose members match the spectral radii of EchoKoop's prescribed operators. For each scale s and half-life τ_q, we first sample a sparse random matrix

$$\widetilde{\mathbf{K}}_q^{(s)} = \mathbf{M}_q^{(s)} \odot \mathbf{U}_q^{(s)}, \qquad \left[\mathbf{M}_q^{(s)} \right]_{ij} \sim \mathrm{Bernoulli}(c), \qquad \left[\mathbf{U}_q^{(s)} \right]_{ij} \sim \mathcal{U}(-0.5, 0.5).$$

where c is the connection probability and $\odot$ denotes element-wise multiplication. We then rescale the sampled matrix to match the spectral radius of the corresponding normal operator:

$$\mathbf{K}_{\mathrm{rand},q}^{(s)} = \frac{2^{-s/\tau_q}}{\rho\left(\widetilde{\mathbf{K}}_q^{(s)} \right)} \widetilde{\mathbf{K}}_q^{(s)},$$

where $\rho(\cdot)$ denotes the spectral radius. Therefore,

$$\rho\left(\mathbf{K}_{\mathrm{rand},q}^{(s)} \right) = 2^{-s/\tau_q} = \rho\left(\mathbf{K}_q^{(s)} \right).$$

The random operators remain frozen throughout training. The normal-operator and reservoir variants preserve the encoder, decoder, modal width, number of members, temporal resolutions, global selector, and training protocol, and their members are matched in spectral radius. EchoKoop aligns the history using the prescribed phases and decay envelopes of its operators. Matching a random operator's spectral radius does not provide these modal phases, so EchoKoop-R forms each reservoir's initial state with learned temporal weights shared across samples. Its forecast recurrence remains linear and autonomous, without input injection or a leaky-state update. The comparison thus evaluates each frozen operator bank with an initializer suited to its coordinates.

In Algorithm 1, $\mathrm{Rot}(\theta) = \left[\begin{smallmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{smallmatrix} \right]$ denotes the unit-radius two-dimensional rotation matrix.

D Experimental Protocol and Reproducibility Details

D.1 Datasets

The benchmark files for ETTh1, ETTh2, ETTm1, ETTm2, Weather, Exchange, PEMS03, PEMS04, PEMS07, and PEMS08 were downloaded from the Time-Series-Library collection on Hugging

Algorithm 1 EchoKoop Forward Pass

Require: Historical window $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_L]^\top$; forecast horizon H; resolution set $\mathcal{S}$; fixed operator banks $\{\mathbf{K}_q^{(s)}\}_{q=1}^{Q_a}$; selector logits $\{\ell_{p,q}^{(s)}\}$; orthogonal mixing matrices $\{\mathbf{O}^{(s)}\}$.

Ensure: Forecast $\widehat{\mathbf{Y}} \in \mathbb{R}^{H \times C}$.

$\mathbf{x}_{\mathrm{anc}} \leftarrow \mathbf{x}_L$
$\widetilde{\mathbf{X}} \leftarrow \mathbf{X} - \mathbf{1}_L \mathbf{x}_{\mathrm{anc}}^\top$
$\mathbf{F} \leftarrow \mathcal{E}_\theta\left(\widetilde{\mathbf{X}}\right)$
for $t = 1, \ldots, L$ **do**
  $\mathbf{f}_t \leftarrow [\widetilde{\mathbf{x}}_t; \mathrm{LN}_{\mathrm{feat}}(\mathbf{F}_t)]$
end for
for $s \in \mathcal{S}$ **do**
  $N_s^{\mathrm{in}} \leftarrow \lfloor L/s \rfloor, \quad N_s^{\mathrm{out}} \leftarrow \lceil H/s \rceil$
  for $j = 0, \ldots, N_s^{\mathrm{in}} - 1$ **do**
    $\mathbf{g}_j^{(s)} \leftarrow s^{-1/2} \sum_{r=0}^{s-1} \mathbf{f}_{js+r+1}$
    $[\mathbf{v}_{j,1}^{(s)}; \ldots; \mathbf{v}_{j,P}^{(s)}] \leftarrow \mathbf{W}_{\mathrm{in}}^{(s)} \mathbf{g}_j^{(s)} + \mathbf{b}_{\mathrm{in}}^{(s)}$
  end for
  for $q = 1, \ldots, Q_a$ **do**
    for $p = 1, \ldots, P$ **do**
      $\mathbf{z}_{0,q,p}^{(s)} \leftarrow \sum_{j=0}^{N_s^{\mathrm{in}}-1} \eta_{j,q}^{(s)} \, \mathrm{Rot}(a_j^{(s)} \omega_p^{(s)}) \mathbf{v}_{j,p}^{(s)}$
    end for
    $\mathbf{u}_{0,q}^{(s)} \leftarrow [\mathbf{z}_{0,q,1}^{(s)}; \ldots; \mathbf{z}_{0,q,P}^{(s)}]$
  end for
  $\boldsymbol{\alpha}_p^{(s)} \leftarrow \mathrm{softmax}(\boldsymbol{\ell}_p^{(s)}), \quad p = 1, \ldots, P$
  for $n = 0, \ldots, N_s^{\mathrm{out}} - 1$ **do**
    for $q = 1, \ldots, Q_a$ **do**
      $\mathbf{u}_{n+1,q}^{(s)} \leftarrow \mathbf{K}_q^{(s)} \mathbf{u}_{n,q}^{(s)}$
    end for
  end for
  for $n = 0, \ldots, N_s^{\mathrm{out}}$ **do**
    for $p = 1, \ldots, P$ **do**
      $\mathbf{z}_{n,p}^{(s)} \leftarrow \sum_{q=1}^{Q_a} \alpha_{p,q}^{(s)} [\mathbf{u}_{n,q}^{(s)}]_p$
    end for
    $\overline{\mathbf{z}}_n^{(s)} \leftarrow \mathbf{O}^{(s)} [\mathbf{z}_{n,1}^{(s)}; \ldots; \mathbf{z}_{n,P}^{(s)}]$
  end for
  $\mathbf{Z}_{1:H}^{(s)} \leftarrow \mathrm{Interp}\left(\{(ns, \overline{\mathbf{z}}_n^{(s)})\}_{n=0}^{N_s^{\mathrm{out}}}, \{0, 1, \ldots, H\}\right)_{1:H}$
  $\widehat{\mathbf{Y}}^{(s)} \leftarrow \mathcal{D}_\theta^{(s)}\left(\mathbf{Z}_{1:H}^{(s)}\right)$
end for
$\widehat{\mathbf{Y}}_{\mathrm{base}} \leftarrow \mathbf{1}_H \mathbf{x}_{\mathrm{anc}}^\top + \sum_{s \in \mathcal{S}} \widehat{\mathbf{Y}}^{(s)}$
if cross-scale correction is enabled **then**
  $\Delta\mathbf{Y} \leftarrow \mathcal{C}_\theta\left(\mathrm{QueryPool}\left(\mathrm{MHA}\left(\{\mathbf{Z}_{1:H}^{(s)}\}_{s \in \mathcal{S}}\right)\right)\right)$
else
  $\Delta\mathbf{Y} \leftarrow \mathbf{0}$
end if
$\widehat{\mathbf{Y}} \leftarrow \widehat{\mathbf{Y}}_{\mathrm{base}} + \Delta\mathbf{Y}$
return $\widehat{\mathbf{Y}}$

Face[1]. Their preprocessing and evaluation settings follow the established long-horizon forecasting protocol used by Informer, TimesNet, and iTransformer (Zhou et al., 2021; Wu et al., 2023; Liu et al., 2024).

[1] https://huggingface.co/datasets/thuml/Time-Series-Library

Table 5: Summary of datasets used in the experiments. C denotes the number of variables, T denotes the number of time steps, and Fr denotes the sampling frequency.

Dataset	C	T	Fr	Domain
ETTh1	7	17,420	1 hour	electricity transformer temperature
ETTh2	7	17,420	1 hour	electricity transformer temperature
ETTm1	7	69,680	15 min	electricity transformer temperature
ETTm2	7	69,680	15 min	electricity transformer temperature
Weather	21	52,696	10 min	meteorology
Exchange	8	7,588	1 day	exchange rate
PEMS03	358	26,208	5 min	traffic flow
PEMS04	307	16,992	5 min	traffic flow
PEMS07	883	28,224	5 min	traffic flow
PEMS08	170	17,856	5 min	traffic flow

Table 5 summarizes the datasets and their sampling frequencies. For ETTh1, ETTh2, ETTm1, and ETTm2, the first 12 months are used for training, the next 4 months for validation, and the final 4 months for testing. Weather and Exchange use chronological 70%/10%/20% partitions, whereas each PEMS dataset uses chronological 60%/20%/20% partitions. Observations are never shuffled across partition boundaries. Validation and test loaders prepend only the preceding L observations as historical context; all forecast targets remain inside their assigned partition. Normalization statistics are estimated exclusively from the training partition and then applied unchanged to validation and test data.

All experiments use multivariate inputs and predict all variables, with a sliding-window stride of one. For PEMS, we use the first traffic-flow feature from each public array and treat sensor locations as variables. ETT and Weather use a look-back length of $L = 96$ and prediction horizons $H \in \{96, 192, 336, 720\}$; Exchange uses $L = 96$ and $H \in \{48, 96, 144, 192\}$; and PEMS uses $L = 12$ and $H \in \{12, 24, 48, 96\}$. Each horizon is trained and evaluated as a separate forecasting task.

D.2 Model and Optimization Settings

Model implementation. We implement EchoKoop in PyTorch within the Time-Series-Library forecasting pipeline. The causal encoder in Appendix C.1 has five convolutional layers with kernel size 3 and dilations $1, 2, 4, 8, 16$. Time covariates and decoder inputs are accepted for interface compatibility but are not used by the EchoKoop forward pass. Each run fixes its operator family and active resolution branches at model construction. Table 7 gives the optimization settings for the standard forecasting benchmarks.

Normal-operator configuration. Unless explicitly identified as EchoKoop-R, the reported EchoKoop results use the normal-operator bank with $Q_a = 6$ members at each of the three resolutions $\mathcal{S} = \{1, 2, 4\}$. The operators are stored as persistent non-trainable buffers. The selector described in Appendix C.5 is initialized from the profile $(0.025, 0.075, 0.30, 0.25, 0.20, 0.15)$. The uniform alternative initializes all six weights to $1/6$. Both remain learnable, and the observed forecasting results are insensitive to this initialization choice.

Reservoir comparison. EchoKoop-R uses $Q_r = 6$ sparse operators per branch with nonzero density $c = 0.1$, constructed as in Appendix C.9. The expected mean nonzero degree is $cd_r = 9.6$ for $d_r = 96$ and 12.8 for $d_r = 128$. The target radius of each operator matches that of its normal-bank counterpart, and the reservoir matrices are likewise persistent non-trainable buffers.

Randomness control. The normal transition matrices are deterministic functions of the prescribed frequencies, half-lives, and resolutions. Their fixed orthogonal mixing matrices use branch-local CPU generators with seeds $4201 + b$, where $b \in \{0, 1, 2\}$ is the zero-based resolution-branch index; these matrices are therefore identical across the three training seeds. For EchoKoop-R, experiment seed $u \in \{0, 1, 2\}$ sets the reservoir base seed to $14201 + u$. Branch b adds an offset $10000b$, member q adds $1009q$, and the learned temporal-weight initialization uses the corresponding branch

Table 6: Dataset-dependent configurations of EchoKoop.

Dataset family	L/ℓ_{dec}	Horizons H	d_m	Half-lives $\{\tau_q\}$	Batch
ETT/Weather	$96/48$	$96, 192,$ $336, 720$	96	$24, 96, 360,$ $1440, 5760, 23040$	32
Exchange	$96/48$	$48, 96,$ $144, 192$	96	$24, 96, 360,$ $1440, 5760, 23040$	32
PEMS	$12/6$	$12, 24,$ $48, 96$	128	$3, 12, 48,$ $192, 768, 3072$	$8, 8, 4, 16$ †

Here, ℓ_{dec} denotes the decoder label length. All configurations use lift width $d_\ell = 256$, decoder width $d_d = 96$, native resolutions $\mathcal{S} = \{1, 2, 4\}$, $Q_a = 6$ operators per branch, and frequency scales $\{1, 1, 1\}$. The automatic attention backend is used for ETT, Weather, and Exchange, whereas the math backend is used for PEMS. † Batch sizes correspond to PEMS03, PEMS04, PEMS07, and PEMS08, respectively.

Table 7: Optimization and runtime settings for EchoKoop on the standard forecasting benchmarks. Physical-field training is specified in Appendix J.2.

Hyperparameter	Value
Operator mode	Fixed normal
Selector	Learned per modal pair
Selector initialization	Prescribed mixture; uniform in stated ablation
Decoder normalization	LayerNorm
Cross-branch correction	Enabled
Dropout	0.1
Output-head initialization gain	5×10^{-3}
Training objective	Huber, $\delta_{\mathrm{H}} = 0.15$
Validation objective	MSE
Optimizer	Adam
Initial learning rate	5×10^{-5}
Learning-rate schedule	0.9 decay per epoch after epoch 3
Weight decay	10^{-5}
Gradient clipping norm	1.0
Maximum epochs	25
Early-stopping patience	3
Random seeds	$\{0, 1, 2\}$
Mixed precision	Disabled
PyTorch version	2.4.0
PyTorch CUDA build	11.8
GPU model	PPU-ZW810E

seed plus 97. Consequently, reservoir operators are independently redrawn across experiment seeds, while all remaining trainable layers follow the same experiment seed.

Decoding and correction. Each branch decoder uses LayerNorm, a width-96 projection, two kernel-size-5 temporal residual blocks with dilations 1 and 2, and a bias-free output projection. The cross-scale correction uses width 96, six-head attention, a 512-dimensional feed-forward layer, learned-query pooling, and kernel-size-5 residual blocks with dilations $1, 2, 4, 8$. Its final projection is initialized to zero, as described in Appendix C.6.

Table 6 reports the configurations used in the reported EchoKoop experiments. These values describe the experimental profiles rather than the generic command-line defaults.

Writing k for the one-based epoch index and $\mathrm{lr}_0 = 5 \times 10^{-5}$, the implemented schedule is

$$\mathrm{lr}_k = \begin{cases} \mathrm{lr}_0, & k \le 3, \\ \mathrm{lr}_0 \ 0.9^{k-3}, & k > 3. \end{cases}$$

Table 8: Public code sources used for the forecasting baselines.

Method	Repository	Revision
PatchTST, iTransformer TimesNet, DLinear	https://github.com/thuml/Time-Series-Library	—
Koopa	https://github.com/thuml/Time-Series-Library	—
KUMA	https://github.com/SuperBearcafedo/Review/tree/KUMA-Official	8ed21081
K^2VAE	https://github.com/decisionintelligence/K2VAE	19c602a8
SKOLR	https://github.com/networkslab/SKOLR	650de539
FlowMixer	https://github.com/FaresBMehouachi/FlowMixer	91566591
AIKAE	https://github.com/anthony-frion/AIKAE	fea1282a

D.3 Baseline Code Sources and Integration Notes

Table 8 lists the public repositories used for the forecasting baselines. Unless noted otherwise, the model architectures and training objectives follow the corresponding implementations. All scores are obtained locally under the evaluation protocol of this paper.

We add minor numerical-stability safeguards to K^2VAE for hardware backends without native CUDA support and make minor interface changes to integrate SKOLR into the shared forecasting pipeline. These changes do not alter either model's architecture or training objective.

D.4 Evaluation Metrics

Each variable is standardized using statistics estimated exclusively from the training partition. Predictions are evaluated in this standardized space; inverse transformation is disabled. Let N denote the number of test windows, H the forecast horizon, and C the number of forecast variables. We report mean squared error (MSE) and mean absolute error (MAE):

$$\mathrm{MSE} = \frac{1}{NHC} \sum_{i=1}^{N} \sum_{h=1}^{H} \sum_{c=1}^{C} \left(\widehat{y}_{h,c}^{(i)} - y_{h,c}^{(i)} \right)^2,$$

and

$$\mathrm{MAE} = \frac{1}{NHC} \sum_{i=1}^{N} \sum_{h=1}^{H} \sum_{c=1}^{C} \left| \widehat{y}_{h,c}^{(i)} - y_{h,c}^{(i)} \right|.$$

Lower values indicate better forecasting performance. Each configuration is run with three random seeds, and aggregate results are computed across these runs. Trainable parameters are counted only for tensors with `requires_grad=True`. Persistent operator-buffer elements and, for EchoKoop-R, the realized nonzero connection density are reported separately; they are not included in the trainable-parameter count.

E Complete Forecasting Results

Table 9 reports the per-horizon results underlying the dataset-level comparisons in Section 4.1.

F Cross-Architecture Dynamics Transfer

This appendix specifies the intervention used in Section 4.3. The main text reports aggregate forecasting outcomes, whereas this section defines which recurrent operator is replaced in each host architecture and how rollout diagnostics are computed.

F.1 Transfer Intervention

We call the replacement of a host's targeted recurrent operator with a prescribed normal-operator bank the *Frozen Dynamics* intervention. For each host, we replace only the operator recursively

Table 9: Main forecasting results. We report MSE/MAE for each dataset and prediction horizon. Lower is better. EchoKoop and every baseline are rerun in the shared framework, and each entry is averaged over three runs with seeds $\{0, 1, 2\}$. The best results are in bold and the second-best results are underlined. Settings without valid results for all three seeds are denoted by "– / –".

Dataset	Horizon	EchoKoop	Koopa	KUMA	K2VAE	SKOLR	FlowMixer	AIKAE	PatchTST	iTransformer	TimesNet	DLinear
ETTh1	96	0.385 / 0.407	0.387 / 0.404	0.390 / 0.407	0.418 / 0.429	0.386 / 0.404	0.384 / **0.393**	0.397 / 0.405	**0.379** / 0.399	0.388 / 0.406	0.415 / 0.430	0.386 / 0.398
	192	**0.422** / 0.431	0.436 / 0.431	0.438 / 0.436	0.500 / 0.472	0.433 / 0.432	0.436 / **0.425**	0.444 / 0.436	0.427 / 0.431	0.443 / 0.438	0.467 / 0.458	0.447 / 0.442
	336	**0.447** / 0.452	0.493 / 0.459	0.473 / 0.456	0.520 / 0.486	0.482 / 0.460	0.481 / **0.447**	0.490 / 0.463	0.474 / 0.459	0.490 / 0.462	0.504 / 0.478	0.507 / 0.483
	720	0.500 / 0.500	0.518 / 0.488	0.500 / **0.484**	0.522 / 0.498	**0.494** / 0.486	– / –	0.562 / 0.512	0.521 / 0.504	0.510 / 0.494	0.511 / 0.493	0.512 / 0.507
	Avg	**0.439** / 0.448	0.459 / **0.446**	0.450 / 0.446	0.490 / 0.471	0.449 / 0.446	– / –	0.473 / 0.454	0.450 / 0.448	0.458 / 0.450	0.474 / 0.465	0.463 / 0.458
ETTh2	96	0.301 / 0.354	0.307 / 0.354	0.321 / 0.360	0.329 / 0.368	0.294 / 0.345	**0.289** / **0.338**	0.302 / 0.347	– / –	0.299 / 0.350	0.327 / 0.368	0.320 / 0.376
	192	0.387 / 0.408	0.381 / 0.401	0.383 / 0.400	0.416 / 0.418	**0.376** / 0.399	0.379 / **0.392**	0.377 / 0.396	0.380 / 0.403	0.379 / 0.398	0.405 / 0.415	0.525 / 0.494
	336	0.440 / 0.449	0.426 / 0.438	0.426 / 0.435	0.457 / 0.450	0.421 / 0.432	0.426 / **0.431**	0.428 / 0.436	**0.419** / 0.434	0.424 / 0.432	0.440 / 0.443	0.511 / 0.493
	720	0.437 / 0.458	0.438 / 0.450	0.436 / 0.449	0.463 / 0.464	0.437 / 0.453	– / –	0.434 / 0.449	0.438 / 0.456	0.434 / 0.450	**0.431** / **0.447**	0.644 / 0.576
	Avg	0.391 / 0.417	0.388 / 0.411	0.392 / 0.411	0.416 / 0.425	**0.382** / 0.407	– / –	0.385 / **0.407**	– / –	0.384 / 0.408	0.401 / 0.418	0.500 / 0.485
ETTm1	96	**0.324** / **0.359**	0.333 / 0.370	0.341 / 0.373	0.350 / 0.381	0.349 / 0.380	0.345 / 0.373	0.328 / 0.364	0.325 / 0.366	0.341 / 0.376	0.339 / 0.379	0.345 / 0.373
	192	**0.358** / **0.383**	0.370 / 0.389	0.399 / 0.399	0.411 / 0.409	0.396 / 0.403	0.390 / 0.394	0.371 / 0.388	0.371 / 0.391	0.382 / 0.395	0.408 / 0.408	0.381 / 0.391
	336	**0.386** / **0.402**	0.402 / 0.411	0.413 / 0.414	0.479 / 0.441	0.431 / 0.424	0.421 / 0.414	0.407 / 0.414	0.398 / 0.407	0.417 / 0.418	0.427 / 0.428	0.414 / 0.416
	720	**0.431** / **0.430**	– / –	0.491 / 0.454	0.573 / 0.488	0.484 / 0.453	0.487 / 0.448	0.468 / 0.450	0.461 / 0.446	0.491 / 0.458	0.485 / 0.456	0.472 / 0.450
	Avg	**0.375** / **0.394**	– / –	0.411 / 0.410	0.453 / 0.430	0.415 / 0.415	0.411 / 0.407	0.394 / 0.404	0.389 / 0.403	0.408 / 0.412	0.415 / 0.418	0.403 / 0.408
ETTm2	96	0.182 / 0.269	**0.179** / **0.261**	0.179 / 0.263	0.195 / 0.272	0.186 / 0.271	0.183 / 0.265	0.185 / 0.267	0.181 / 0.265	0.186 / 0.271	0.188 / 0.267	0.192 / 0.290
	192	0.255 / 0.318	**0.243** / **0.304**	0.248 / 0.307	0.264 / 0.318	0.249 / 0.309	0.250 / 0.306	0.252 / 0.311	0.247 / 0.308	0.252 / 0.313	0.260 / 0.310	0.276 / 0.352
	336	0.319 / 0.358	**0.305** / **0.344**	0.309 / 0.347	0.316 / 0.348	0.310 / 0.348	0.312 / 0.345	0.307 / 0.344	0.311 / 0.349	0.315 / 0.352	0.314 / 0.346	0.368 / 0.411
	720	0.433 / 0.424	**0.399** / **0.398**	0.408 / 0.404	0.413 / 0.402	0.408 / 0.403	0.409 / 0.401	0.405 / 0.402	0.417 / 0.411	0.413 / 0.407	0.424 / 0.407	0.475 / 0.476
	Avg	0.297 / 0.342	**0.282** / **0.327**	0.286 / 0.330	0.297 / 0.335	0.288 / 0.333	0.289 / 0.329	0.287 / 0.331	0.289 / 0.333	0.292 / 0.336	0.297 / 0.333	0.328 / 0.382
Weather	96	0.180 / 0.223	0.160 / 0.207	**0.158** / **0.204**	0.174 / 0.221	0.181 / 0.225	0.169 / 0.215	0.172 / 0.218	0.174 / 0.215	0.176 / 0.216	0.171 / 0.221	0.195 / 0.255
	192	0.224 / 0.268	0.209 / **0.250**	**0.208** / 0.250	0.226 / 0.265	0.217 / 0.256	0.216 / 0.256	0.224 / 0.261	0.220 / 0.256	0.225 / 0.258	0.227 / 0.265	0.236 / 0.295
	336	0.309 / 0.330	0.267 / **0.292**	**0.265** / 0.293	0.278 / 0.300	0.275 / 0.300	0.275 / 0.299	0.282 / 0.302	0.279 / 0.297	0.283 / 0.300	0.286 / 0.307	0.282 / 0.333
	720	0.372 / 0.370	**0.346** / **0.345**	0.349 / 0.348	0.371 / 0.357	0.351 / 0.349	0.353 / 0.350	0.365 / 0.356	0.355 / 0.348	0.362 / 0.352	0.360 / 0.355	0.347 / 0.384
	Avg	0.271 / 0.298	0.246 / **0.274**	**0.245** / 0.274	0.262 / 0.286	0.256 / 0.283	0.253 / 0.280	0.261 / 0.284	0.257 / 0.279	0.262 / 0.282	0.261 / 0.287	0.265 / 0.317
Exchange	48	0.049 / 0.156	0.047 / 0.151	0.047 / 0.150	0.077 / 0.193	0.045 / 0.146	0.043 / 0.144	0.048 / 0.151	0.044 / 0.144	0.046 / 0.150	0.061 / 0.177	**0.042** / **0.142**
	96	0.094 / 0.218	0.094 / 0.212	0.091 / 0.210	0.130 / 0.250	0.087 / 0.206	0.084 / 0.203	0.099 / 0.220	0.089 / 0.207	0.088 / 0.209	0.117 / 0.244	**0.078** / **0.199**
	144	0.141 / 0.271	0.141 / 0.263	0.138 / 0.262	0.185 / 0.300	0.133 / 0.257	0.129 / 0.252	0.146 / 0.269	0.134 / 0.257	0.132 / 0.257	0.165 / 0.293	**0.115** / **0.245**
	192	0.189 / 0.314	0.195 / 0.315	0.193 / 0.312	0.218 / 0.333	0.189 / 0.311	0.172 / 0.295	0.192 / 0.311	0.189 / 0.308	0.180 / 0.303	0.214 / 0.335	**0.157** / **0.293**
	Avg	0.118 / 0.240	0.119 / 0.235	0.117 / 0.234	0.153 / 0.269	0.114 / 0.230	0.107 / 0.224	0.121 / 0.238	0.114 / 0.229	0.112 / 0.230	0.139 / 0.262	**0.098** / **0.220**
PEMS03	12	**0.066** / **0.172**	0.085 / 0.192	0.069 / 0.173	0.075 / 0.176	0.113 / 0.224	0.097 / 0.206	0.127 / 0.243	0.112 / 0.223	0.086 / 0.193	0.080 / 0.185	0.147 / 0.271
	24	**0.088** / **0.197**	0.143 / 0.252	0.108 / 0.216	0.116 / 0.217	0.226 / 0.315	0.190 / 0.289	0.255 / 0.345	0.225 / 0.315	0.154 / 0.258	0.130 / 0.236	0.261 / 0.373
	48	**0.138** / **0.247**	0.330 / 0.388	0.241 / 0.312	0.257 / 0.316	0.530 / 0.495	0.466 / 0.464	0.593 / 0.544	0.526 / 0.495	0.332 / 0.384	0.287 / 0.348	0.494 / 0.542
	96	**0.246** / **0.333**	0.905 / 0.665	0.608 / 0.494	0.644 / 0.500	1.191 / 0.798	1.137 / 0.781	1.268 / 0.849	1.164 / 0.786	0.771 / 0.608	0.632 / 0.538	0.714 / 0.679
	Avg	**0.135** / **0.237**	0.366 / 0.374	0.257 / 0.299	0.273 / 0.302	0.515 / 0.458	0.473 / 0.435	0.561 / 0.495	0.507 / 0.455	0.336 / 0.361	0.282 / 0.327	0.404 / 0.466
PEMS04	12	0.083 / 0.189	0.093 / 0.203	**0.082** / 0.188	0.082 / **0.184**	0.125 / 0.237	0.106 / 0.217	0.138 / 0.258	0.124 / 0.236	0.105 / 0.212	0.095 / 0.204	0.178 / 0.305
	24	**0.099** / **0.209**	0.145 / 0.259	0.116 / 0.228	0.114 / 0.220	0.241 / 0.331	0.192 / 0.296	0.275 / 0.365	0.239 / 0.330	0.178 / 0.279	0.148 / 0.256	0.297 / 0.406
	48	**0.131** / **0.243**	0.307 / 0.379	0.201 / 0.305	0.196 / 0.287	0.564 / 0.520	0.448 / 0.461	0.637 / 0.573	0.552 / 0.511	0.379 / 0.414	0.315 / 0.369	0.547 / 0.575
	96	**0.195** / **0.306**	0.727 / 0.602	0.359 / 0.420	0.431 / 0.418	1.270 / 0.836	1.141 / 0.787	1.358 / 0.888	1.245 / 0.818	0.784 / 0.628	0.620 / 0.544	0.812 / 0.735
	Avg	**0.127** / **0.237**	0.318 / 0.361	0.190 / 0.285	0.206 / 0.277	0.550 / 0.481	0.472 / 0.440	0.602 / 0.521	0.540 / 0.474	0.362 / 0.383	0.295 / 0.343	0.459 / 0.505
PEMS07	12	0.066 / 0.166	0.081 / 0.187	**0.066** / **0.163**	0.076 / 0.169	0.104 / 0.215	0.089 / 0.199	0.114 / 0.227	0.100 / 0.208	0.081 / 0.187	0.078 / 0.183	0.144 / 0.272
	24	**0.084** / **0.186**	0.142 / 0.249	0.112 / 0.205	0.132 / 0.210	0.224 / 0.312	0.181 / 0.284	0.256 / 0.339	0.208 / 0.297	0.153 / 0.256	0.130 / 0.233	0.268 / 0.378
	48	**0.115** / **0.220**	0.336 / 0.388	0.241 / 0.282	0.272 / 0.298	0.567 / 0.506	0.466 / 0.462	0.616 / 0.536	0.523 / 0.475	0.356 / 0.396	0.270 / 0.332	0.514 / 0.546
	96	**0.177** / **0.275**	0.681 / 0.580	0.457 / 0.396	0.525 / 0.417	1.264 / 0.818	1.079 / 0.753	1.319 / 0.843	1.130 / 0.753	0.741 / 0.603	0.525 / 0.490	0.743 / 0.680
	Avg	**0.111** / **0.212**	0.310 / 0.351	0.219 / 0.262	0.251 / 0.274	0.540 / 0.463	0.454 / 0.425	0.576 / 0.486	0.490 / 0.433	0.333 / 0.361	0.251 / 0.310	0.417 / 0.469
PEMS08	12	**0.078** / **0.182**	0.098 / 0.206	0.080 / 0.183	0.088 / 0.188	0.121 / 0.233	0.106 / 0.214	0.131 / 0.250	0.116 / 0.226	0.099 / 0.206	0.098 / 0.204	0.175 / 0.303
	24	**0.109** / **0.216**	0.168 / 0.271	0.127 / 0.231	0.135 / 0.231	0.238 / 0.328	0.198 / 0.295	0.268 / 0.361	0.226 / 0.318	0.182 / 0.282	0.164 / 0.264	0.300 / 0.406
	48	**0.184** / **0.280**	0.384 / 0.418	0.275 / 0.341	0.282 / 0.330	0.578 / 0.526	0.479 / 0.473	0.642 / 0.572	0.528 / 0.494	0.449 / 0.451	0.351 / 0.391	0.575 / 0.580
	96	**0.404** / **0.411**	0.918 / 0.683	0.616 / 0.523	0.643 / 0.514	1.374 / 0.864	1.304 / 0.841	1.446 / 0.903	1.202 / 0.791	1.103 / 0.744	0.776 / 0.614	0.876 / 0.739
	Avg	**0.194** / **0.272**	0.392 / 0.395	0.275 / 0.320	0.287 / 0.316	0.578 / 0.488	0.522 / 0.456	0.622 / 0.522	0.518 / 0.457	0.458 / 0.421	0.347 / 0.368	0.482 / 0.507

applied across forecast steps. All remaining host-specific encoders, decoders, losses, optimization settings, data splits, and observation contexts are retained. The $Q = 1$ and $Q = 6$ variants are trained independently. For $Q = 6$, branch outputs are combined using learned per-pair selection.

Koopa. Koopa estimates a sample-dependent DMD transition in each time-varying block and recursively applies this transition to forecast the time-varying component. We replace this rollout in all three time-varying blocks with the fixed operator bank $\{\mathbf{K}_q\}_{q=1}^{Q}$. The final segment latent is copied to all branches before propagation. We retain Koopa's learned time-invariant latent map because it is applied once to produce a full forecast, rather than recursively at a defined temporal step. Replacing this one-shot map would therefore extend the intervention beyond recurrent latent dynamics.

SKOLR. The native SKOLR recurrence is

$$\mathbf{h}_t = \mathbf{B}\mathbf{x}_t + \mathbf{K}\mathbf{h}_{t-1}.$$

Under Frozen Dynamics, SKOLR retains the learned input map $\mathbf{B}$ and replaces only the autonomous transition:

$$\mathbf{h}_{t,q} = \mathbf{B}\mathbf{x}_t + \mathbf{K}_q\mathbf{h}_{t-1,q}, \qquad q = 1, \ldots, Q.$$

After the observed context, each branch is propagated autonomously by $\mathbf{h}_{t,q} = \mathbf{K}_q\mathbf{h}_{t-1,q}$.

K^2VAE. For K^2VAE, we replace only the Koopman transition acting on patch latents. The original history-chunked matrix-power schedule is retained. The auxiliary Transformer and the Kalman state–covariance recurrence remain unchanged, because neither is the recurrent patch-latent Koopman transition targeted by the intervention.

Table 10: Changes in trainable parameters caused by replacing the native recurrent transition. Frozen operator matrices are registered as non-trainable buffers and are therefore excluded. Long denotes ETTh1, ETTm1, Weather, and Exchange.

Protocol	Host	Native	$Q = 1$	$\Delta_{Q=1}$	$Q = 6$	$\Delta_{Q=6}$
Long	Koopa	0	0	0	1,152	$+1,152$
	SKOLR	131,072	0	$-131,072$	1,536	$-129,536$
	K2VAE	16,384	0	$-16,384$	384	$-16,000$
PEMS	Koopa	0	0	0	576	$+576$
	SKOLR	8,192	0	$-8,192$	384	$-7,808$
	K2VAE	1,024	0	$-1,024$	96	-928

This design tests transfer of frozen recurrent dynamics rather than the claim that every learned linear map in a host architecture should be interpreted as a temporal Koopman operator.

F.2 Parameterization and Capacity of Transferred Dynamics

Our transfer experiment changes only the parameterization of the targeted recurrent operator. All encoders, decoders, decomposition modules, normalization layers, auxiliary components, and training objectives are identical between each original host and its Frozen Dynamics variant. Consequently, the difference in the total number of trainable parameters is exactly the difference introduced by the replaced operator.

Let d denote the latent-state dimension, B_{host} the number of recurrent transition modules in the host, and Q the number of structured operators. Each transferred normal operator is frozen and registered as a non-trainable buffer. In the multiple-operator setting, the only trainable component introduced by Frozen Dynamics is a per-modal-pair selector with

$$N_{\mathrm{param}}^{\mathrm{sel}} = B_{\mathrm{host}} \frac{d}{2} Q$$

trainable logits. For $Q = 1$, the selector is removed and the transferred transition contains no trainable parameters.

The native time-varying DMD transition in Koopa is fitted independently from each input sample and therefore contains no trainable parameters. Frozen Dynamics with $Q = 6$ adds only the selector parameters in this case. In contrast, SKOLR and K^2VAE originally employ trainable dense recurrent matrices. Frozen Dynamics removes these matrices and replaces them with a substantially smaller selector. The resulting host- and protocol-specific changes in trainable parameters are summarized in Table 10.

F.3 Transition Statistics

For each targeted recurrent operator $\mathbf{K}$, we report its spectral radius,

$$\rho(\mathbf{K}) = \max_i |\lambda_i(\mathbf{K})|,$$

the Henrici departure from normality,

$$d_{\mathrm{H}}(\mathbf{K}) = \frac{\sqrt{\|\mathbf{K}\|_{\mathrm{F}}^2 - \sum_i |\lambda_i(\mathbf{K})|^2}}{\|\mathbf{K}\|_{\mathrm{F}}},$$

and a sampled finite-horizon relative power gain. Define $\mathcal{N} = \{1, 2, 4, 8, 16, 32\}$; for $\rho(\mathbf{K}) > 0$,

$$G_{\mathrm{rel}}^{(\mathcal{N})}(\mathbf{K}) = \max_{n \in \mathcal{N}} \frac{\|\mathbf{K}^n\|_2}{\rho(\mathbf{K})^n}.$$

We additionally record $\kappa_2(\mathbf{V})$, where $\mathbf{K} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1}$ when the eigendecomposition is numerically reliable.

Table 11: Host-wise diagnostics of the recurrent operators targeted by Frozen Dynamics. Each run is first reduced to the maximum over all recorded matrices having the targeted recurrent role. Entries then report the median $[P_{95}]$ of these per-run maxima over 36 dataset–horizon–seed runs (six datasets, two horizons, and three seeds). The $n_{\rho(\mathbf{K})>1}$ column counts runs whose targeted operator has spectral radius greater than one. Koopa's time-invariant maps and K2VAE's Kalman transition are excluded.

Host	Transition	Q	$n_{\rho(\mathbf{K})>1}/36$	$\rho^{\mathrm{run}}_{\max}(\mathbf{K})$	$\kappa^{\mathrm{run}}_{2,\max}(\mathbf{V})$	$d^{\mathrm{run}}_{\mathrm{H},\max}$	$G^{(\mathcal{N}),\mathrm{run}}_{\mathrm{rel},\max}$
Koopa	Native DMD	–	13/36	0.9532 [1.3871]	9.64×10^{2} $[1.03\times10^{4}]^{\dagger}$	0.9213 [0.9933]	7.93 [82.52]
	Frozen Dynamics	1	0/36	0.9548 [0.9923]	1.00^{*}	0^{*}	1.00
	Frozen Dynamics	6	0/36	0.9993 [0.9999]	1.00^{*}	0^{*}	1.00
SKOLR	Native learned Koopman	–	14/36	0.9376 [1.2311]	48.9 [409.2]	0.2886 [0.6508]	1.72 [5.16]
	Frozen Dynamics	1	0/36	0.9402 [0.9923]	1.00^{*}	0^{*}	1.00
	Frozen Dynamics	6	0/36	0.9990 [0.9999]	1.00^{*}	0^{*}	1.00
K2VAE	Native learned Koopman	–	35/36	1.0462 [1.1658]	7.11 [17.35]	0.1076 [0.1828]	1.63 [3.54]
	Frozen Dynamics	1	0/36	0.9548 [0.9923]	1.00^{*}	0^{*}	1.00
	Frozen Dynamics	6	0/36	0.9993 [0.9999]	1.00^{*}	0^{*}	1.00

† All 864 numerical eigenvector-condition estimates for Koopa's sample-fitted DMD matrices were flagged as unreliable because of repeated or clustered eigenvalues; they are therefore qualitative only. * The structured damped-rotation operators are normal by construction, so $\kappa_2(\mathbf{V}) = 1$ and $d_{\mathrm{H}} = 0$ analytically. The largest observed numerical Henrici residual was 1.51×10^{-8}.

For $Q = 6$, diagnostics are evaluated on individual branch operators $\mathbf{K}_q$, rather than on an effective matrix fitted after branch selection. Each prescribed damped-rotation operator is normal by construction. Consequently,

$$d_{\mathrm{H}}(\mathbf{K}_q) = 0, \qquad G^{(\mathcal{N})}_{\mathrm{rel}}(\mathbf{K}_q) = 1$$

in exact arithmetic. Eigenvector condition numbers are treated as auxiliary diagnostics because repeated or clustered eigenvalues can make their numerical estimates unreliable.

The reported host–Frozen Dynamics comparisons include only the targeted recurrent operator: Koopa's time-varying DMD map, SKOLR's recurrent Koopman map, and the K^2VAE patch-latent Koopman map. Koopa's one-shot invariant map and the K^2VAE Kalman transition are excluded. Diagnostic tensors are detached from the optimization graph and do not affect training.

Table 11 summarizes host-wise transition diagnostics. Figures 5 and 6 relate power gain to forecasting error and show its rollout envelope, respectively. Figure 7 shows representative spectral placements and training-time drift. These diagnostics are descriptive: they do not establish causal effects on forecasting accuracy or identify the ground-truth Koopman operator.

Rollout diagnostics. The operator diagnostics provide a possible explanation for the host-dependent transfer results. Koopa's native DMD transitions exhibit strong non-normality, with a median normalized Henrici departure of 0.9213, and substantial transient amplification, with a median maximum relative power gain of 7.93 and a 95th percentile of 82.52. In contrast, the structured frozen operators are normal by construction and satisfy $d_{\mathrm{H}} = 0$ and $G^{(\mathcal{N})}_{\mathrm{rel}} = 1$. This reduction is consistent with Koopa's stronger long-horizon gains. However, the mixed K^2VAE results show that controlled rollout behavior alone does not guarantee improved forecasting accuracy.

Spectra and training-time drift. Figure 7 compares representative spectral placement with the drift of native recurrent operators during training. Diagnostics are recorded at epoch 1, every five epochs, and the final early-stopping epoch; drift is normalized by each run's epoch-1 value. The drift panels retain Native runs with finite values at epochs 1, 5, and 10: $n = 20$ for Koopa, $n = 11$ for SKOLR, and $n = 15$ for K^2VAE. Frozen Dynamics is omitted from these panels because its operators are fixed. The spectral panels characterize placement rather than numerical rank, and the drift panels are descriptive diagnostics rather than evidence of causal architecture effects or Koopman-operator identification. The local drift axes differ by host and therefore support only within-host comparisons.

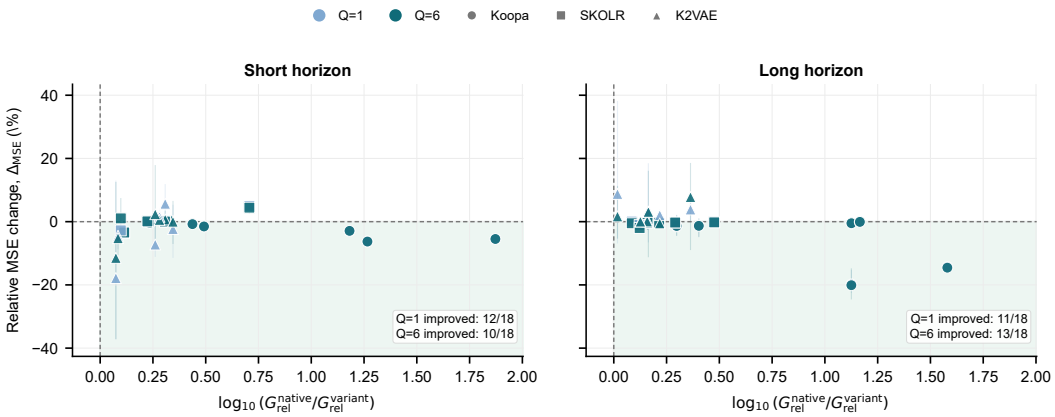

Figure 5: Stability–forecasting trade-off after replacing each native recurrent operator through Frozen Dynamics. Each point represents one host–dataset task. Forecasting MSE is first averaged over seeds 0, 1, and 2, after which the relative change is computed against the corresponding Native host. The horizontal coordinate is the median paired log-scale reduction in maximum relative power gain over the same seeds; positive values indicate reduced transient amplification. Vertical lines span the seed-level relative MSE changes. The shaded lower-right region denotes simultaneous reduction in transient amplification and improvement in forecasting accuracy. The improvement counts for $Q = 1$ and $Q = 6$ are computed independently against each host model's own Native baseline.

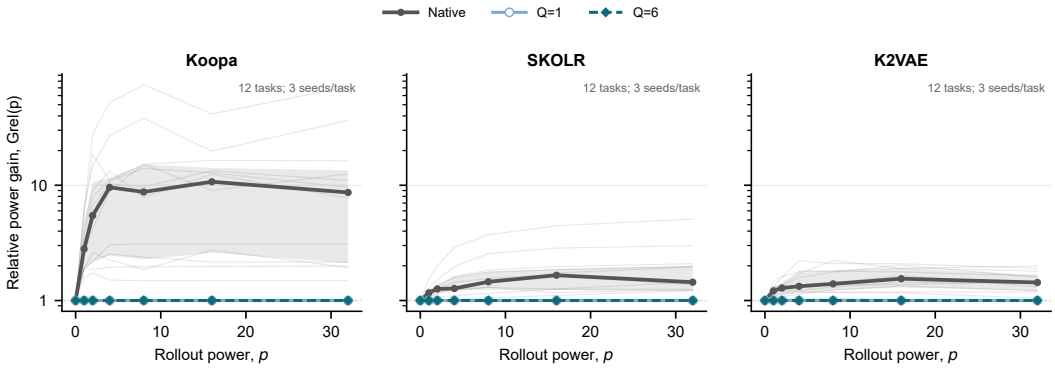

Figure 6: Relative power-gain envelopes of the recurrent operators targeted by Frozen Dynamics. Thin gray curves denote individual Native dataset–horizon tasks after taking the pointwise median over seeds 0, 1, and 2, yielding 12 tasks for each host architecture. Thick curves report the median across these tasks, while the gray shaded region shows the interquartile range of the native operators. The $Q = 1$ and $Q = 6$ Frozen Dynamics variants remain at the normal-operator bound $G_{\mathrm{rel}}^{(\mathcal{N})} = 1$. All host-model panels share the same horizontal and vertical axes.

F.4 Mechanism Tests for the Stability–Misspecification Trade-off

Table 11 shows that the Frozen Dynamics operator bank suppresses non-normal transient amplification, yet forecast accuracy varies across hosts and tasks. Its benefit also depends on whether the prescribed decay and frequency modes suit the task. We examine this balance by extending the decay-scale bank, shifting its frequency grid, and adding a low-rank correction.

All three tests use seeds $0, 1, 2$ and retain the host protocols. Together, they comprise 72 runs—9 for decay-scale coverage, 54 for frequency alignment, and 9 for low-rank adaptivity—and 24 direct contrast groups. For two MSE values a and b, with b as the reference, define

$$\Delta_{\mathrm{MSE}}(a, b) = 100\frac{a - b}{b}. \tag{14}$$

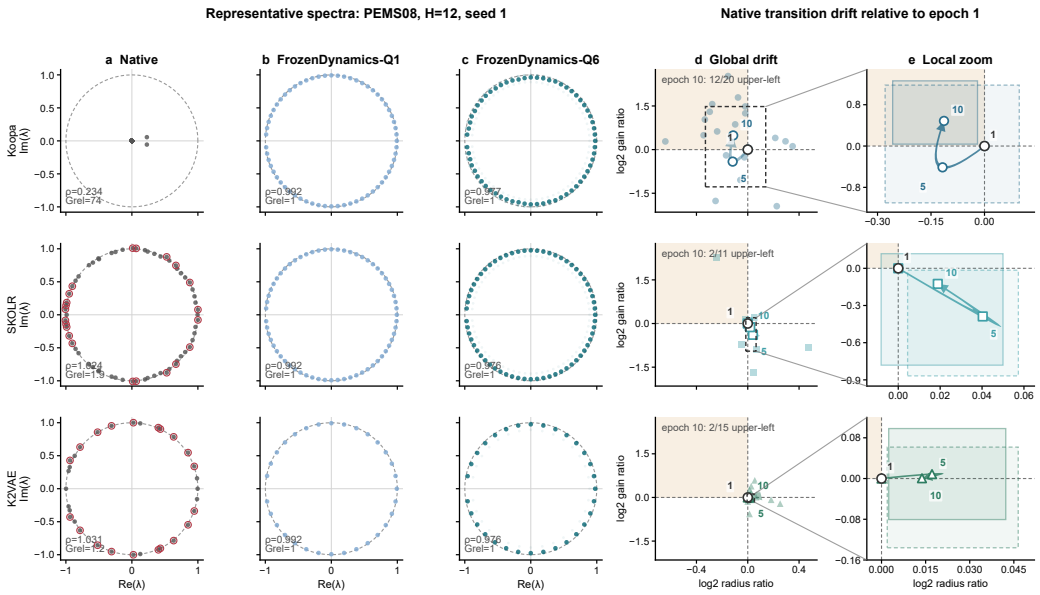

Figure 7: Spectral placement and training-time drift of recurrent operators targeted by Frozen Dynamics (rows: Koopa, SKOLR, and K^2VAE). **a–c**, PEMS08 spectra at $H = 12$, seed 1. Native shows the targeted operator with maximum relative power gain; $Q = 1$ shows the single frozen spectrum. For $Q = 6$, the plotted points are one-step selector-weighted modal locations and faint markers are bank candidates; they are not the spectrum of one operator governing the combined multi-step rollout. Dashed circles mark the unit circle; red outlines mark points outside it. **d–e**, Native-operator drift relative to epoch 1. The horizontal and vertical axes show $\log_2(\rho(\mathbf{K}_e)/\rho(\mathbf{K}_1))$ and $\log_2(G^{(\mathcal{N})}_{\mathrm{rel}}(\mathbf{K}_e)/G^{(\mathcal{N})}_{\mathrm{rel}}(\mathbf{K}_1))$, respectively. **d**, Global view on common axes, with median checkpoints at epochs 1, 5, and 10 and dashed boxes linked to the local views. **e**, Host-specific views with marginal interquartile-range rectangles at epochs 5 (dashed) and 10 (solid). Arrows guide the observed median checkpoints, not continuous trajectories; the shaded upper-left quadrant marks lower spectral radius but higher finite-horizon relative power gain.

A negative value favors the first condition. We calculate this change for each paired comparison and summarize the results in Table 3 of Section 4.3. Because each individual condition contains three seeds, these experiments are interpreted as mechanism probes rather than formal significance tests.

Decay-scale coverage. The native SKOLR transition on Exchange at $H = 48$ has a diagnostic reference decay scale of approximately 13.1 raw samples, whereas the original transferred normal-operator bank uses

$$\mathcal{T}_{Q=6} = \{24, 96, 360, 1440, 5760, 23040\}.$$

We therefore construct an augmented bank

$$\mathcal{T}_{Q=8} = \{6, 12, 24, 96, 360, 1440, 5760, 23040\},$$

which retains every original operator while adding two shorter decay scales.

For host block b, let Δ_b denote its operator step and $\alpha^{(b)}_{p,q}$ the selector weight for modal pair p. Define the member radii

$$\rho_{b,q} = \exp\left(-\frac{\log 2}{\tau_q}\Delta_b\right),$$

and summarize the selected decay scale for each block and modal pair through

$$\bar{\rho}_{b,p} = \sum_q \alpha^{(b)}_{p,q}\rho_{b,q}, \qquad \tau^{(b,p)}_{\mathrm{eff}} = -\frac{\Delta_b \log 2}{\log \bar{\rho}_{b,p}}.$$

Adding $\tau = 6$ and 12 changes the mean, taken over host blocks, modal pairs, and seeds, of the selector-induced effective half-life from 122.45 to 39.34 raw samples and assigns approximately 0.25 of the selector mass to $\tau \leq 12$.

This improved decay-scale coverage does not recover the native performance. The original $Q = 6$ model increases MSE by 9.56% relative to native SKOLR, while the augmented $Q = 8$ model increases it by 10.16%. The direct $Q = 8$-versus-$Q = 6$ contrast is also slightly unfavorable on average, despite two small seed-level wins.

The selector remains close to its uniform initialization in both conditions: its normalized entropy is approximately one, and its mean ℓ_1 displacement is only 6×10^{-6} for $Q = 6$ and 7×10^{-6} for $Q = 8$. Thus, making an appropriate short decay scale available is not sufficient by itself. Either decay mismatch is not the dominant limitation in this task, or the host does not learn to isolate the newly introduced short-decay modes.

Frequency alignment. ETTm1 is sampled every 15 minutes and therefore has a daily period of 96 raw samples. We compare a frequency grid containing the daily fundamental and its harmonics,

$$\Omega_{\text{align}} = \left\{0, \frac{1}{96}, \frac{2}{96}, \ldots, \frac{47}{96}\right\},$$

with a grid having the same frequency budget but half-bin-shifted non-DC frequencies,

$$\Omega_{\text{shift}} = \left\{0, \frac{0.5}{96}, \frac{1.5}{96}, \ldots, \frac{46.5}{96}\right\}.$$

The raw-frequency distance to the daily fundamental is therefore zero for Ω_{align} and $1/192$ for Ω_{shift}. All half-lives, selectors, host components, optimization settings, and parameter counts are held fixed.

Both frequency conditions retain the same structural stability:

$$\rho(\mathbf{K}) \approx 0.9989, \qquad \kappa_2(\mathbf{V}) = 1, \qquad d_{\text{H}} \approx 0, \qquad G_{\text{rel}}^{(\mathcal{N})} = 1.$$

Consequently, their forecasting difference isolates modal alignment rather than recurrent amplification.

At $H = 720$, the aligned grid improves eight of nine host–seed pairs and reduces MSE by 1.15% on average. The corresponding host-wise mean reductions are 2.30%, 0.24%, and 0.90% for K^2VAE, Koopa, and SKOLR, respectively. At $H = 96$, however, the aligned grid is 0.30% worse on average and wins only four of nine comparisons. This horizon dependence is consistent with modal phase errors accumulating over longer recursive rollouts. It provides the clearest evidence that task-aligned modal support matters independently of operator normality.

Controlled low-rank adaptivity. The preceding experiments indicate that a completely prescribed operator bank may be too restrictive in some tasks. We call the resulting Koopa variant *Frozen Dynamics + rank-4 correction* and augment each frozen operator with a shared low-rank correction,

$$\widetilde{\mathbf{K}}_q = \mathbf{K}_q + \frac{\alpha_{\text{lr}}}{r_{\text{lr}}} \mathbf{U}_{\text{lr}} \mathbf{V}_{\text{lr}}, \qquad r_{\text{lr}} = 4, \quad \alpha_{\text{lr}} = 4,$$

where $\mathbf{V}_{\text{lr}} \in \mathbb{R}^{r_{\text{lr}} \times d}$ and $\mathbf{U}_{\text{lr}} \in \mathbb{R}^{d \times r_{\text{lr}}}$. The same correction is shared across all $Q = 6$ branches within a time-varying block. The matrix $\mathbf{U}_{\text{lr}}$ is initialized to zero, so $\widetilde{\mathbf{K}}_q = \mathbf{K}_q$ at initialization. Across Koopa's three time-varying blocks, the correction adds $3{,}072$ trainable parameters. The denominator for the reported ratio is the complete trainable parameter count of the uncorrected Koopa–Frozen Dynamics $Q = 6$ host model, $126{,}976$, so the increment is $3{,}072/126{,}976 = 2.42\%$. Koopa's sample-fitted DMD transitions are non-trainable; the ratio is not taken relative to the $1{,}152$ selector logits alone.

On Exchange at $H = 192$, native Koopa obtains a mean MSE of 0.19511. The uncorrected $Q = 6$ model reduces this to 0.19121, corresponding to a 1.96% mean improvement. The rank-4 corrected variant further reduces the mean MSE to 0.18926, corresponding to a 2.94% improvement over native Koopa. Its direct improvement over the uncorrected model is 0.84%, but the seed-level variance is substantial.

Table 12: Diagnostics of Koopa's time-varying transition on Exchange at $H = 192$. Entries are mean [maximum] across three blocks and three seeds, except for $\kappa_{2,\max}(\mathbf{V})$ and MSE. Both Frozen Dynamics variants use $Q = 6$.

Transition	Residual/base ratio	$\rho(\mathbf{K})$	$\kappa_{2,\max}(\mathbf{V})$	d_{H}	$G_{\mathrm{rel}}^{(\mathcal{N})}$	MSE
Native DMD	–	1.348 [1.432]	$3.71\times10^{3\dagger}$	0.849 [0.905]	2.00 [2.31]	0.19511
Frozen Dynamics	0	0.9993 [0.9993]	1.00^{*}	1.25×10^{-8}	1.00	0.19121
Frozen Dynamics + rank-4 correction	0.104 [0.260]	1.072 [1.171]	$2.02\times10^{3\dagger}$	0.094 [0.201]	2.40 [4.59]	**0.18926**

† Eigenvector-condition estimates are qualitative for clustered eigenvalues; * the uncorrected frozen operators have an orthonormal eigenbasis by construction.

Table 12 places these accuracy changes alongside the transition diagnostics. The uncorrected frozen operator is normal and contractive, whereas the rank-4 correction raises the mean spectral radius above one and reintroduces non-normal power growth. Its modest average accuracy gain therefore comes with a less controlled rollout.

The seed-level behavior is consistent with this trade-off. For seeds 0 and 2, the mean residual-to-base Frobenius ratios are 0.067 and 0.065, and the rank-4 corrected variant improves over the uncorrected $Q = 6$ variant by 7.48% and 0.97%, respectively. For seed 1, the ratio increases to 0.181, the mean spectral radius reaches 1.122, and MSE degrades by 5.93%. With only three seeds, this pattern does not establish a quantitative threshold. It suggests that added flexibility can improve fit, but larger corrections may also amplify errors.

Summary. The three tests show that transfer depends on both rollout stability and whether the prescribed modes suit the task. Adding short decay scales alone does not improve SKOLR on Exchange, whereas aligning the frequency grid helps more consistently at the longer ETTm1 horizon. The rank-4 correction yields a small average gain on Koopa–Exchange but can also increase amplification. Frozen Dynamics is therefore most useful when its stable operator bank contains modes that the host can exploit.

G Ablation Study of EchoKoop

This section provides the complete controlled ablation study underlying Section 4.2. We use the prespecified core suite of ETTh1, ETTm1, Exchange, Weather, PEMS04, and PEMS08, denoted by $\mathcal{D}_{\mathrm{abl}}$. It spans hourly and minute-level measurements, financial and meteorological series, and high-dimensional traffic networks. This suite is used consistently for every ablation and is distinct from the six best-performing datasets selected for compact presentation in Table 1. EchoKoop denotes the default model with frozen normal operators. EchoKoop-R (R for reservoir) denotes the random-reservoir alternative with member-wise matched spectral radii.

G.1 Protocol and Evaluation

Each variant is trained from scratch with seeds $0, 1, 2$ on four prediction horizons per dataset. Except for the named intervention, we retain the same data splits, preprocessing, look-back length, latent width, temporal resolutions, decoder, optimizer, learning-rate schedule, early-stopping protocol, and evaluation code as default EchoKoop. The loss-robustness rows change only the Huber threshold. The reservoir rows preserve the encoder–decoder architecture and spectral-radius schedule while using learned temporal weights to form their initial states, as detailed in Appendix C.9.

Let $\mathcal{H}_d$ denote the four horizons used for dataset d. For variant v, we first average over the three seeds and then macro-average the resulting 24 dataset–horizon cells:

$$\overline{\mathrm{MSE}}_v = \frac{1}{24} \sum_{d \in \mathcal{D}_{\mathrm{abl}}} \sum_{h \in \mathcal{H}_d} \left(\frac{1}{3} \sum_{s=0}^{2} \mathrm{MSE}_{v,d,h,s} \right).$$

Table 13: Controlled dataset-wise ablation study of EchoKoop. Each dataset column reports test MSE averaged over four prediction horizons and three random seeds. "Avg." reports MSE/MAE macro-averaged over the $6 \times 4 = 24$ dataset–horizon settings. Δ_{MSE} is the relative change in macro-average MSE against the frozen normal-operator $Q_a = 6$ reference unless marked otherwise and is computed from unrounded cell means. Negative values indicate improvement. Lower is better.

Variant	ETTh1	ETTm1	Exchange	Weather	PEMS04	PEMS08	Avg. MSE/MAE	Δ_{MSE}
EchoKoop (default, $Q_a = 6$)	0.4391	0.3743	0.1170	0.2700	0.1271	0.1937	0.2535 / 0.3142	Ref.
A. Normal-operator bank size								
Frozen normal operators, $Q_a = 3$	0.4394	0.3742	0.1171	0.2703	0.1310	0.1919	0.2540 / 0.3149	$+0.18\%$
Frozen normal operator, $Q_a = 1$	0.4396	0.3743	0.1172	0.2711	0.1292	0.1913	0.2538 / 0.3148	$+0.09\%$
B. Multiscale state construction and decoding								
Fine + mid resolutions, $\mathcal{S} = \{1, 2\}$	0.4404	0.3776	0.1166	0.2810	0.1380	0.1994	0.2588 / 0.3183	$+2.09\%$
Fine resolution only, $\mathcal{S} = \{1\}$	0.4589	0.4099	0.1176	0.3230	0.1434	0.2038	0.2761 / 0.3300	$+8.90\%$
w/o phase-aware history alignment	0.4852	0.4036	0.1149	0.2744	0.1306	0.1946	0.2672 / 0.3223	$+5.39\%$
w/o cross-scale residual correction	0.4435	0.3771	0.1136	0.2783	0.1343	0.2028	0.2583 / 0.3192	$+1.87\%$
C. Operator family and trainability								
Trainable operators, $Q_a = 6$	0.4417	0.3743	0.1225	0.2697	0.1243	0.1870	0.2532 / 0.3133	-0.11%
Trainable operator, $Q_a = 1$	0.4428	0.3747	0.1215	0.2672	0.1242	0.1874	0.2530 / 0.3133	$-0.32\%^{\dagger}$
EchoKoop-R, $Q_r = 6$	0.4738	0.4063	0.1200	0.2839	0.1377	0.1952	0.2695 / 0.3223	$+6.30\%$
EchoKoop-R, $Q_r = 1$	0.5130	0.4248	0.1226	0.3062	0.1405	0.2004	0.2846 / 0.3295	$+5.60\%^{\ddagger}$
D. Training-loss robustness								
Huber loss, $\delta_{\mathrm{H}} = 0.10$	0.4392	0.3754	0.1185	0.2715	0.1292	0.1898	0.2539 / 0.3141	$+0.16\%$
Huber loss, $\delta_{\mathrm{H}} = 0.30$	0.4385	0.3772	0.1195	0.2724	0.1286	0.1897	0.2543 / 0.3160	$+0.30\%$

Default EchoKoop uses Huber loss with $\delta_{\mathrm{H}} = 0.15$. † Relative to the frozen normal-operator $Q_a = 1$ model. ‡ Relative to EchoKoop-R with $Q_r = 6$. EchoKoop-R matches the encoder, decoder, modal dimensions, temporal resolutions, training protocol, and member-wise spectral radii; it initializes its reservoirs with learned temporal weights.

The relative change in Table 13 is $\Delta_{\mathrm{MSE}}(\overline{\mathrm{MSE}}_v, \overline{\mathrm{MSE}}_r)$, as defined in Equation (14), where r is the frozen normal-operator $Q_a = 6$ reference unless noted otherwise. Unlike the paired mechanism tests, this percentage is calculated after averaging over seeds and dataset–horizon cells. Macro-averaging gives each cell equal weight and prevents the larger PEMS datasets from dominating the aggregate.

G.2 Results and Interpretation

Size of the normal-operator bank. Reducing the number of frozen normal operators from $Q_a = 6$ to $Q_a = 3$ or $Q_a = 1$ changes macro-average MSE by only $+0.18\%$ and $+0.09\%$, respectively. The response is small and non-monotonic across these dictionary sizes: $Q_a = 6$ is the reference configuration, but the aggregate results do not support a claim that more half-life candidates are uniformly better. Instead, a single prescribed operator already captures much of the forecasting benefit, while the larger bank supplies a modest amount of task-level flexibility. The per-dataset columns also show that this flexibility is heterogeneous: $Q_a = 3$ is slightly better on PEMS08, whereas $Q_a = 6$ is better on PEMS04.

Multiscale state construction. The strongest degradations arise when temporal structure is removed. Dropping the coarsest branch increases macro-average MSE by 2.09%, and retaining only the fine branch increases it by 8.90%. The larger second change indicates that the middle and coarse native grids are complementary rather than interchangeable replicas of the fine-resolution rollout. Removing phase-aware history alignment increases MSE by 5.39%, the largest single-component penalty among the matched normal-operator ablations. This supports aligning historical latent cells with the forecast origin before propagation, although the one-factor-at-a-time design does not identify interactions between phase alignment and resolution. Removing the cross-scale residual correction has a smaller but consistent aggregate cost of 1.87%, indicating that most predictive capacity resides in the branch rollouts while learned cross-scale exchange provides an additional refinement.

Figure 8 shows the PEMS04 results by forecast horizon. Retaining only the fine resolution gives the largest paired degradation, while removing the coarse branch also increases MSE at all four

Table 14: Test MSE averaged over four horizons and three seeds. The two sparse variants use the same operators and differ in history initialization.

Design	ETTh1	ETTm1	PEMS04
EchoKoop	0.4391	0.3743	0.1271
Sparse, phase-based history	0.4878	0.4161	0.1385
EchoKoop-R	0.4738	0.4063	0.1377

horizons. Reducing the bank from $Q_a = 6$ to $Q_a = 1$ or 3 has little effect through $H = 48$, but increases error at $H = 96$. This single-dataset illustration complements the six-dataset results in Table 13.

Transition trainability. Making all six initially normal transition matrices trainable changes macro-average MSE from 0.2535 to 0.2532, a relative difference of -0.11%. Across the 24 dataset–horizon cells, the frozen and trainable variants each win 12. The $Q_a = 1$ comparison is similar: trainability improves over its matched frozen $Q_a = 1$ reference by 0.32%. These differences are small relative to the penalties from removing multiscale rollout or phase-aware initialization and vary in sign across datasets. The controlled evidence therefore supports the narrower conclusion that transition trainability is not necessary for competitive performance under this architecture and protocol; it does not establish that frozen transitions dominate every trainable alternative.

Normal-operator and random-reservoir designs. EchoKoop-R with $Q_r = 6$ is 6.30% worse than default EchoKoop, despite matching the number of members, modal width, temporal resolutions, training protocol, and member-wise spectral radii. Reducing the reservoir bank from $Q_r = 6$ to $Q_r = 1$ incurs a further 5.60% penalty relative to the six-member reservoir. This contrasts with the weak sensitivity to Q_a in the normal-operator family and suggests that an unstructured random bank relies more heavily on member diversity. Overall, these results favor EchoKoop's prescribed bank with phase-aligned initialization over a random bank initialized with learned temporal weights.

To examine the role of initialization within the random bank, we also replace only the six normal operators on ETTh1, ETTm1, and PEMS04, leaving EchoKoop's phase-based history construction unchanged. This control uses the same sparse operators and training settings as EchoKoop-R, which instead learns temporal weights for the initial state. Table 14 compares the three designs on this subset.

Learned temporal weights lower the sparse-bank MSE on all three datasets, although neither sparse variant matches the normal-operator reference. The improvement indicates that the initializer matters when the operator family changes; the remaining gap favors the prescribed normal bank in this comparison.

Robustness to the training loss. Default EchoKoop uses Huber loss with $\delta_{\mathrm{H}} = 0.15$. Changing the threshold to 0.10 or 0.30 changes macro-average MSE by only $+0.16\%$ and $+0.30\%$, respectively. The principal ablation ordering is therefore not an artifact of a narrowly tuned Huber threshold.

H Statistical Significance and Uncertainty

Unit of inference. We treat the dataset, rather than an individual prediction horizon or random seed, as the primary independent unit. Horizons from the same dataset share the underlying series, chronological split, and overlapping forecast windows, while seeds are repeated fits of the same task. Treating all dataset–horizon–seed records as independent would therefore constitute pseudoreplication. For the benchmark analysis, we first average each method over the four horizons of a dataset. For the controlled ablations, we first average the three matched seeds within each dataset–horizon cell and then the four horizons within each dataset. All tests are two-sided, and all pairwise families are corrected by Holm's step-down procedure (Holm, 1979). We accompany the tests with win/tie/loss counts, rank-biserial effect sizes, and dataset-clustered bootstrap intervals in the archived statistical supplement. The available records do not include aligned per-origin losses or

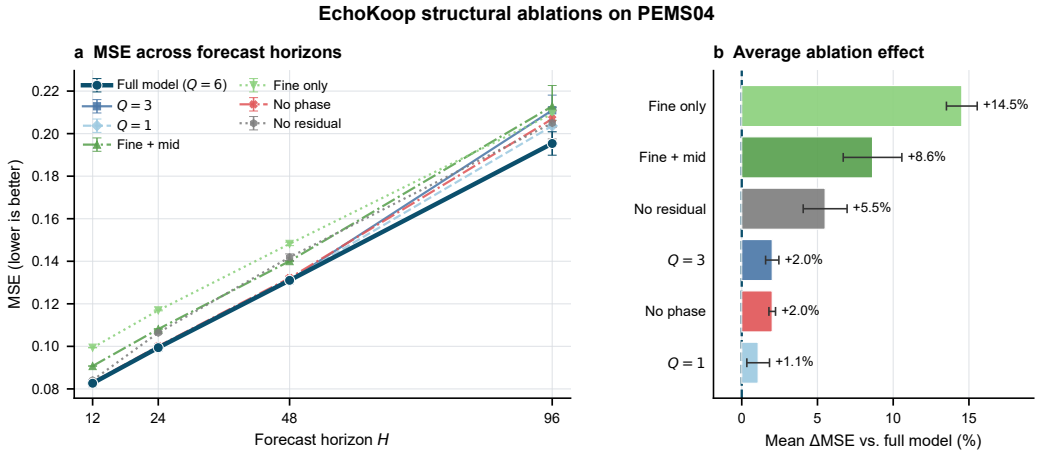

Figure 8: **Structural ablation on PEMS04. a**, Test MSE across forecasting horizons. **b**, Average MSE change relative to the full $Q = 6$ model. Results are mean $\pm$ standard deviation over three seeds; positive changes indicate degradation.

Table 15: Friedman average ranks and control-versus-EchoKoop post-hoc rank contrasts for eight complete-case methods over ten datasets. Lower ranks are better; dashes identify EchoKoop as the reference method.

	MSE				MAE			
Model	Rank	z	unadjusted p	p_{Holm}	Rank	z	unadjusted p	p_{Holm}
KUMA	2.70	-0.274	0.784	0.784	2.60	-0.639	0.523	1.000
K^2VAE	5.60	2.373	0.0176	0.123	5.10	1.643	0.100	0.502
SKOLR	4.60	1.461	0.144	0.432	4.30	0.913	0.361	1.000
AIKAE	5.40	2.191	0.0285	0.142	5.10	1.643	0.100	0.502
iTransformer	4.10	1.004	0.315	0.631	4.20	0.822	0.411	1.000
TimesNet	5.00	1.826	0.0679	0.272	5.20	1.734	0.0828	0.497
DLinear	5.60	2.373	0.0176	0.123	6.20	2.647	0.0081	0.0568
EchoKoop	3.00	–	–	–	3.30	–	–	–

predictions for forecast-origin tests. We do not test equivalence without a prespecified equivalence margin.

Overall benchmark comparison. The complete-case analysis retains the eight methods with results on all 40 dataset–horizon settings: EchoKoop, KUMA, K^2VAE, SKOLR, AIKAE, iTransformer, TimesNet, and DLinear. Friedman tests over the ten dataset-level blocks reject the null of identical method ranks for both MSE ($\chi_F^2(7) = 15.23$, $p = 0.033$) and MAE ($\chi_F^2(7) = 15.47$, $p = 0.030$). On MSE, EchoKoop has the second-best average rank (3.00), behind KUMA (2.70); its MAE rank is 3.30. These omnibus results establish overall rank differences, but do not identify a particular EchoKoop–baseline pair. Table 15 reports the complete ranks and control-versus-EchoKoop rank contrasts following the standard multi-dataset comparison protocol (Demšar, 2006). For metric m, the contrast is $z = (\bar{R}_m - \bar{R}_{\text{EchoKoop}})/\sqrt{k(k+1)/(6N)}$, with $k = 8$ methods and $N = 10$ datasets; positive z indicates a worse average rank than EchoKoop. The seven two-sided comparisons are Holm-adjusted separately for MSE and MAE.

None of these average-rank contrasts remains significant after Holm adjustment. Because the rank contrasts retain only each dataset's within-block ordering, we also use paired Wilcoxon tests, which incorporate the ordering of the paired dataset-level difference magnitudes.

We therefore compare EchoKoop with each of the ten baselines using paired Wilcoxon signed-rank tests on the available complete dataset means and apply Holm correction across the ten comparisons.

Table 16: Summary of statistical evidence. Dataset-level tests are the primary analysis; setting-level tests are dependent-horizon sensitivity analyses. All pairwise p-values are Holm-adjusted within their stated family. Positive r_{rb} favors EchoKoop or the full ablation reference.

Comparison	$r_{\mathrm{rb},D}$	Dataset-level p	Setting-level p
EchoKoop vs. K^2VAE	0.89	0.098	< 0.001
EchoKoop vs. TimesNet	0.85	0.109	< 0.001
EchoKoop vs. DLinear	0.89	0.098	< 0.001
Fine + middle scales vs. full	0.90	0.563	0.004
Fine scale only vs. full	1.00	0.375	< 0.001
w/o history phase vs. full	0.81	0.750	0.014
Trainable $Q_a = 6$ vs. frozen	-0.24	1.000	1.000
Reservoir $Q_r = 1$ vs. normal	1.00	0.375	< 0.001
Reservoir $Q_r = 6$ vs. normal	1.00	0.375	0.001

No pair remains significant at the 0.05 level. The smallest adjusted MSE values are $p = 0.098$ for K^2VAE and DLinear, followed by $p = 0.109$ for TimesNet. These comparisons nevertheless have large positive rank-biserial effects (0.89, 0.89, and 0.85, respectively), consistent with the favorable win/tie/loss counts, but the ten available datasets do not provide enough independent blocks for a multiplicity-corrected pairwise superiority claim.

Controlled ablations. The primary ablation tests use six dataset-level differences and apply Holm correction across the twelve eligible variants. None reaches the corrected 0.05 threshold. This outcome should be interpreted together with the discrete resolution of the test: with only six nonzero paired differences, the smallest attainable two-sided exact Wilcoxon value is 0.03125 before correction. In particular, the EchoKoop-R variants have the maximum dataset-level rank-biserial effect ($r_{\mathrm{rb}} = 1.00$), whereas the trainable $Q_a = 6$ comparison has a small effect of the opposite sign ($r_{\mathrm{rb}} = -0.24$). Thus, the absence of a corrected dataset-level rejection does not imply that these effects are equal in magnitude.

As a sensitivity analysis, we also apply the same paired test to the 24 seed-averaged dataset–horizon cells. Because the four horizons within each dataset are dependent, these values are exploratory rather than the primary inferential result. After Holm correction, the setting-level evidence is significant for retaining only the fine and middle scales ($p = 0.004$), retaining only the fine scale ($p < 0.001$), removing history-phase alignment ($p = 0.014$), and using EchoKoop-R with $Q_r = 1$ ($p < 0.001$) or $Q_r = 6$ ($p = 0.001$). It is not significant for making the six initially normal operators trainable ($p = 1.000$). Table 16 summarizes the distinction between the primary and sensitivity analyses.

Interpretation. The ten-dataset benchmark shows differences in overall ranks, although no pairwise difference survives correction for multiple comparisons. In the ablations, multiscale rollout and history alignment have the largest observed effects, while training the transition entries offers no consistent gain. These one-component comparisons do not resolve interactions among components.

I Low-Dimensional Dynamical Systems

To complement the standard forecasting benchmarks, we study Lorenz–63, Van der Pol, and the double pendulum, representing chaotic, limit-cycle, and conservative nonlinear dynamics, respectively. We first compare finite-horizon accuracy and long-run structure under a shared protocol, then examine Lorenz–63 distributional recovery under a separate conditional-prediction protocol.

I.1 Experimental Protocol

The Lorenz–63 system is defined by

$$\begin{cases} \dot{x} = 10(y - x), \\ \dot{y} = x(28 - z) - y, \\ \dot{z} = xy - \dfrac{8}{3}z. \end{cases}$$

The Van der Pol oscillator is defined by

$$\begin{cases} \dot{q} = p, \\ \dot{p} = (1 - q^2)p - q. \end{cases}$$

For the double pendulum, we use the standard frictionless dynamics

$$\mathbf{M}(\boldsymbol{\theta})\ddot{\boldsymbol{\theta}} + \mathbf{C}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}})\dot{\boldsymbol{\theta}} + \mathbf{G}(\boldsymbol{\theta}) = \mathbf{0},$$

where $\boldsymbol{\theta} = [\theta_1, \theta_2]^\top$, $v_i = \dot{\theta}_i$, $m_1 = m_2 = \ell_1 = \ell_2 = 1$, and $g = 9.81$. To avoid angular discontinuities, its observable is

$$\mathbf{s}_t = [\sin\theta_{1,t}, \cos\theta_{1,t}, \sin\theta_{2,t}, \cos\theta_{2,t}, v_{1,t}, v_{2,t}]^\top. \tag{15}$$

Trajectories are generated by fixed-step RK4. The time steps are 0.1, 0.05, and 0.01 for Lorenz–63, Van der Pol, and the double pendulum, respectively. Each trajectory contains 1,201 retained samples. The train/validation/test trajectory counts are $128/32/32$, $64/16/16$, and $96/24/24$, respectively. All splits use disjoint initial conditions, and normalization statistics are computed from training trajectories only.

All methods share the same forecast origins and 50-step targets while retaining their designated observation contexts. In model labels here, the suffix C1 or C12 specifies the number of observed context states; the mathematical C denotes the number of observation channels. EchoKoop-C1 uses one observed state, whereas EchoKoop-C12 uses the preceding 12 states; the two variants otherwise share the same architecture and fixed-operator dictionary. FlowMixer-C12 is trained with teacher-forced $12 \to 12$ blocks and is evaluated recursively. Pointwise NRMSE is reported at steps $\{5, 20, 50\}$. Long-term metrics are computed from uninterrupted 1,000-step autonomous rollouts over four held-out trajectories. Learned-model results are averaged over three seeds; DMD and Delay-DMD are deterministic under the fixed split.

Valid prediction time. For rollout $n \in \{1, \ldots, N_{\mathrm{roll}}\}$ and step $t \in \{1, \ldots, T\}$, with $N_{\mathrm{roll}} = 4$, the range-normalized trajectory error is

$$e_{n,t} = \sqrt{\frac{1}{C}\sum_{c=1}^{C}\left(\frac{\hat{s}_{n,t,c} - s_{n,t,c}}{R_c}\right)^2}.$$

where $R_c = s_{c,\max}^{\mathrm{train}} - s_{c,\min}^{\mathrm{train}}$ is the range of coordinate c computed on the training split. Valid prediction time (VPT) is the number of consecutive rollout steps before $e_{n,t}$ first exceeds 0.1, averaged across the four held-out trajectories. A rollout that never exceeds the threshold receives the full 1,000-step horizon.

System-specific structure metrics. For Lorenz–63, Occ.-JS is the Jensen–Shannon divergence between reference and predicted occupancies on a 40×40 histogram of the (x, z) projection. Std.-Err is the mean relative error of the coordinate-wise attractor standard deviations,

$$\mathrm{StdErr} = \frac{1}{C}\sum_{c=1}^{C}\frac{|\hat{\sigma}_c - \sigma_c|}{\sigma_c},$$

where σ_c and $\hat{\sigma}_c$ are computed over the reference and predicted long rollouts, respectively.

Table 17: Pointwise forecast NRMSE ($\times 10^3$; lower is better). Learned-model results are mean $\pm$ sample standard deviation over three seeds; DMD-based methods are deterministic. The best result is bold and the second-best is underlined.

	Lorenz–63			Van der Pol			Double pendulum		
Method	5	20	50	5	20	50	5	20	50
VAE	$\underline{11.726 \pm 1.306}$	65.420 ± 1.856	167.883 ± 0.018	2.416 ± 0.241	2.012 ± 0.157	3.135 ± 0.620	$\underline{2.497 \pm 0.039}$	$\underline{3.907 \pm 0.068}$	$\mathbf{9.960 \pm 0.381}$
KAE	120.117 ± 1.783	176.370 ± 0.129	186.123 ± 0.123	11.217 ± 1.027	10.990 ± 0.963	20.269 ± 1.530	42.649 ± 0.280	113.927 ± 0.411	211.389 ± 0.506
DMD	204.612 ± 0.000	194.742 ± 0.000	192.721 ± 0.000	37.220 ± 0.000	126.719 ± 0.000	125.572 ± 0.000	47.028 ± 0.000	164.970 ± 0.000	278.924 ± 0.000
Delay-DMD	166.930 ± 0.000	187.132 ± 0.000	191.087 ± 0.000	7.739 ± 0.000	86.850 ± 0.000	114.412 ± 0.000	9.401 ± 0.000	84.882 ± 0.000	215.989 ± 0.000
FlowMixer-C12	33.521 ± 1.598	78.310 ± 3.955	205.628 ± 6.020	1.411 ± 0.864	1.680 ± 0.594	2.654 ± 0.678	10.220 ± 0.407	29.898 ± 1.832	90.524 ± 3.945
EchoKoop-C1	$\mathbf{7.486 \pm 0.701}$	$\underline{42.526 \pm 0.234}$	$\underline{153.101 \pm 1.543}$	$\mathbf{0.879 \pm 0.066}$	$\mathbf{1.245 \pm 0.099}$	$\mathbf{1.556 \pm 0.065}$	$\mathbf{2.229 \pm 0.058}$	$\mathbf{3.482 \pm 0.047}$	$\underline{13.717 \pm 0.676}$
EchoKoop-C12	14.472 ± 0.370	$\mathbf{36.090 \pm 1.494}$	$\mathbf{139.707 \pm 1.733}$	$\underline{1.387 \pm 0.186}$	$\underline{1.571 \pm 0.475}$	$\underline{1.697 \pm 0.308}$	2.615 ± 0.101	4.375 ± 0.143	16.103 ± 1.102

Table 18: Long-term metrics from 1,000-step autonomous rollouts. Lower is better except for VPT. "div." denotes an unbounded rollout for which the corresponding structural statistic is invalid. The best result is bold and the second-best is underlined; tied second-best results are all underlined.

	Lorenz–63			Van der Pol		Double pendulum	
Method	Occ.-JS ↓	Std.-Err ↓	VPT ↑	Freq.-Err ↓	VPT ↑	Energy Drift ↓	VPT ↑
VAE	0.3811 ± 0.0089	div.	29.33 ± 1.63	0.6324 ± 0.4124	$\underline{128.75 \pm 18.01}$	$(5.14 \pm 8.90) \times 10^{12}$	$\mathbf{166.83 \pm 21.04}$
KAE	0.5986 ± 0.0008	0.8780 ± 0.0020	1.50 ± 0.43	0.6473 ± 1.0411	117.83 ± 42.97	4.0636 ± 0.0930	35.75 ± 1.09
DMD	0.6695 ± 0.0000	0.9729 ± 0.0000	0.75 ± 0.00	$\underline{0.0357 \pm 0.0000}$	17.75 ± 0.00	1.4201 ± 0.0000	49.00 ± 0.00
Delay-DMD	0.6393 ± 0.0000	0.9485 ± 0.0000	1.25 ± 0.00	$\underline{0.0357 \pm 0.0000}$	38.75 ± 0.00	4.3061 ± 0.0000	28.25 ± 0.00
FlowMixer-C12	0.5678 ± 0.0145	0.6513 ± 0.2070	26.75 ± 1.09	$\mathbf{0.0313 \pm 0.0541}$	$\mathbf{995.75 \pm 7.36}$	3.1930 ± 0.2822	$\underline{144.08 \pm 2.67}$
EchoKoop-C1	$\mathbf{0.2065 \pm 0.0094}$	$\mathbf{0.2139 \pm 0.0029}$	$\mathbf{36.92 \pm 1.15}$	1.2009 ± 0.0000	50.50 ± 0.00	$\underline{0.2576 \pm 0.0796}$	53.67 ± 0.14
EchoKoop-C12	$\underline{0.2180 \pm 0.0095}$	$\underline{0.2408 \pm 0.0226}$	$\underline{34.08 \pm 0.38}$	1.2009 ± 0.0000	47.83 ± 1.01	$\mathbf{0.2329 \pm 0.1071}$	53.17 ± 0.52

For Van der Pol, Freq.-Err compares the dominant nonzero frequencies of the q coordinate,

$$\mathrm{FreqErr} = \frac{1}{N_{\mathrm{roll}}} \sum_{n=1}^{N_{\mathrm{roll}}} \frac{\left| \hat{f}_n^{(q)} - f_n^{(q)} \right|}{f_n^{(q)}},$$

where the dominant frequency is obtained from the discrete Fourier spectrum of the complete rollout.

For the double pendulum, angles are recovered from the sine–cosine observation in Equation (15). The mechanical energy is

$$E(\theta_1, \theta_2, v_1, v_2) = v_1^2 + \frac{1}{2} v_2^2 + v_1 v_2 \cos(\theta_1 - \theta_2) - 2g \cos\theta_1 - g \cos\theta_2.$$

Energy Drift is the rollout-averaged relative deviation from the initial energy,

$$\mathrm{EnergyDrift} = \frac{1}{N_{\mathrm{roll}} T} \sum_{n=1}^{N_{\mathrm{roll}}} \sum_{t=1}^{T} \frac{\left| \hat{E}_{n,t} - E_{n,0} \right|}{|E_{n,0}|},$$

where $T = 1000$.

Lower values are better for NRMSE, Occ.-JS, Std.-Err, Freq.-Err, and Energy Drift, whereas higher VPT is better. A result marked "div." denotes an unbounded rollout for which the corresponding long-term structural statistic is not valid.

I.2 Results and Discussion

Finite-horizon accuracy. An EchoKoop variant achieves the lowest mean NRMSE in eight of the nine system–horizon settings. EchoKoop-C1 is strongest on Van der Pol and at the shorter double-pendulum horizons, whereas EchoKoop-C12 performs better at the longer Lorenz–63 horizons. The VAE remains best on the 50-step double-pendulum forecast.

Long-term dynamical structure. EchoKoop best preserves the Lorenz attractor statistics and produces the lowest double-pendulum energy drift. The Van der Pol results expose a complementary limitation: FlowMixer-C12 more accurately preserves the dominant frequency and maintains a substantially longer valid rollout. The prescribed flows thus perform well on selected long-run statistics, while oscillatory dynamics leave room for improvement.

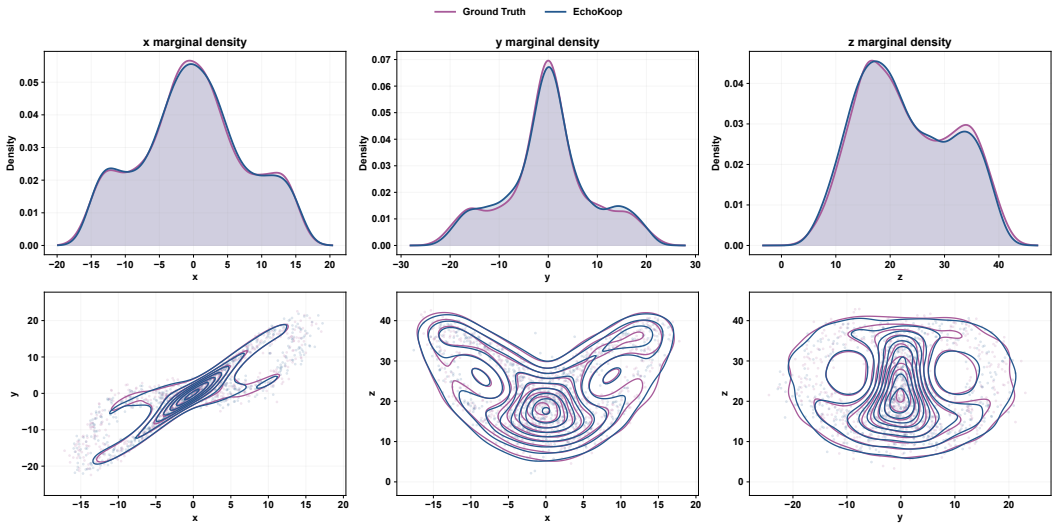

Figure 9: Lorenz–63 marginal and projected distributions under conditional prediction. Ground truth is magenta and EchoKoop is blue. The top row shows the marginal densities of x, y, and z; the bottom row shows density contours in the (x, y), (x, z), and (y, z) projections.

I.3 Statistical and Geometric Recovery of the Lorenz–63 Attractor

We also compare the distribution of Lorenz–63 states predicted from ground-truth history windows with the reference attractor. This separate conditional-prediction protocol assesses statistical and geometric agreement outside the shared cross-method ranking.

Specifically, EchoKoop is conditioned on 4,167 ground-truth history windows of length $L = 96$ and predicts the subsequent $H = 24$ states from each window. This produces 100,008 candidate states; following the released evaluation code, the first 100,000 are retained for comparison with the ground-truth distribution. As shown in Figure 9, the predicted marginal densities of x, y, and z closely match their ground-truth counterparts. The projected density contours also recover the principal modes and the characteristic two-lobed organization of the Lorenz–63 attractor.

Figure 10 provides a complementary comparison of the two-dimensional joint distributions. EchoKoop reproduces the overall support and major density structures in the (x, y), (x, z), and (y, z) projections. Quantitatively, the pooled comparison yields a mean Jensen–Shannon divergence of 0.0037, an MMD^2 of 2.71×10^{-4}, and a mean ground-truth-to-prediction KL divergence of 0.0684. These results demonstrate that EchoKoop can recover the finite-horizon state distribution and coarse geometric structure of the Lorenz–63 attractor under conditional prediction.

The joint distributions show larger discrepancies than the marginals, particularly in the (y, z) projection. Because each forecast starts from a ground-truth history window, these results describe finite-horizon distributional agreement rather than recovery of the invariant measure from an autonomous rollout.

J High-Dimensional Physical Fields

We evaluate EchoKoop on cylinder-wake, dam-flow, and ERA5 fields under the protocol specified below. Cylinder and Dam are evaluated at horizons 5, 20, and 50, while ERA5 Z500+T850 is evaluated at horizons 5, 10, and 50. All learned-model results are averaged over seeds $\{0, 1, 2\}$.

EchoKoop-1f uses one input frame, while EchoKoop uses 16 frames for Cylinder and 12 for Dam and ERA5. Both use resolutions $\mathcal{S} = \{1, 2, 4\}$ for future latent rollout. The CFD training batch sizes are 4 for EchoKoop-1f and 16/12 for EchoKoop on Cylinder/Dam, respectively. The comparison includes persistence, DMD, Delay-DMD, KAE, VAE, FlowMixer, and ConvLSTM.

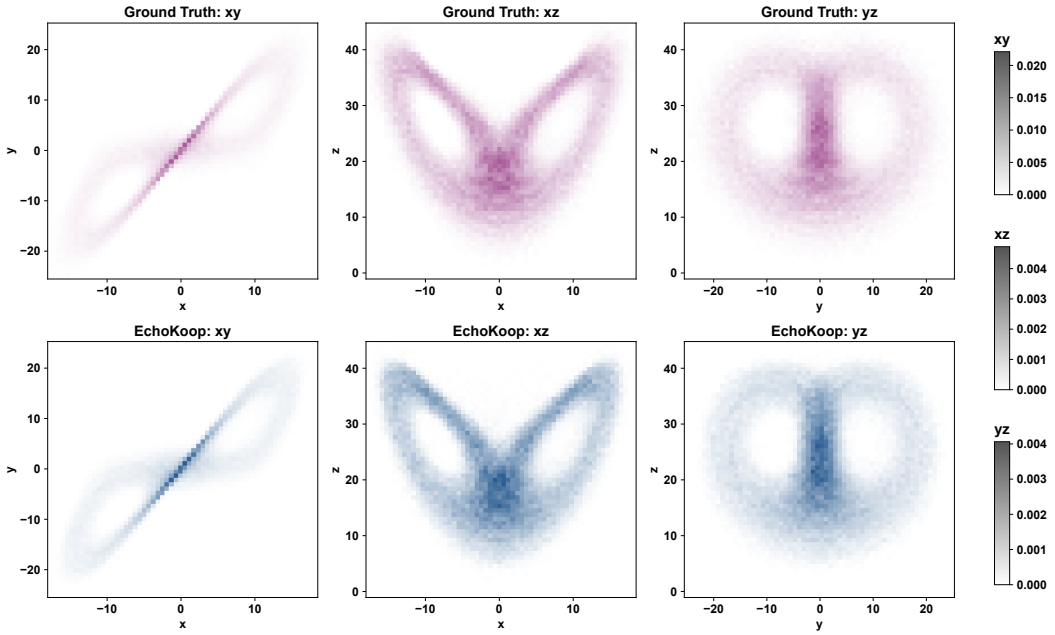

Figure 10: Lorenz–63 joint distributions under conditional prediction. Columns show the (x, y), (x, z), and (y, z) projections; rows show ground truth and EchoKoop, respectively. Each projection uses a shared density scale, shown by the color bars at right.

J.1 Physical Systems and Datasets

Cylinder wake and dam flow. The Cylinder and Dam datasets contain pre-generated CFDBench velocity fields. A schematic, nondimensional balance for an incompressible velocity field is

$$\begin{cases} \dfrac{\partial u}{\partial t} + u\dfrac{\partial u}{\partial x} + v\dfrac{\partial u}{\partial y} = -\dfrac{\partial p}{\partial x} + \dfrac{1}{\mathrm{Re}}\nabla^2 u, \\ \dfrac{\partial v}{\partial t} + u\dfrac{\partial v}{\partial x} + v\dfrac{\partial v}{\partial y} = -\dfrac{\partial p}{\partial y} + \dfrac{1}{\mathrm{Re}}\nabla^2 v, \\ \dfrac{\partial u}{\partial x} + \dfrac{\partial v}{\partial y} = 0, \end{cases} \tag{16}$$

where (u, v) is velocity, p is pressure, and Re is the Reynolds number. Cylinder captures flow around a circular cylinder; Dam captures flow over a step-like barrier. The latter also involves gravity and an air–water interface, so the displayed equations do not specify its full numerical model. Both datasets provide two velocity components on a 64×64 grid.

Cylinder contains 39 development trajectories and 10 held-out trajectories, each with 1,000 frames sampled at $\Delta t = 0.001$. Dam contains 55 development trajectories and 12 held-out trajectories, each with 100 frames sampled at $\Delta t = 0.1$. We reserve 10% of the development trajectories for validation using a fixed split seed, giving train/validation/test counts of $35/4/10$ for Cylinder and $49/6/12$ for Dam. Complete trajectories, rather than overlapping temporal windows, are assigned to each split. Channel statistics are estimated from the training trajectories only.

ERA5 reanalysis. ERA5 is a global atmospheric reanalysis rather than a CFD simulation (Hersbach et al., 2020). The released benchmark contains five variables—Z500, T850, Q700, U850, and V850—on a 64×32 latitude–longitude grid. We evaluate joint forecasting of Z500 geopotential at 500 hPa and T850 temperature at 850 hPa. The supplied chronological train, validation, and test partitions contain 50,258, 6,282, and 6,283 frames, respectively. We retain the benchmark min–max normalization metadata and latitude-dependent spatial weights, and use identical temporal partitions for all methods.

Equation (16) is provided for physical context, not imposed as a training constraint. EchoKoop is trained on field observations and does not explicitly enforce momentum conservation, incompressibility, atmospheric balance, or boundary conditions.

J.2 Physical-Field Training and Metrics

The physical-field experiments use a separate training budget from the standard time-series benchmarks. EchoKoop is trained for at most 300 epochs with Adam (learning rate 10^{-3}, no weight decay), a factor-0.8 learning rate reduction every 20 epochs, gradient clipping at norm 1, and early stopping after 20 epochs without validation improvement. The checkpoint with the lowest validation objective is evaluated. EchoKoop variants are trained on 20-step targets and rolled out freely to the reported horizons. On CFD, the training objective is field MSE plus 0.05 times vorticity MSE, with vorticity scaled by its training-set RMS; ERA5 uses latitude-weighted field MSE. Multi-frame EchoKoop batch sizes are 16, 12, and 8 for Cylinder, Dam, and ERA5, respectively.

Metrics are computed after restoring physical units, at the exact forecast step h. For test window i, channel c, and spatial position x, let R_c be the fixed training-data (or supplied benchmark) range and w_x the spatial weight. We report

$$\mathrm{NRMSE}_h = \frac{1}{N}\sum_{i=1}^{N}\left[\frac{1}{C}\sum_{c=1}^{C}\frac{\sum_x w_x\left((\hat{y}_{i,h,c,x}-y_{i,h,c,x})/R_c\right)^2}{\sum_x w_x}\right]^{1/2}.$$

The CFD grids use $w_x = 1$; ERA5 uses the supplied latitude weights. SSIM is computed per channel with an 11×11 Gaussian window ($\sigma = 1.5$), valid spatial support, and constants $(0.01R_c)^2$ and $(0.03R_c)^2$. Channel and window scores are averaged without latitude weighting. Reported values are then averaged over the three model seeds.

J.3 Dataset-Level Results

The horizon-wise results show different patterns across the three fields. ConvLSTM achieves the lowest NRMSE and highest SSIM at every reported endpoint. Among the remaining methods, DMD or Delay-DMD has the lowest NRMSE at the shortest lead on Cylinder, Dam, and ERA5. At longer leads, multi-frame EchoKoop has the lowest NRMSE on Cylinder at $H = 50$. On Dam at $H = 20$ and $H = 50$, its NRMSE matches EchoKoop-1f to the reported precision, while its SSIM is higher.

On ERA5, the lowest NRMSE among methods other than ConvLSTM is obtained by DMD at $H = 5$, EchoKoop-1f at $H = 10$, and VAE at $H = 50$. These comparisons do not isolate the effects of multiresolution dynamics or input-history length: both EchoKoop variants use multiresolution operators, and their CFD training batch sizes also differ.

KAE-Info adapts the released InformationKoopman latent-dynamics core (Cheng et al., 2026) within the common spatial architecture, using the same splits, normalization, training budget, and evaluation pipeline as the other learned baselines. Its long-horizon results on Cylinder and ERA5 vary substantially across seeds, as reflected in the reported uncertainty.

J.4 Qualitative Forecasts

Figures 11 and 12 show Cylinder and Dam forecasts. Reference and predicted fields share color limits within each figure.

The Cylinder visualization shows that EchoKoop preserves the principal vortex pattern at horizon 50, consistent with its lower long-horizon NRMSE and higher SSIM. In the ERA5 T850 panel, large-scale anomaly structure is retained at horizon 10, although local amplitude and contour differences remain visible.

J.5 Resource Reporting

Table 20 reports the computational resources used by EchoKoop in the physical-field experiments. These measurements are provided for reproducibility rather than as evidence of universal runtime or memory efficiency.

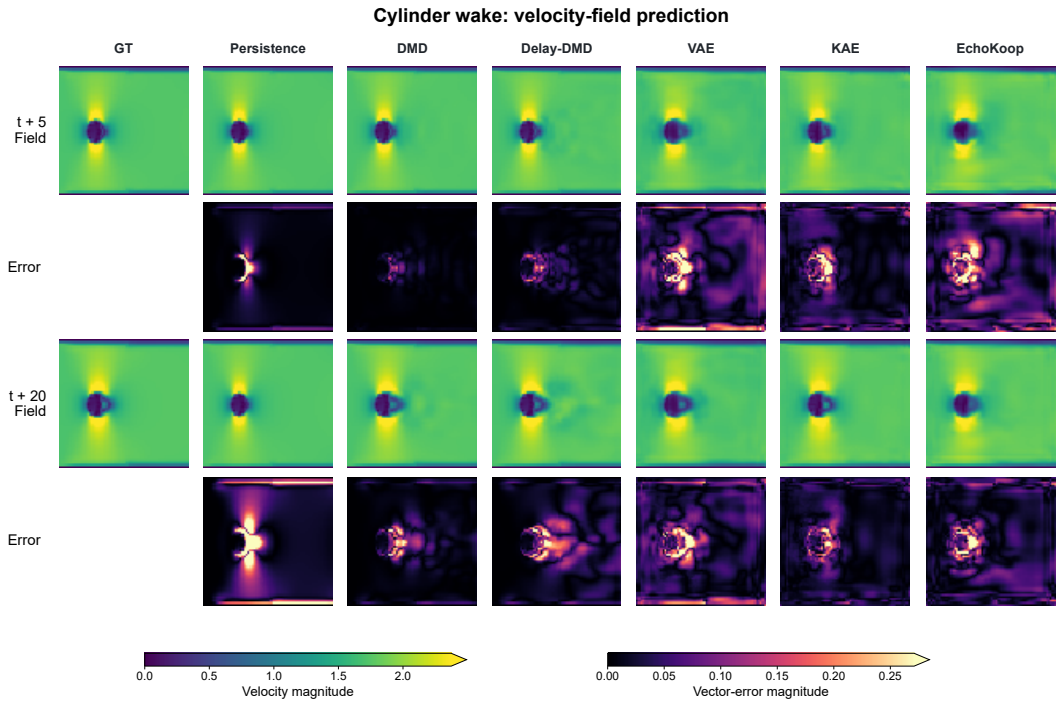

Figure 11: Long-horizon velocity-field forecasting on the Cylinder system. Columns show ground truth (GT), Persistence, DMD, Delay-DMD, VAE, KAE and EchoKoop at prediction horizons $t + 5$ and $t + 20$. The corresponding vector-error magnitudes are shown below each predicted field. All methods share the same field and error color scales within the figure.

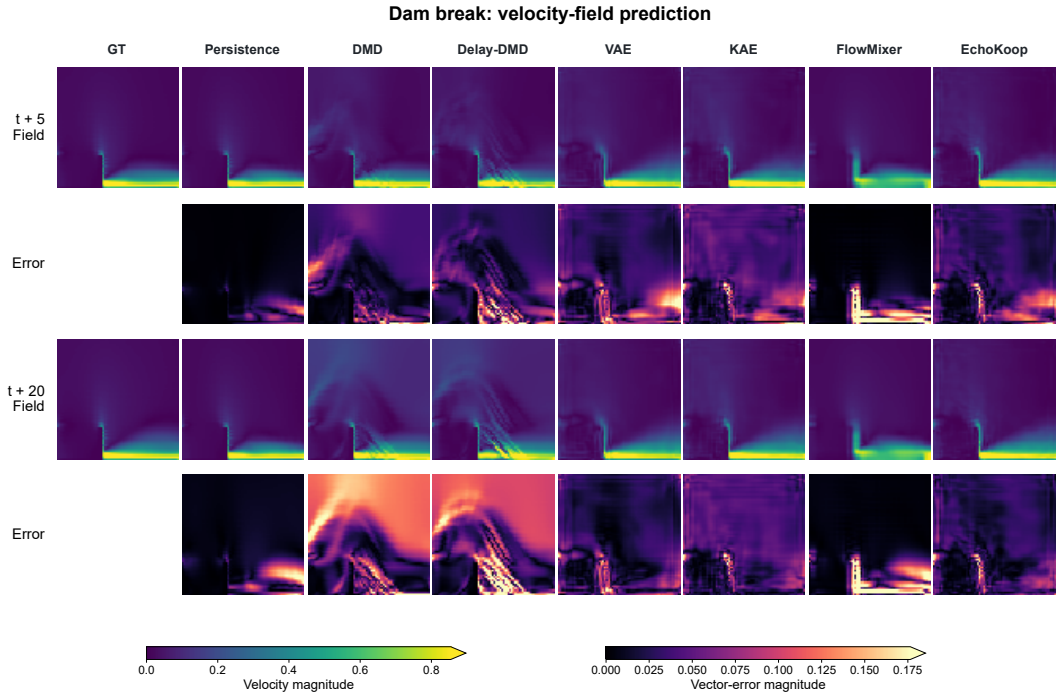

Figure 12: Long-horizon velocity-field forecasting on the Dam-break system. Columns show ground truth (GT), Persistence,DMD, Delay-DMD, VAE, KAE, FlowMixer, and EchoKoop at prediction horizons $t + 5$ and $t + 20$. The lower rows show vector-error magnitudes under a common color scale for all methods.

Table 19: Complete multi-horizon forecasting performance on three high-dimensional physical-field datasets. Each value is the mean ± sample standard deviation over seeds 0/1/2 ($n = 3$). NRMSE is lower-is-better and SSIM is higher-is-better. Context reports the effective inference frames as Cylinder/Dam/ERA5. The detailed values are presented without cross-context ranking.

Method	Context (C/D/E)	Metric	Cylinder			Dam			ERA5 (Z500+T850)		
			5	20	50	5	20	50	5	10	50
Single-frame inference											
Persistence	1/1/1	NRMSE ↓	0.0215 ± 0.0000	0.0242 ± 0.0000	0.0334 ± 0.0000	0.0045 ± 0.0000	0.0092 ± 0.0000	0.0130 ± 0.0000	0.0366 ± 0.0000	0.0474 ± 0.0000	0.0590 ± 0.0000
		SSIM ↑	0.9183 ± 0.0000	0.8950 ± 0.0000	0.8560 ± 0.0000	0.9830 ± 0.0000	0.9465 ± 0.0000	0.9052 ± 0.0000	0.8997 ± 0.0000	0.8431 ± 0.0000	0.7908 ± 0.0000
DMD	1/1/1	NRMSE ↓	0.0015 ± 0.0000	0.0079 ± 0.0000	0.0212 ± 0.0000	0.0033 ± 0.0000	0.0055 ± 0.0000	0.0086 ± 0.0000	0.0260 ± 0.0000	0.0361 ± 0.0000	0.0463 ± 0.0001
		SSIM ↑	0.9988 ± 0.0000	0.9831 ± 0.0000	0.9214 ± 0.0000	0.9846 ± 0.0001	0.9582 ± 0.0003	0.9162 ± 0.0003	0.9376 ± 0.0000	0.8998 ± 0.0001	0.8615 ± 0.0003
KAE	1/1/1	NRMSE ↓	0.0046 ± 0.0002	0.0049 ± 0.0001	0.0253 ± 0.0010	0.0034 ± 0.0003	0.0031 ± 0.0003	0.0044 ± 0.0003	0.0321 ± 0.0068	0.0375 ± 0.0054	0.0467 ± 0.0030
		SSIM ↑	0.9965 ± 0.0002	0.9960 ± 0.0003	0.9083 ± 0.0049	0.9891 ± 0.0021	0.9912 ± 0.0017	0.9769 ± 0.0025	0.9118 ± 0.0250	0.8951 ± 0.0207	0.8662 ± 0.0062
VAE	1/1/1	NRMSE ↓	0.0046 ± 0.0007	0.0058 ± 0.0006	0.0301 ± 0.0008	0.0040 ± 0.0006	0.0037 ± 0.0006	0.0048 ± 0.0003	0.0440 ± 0.0002	0.0440 ± 0.0002	0.0450 ± 0.0002
		SSIM ↑	0.9956 ± 0.0012	0.9929 ± 0.0012	0.8818 ± 0.0029	0.9869 ± 0.0040	0.9897 ± 0.0033	0.9855 ± 0.0023	0.8695 ± 0.0014	0.8697 ± 0.0014	0.8689 ± 0.0017
KAE-Info†	1/1/1	NRMSE ↓	0.0623 ± 0.0152	0.1215 ± 0.0692	0.1341 ± 0.0683	0.0224 ± 0.0005	0.0255 ± 0.0002	0.0264 ± 0.0005	0.1008 ± 0.0189	0.1286 ± 0.0164	0.1782 ± 0.0737
		SSIM ↑	0.8355 ± 0.0218	0.7468 ± 0.0944	0.7487 ± 0.0966	0.7866 ± 0.0110	0.7535 ± 0.0065	0.7425 ± 0.0131	0.6882 ± 0.0190	0.5831 ± 0.0748	0.4893 ± 0.1436
EchoKoop-1f	1/1/1	NRMSE ↓	0.0075 ± 0.0006	0.0076 ± 0.0005	0.0131 ± 0.0010	0.0036 ± 0.0001	0.0030 ± 0.0001	0.0034 ± 0.0003	0.0264 ± 0.0003	0.0327 ± 0.0001	0.0478 ± 0.0012
		SSIM ↑	0.9918 ± 0.0014	0.9914 ± 0.0015	0.9815 ± 0.0074	0.9884 ± 0.0009	0.9914 ± 0.0011	0.9897 ± 0.0026	0.9358 ± 0.0023	0.9154 ± 0.0015	0.8687 ± 0.0010
Multi-frame inference											
Delay-DMD	16/12/12	NRMSE ↓	0.0020 ± 0.0000	0.0087 ± 0.0000	0.0233 ± 0.0001	0.0029 ± 0.0000	0.0046 ± 0.0000	0.0074 ± 0.0000	0.0301 ± 0.0000	0.0382 ± 0.0001	0.0464 ± 0.0002
		SSIM ↑	0.9983 ± 0.0000	0.9814 ± 0.0000	0.9115 ± 0.0003	0.9919 ± 0.0001	0.9752 ± 0.0002	0.9419 ± 0.0003	0.9157 ± 0.0002	0.8902 ± 0.0004	0.8631 ± 0.0008
FlowMixer	16/12/12	NRMSE ↓	–	–	–	0.0089 ± 0.0000	0.0116 ± 0.0000	0.0147 ± 0.0000	0.0441 ± 0.0018	0.0498 ± 0.0013	0.0558 ± 0.0013
		SSIM ↑	–	–	–	0.9657 ± 0.0000	0.9346 ± 0.0005	0.8948 ± 0.0009	0.8693 ± 0.0046	0.8395 ± 0.0026	0.8122 ± 0.0008
ConvLSTM	16/12/12	NRMSE ↓	0.0003 ± 0.0000	0.0004 ± 0.0000	0.0029 ± 0.0004	0.0018 ± 0.0001	0.0019 ± 0.0001	0.0027 ± 0.0002	0.0146 ± 0.0000	0.0228 ± 0.0001	0.0442 ± 0.0004
		SSIM ↑	1.0000 ± 0.0000	0.9999 ± 0.0000	0.9982 ± 0.0012	0.9964 ± 0.0004	0.9964 ± 0.0005	0.9938 ± 0.0014	0.9793 ± 0.0003	0.9582 ± 0.0006	0.8705 ± 0.0015
EchoKoop	16/12/12	NRMSE ↓	0.0048 ± 0.0002	0.0050 ± 0.0002	0.0089 ± 0.0017	0.0033 ± 0.0001	0.0030 ± 0.0001	0.0034 ± 0.0003	0.0280 ± 0.0008	0.0337 ± 0.0003	0.0486 ± 0.0002
		SSIM ↑	0.9956 ± 0.0010	0.9953 ± 0.0011	0.9903 ± 0.0005	0.9903 ± 0.0014	0.9924 ± 0.0014	0.9911 ± 0.0017	0.9290 ± 0.0033	0.9110 ± 0.0017	0.8651 ± 0.0022

Notes. C/D/E denotes Cylinder/Dam/ERA5. All methods use shared task-specific splits, normalization, and exact-endpoint evaluation. KAE-Info† follows the released InformationKoopman latent-dynamics mechanism (Cheng et al., 2026), including neighborhood InfoNCE, VNE regularization, and a detached ridge-fitted shared latent linear operator estimated solely from training trajectories. The operator is frozen during validation and test rollout. Its training windows contain 16/16/12 frames, whereas test-time rollout begins from one encoded forecast-origin state; it is therefore listed under single-frame inference. "–" denotes that all three Cylinder FlowMixer runs terminated with non-finite gradients.

Table 20: Resource measurements for EchoKoop on the physical-field datasets across three seeds. Standard deviations are shown where reported.

Dataset	Params. (M)	Epochs	Train (min)	Peak memory (GiB)
Cylinder	7.482	52.3 ± 22.0	154.5 ± 63.7	1.458
Dam	7.482	148.3 ± 61.2	51.5 ± 20.7	1.092
ERA5	6.744	59.0 ± 13.9	428.7 ± 76.2	0.351

Peak memory is the mean, over seeds, of the largest PyTorch-reported accelerator allocation across recorded phases; it excludes driver-resident memory. The current implementation executes the frozen transitions densely and does not exploit their 2×2 block structure through specialized kernels. Accordingly, we claim parameter-efficient latent dynamics rather than minimal end-to-end runtime or memory usage.

J.6 Limitations of the Physical-Field Evaluation

The physical-field evaluation uses the public CFDBench and ERA5 datasets[2]. The experiments evaluate predictive similarity using NRMSE and SSIM. These metrics do not directly measure conservation, equation residuals, boundary-condition violations, spectral energy transfer, or long-term statistical stationarity. The results therefore show that EchoKoop can forecast high-dimensional spatial fields accurately under the evaluated distributions, but they do not establish that its predictions are physically valid outside the observed regime.

[2] `https://github.com/luo-yining/CFDBench`; `https://cds.climate.copernicus.eu/datasets/reanalysis-era5-pressure-levels`