LEARNING LATENT REPRESENTATIONS FOR LONG-TERM CHAOTIC FORECASTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Long-term autonomous forecasting of chaotic dynamical systems is challenging because small errors can progressively influence subsequent evolution under chaotic dynamics. Latent-space prediction has become an important paradigm for modeling complex dynamical systems. However, achieving long-term chaotic forecasting in latent space remains an open problem: models often struggle to preserve long-term attractor structure and dynamical characteristics during sustained autonomous rollout. Forecasting performance depends not only on the learned representation itself, but also on whether that representation can support sustained autonomous evolution under a downstream prediction model. We therefore propose Latent Chaotic Time Series Forecasting (LatentCTSF), a two-stage latent forecasting framework for chaotic systems. In the first stage, LatentCTSF constructs a Robust–Residual latent representation from multi-scale histories. Future prediction and autonomous rollout are used to improve its forecastability. We further introduce finite-horizon relative dynamics with gradient protection to capture the sensitivity of chaotic systems while limiting conflicts with the primary objectives. In the second stage, the encoder and decoder are frozen. We integrate various time-series forecasting backbones into a unified incremental forecasting framework within the latent space. For high-dimensional systems, cross-variable and spatial dependencies are further modeled explicitly. Experiments on representative chaotic systems show that LatentCTSF consistently improves short-term prediction accuracy, substantially extends the valid prediction time across different forecasting backbones, and better preserves long-term dynamical statistics.

1 INTRODUCTION

Chaotic dynamical systems arise throughout atmospheric science, climate modeling, fluid dynamics, and other nonlinear processes (Lorenz, 1963; Kochkov et al., 2024; Racca et al., 2023). Their sensitivity to initial conditions and local perturbations makes long-range prediction intrinsically difficult (Lorenz, 1963; Aurell et al., 1997). In low-dimensional chaotic systems, finite prediction errors can grow rapidly even when the state is represented by only a few variables (Aurell et al., 1997). High-dimensional chaotic systems introduce an additional challenge because their evolution can depend on complex coupling across state variables or spatial locations (Pathak et al., 2018; Vlachas et al., 2020; Gilpin, 2023). Existing approaches address these difficulties from several directions. Recurrent neural networks, including LSTMs, learn temporal evolution directly from state histories (Vlachas et al., 2018; 2020); Transformer and attractor-memory models capture long-range or multiscale interactions (He et al., 2025; Hu et al., 2024); and reservoir computing learns autonomous dynamics through a recurrent state expansion (Pathak et al., 2017; 2018). Stabilized neural ODEs and invariant-measure objectives address rollout stability and long-run statistics (Linot et al., 2023; Jiang et al., 2023; Schiff et al., 2024). Koopman learning and PFNN instead impose structured evolution in a lifted space, with PFNN separating contraction and measure-invariant phases (Lusch et al., 2018; Cheng et al., 2025).

In recent years, nonlinear manifolds, multiscale effective dynamics, neural ODE reduction, spectral submanifolds, and generative effective dynamics have expanded latent-space forecasting (Lee & Carlberg, 2020; Vlachas et al., 2022; Linot & Graham, 2022; Cenedese et al., 2022; Gao et al., 2024). LatentTSF motivates this direction from the observation that measurements may be incomplete or noisy, and shifts the forecasting task into a learned latent state space (Yang et al., 2026). CAE–ESN

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.

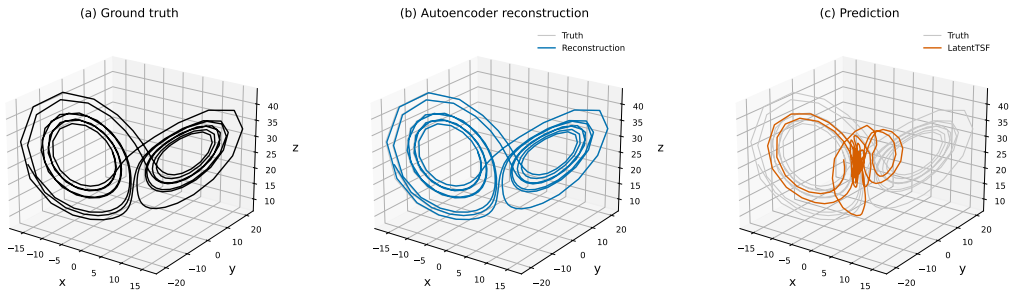

Figure 1: Reconstruction–forecast mismatch for LatentTSF on the same Lorenz–63 system. (a) Ground-truth evolution over the 300-step forecast interval. (b) Reconstruction obtained by encoding and decoding those true states with the frozen pointwise autoencoder. (c) The native LatentTSF prediction of all 300 future states from a 30-step observed context; gray denotes the target trajectory. Accurate state reconstruction does not make the resulting latent sequence equally easy to predict over an observation-free horizon.

and β-VAE–Transformer demonstrate the effectiveness of learning representations and temporal evolution models separately, while LE-PDE and Koopman autoencoders further incorporate prediction and evolution constraints into representation learning (Racca et al., 2023; Solera-Rico et al., 2024; Wu et al., 2022; Lusch et al., 2018; Azencot et al., 2020). These advances also reveal an issue that warrants further study: latent representations with similar reconstruction accuracy can exhibit different forecasting capabilities. Our Lorenz–63 experiment provides related evidence: LatentTSF accurately reconstructs unseen states, yet its direct multi-step forecasts deviate from the attractor trajectory (Figure 1). In LatentTSF, the latent space is fixed after reconstruction pretraining, while subsequent prediction and alignment objectives optimize only the forecasting backbone. This leaves the representation itself unchanged by the downstream forecasting task, motivating us to investigate whether forecasting supervision can be introduced earlier, during representation learning, before training the final predictor. Moreover, the rolling forecast continuously advances model-generated states, so prediction errors affect subsequent inputs, and error amplification in chaotic dynamics makes this issue more pronounced. Therefore, building on existing prediction-oriented representation learning, we further investigate how short-horizon autonomous training can shape latent states that support multiple independently trained forecasting backbones.

We introduce LatentCTSF, a two-stage latent modeling framework for autonomous chaotic forecasting. Stage 1 targets autonomous predictability at the representation level. A gated multi-scale history encoder extracts evolution patterns over different temporal ranges and organizes the latent state into functionally complementary robust and predictive-residual components. Beyond reconstruction, multi-horizon future prediction and a low-capacity autonomous probe directly supervise whether the representation can support future evolution. We further learn the finite-horizon relative evolution of recurrent state pairs and protect the base representation objectives through gradient-conflict projection and a module-wise magnitude budget (Aurell et al., 1997; Yu et al., 2020). Stage 2 removes the auxiliary prediction modules, freezes the encoder and decoder, and trains an independent latent dynamics model from scratch. The predictor evolves latent states autonomously through an incremental parameterization. For high-dimensional systems, the transition retains a structured latent map and models coupling across variables in multivariate states or across locations in spatial fields. The framework therefore improves autonomous forecasting through two separately testable components: construction of forecast-compatible latent states and structure-matched latent dynamics.

Our contributions are:

- We investigate how latent representations can be explicitly shaped for autonomous chaotic evolution before training the final forecasting model. We further examine whether the learned representations remain effective across independently trained predictors in both low- and high-dimensional systems.
- We propose LatentCTSF, which explicitly shapes latent representations with future prediction and autonomous-evolution supervision before training the final forecasting model. We further introduce protected finite-horizon relative-evolution learning for chaotic dynamics, while the final predictor advances latent states through incremental autonomous rollout.

- We establish a three-tier evaluation comprising paired Raw–Latent comparisons across temporal backbones, representation controls under a shared Transformer backbone, and complete-pipeline comparisons. The evaluation covers short-term trajectory accuracy, valid prediction time, and long-run dynamical and statistical behavior.

2 Related Work

2.1 Chaotic Time Series Forecasting

Data-driven chaotic forecasting spans recurrent networks (Vlachas et al., 2018; 2020), reservoir computing (Pathak et al., 2017; 2018), and attention models (He et al., 2025), with systematic benchmarks comparing these model families across diverse chaotic systems (Gilpin, 2023). Multi-step training, generalized teacher forcing, Horizon Forcing, and attractor-aware memory address recursive error growth or multiscale temporal structure (Vlachas et al., 2020; Hess et al., 2023; Zhuang et al., 2025; Hu et al., 2024). Structured alternatives include stabilized neural ODEs (Linot et al., 2023), Fourier neural operators (Li et al., 2021), invariant-measure training (Jiang et al., 2023; Schiff et al., 2024), MNO (Li et al., 2022), and PFNN (Cheng et al., 2025). Recurrence-based forecasting and finite-size predictability instead characterize finite-amplitude error growth (Sugihara & May, 1990; Aurell et al., 1997). Existing methods have not systematically studied autonomous chaotic forecasting in latent space across different predictive backbones.

2.2 Latent Dynamics Forecasting

Latent forecasting separates representation from temporal evolution through nonlinear manifolds, multiscale effective dynamics, neural ODE reduction, spectral submanifolds, and generative reconstruction (Lee & Carlberg, 2020; Vlachas et al., 2022; Linot & Graham, 2022; Cenedese et al., 2022; Gao et al., 2024). CAE–ESN combines convolutional reduction with reservoir dynamics (Racca et al., 2023); β-VAE–Transformer pairs variational compression with attention (Solera-Rico et al., 2024); and LatentTSF trains a separate forecaster using latent prediction and alignment objectives (Yang et al., 2026). LE-PDE jointly optimizes its encoder and latent evolution model (Wu et al., 2022), whereas Koopman autoencoders favor coordinates with approximately linear recurrence (Lusch et al., 2018; Azencot et al., 2020). These objectives impose different latent geometries, so reconstruction quality alone does not establish suitability for repeated autonomous evolution.

2.3 Autonomous Dynamics Modeling

Autonomous forecasting repeatedly feeds predictions back into the transition, making rollout training and stable long-horizon optimization essential (Vlachas et al., 2020; Hess et al., 2023; Zhuang et al., 2025). Increment prediction parameterizes each step as a correction to the current state. For high-dimensional dynamics, Fourier operators model global modes (Li et al., 2021), graph simulators encode local interactions (Sanchez-Gonzalez et al., 2020), and reservoir or attention models preserve coupling across variables (Pathak et al., 2018; He et al., 2025). The remaining challenge is to combine closed-loop training with transitions matched to vector or structured latent states without conflating representation quality with predictor capacity.

3 Method

3.1 Overview

Given an observed history $X_t = (x_{t-L+1}, \ldots, x_t)$ of length L, where x_τ denotes the system state at time τ, we learn an autonomous chaotic forecasting model in two stages.

Stage 1 constructs a forecast-compatible latent sequence $\mathcal{Z}_t = (z_{t-L+1}, \ldots, z_t)$. Each latent state is decomposed into a robust component z_τ^{rob} and a predictive-residual component z_τ^{res}. Representation learning is jointly driven by reconstruction, representation regularization, and predictive dynamics. An auxiliary guarded-response branch further constrains finite-horizon relative state evolution.

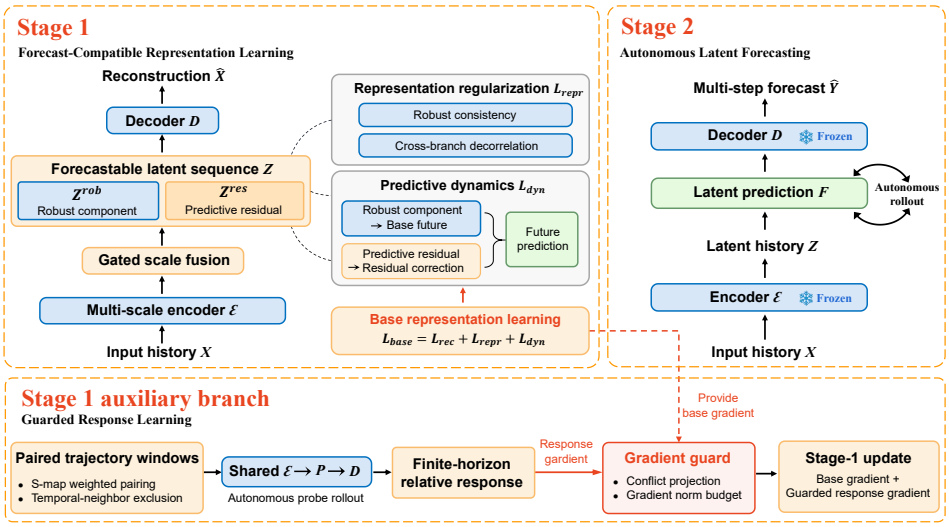

Figure 2: Overview of the proposed framework. Stage 1 constructs a Robust–Residual latent sequence through multi-scale history encoding and predictive representation learning. The guarded-response branch introduces auxiliary supervision on finite-horizon relative evolution. Stage 2 freezes the encoder and decoder and trains an independent latent predictor for autonomous multi-step forecasting.

After Stage 1, all auxiliary prediction modules are removed. The encoder E_θ and decoder D_ϕ are frozen, and Stage 2 trains an independent latent dynamics model F_ψ through incremental autonomous rollout.

3.2 Stage 1: Forecast-Compatible Representation Learning

Multi-scale latent construction. For each time index $\tau \in \{t-L+1, \dots, t\}$, let $X_\tau = (x_{t-L+1}, \dots, x_\tau)$ denote the available history prefix. The encoder extracts S features $\{h_\tau^{(s)}\}_{s=1}^S$ using temporal branches with different receptive fields. A learnable gate G_θ assigns scale-dependent weights $a_{\tau,s}$ and produces the fused representation

$$a_{\tau,1:S} = \mathrm{softmax}\, G_\theta([h_\tau^{(1)}; \dots; h_\tau^{(S)}]), \qquad h_\tau = \sum_{s=1}^{S} a_{\tau,s} h_\tau^{(s)}, \tag{1}$$

where $[\cdot;\cdot]$ denotes feature concatenation. Only observations available up to time τ are used.

The encoder E_θ maps the fused history representation to

$$z_\tau = E_\theta(X_\tau) = [z_\tau^{\mathrm{rob}}, z_\tau^{\mathrm{res}}], \tag{2}$$

where z_τ^{rob} denotes the robust component and z_τ^{res} the predictive-residual component. For vector-valued systems, z_τ is a latent vector. For periodic or spatially indexed systems, the latent state retains its structured map representation.

Base representation objective. Stage 1 optimizes

$$\mathcal{L}_{\mathrm{base}} = \mathcal{L}_{\mathrm{rec}} + \mathcal{L}_{\mathrm{repr}} + \mathcal{L}_{\mathrm{dyn}}, \tag{3}$$

where $\mathcal{L}_{\mathrm{rec}}$ reconstructs all states in the encoded history, $\mathcal{L}_{\mathrm{repr}}$ regularizes the Robust–Residual representation, and $\mathcal{L}_{\mathrm{dyn}}$ introduces predictive supervision.

The reconstruction objective is

$$\mathcal{L}_{\mathrm{rec}} = \frac{1}{L} \sum_{\ell=1}^{L} \| D_\phi(z_{t-L+\ell}) - x_{t-L+\ell} \|_2^2, \tag{4}$$

where D_ϕ is the decoder parameterized by ϕ.

For representation regularization, we construct a second history view by masking part of the earlier observations while retaining the most recent states. The robust component is encouraged to remain consistent across the two views, with variance and covariance regularization used to prevent collapse (Bardes et al., 2022). A cross-branch decorrelation term further discourages duplicated directions between z_t^{rob} and z_t^{res}. We group these terms as $\mathcal{L}_{\mathrm{repr}}$, their detailed definitions are given in Appendix A.1.

Predictive dynamics. To make the latent state informative for future evolution, we introduce multi-horizon future prediction and a low-capacity autonomous probe.

Let $\mathcal{K}$ denote the set of supervised forecast offsets. For each $k \in \mathcal{K}$, a base head B_k predicts the future state from z_t^{rob}, while a correction head C_k refines the detached base prediction:

$$\mathcal{L}_{\mathrm{future}} = \frac{1}{2|\mathcal{K}|} \sum_{k\in\mathcal{K}} \Big[\|B_k(z_t^{\mathrm{rob}}) - x_{t+k}\|_2^2 + \| \operatorname{sg}(B_k(z_t^{\mathrm{rob}})) + C_k(\chi_t) - x_{t+k}\|_2^2 \Big], \tag{5}$$

where χ_t denotes the correction input and $\operatorname{sg}(\cdot)$ blocks gradient propagation through the base prediction.

Future heads are conditioned on encoded observations and therefore do not experience their own prediction errors. We additionally introduce a low-capacity latent transition P_ω, parameterized by ω, and recursively apply it from z_t. The decoded rollout is supervised in physical space:

$$\mathcal{L}_{\mathrm{auto}} = \frac{1}{|\mathcal{K}|} \sum_{h\in\mathcal{K}} \frac{1}{h} \sum_{k=1}^{h} \left\| D_\phi(P_\omega^k(z_t)) - x_{t+k} \right\|_2^2, \tag{6}$$

where P_ω^k denotes k recursive applications of the same probe.

The predictive-dynamics objective is

$$\mathcal{L}_{\mathrm{dyn}} = \lambda_{\mathrm{future}} \mathcal{L}_{\mathrm{future}} + \lambda_{\mathrm{auto}} \mathcal{L}_{\mathrm{auto}}, \tag{7}$$

where $\lambda_{\mathrm{future}}$ and λ_{auto} control the two terms. All Stage 1 prediction heads and P_ω are removed after pretraining.

3.3 Guarded Response Learning

Pointwise prediction constrains each future state independently. We additionally supervise how the difference between paired states changes over a finite horizon. Candidate pairs are sampled from the training trajectories using temporal-neighbor exclusion and S-map-weighted recurrence sampling; the detailed pairing procedure is provided in Appendix A.2.

For a selected pair indexed by i and j, let $z_i = E_\theta(X_i)$ and $z_j = E_\theta(X_j)$ denote their terminal latent states. At forecast offset k, the observed and predicted relative changes are

$$\Delta x_k = x_{j+k} - x_{i+k}, \qquad \Delta \widehat{x}_k = D_\phi(P_\omega^k(z_j)) - D_\phi(P_\omega^k(z_i)). \tag{8}$$

Let $\mathcal{K}_r$ denote the response horizons. We define

$$\mathcal{L}_{\mathrm{resp}} = \frac{1}{|\mathcal{K}_r|} \sum_{k\in\mathcal{K}_r} \frac{\mathrm{MSE}(\Delta\widehat{x}_k, \Delta x_k)}{c_k}, \tag{9}$$

where $c_k > 0$ is a horizon-specific normalization factor estimated from the training data. This objective models finite-horizon relative evolution rather than an infinitesimal Lyapunov quantity (Aurell et al., 1997).

Because response supervision may conflict with the base representation objectives, its gradient is not directly added to $\mathcal{L}_{\mathrm{base}}$. For each parameter group $m \in \{E, D, P\}$, let g_r^m and g_{base}^m denote the response and base gradients. We define a guarded response gradient

$$\widetilde{g}_r^m = \mathrm{Guard}\left(g_r^m; \mathcal{G}_m, b_m\right), \qquad g_{\mathrm{update}}^m = g_{\mathrm{base}}^m + \widetilde{g}_r^m, \tag{10}$$

where $\mathcal{G}_m$ contains the protected gradients for parameter group m, and b_m is a module-wise magnitude budget. $\mathrm{Guard}(\cdot)$ projects the response gradient onto directions that do not locally conflict with the protected objectives and then bounds its norm. The exact projection and budget definitions are given in Appendix A.3. The future-prediction heads receive only the base gradient.

3.4 Stage 2: Autonomous Latent Forecasting

After Stage 1, the auxiliary prediction modules and response branch are discarded. The pretrained encoder E_θ and decoder D_ϕ are frozen, and an independently initialized latent dynamics model F_ψ is trained.

Let $S_t = (z_{t-L_z+1}, \ldots, z_t)$ denote a latent history of length L_z. Rather than directly predicting the next latent state, F_ψ predicts its increment:

$$\Delta\widehat{z}_{t+1} = F_\psi(S_t), \qquad \widehat{z}_{t+1} = z_t + \Delta\widehat{z}_{t+1}. \tag{11}$$

For vector-valued systems, F_ψ can be instantiated with different temporal forecasting backbones. For high-dimensional periodic or spatial states, the latent map is retained and the transition additionally models coupling across variables or spatial locations using shared temporal modeling and circular spatial operators. The detailed structured transition is provided in Appendix A.4.

Training uses autonomous composition: each predicted latent state is appended to the context and reused for the next step. Let $\widehat{\mathcal{Z}}$ denote the recursively predicted latent trajectory and $\mathcal{Z}$ the corresponding target trajectory encoded by the frozen E_θ. The Stage 2 objective is

$$\mathcal{L}_{\mathrm{stage2}} = \mathrm{MSE}(\widehat{\mathcal{Z}}, \mathcal{Z}). \tag{12}$$

Finally, each predicted latent state is mapped back to physical space by the frozen decoder:

$$\widehat{x}_{t+k} = D_\phi(\widehat{z}_{t+k}), \qquad \widehat{Y}_t = (\widehat{x}_{t+1}, \ldots, \widehat{x}_{t+H}), \tag{13}$$

where H is the forecasting horizon and $\widehat{Y}_t$ denotes the multi-step physical prediction. Only the initial history is observed; all subsequent states are generated autonomously.

4 Experiments

4.1 Experimental Setup

Datasets. We evaluate Lorenz–63, Lorenz–96, and Kuramoto–Sivashinsky (KS), spanning low-dimensional, high-dimensional multivariate, and spatially extended chaos. Thirty sampled steps correspond to one nominal Lyapunov time in each system. All methods use the same train, validation, and test trajectories, with normalization fitted only on the training partition. The governing equations, numerical solvers, sampling procedures, and splits are provided in Appendices B.1 and B.2.

Baselines. We organize comparisons into three groups. Complete-pipeline baselines include CAE–ESN (Racca et al., 2023), LatentTSF (Yang et al., 2026), and PFNN (Cheng et al., 2025). Representation controls compare raw states, point autoencoding (Hinton & Salakhutdinov, 2006), β-VAE (Solera-Rico et al., 2024), time-lagged AE (Wehmeyer & Noé, 2018), Koopman AE (Lusch et al., 2018; Azencot et al., 2020), and our representation under the same incremental Transformer. Backbone controls pair GRU, LSTM, TCN, and Transformer predictors with either raw observations or the frozen proposed representation. Their comparison roles and shared interfaces are detailed in Appendix B.3.

Implementation details. Our method follows the two-stage procedure: Stage 1 learns the representation, after which the encoder and decoder are frozen and Stage 2 trains the autonomous latent predictor. The input history spans 30 time steps. Stage 2 is trained over 10 autoregressive steps, whereas evaluation extends the autonomous rollout to 300 steps. All experiments are conducted on an NVIDIA RTX 4080 SUPER-based virtual GPU with a 32 GB memory profile. Architecture, optimization, loss-weight, and checkpoint details are given in Appendix B.4.

Table 1: Paired state-interface comparison across temporal backbones. Raw predicts observations; Ours predicts from the frozen proposed latent state. $\bar{D}_{\mathrm{stsp}}$ is the dimension-normalized joint-state divergence over the complete 300-step rollout. Higher VPT is better; lower MSE@30 and $\bar{D}_{\mathrm{stsp}}$ are better. Within each Raw–Ours pair, the better result is bold.

		Lorenz–63			Lorenz–96			KS		
Backbone	State	VPT (LT)↑	MSE@30↓	$\bar{D}_{\mathrm{stsp}}$ ↓	VPT (LT)↑	MSE@30↓	$\bar{D}_{\mathrm{stsp}}$ ↓	VPT (LT)↑	MSE@30↓	$\bar{D}_{\mathrm{stsp}}$ ↓
GRU	Raw	4.71	0.0013	**0.0082**	0.00	0.5564	0.4485	0.01	0.5208	0.4202
	Ours	**5.04**	**0.0002**	0.0131	**2.14**	**0.0074**	**0.3586**	**1.03**	**0.0576**	**0.3844**
LSTM	Raw	4.92	**0.0042**	0.0125	0.00	0.7940	0.4598	0.00	0.7738	0.4371
	Ours	**5.05**	0.0206	**0.0101**	**2.32**	**0.0060**	**0.3390**	**1.02**	**0.0566**	**0.3901**
TCN	Raw	2.94	0.0701	0.0245	0.01	0.5156	0.4444	0.04	0.4369	0.3788
	Ours	**4.50**	**0.0146**	**0.0106**	**2.34**	**0.0062**	**0.3502**	**1.01**	**0.0489**	**0.3590**
Transformer	Raw	2.50	0.0175	0.0133	0.02	0.3310	0.4627	0.37	0.1365	0.4334
	Ours	**4.71**	**0.0053**	**0.0132**	**2.36**	**0.0055**	**0.3504**	**1.36**	**0.0287**	**0.3678**

Metrics. For each 300-step autonomous rollout, the normalized Mean Squared Error over the first 30 steps (MSE@30) measures short-term prediction accuracy, while the Valid Prediction Time (VPT) is defined as the first time at which the cumulative symmetric Mean Absolute Percentage Error (sMAPE) exceeds 30%. VPT is reported in Lyapunov Times (LT), with higher values indicating better performance. We further report the dimension-normalized state-space divergence $\bar{D}_{\mathrm{stsp}}$ over all 300 predicted states. This metric is based on the Kullback–Leibler (KL) divergence $D_{\mathrm{KL}}(p_{\mathrm{true}} \| p_{\mathrm{pred}})$ between Gaussian Mixture Model (GMM) estimates of the target and predicted finite-window state distributions.

4.2 Forecasting Performance

Transfer across forecasting backbones. Table 1 first tests whether the latent representation benefits different times series forecasting models. For each GRU, LSTM, TCN, or Transformer pair, the Raw model predicts normalized observations directly, whereas LatentCTSF predicts increments from the frozen latent state. The proposed representation consistently improves autonomous forecasting across all four backbone predictors, with the gains being particularly pronounced on the two high-dimensional systems. On Lorenz–96 and KS, every backbone benefits from substantially longer VPT and lower short-term errors after replacing the raw state with the learned latent state. This trend holds for recurrent, convolutional, and attention-based predictors, indicating that the improvement does not depend on a particular forecasting architecture. The gains are smaller on Lorenz–63, where direct prediction from the low-dimensional physical state is already strong, but VPT still improves for all four backbones. Overall, these results show that the proposed representation provides a more favorable state interface for long-term forecasting rather than merely improving a specific downstream predictor.

Complete pipelines and representation controls. Table 2 further evaluates the method from two complementary perspectives. In this experiment, LatentCTSF is instantiated with the Transformer as its Stage 2 forecasting backbone. The upper block compares complete forecasting pipelines using their native representations and forecasting models, whereas the lower block fixes the same incremental Transformer for all methods and changes only the state representation. The latter therefore isolates the contribution of representation learning from that of the forecasting backbone.

Compared with existing complete pipelines, LatentCTSF achieves the strongest trajectory forecasting performance across all three systems, with the advantage becoming especially clear on Lorenz–96 and KS. Existing latent forecasting pipelines deteriorate rapidly under long-term rollout on these higher-dimensional systems, whereas the proposed framework maintains substantially longer trajectory agreement and lower early-horizon prediction error. This suggests that explicitly shaping the latent state for long-term chaotic forecasting becomes increasingly important as the underlying dynamics grow more complex.

The controlled representation comparison further supports this conclusion. Under the same incremental Transformer and identical Stage 2 training protocol, the proposed representation achieves the best VPT and MSE@30 on all three systems. The finite-window state-space divergence also

Table 2: Autonomous forecasting under the $H = 300$ test protocol. The upper block compares complete forecasting pipelines. The lower block freezes each representation and uses the same incremental Transformer and Stage 2 training protocol.

	Lorenz–63			Lorenz–96			KS		
Method	VPT (LT)↑	MSE@30↓	$\bar{D}_{\text{stsp}}$ ↓	VPT (LT)↑	MSE@30↓	$\bar{D}_{\text{stsp}}$ ↓	VPT (LT)↑	MSE@30↓	$\bar{D}_{\text{stsp}}$ ↓
Complete forecasting pipelines									
CAE–ESN	4.62	0.0114	0.0181	0.26	0.3545	**0.3494**	0.21	0.5082	0.4259
LatentTSF	0.99	0.0972	0.0669	0.00	0.9683	0.4140	0.00	1.1198	0.5024
PFNN	0.34	1.0059	0.0589	0.05	0.9800	0.7484	0.15	0.5316	0.4399
Transformer-backed representation controls									
Raw state	2.50	0.0175	0.0133	0.02	0.3310	0.4627	0.37	0.1365	0.4334
Point AE	3.94	0.0424	**0.0119**	0.25	0.2899	0.3750	0.27	0.2529	**0.3196**
β-VAE	2.88	0.0149	0.0372	0.73	0.0294	0.3861	0.91	0.0327	0.3697
Time-lagged AE	4.05	0.0874	0.0345	0.01	0.4389	0.3616	0.14	0.3405	0.3450
Koopman AE	3.75	0.1024	0.0236	0.00	0.4869	0.3895	0.32	0.2276	0.3358
Ours	**4.71**	**0.0053**	0.0132	**2.36**	**0.0055**	0.3504	**1.36**	**0.0287**	0.3678

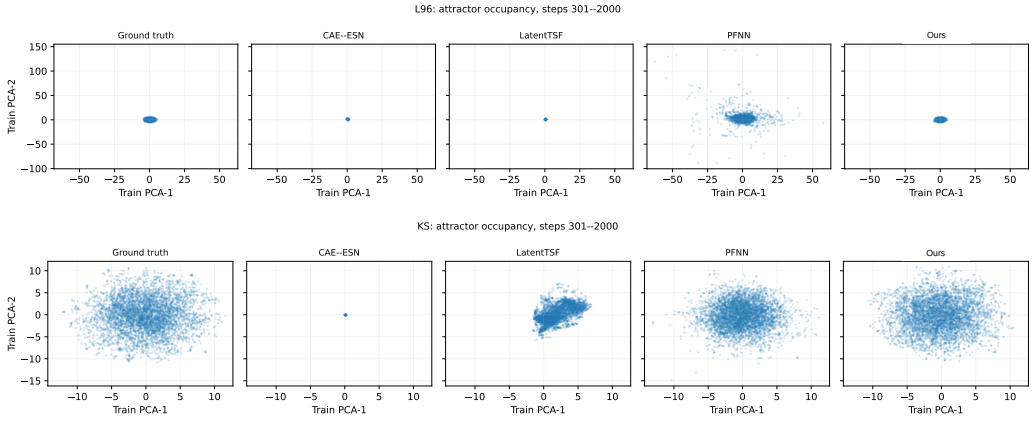

Figure 3: Long-term state-space occupancy over steps 301–2000 for Lorenz–96 (top) and KS (bottom), projected onto the first two components of a common train-only PCA basis.

remains competitive, showing that the gain in trajectory prediction is achieved without a substantial degradation of the overall state-space behavior. Together with the cross-backbone results, these experiments support the central premise of LatentCTSF: latent representations explicitly trained for autonomous predictability provide a more effective basis for long-term chaotic forecasting.

4.3 Long-horizon statistical behavior

Long-horizon evaluation examines whether the autonomous model continues to generate states consistent with the underlying chaotic dynamics after exact trajectory tracking is no longer expected. We therefore visualize the state-space occupancy over steps 301–2000 of independent autonomous rollouts in Figure 3. Lorenz–96 and the KS system are projected onto a common Principal Component Analysis (PCA) basis fitted using the training data only. Unlike trajectory-based metrics, these plots characterize the geometry and coverage of the generated states after phase agreement with the reference trajectory has been lost. Figure 3 shows distinct long-term behaviors across methods. CAE–ESN and LatentTSF exhibit pronounced state-space collapse, while PFNN preserves a broader occupancy but introduces excessive dispersion; in comparison, LatentCTSF more closely matches the ground-truth distribution on both Lorenz–96 and KS. These plots complement trajectory-based metrics by revealing long-term collapse, missing regions, and spurious excursions. Appendix Table A3 reports the Maximum Mean Discrepancy (MMD) between long-term state distributions and the relative spectral error over the same post-transient interval. The results are consistent with the occupancy analysis, with LatentCTSF showing stronger agreement with the target long-term statistics across the evaluated systems. Complete definitions and implementation details are provided in Appendix C.

Table 3: Stage-wise ablations under the common $H = 300$ protocol. Stage 1 progressively adds objective groups while fixing the encoder architecture and predictor configuration. Stage 2 uses the full frozen representation and varies either the output parameterization on Lorenz–63 or the treatment of the 2×20 latent map on Lorenz–96. The best result within each ablation block is bold.

Stage	System	Configuration	VPT (LT)↑	MSE@30↓	$\bar{D}_{\mathrm{stsp}}$ ↓
Stage 1	Lorenz–63	Reconstruction only	3.07	0.0704	0.0346
		+ representation regularization	3.03	0.0300	0.0269
		+ predictive dynamics	4.58	0.0623	0.0136
		+ guarded response	**4.71**	**0.0053**	**0.0132**
	Lorenz–96	Reconstruction only	1.70	0.0110	0.3631
		+ representation regularization	1.97	0.0085	0.3685
		+ predictive dynamics	2.12	0.0075	**0.3500**
		+ guarded response	**2.36**	**0.0055**	0.3504
Stage 2	Lorenz–63	Direct latent prediction	3.50	**0.0045**	0.0142
		Incremental latent prediction	**4.71**	0.0053	**0.0132**
	Lorenz–96	Incremental, flattened vector Transformer	0.21	0.2904	0.3854
		Incremental, periodic-map Transformer	**2.36**	**0.0055**	**0.3504**

4.4 Stage-wise ablations

Ablation settings. We examine five components of LatentCTSF. For Stage 1, we start from reconstruction-only training and cumulatively introduce: **(1) representation regularization**, including history-view regularization and cross-branch decorrelation; **(2) predictive dynamics**, comprising multi-horizon future prediction and the low-capacity autonomous probe; **(3) guarded response**, combining paired-state supervision with gradient-conflict projection and a module-wise norm budget. The encoder architecture and predictor configuration remain fixed within each system. For Stage 2, we freeze the full pretrained representation and separately evaluate: **(4) output parameterization**, comparing direct and incremental latent prediction on Lorenz–63; and **(5) structured transition modeling**, comparing flattened-vector and periodic-map Transformers on Lorenz–96, both using incremental prediction.

Table 3 shows that the full Stage 1 configuration achieves the best VPT and MSE@30 on both systems. Representation regularization mainly improves short-term accuracy, predictive dynamics strengthens sustained trajectory tracking, and guarded response provides further gains; since the encoder and downstream predictor remain unchanged, these results support the contribution of the Stage 1 objectives to the forecastability of the learned representation. More detailed component ablations are reported in Appendix E. With the full representation fixed, the Stage 2 ablations further show that incremental prediction improves long-horizon tracking on Lorenz–63, while preserving the periodic latent map is critical on Lorenz–96, where the structured Transformer clearly outperforms the flattened alternative. Overall, the results indicate that effective autonomous forecasting depends on both forecast-compatible latent representations and transition designs matched to the latent-state structure.

5 Conclusion

We introduced LatentCTSF, a two-stage latent forecasting framework for chaotic dynamics. Stage 1 learns forecast-oriented latent representations through reconstruction, representation regularization, predictive pretraining, and guarded finite-horizon response learning, while Stage 2 freezes the learned representation and trains an independent transition model through autonomous rollout. Experiments on Lorenz–63, Lorenz–96, and KS show that LatentCTSF improves short-term forecasting accuracy and sustains more reliable autonomous prediction over longer horizons, while better preserving the long-term state-space and statistical behavior of the underlying dynamics. Future work will investigate which information in the learned latent representation supports accurate trajectory prediction and which contributes to preserving long-term statistical behavior after trajectory-level agreement is lost. We will further examine whether these two capabilities rely on shared or distinct representational information.

AI USE STATEMENT

Generative AI tools were used in parts of this study to assist with literature search, code debugging, and language polishing of the manuscript. All AI-assisted suggestions, reference information, code, and explanations were independently reviewed and verified by the authors. All experimental results reported in this paper were obtained from the authors' own model implementations and experiments, rather than generated by AI. The authors take full responsibility for the methodology, experimental results, research conclusions, and final content of this work.

REPRODUCIBILITY STATEMENT

To support reproducibility, we provide detailed descriptions of the model architecture, training procedure, datasets, preprocessing steps, evaluation protocol, and hyperparameter settings in the main paper and appendix. The Supplementary Material contains the complete source code, experiment configurations, data generation and preprocessing scripts for reproducing the main results.

REFERENCES

E Aurell, Guido Boffetta, Andrea Crisanti, G Paladin, and Angelo Vulpiani. Predictability in the large: an extension of the concept of lyapunov exponent. *Journal of physics A: Mathematical and general*, 30(1):1–26, 1997.

Omri Azencot, N. Benjamin Erichson, Vanessa Lin, and Michael W. Mahoney. Forecasting sequential data using consistent Koopman autoencoders. In *International conference on machine learning*, pp. 475–485. PMLR, 2020.

Adrien Bardes, Jean Ponce, and Yann LeCun. VICReg: Variance-invariance-covariance regularization for self-supervised learning. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=xm6YD62D1Ub.

Mattia Cenedese, Joar Axås, Bastian Bäuerlein, Kerstin Avila, and George Haller. Data-driven modeling and prediction of non-linearizable dynamics via spectral submanifolds. *Nature communications*, 13(1):872, 2022.

Xiaoyuan Cheng, Yi He, Yiming Yang, Xiao Xue, Sibo Cheng, Daniel Giles, Xiaohang Tang, and Yukun Hu. Learning chaos in a linear way. In *International Conference on Learning Representations*, volume 2025, pp. 52336–52378, 2025.

Han Gao, Sebastian Kaltenbach, and Petros Koumoutsakos. Generative learning for forecasting the dynamics of high-dimensional complex systems. *Nature Communications*, 15(1):8904, 2024.

William Gilpin. Model scale versus domain knowledge in statistical forecasting of chaotic systems. *Physical Review Research*, 5(4):043252, 2023.

Yi He, Yiming Yang, Xiaoyuan Cheng, Hai Wang, Xiao Xue, Boli Chen, and Yukun Hu. Chaos meets attention: Transformers for large-scale dynamical prediction. *arXiv preprint arXiv:2504.20858*, 2025.

Florian Hess, Zahra Monfared, Manuel Brenner, and Daniel Durstewitz. Generalized teacher forcing for learning chaotic dynamics. *arXiv preprint arXiv:2306.04406*, 2023.

Geoffrey E. Hinton and Ruslan R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

Jiaxi Hu, Yuehong Hu, Wei Chen, Ming Jin, Shirui Pan, Qingsong Wen, and Yuxuan Liang. Attractor memory for long-term time series forecasting: A chaos perspective. *Advances in Neural Information Processing Systems*, 37:20786–20818, 2024.

Ruoxi Jiang, Peter Y Lu, Elena Orlova, and Rebecca Willett. Training neural operators to preserve invariant measures of chaotic attractors. *Advances in Neural Information Processing Systems*, 36: 27645–27669, 2023.

Dmitrii Kochkov, Janni Yuval, Ian Langmore, Peter Norgaard, Jamie Smith, Griffin Mooers, Milan Klöwer, James Lottes, Stephan Rasp, Peter Düben, et al. Neural general circulation models for weather and climate. *Nature*, 632(8027):1060–1066, 2024.

Kookjin Lee and Kevin T Carlberg. Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders. *Journal of Computational Physics*, 404:108973, 2020.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=c8P9NQVtmnO.

Zongyi Li, Miguel Liu-Schiaffini, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Learning chaotic dynamics in dissipative systems. *Advances in Neural Information Processing Systems*, 35:16768–16781, 2022.

Alec J Linot and Michael D Graham. Data-driven reduced-order modeling of spatiotemporal chaos with neural ordinary differential equations. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 32(7), 2022.

Alec J Linot, Joshua W Burby, Qi Tang, Prasanna Balaprakash, Michael D Graham, and Romit Maulik. Stabilized neural ordinary differential equations for long-time forecasting of dynamical systems. *Journal of Computational Physics*, 474:111838, 2023.

Edward N. Lorenz. Deterministic nonperiodic flow. *Journal of the Atmospheric Sciences*, 20(2): 130–141, 1963.

Bethany Lusch, J. Nathan Kutz, and Steven L. Brunton. Deep learning for universal linear embeddings of nonlinear dynamics. *Nature Communications*, 9(1):4950, 2018.

Jaideep Pathak, Zhixin Lu, Brian R. Hunt, Michelle Girvan, and Edward Ott. Using machine learning to replicate chaotic attractors and calculate Lyapunov exponents from data. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 27(12), 2017.

Jaideep Pathak, Brian Hunt, Michelle Girvan, Zhixin Lu, and Edward Ott. Model-free prediction of large spatiotemporally chaotic systems from data: A reservoir computing approach. *Physical Review Letters*, 120(2):024102, 2018.

Alberto Racca, Nguyen Anh Khoa Doan, and Luca Magri. Predicting turbulent dynamics with the convolutional autoencoder echo state network. *Journal of Fluid Mechanics*, 975:A2, 2023.

Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter Battaglia. Learning to simulate complex physics with graph networks. In *International conference on machine learning*, pp. 8459–8468. Pmlr, 2020.

Yair Schiff, Zhong Yi Wan, Jeffrey B Parker, Stephan Hoyer, Volodymyr Kuleshov, Fei Sha, and Leonardo Zepeda-Núñez. Dyslim: Dynamics stable learning by invariant measure for chaotic systems. *arXiv preprint arXiv:2402.04467*, 2024.

Alberto Solera-Rico, Carlos Sanmiguel Vila, Miguel Gómez-López, Yuning Wang, Abdulrahman Almashjary, Scott TM Dawson, and Ricardo Vinuesa. β-variational autoencoders and transformers for reduced-order modelling of fluid flows. *Nature Communications*, 15(1):1361, 2024.

George Sugihara and Robert M May. Nonlinear forecasting as a way of distinguishing chaos from measurement error in time series. *Nature*, 344(6268):734–741, 1990.

Pantelis R Vlachas, Wonmin Byeon, Zhong Y Wan, Themistoklis P Sapsis, and Petros Koumoutsakos. Data-driven forecasting of high-dimensional chaotic systems with long short-term memory networks. *Proceedings. Mathematical, physical, and engineering sciences*, 474(2213):20170844, 2018.

Pantelis R. Vlachas, Jaideep Pathak, Brian R. Hunt, Themistoklis P. Sapsis, Michelle Girvan, Edward Ott, and Petros Koumoutsakos. Backpropagation algorithms and reservoir computing in recurrent neural networks for the forecasting of complex spatiotemporal dynamics. *Neural Networks*, 126: 191–217, 2020.

Pantelis R Vlachas, Georgios Arampatzis, Caroline Uhler, and Petros Koumoutsakos. Multiscale simulations of complex systems by learning their effective dynamics. *Nature Machine Intelligence*, 4(4):359–366, 2022.

Christoph Wehmeyer and Frank Noé. Time-lagged autoencoders: Deep learning of slow collective variables for molecular kinetics. *The Journal of Chemical Physics*, 148(24), 2018.

Tailin Wu, Takashi Maruyama, and Jure Leskovec. Learning to accelerate partial differential equations via latent global evolution. In *Advances in Neural Information Processing Systems*, volume 35, pp. 2240–2253, 2022.

Jie Yang, Yifan Hu, Yuante Li, Kexin Zhang, Kaize Ding, and Philip S Yu. From observations to states: Latent time series forecasting. *arXiv preprint arXiv:2602.00297*, 2026.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. In *Advances in Neural Information Processing Systems*, volume 33, pp. 5824–5836, 2020.

Yong Zhuang, Matthew Almeida, Wei Ding, Shafiqul Islam, Zihan Li, and Ping Chen. Horizon forcing: Improving the recurrent forecasting of chaotic systems. *ACM Transactions on Intelligent Systems and Technology*, 16(5):1–22, 2025.

A Additional Method Details

A.1 Representation regularization

For a minibatch of size B, let $u_b = z_{t,b}^{\mathrm{rob}}$ and $\widetilde{u}_b = \widetilde{z}_{t,b}^{\mathrm{rob}}$ denote the robust codes of the original and masked views of sample b. Stacking the codes row-wise gives $U, \widetilde{U} \in \mathbb{R}^{B \times d_{\mathrm{rob}}}$. Their view-consistency loss is

$$\mathcal{L}_{\mathrm{inv}} = \frac{1}{B} \sum_{b=1}^{B} \|u_b - \widetilde{u}_b\|_2^2. \tag{14}$$

Let $\sigma_q(U)$ be the minibatch standard deviation of coordinate q, $\mathrm{Cov}(U)$ the centered sample covariance matrix, and $[a]_+ = \max(0, a)$. The indices $p, q \in \{1, \ldots, d_{\mathrm{rob}}\}$ refer to latent coordinates. For target standard deviation $\gamma > 0$, the variance and covariance penalties are

$$\begin{aligned} \mathcal{V}(U, \widetilde{U}) &= \frac{1}{2d_{\mathrm{rob}}} \sum_{q=1}^{d_{\mathrm{rob}}} \left([\gamma - \sigma_q(U)]_+ + [\gamma - \sigma_q(\widetilde{U})]_+ \right), \\ \mathcal{C}(U, \widetilde{U}) &= \frac{1}{2d_{\mathrm{rob}}} \sum_{p \neq q} \left(\mathrm{Cov}(U)_{pq}^2 + \mathrm{Cov}(\widetilde{U})_{pq}^2 \right). \end{aligned} \tag{15}$$

Let Z^{rob} and Z^{res} denote the minibatch matrices of robust and residual codes, and let $\mathrm{Corr}(Z^{\mathrm{rob}}, Z^{\mathrm{res}})$ be their cross-correlation matrix. The complete representation objective is

$$\begin{aligned} \mathcal{L}_{\mathrm{rob}} &= \mathcal{L}_{\mathrm{inv}} + \alpha_v \mathcal{V}(U, \widetilde{U}) + \alpha_c \mathcal{C}(U, \widetilde{U}), \\ \mathcal{L}_{\mathrm{decorr}} &= \left\| \mathrm{Corr}(Z^{\mathrm{rob}}, Z^{\mathrm{res}}) \right\|_F^2, \\ \mathcal{L}_{\mathrm{repr}} &= \lambda_{\mathrm{rob}} \mathcal{L}_{\mathrm{rob}} + \lambda_{\mathrm{decorr}} \mathcal{L}_{\mathrm{decorr}}. \end{aligned} \tag{16}$$

Here α_v and α_c weight variance preservation and within-branch decorrelation. The covariance and correlation statistics are computed over centered minibatch samples. In Equation 5, $\chi_t = z_t^{\mathrm{res}}$ for a vector state; for a structured latent map, χ_t retains the complete map so that the correction head can use neighboring channels and positions.

A.2 Response-pair construction and normalization

For an anchor window ending at index i, the candidate set $\mathcal{J}_i$ excludes samples whose input or target intervals overlap those of the anchor. Distances d_{ij} are measured between the current physical states after train-set standardization. With $d_{i,\min} = \min_{j\in\mathcal{J}_i} d_{ij}$ and candidate-set mean $\bar{d}_i$, the paired index is drawn according to

$$p(j \mid i) \propto \exp\left[-2\theta \frac{d_{ij} - d_{i,\min}}{\bar{d}_i}\right]. \tag{17}$$

The locality parameter $\theta \geq 0$ controls how strongly sampling favors recurrent states near the anchor. For state dimension d_x, the response scale at horizon k is the training-pair mean squared separation

$$c_k = \mathbb{E}_{(i,j)}\left[\frac{1}{d_x}\|x_{j+k} - x_{i+k}\|_2^2\right]. \tag{18}$$

The candidate set excludes temporal neighbors whose input or target windows overlap. Pair eligibility, distance normalization, θ, and c_k are determined from training trajectories only.

A.3 Gradient projection and norm budget

For parameter group $m \in \{E, D, P\}$, corresponding to the encoder, decoder, and autonomous probe, respectively, the response gradient is normalized as

$$\bar{g}_r^m = \begin{cases} g_r^m / \|g_r^m\|_2, & \|g_r^m\|_2 > 0, \\ 0, & \text{otherwise.} \end{cases} \tag{19}$$

Let $\mathcal{G}_m$ contain the protected base-objective gradients that act on the same parameter group. The closest response direction that is first-order non-conflicting with every protected gradient is

$$q^m = \arg\min_q \|q - \bar{g}_r^m\|_2^2 \quad \text{s.t.} \quad \langle q, g\rangle \geq 0 \quad \forall g \in \mathcal{G}_m. \tag{20}$$

The projected direction is then scaled by a module-wise budget:

$$\begin{aligned} b_m &= \min\left(\|g_{\text{auto}}^m\|_2, \|g_{\text{base}}^m\|_2\right), \\ \widetilde{g}_r^m &= \begin{cases} \min(\|g_r^m\|_2 \|q^m\|_2, b_m) q^m / \|q^m\|_2, & \|q^m\|_2 > 0, \\ 0, & \text{otherwise.} \end{cases} \end{aligned} \tag{21}$$

Here g_{auto}^m is the gradient of the weighted autonomous-probe loss and g_{base}^m is the gradient of the complete base objective. The resulting update is the one stated in Equation 10. The future heads are outside the response path and receive only their base gradients.

A.4 Structured latent transition

For a structured latent state $Z_t \in \mathbb{R}^{C\times P}$, C is the number of latent channels and P the number of periodic positions. Let $\mathcal{C}_{\text{in}}$ and $\mathcal{C}_{\text{out}}$ denote learned circular spatial operators. At position p and time τ,

$$\begin{aligned} h_{\tau,p} &= \left[\mathcal{C}_{\text{in}}(Z_\tau)\right]_{:,p} + e_\tau, \\ r_{t,p} &= \mathcal{T}_\psi(h_{t-L_z+1,p}, \ldots, h_{t,p}), \end{aligned} \tag{22}$$

where e_τ is a temporal positional encoding and $\mathcal{T}_\psi$ is the temporal backbone shared across positions. The vector $r_{t,p}$ summarizes the latent history at position p. Collecting these vectors as $R_t = [r_{t,1}, \ldots, r_{t,P}]$, the output operator produces the latent-map increment:

$$\Delta\widehat{Z}_{t+1} = \mathcal{C}_{\text{out}}(R_t), \qquad \widehat{Z}_{t+1} = Z_t + \Delta\widehat{Z}_{t+1}. \tag{23}$$

The temporal model therefore applies a shared evolution law, while the circular operators exchange information between neighboring variables or spatial positions. During multi-step training, each $\widehat{Z}_{t+k}$ replaces the oldest state in the latent context and is reused to predict $\widehat{Z}_{t+k+1}$, matching the autonomous inference procedure.

B DATASETS AND EXPERIMENTAL PROTOCOL

Table A1 summarizes the observation and latent-state configurations used in the experiments. Here d_x and d_z are the physical and latent dimensions, L is the observed history length, K is the differentiable training-rollout length, and H is the autonomous evaluation horizon.

Table A1: Dataset and representation protocol. "Map" denotes a periodic spatial latent map.

System	d_x	d_z	L	K/H	Latent geometry
Lorenz–63	3	8	30	10/300	vector, $4+4$ split
Lorenz–96	40	40	30	10/300	2×20 periodic map
KS	128	48	30	10/300	6×8 periodic map

B.1 DYNAMICAL SYSTEMS AND DATA GENERATION

Lorenz–63. The low-dimensional benchmark is the three-variable system

$$\dot{x} = \sigma(y - x), \qquad \dot{y} = x(\rho - z) - y, \qquad \dot{z} = xy - \beta z, \tag{24}$$

with the standard chaotic parameters $\sigma = 10$, $\rho = 28$, and $\beta = 8/3$ (Lorenz, 1963). We integrate with a raw step of 10^{-3}, discard the transient, and retain every 37th state, giving a sampling interval of 0.037. The resulting data comprise one 24,000-state training trajectory and 20 independent on-attractor test trajectories of 2,400 states each.

Lorenz–96. The high-dimensional multivariate benchmark is the cyclic system

$$\dot{x}_i = (x_{i+1} - x_{i-2})x_{i-1} - x_i + F, \qquad i = 1, \ldots, 40, \tag{25}$$

where indices are periodic and $F = 8$. This regime is widely used to evaluate data-driven prediction of high-dimensional chaos (Pathak et al., 2018; Vlachas et al., 2020). We use fourth-order Runge–Kutta integration with four internal steps of approximately 0.00505 per saved state and discard 50 time units before sampling. The dataset contains 40 trajectories, each with 1,024 saved states.

Kuramoto–Sivashinsky. The second spatial benchmark evolves the scalar field $u(x, t)$ according to

$$u_t + uu_x + u_{xx} + u_{xxxx} = 0 \tag{26}$$

on a periodic domain of length 20π, discretized at 128 spatial points. We integrate the Fourier representation with ETDRK4, use a $2/3$ de-aliasing rule, and save one state after every ten internal steps of size $1/24$. After a 25-time-unit transient, we retain 40 trajectories of 1,024 states. This setup tests autonomous prediction when local nonlinear interactions coexist with spatially extended instability (Pathak et al., 2018; Vlachas et al., 2020).

B.2 SPLITS, PREPROCESSING, AND PAIR CONSTRUCTION

Samples are spaced so that 30 observations correspond to one nominal Lyapunov time. The Lorenz–63 training trajectory is divided chronologically into 85% training and 15% validation data; its 20 independently initialized trajectories are reserved for testing. For Lorenz–96 and KS, the first 15 trajectories are used for training, the next five for validation, and the final 20 for testing. Normalization statistics are estimated from the training partition and then fixed.

Recurrence distances are computed in train-standardized physical coordinates. Pair construction excludes overlapping input and target intervals. The locality parameter θ and response scales c_k are estimated from training trajectories only.

B.3 BASELINE AND CONTROL CONFIGURATIONS

The complete-pipeline comparison preserves each method's native representation and transition principle. CAE–ESN couples convolutional autoencoding with reservoir dynamics; LatentTSF uses a

frozen pointwise autoencoder followed by its latent forecasting and alignment objectives; and PFNN combines nonlinear lifting with contraction- and invariant-measure-based evolution.

The representation controls isolate the state interface supplied to a common predictor. Raw observations, point AE, β-VAE, time-lagged AE, Koopman AE, and the proposed encoder use the same incremental Transformer family and Stage 2 rollout objective. The backbone controls instead hold the raw or frozen latent interface fixed while changing the predictor among GRU, LSTM, TCN, and Transformer. Raw-state neural controls predict the next normalized observation directly, whereas latent controls predict an increment of the frozen latent state. All controls use the common data split, history length, autonomous evaluation, and validation-based model selection.

B.4 Parameter settings

Table A2 collects the constants that define the losses and training protocol. The same loss weights and forecast horizons are used for all three systems. The target standard deviation γ, the within-objective coefficients α_v and α_c, and the outer loss weights correspond directly to Equations 15, 16, and 7.

Table A2: Loss and optimization parameters used in the reported experiments.

Stage	Parameter	Setting
Stage 1	Future horizons $\mathcal{K}$	$\{1, 5, 10\}$
	Response horizons $\mathcal{K}_r$	$\{1, \dots, 10\}$
	$\gamma, \alpha_v, \alpha_c$	$1,\ 0.1,\ 0.01$
	$\lambda_{\mathrm{rob}}, \lambda_{\mathrm{decorr}}$	$0.01,\ 0.001$
	$\lambda_{\mathrm{future}}, \lambda_{\mathrm{auto}}$	$0.05,\ 0.025$
	Mask probability / retained tail	0.3 / 5 steps
	Optimizer / initial learning rate	Adam / 5×10^{-4}
	Schedule / maximum epochs	cosine / 300
Stage 2	Latent history L_z	30
	Training rollout K / test horizon H	10 / 300
	Optimizer / initial learning rate	Adam / 10^{-3}
	Schedule	cosine
	Maximum epochs / patience (Lorenz–63)	150 / 60
	Maximum epochs / patience (Lorenz–96, KS)	100 / 15
	Dropout / rollout-MSE weight	0.1 / 10

Stage 1 uses batch size 256 on Lorenz–63 and 128 on Lorenz–96 and KS. The Lorenz–63 multi-scale encoder uses receptive fields 3, 9, and 30. The structured encoders retain the periodic maps listed in Table A1. For the correlation and standard-deviation statistics, 10^{-4} is added only as a numerical stabilizer. The response locality values are $\theta = 128$, 8, and 4 for Lorenz–63, Lorenz–96, and KS, respectively. These values and the response scales c_k are calibrated from training trajectories as described in Appendix A.2; no test trajectory enters their selection or estimation.

The Stage 1 checkpoint minimizes the validation criterion

$$\mathcal{L}_{\mathrm{rec}} + 0.05\mathcal{L}_{\mathrm{future}} + 0.025\mathcal{L}_{\mathrm{auto}}. \tag{27}$$

The response loss is excluded from checkpoint selection so that the auxiliary branch cannot select a representation by its own training target.

The Lorenz–63 Transformer has model width 64, four attention heads, two layers, and feed-forward width 128. The Lorenz–96 and KS Transformers use width 128, eight heads, three layers, and feed-forward width 512. Each Stage 2 checkpoint is selected by validation rollout error. Its increment head is initialized at zero, and no alignment loss is added. Raw-state backbone controls predict the next normalized observation directly; all other history, rollout, split, and validation choices remain matched.

C Metric Definitions

Finite-horizon metrics. All three metrics use the same 300-step autonomous rollout. MSE@30 is the normalized mean squared error averaged over the first 30 predicted steps, trajectories, and state

coordinates. VPT is the first step at which cumulative symmetric mean absolute percentage error exceeds 30%. We average VPT over test trajectories and divide by 30 because 30 samples equal one nominal Lyapunov time. Let N be the number of test trajectories. For test trajectory n, $\widehat{p}_n^{\mathrm{true}}$ and $\widehat{p}_n^{\mathrm{pred}}$ are equally weighted, unit-covariance Gaussian-mixture estimates whose component means are the true and predicted joint states, respectively, over all 300 steps. We define

$$\bar{D}_{\mathrm{stsp}} = \frac{1}{N d_x} \sum_{n=1}^{N} D_{\mathrm{KL}}\big(\widehat{p}_n^{\mathrm{true}} \,\|\, \widehat{p}_n^{\mathrm{pred}}\big)\,. \tag{28}$$

The KL divergence is estimated with 1,000 Monte Carlo samples from $\widehat{p}_n^{\mathrm{true}}$ using a fixed seed. All mixture centers are expressed in train-standardized physical coordinates. Dividing the joint-state divergence by d_x gives a dimension-normalized scale; it is not an average of one-dimensional marginal divergences. Because the distributions use a finite rollout without burn-in, $\bar{D}_{\mathrm{stsp}}$ is a medium-horizon occupancy diagnostic rather than an asymptotic statistic.

Long-run metrics. Long free-running trajectories are evaluated over steps 301–2000, after the initial tracking interval. MMD compares state distributions after common train-only per-coordinate standardization and projection onto the leading $\min(5, d_x)$ components of a train-only PCA basis. Its bandwidth is MMDseparately for each trajectory by the target-only median heuristic. Relative spectral error is

$$E_{\mathrm{spec}} = \frac{\|\overline{P}_{\mathrm{pred}} - \overline{P}_{\mathrm{true}}\|_2}{\|\overline{P}_{\mathrm{true}}\|_2}. \tag{29}$$

For Lorenz–63, $\overline{P}$ is the temporal power spectrum averaged across trajectories and state variables after removing each temporal mean. For the periodic Lorenz–96 and KS fields, it is the spatial power spectrum averaged across trajectories and time. These diagnostics respectively test characteristic time scales and the allocation of variance across spatial Fourier modes (Pathak et al., 2017; Hess et al., 2023; Vlachas et al., 2020). Spectral errors are compared only within the same system.

The occupancy plots use the same steps 301–2000. Lorenz–63 is displayed directly in physical phase space. Lorenz–96 and KS are projected onto the first two components of a common PCA basis fitted only on the training data, so target and predicted states are shown in the same coordinates.

D Long-Horizon Results

Table A3: Long-run diagnostics on steps 301–2000 of autonomous $H = 2000$ rollouts. Lower is better. MMD uses common train-only per-coordinate standardization and a $\min(5, d_x)$-dimensional train-only PCA projection. Spectrum error is computed in physical coordinates: temporal for Lorenz–63 and spatial for Lorenz–96 and KS. Values are comparable across methods within a system, not across systems.

System	Method	MMD↓	Spectrum rel.-ℓ_2 ↓
Lorenz–63	CAE–ESN	0.1047	1.13×10^{4}
	LatentTSF	0.2421	0.8658
	PFNN	0.1672	1.0791
	Ours	**0.1044**	**0.3389**
Lorenz–96	CAE–ESN	0.2709	0.2972
	LatentTSF	0.2756	0.3038
	PFNN	0.5094	31.2408
	Ours	**0.0759**	**0.0173**
KS	CAE–ESN	0.2875	0.9965
	LatentTSF	0.2796	0.8461
	PFNN	0.1435	0.3244
	Ours	**0.1288**	**0.1133**

Across all three systems, LatentCTSF obtains the lowest MMD and spectral error. The Lorenz–63 MMD is close to CAE–ESN, but the latter's rare large-amplitude excursions produce a much

Table A4: Fine-grained Lorenz–63 Stage 1 ablations. Each row changes only the stated representation objective or head structure before retraining the matched Stage 2 predictor.

Stage 1 variant	VPT (LT)↑	MSE@30↓
w/o guarded response	4.58	0.0623
w/o future prediction $\mathcal{L}_{\text{future}}$	3.75	0.0257
w/o robust-view regularization $\mathcal{L}_{\text{rob}}$	4.01	0.1055
w/o cross-branch decorrelation $\mathcal{L}_{\text{decorr}}$	4.43	0.0797
w/o both representation regularizers	2.81	0.0123
w/ unified future head	2.60	0.0131
w/ unified latent and future head	2.90	0.1165
Full method	**4.71**	**0.0053**

larger temporal-spectrum error. The separation is strongest on Lorenz–96, where the proposed model reduces MMD from 0.2709 to 0.0759 and spectral error from 0.2972 to 0.0173 relative to the strongest complete-pipeline alternative on those metrics.

The KS results show the same pattern: PFNN is the closest baseline, while LatentCTSF improves MMD from 0.1435 to 0.1288 and spectral error from 0.3244 to 0.1133. These statistics complement VPT rather than extend it: after trajectories lose phase agreement, they measure whether autonomous samples retain the target state distribution and characteristic temporal or spatial scales.

E Fine-Grained Stage 1 Ablations

Table A4 separates the individual Stage 1 choices that are grouped together in the cumulative ablation of Table 3. The first five variants remove one objective or one objective group from the full Stage 1 model. The *unified future head* variant replaces the detached base-and-correction construction in Equation 5 with one head operating on the complete split latent state. The *unified latent and head* variant additionally removes the robust–residual partition, so both representation and prediction use the undivided latent space. Every variant is followed by a newly trained Stage 2 predictor with the same architecture and protocol.

Removing guarded response preserves most of the valid horizon but raises MSE@30 from 0.0053 to 0.0623, showing that the auxiliary relative-evolution signal mainly improves the early autonomous trajectory in this comparison. Removing either robust-view regularization or decorrelation reduces VPT, and removing both shortens it to 2.81 LT despite a small MSE@30. The two unified variants further reduce VPT to 2.60 and 2.90 LT. Thus, the results support both functional separation of the latent branches and the decomposed future objective; the metrics also show that low early error alone does not guarantee a sustained valid trajectory.

E.1 Point versus multi-scale history encoding

Table A5 isolates the encoder architecture from the Stage 1 objectives. The point variant replaces the multi-scale history encoder with a pointwise multilayer perceptron that maps each current observation directly to the same latent dimension and Robust–Residual split. It retains future prediction, the autonomous probe, representation regularization, guarded response learning, and the same Stage 2 predictor as the full method. This differs from the Point AE representation control in Table 2, which is trained by reconstruction alone. For Lorenz–96 and KS, both variants retain the same map-shaped latent interface and periodic-map Transformer in Stage 2.

The multi-scale history encoder improves all three metrics on every system. The gap is smaller on the directly observed Lorenz–63 state and substantially larger on Lorenz–96 and KS. The point encoders obtain low reconstruction errors but lose the reference trajectories much earlier. Accurate instantaneous encoding therefore does not by itself provide a state that is easy to advance autonomously. Causal histories at multiple temporal scales help organize the latent state around evolution information, with the largest benefit on the two high-dimensional systems.

Table A5: Encoder-architecture ablation under the common $H = 300$ protocol. The point and multi-scale variants use the same latent dimensions, complete Stage 1 objective, and matched Stage 2 predictor. Higher VPT is better; lower MSE@30 and $\bar{D}_{\text{stsp}}$ are better.

System	Encoder	VPT (LT)↑	MSE@30↓	$\bar{D}_{\text{stsp}}$ ↓
Lorenz–63	Point	3.58	0.0558	0.0217
	Multi-scale history	**4.71**	**0.0053**	**0.0132**
Lorenz–96	Point	0.65	0.0518	0.4034
	Multi-scale history	**2.36**	**0.0055**	**0.3504**
KS	Point	0.12	0.7121	0.6297
	Multi-scale history	**1.36**	**0.0287**	**0.3678**

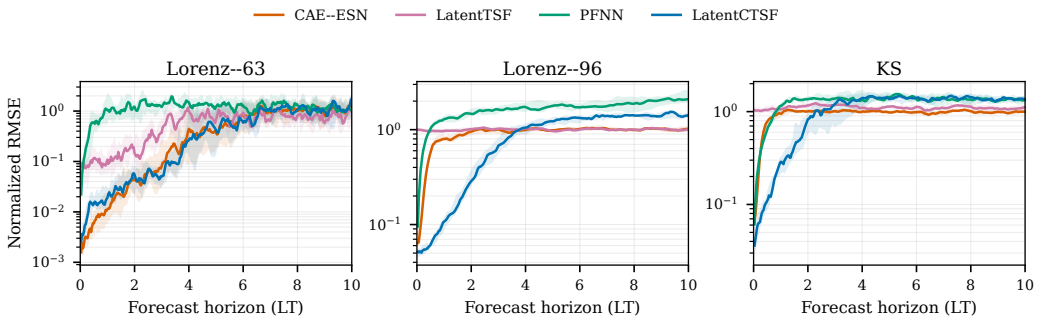

Figure 4: Normalized RMSE over the 300-step autonomous forecast, equivalent to 10 nominal Lyapunov times. Curves show the test-trajectory median and shaded regions show the interquartile range. All methods receive the same 30-step observed context and no future observations.

F Additional Finite-Horizon Diagnostics

The scalar metrics in Table 2 summarize early accuracy, valid forecast duration, and finite-window occupancy. We additionally examine how prediction error develops over the complete 300-step horizon and where it appears in the two high-dimensional state fields. These figures use the same saved autonomous rollouts as the complete-pipeline comparison.

F.1 Error growth over the forecast horizon

Figure 4 reports normalized RMSE at each forecast step. Each solid curve is the median over held-out trajectories, and the shaded region is the interquartile range. The logarithmic vertical axis makes both the initial error and its subsequent growth visible.

Across the three systems, LatentCTSF delays the rapid error increase relative to the complete-pipeline baselines. The separation is especially pronounced on Lorenz–96 and KS, where the baseline errors rise near the start of the autonomous rollout. On Lorenz–63, CAE–ESN and LatentCTSF remain accurate for longer than LatentTSF and PFNN, consistent with the closer VPT values in Table 2. The curves also expose behavior hidden by MSE@30: methods with similar initial errors can diverge at different rates once their predictions are recursively fed back.

F.2 Spatiotemporal field evolution

Figures 5 and 6 visualize the first held-out trajectory over the same 300-step interval. The upper row shows the physical state field, and the lower row shows pointwise absolute error against the ground truth. Within each system, all state panels share one state-value scale and all error panels share one error scale, so differences in pattern persistence and error spread are directly comparable.

The heatmaps show how trajectory error develops across variables rather than only in an aggregate norm. On Lorenz–96, LatentCTSF retains the traveling state pattern for a longer interval and delays

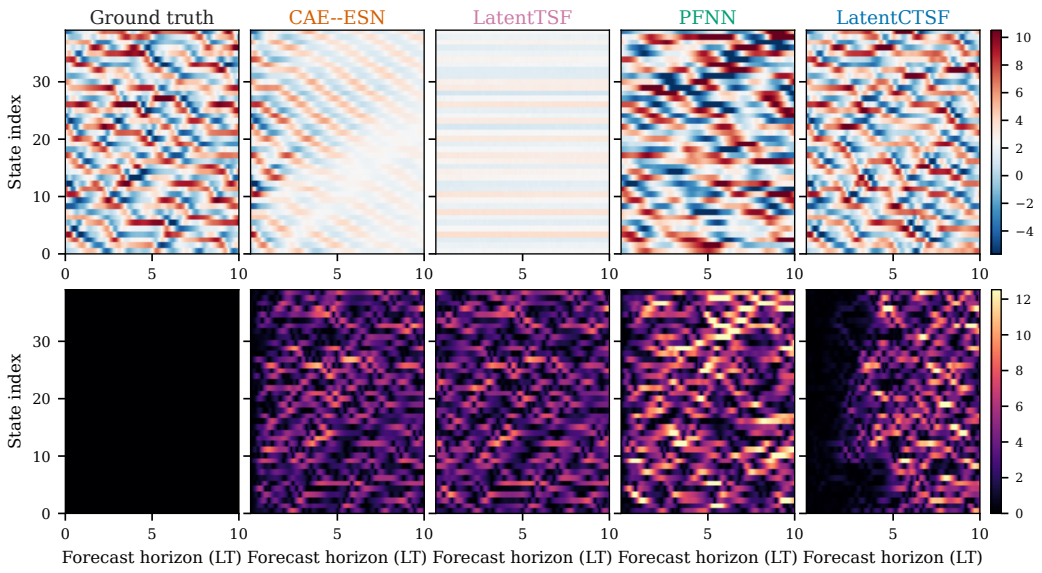

Figure 5: Lorenz–96 state evolution and absolute error over a 300-step autonomous forecast. Columns show the ground truth and the predictions of CAE–ESN, LatentTSF, PFNN, and LatentCTSF. The upper row contains the 40-variable state field; the lower row contains absolute error.

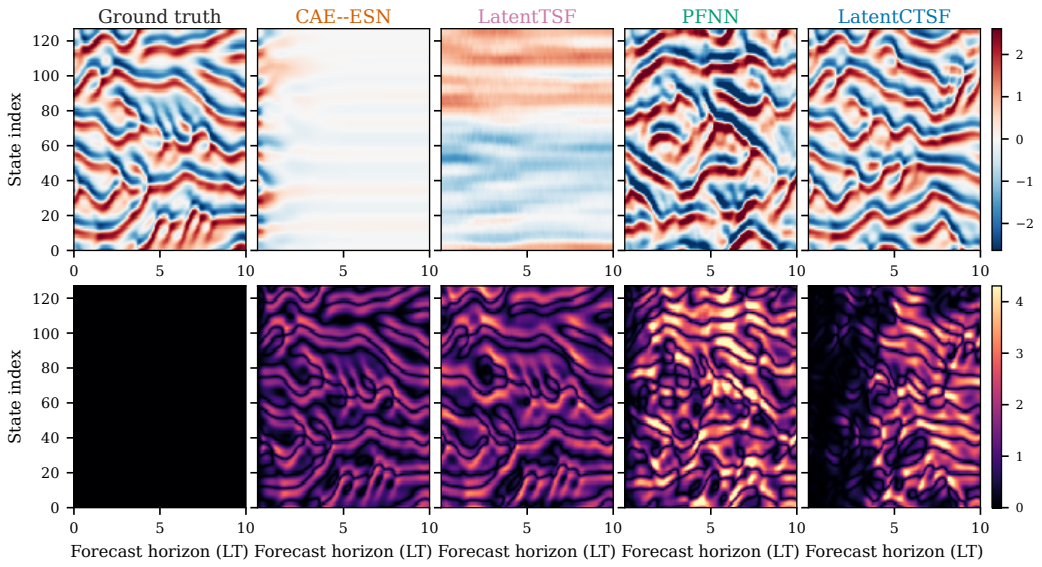

Figure 6: KS state evolution and absolute error over a 300-step autonomous forecast. Columns show the ground truth and the predictions of CAE–ESN, LatentTSF, PFNN, and LatentCTSF. The upper row contains the 128-point spatial field; the lower row contains absolute error.

the system-wide error spread. On KS, it more closely follows the evolving spatial structures during the initial forecast interval, whereas the baseline errors cover the field much earlier. Exact phase agreement eventually degrades in both chaotic systems; the relevant distinction is the duration for which coherent evolution is preserved, matching the VPT and error-growth results.

G GRADIENT-GUARD DIAGNOSTICS

Table A6 reports epoch-averaged gradient geometry for the 300-epoch Lorenz–63 Stage 1 training. It tests whether the projection in Equation 20 and the budget in Equation 21 act as intended.

Table A6: Gradient-guard diagnostics for the full Lorenz–63 Stage 1 run. Active is the fraction of minibatches receiving a nonzero guarded response update; fallback records numerical projection failures.

Module	Raw norm	Budget	Applied norm	Min. cos. before	Min. cos. after	Active	Fallback
Encoder	0.6862	9.16×10^{-4}	9.16×10^{-4}	-0.1243	0.0306	1.000	0.000
Decoder	0.0973	4.82×10^{-4}	4.82×10^{-4}	-0.0760	0.0509	1.000	0.000
Probe	0.9075	1.83×10^{-3}	1.83×10^{-3}	0.4078	0.4796	1.000	0.000

The raw response update conflicts on average with at least one protected objective in the encoder and decoder, whereas the projected update has a nonnegative minimum cosine in every reported module average. The applied norm reaches the module-wise budget, showing that magnitude control is active rather than vacuous. No numerical fallback occurs. These diagnostics verify the intended local action of the guard; they do not imply that a finite optimizer step decreases every protected loss.