DYNAMICAL STATE-SPACE DIFFUSION FOR SPATIOTEMPORAL PREDICTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Diffusion models for temporal prediction typically corrupt each future frame with independent noise, leaving temporal dependence entirely to the denoiser. We propose Dynamical State-Space Diffusion (DSSD), which learns the temporal structure of the corruption process itself. DSSD builds a temporal kernel from the damped oscillatory modes of a stable diagonal state-space model, whose learned decay rates and frequencies let the corruption capture multiple timescales and oscillatory correlations, while the denoiser architecture remains unchanged. Running this model on a finite clip, however, requires the state before the first frame, which is unobserved; starting from a zero state leaves the early frames with less noise than the diffusion schedule prescribes. We avoid this by using the Cholesky factor of the model's stationary covariance as the temporal operator. We prove that this operator is causal, exactly invertible, keeps each frame's noise variance equal to that of standard diffusion, and reduces to the identity at the clean end of the process, and that it is exactly the Kalman innovations form of the state-space model. The denoiser is trained to predict the temporally correlated noise, and sampling removes and reapplies the temporal mixing at each step. On scientific-field and robot-video datasets, DSSD outperforms diffusion baselines that share its denoiser, and the evolution of its learned modes during training shows that the temporal operator adapts to each dataset.

1 INTRODUCTION

Diffusion models generate data by learning to reverse a forward process that gradually corrupts clean samples with Gaussian noise. This denoising view has led to strong image and video generators (Ho et al., 2020; Song et al., 2021; Rombach et al., 2022; Ho et al., 2022; Blattmann et al., 2023), and it has also become attractive for temporal prediction (Voleti et al., 2022; Gao et al., 2023; Rasul et al., 2021; Tashiro et al., 2021). In forecasting, a model must represent not one future but a distribution over plausible futures conditioned on an observed history, making diffusion a natural fit for uncertain spatiotemporal signals such as video, weather, and multivariate time series.

Most diffusion frameworks for temporal prediction use a forward process that adds temporally independent noise to each future frame, relying on the denoising network to learn temporal dependencies (Voleti et al., 2022; Gao et al., 2023). However, neighboring states are often temporally coupled through motion, transport, or oscillation. This motivates incorporating temporal structure into the forward process itself, exposing the denoiser to temporally structured signal and noise.

Dynamical Diffusion (DyDiff) (Guo et al., 2025) incorporates temporal dependence into diffusion by applying an exponential moving average (EMA) operator to both the clean sequence and the diffusion noise. This introduces temporal correlations while preserving a closed-form Gaussian forward process and an invertible temporal operator for sampling. However, at each diffusion step, the EMA operator imposes a single exponential decay across frames. Its decay rate follows a predefined schedule rather than being learned from data, limiting the operator to one memory scale and preventing it from expressing oscillatory correlations.

We propose *Dynamical State-Space Diffusion* (DSSD), a novel diffusion algorithm that incorporates a learnable state-space operator into forward corruption and reverse sampling. Inspired by S4D (Gu et al., 2022b), the model combines multiple damped oscillatory modes whose decay rates, frequencies, and mixture weights are learned jointly with the denoiser. This allows the diffusion process to

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.

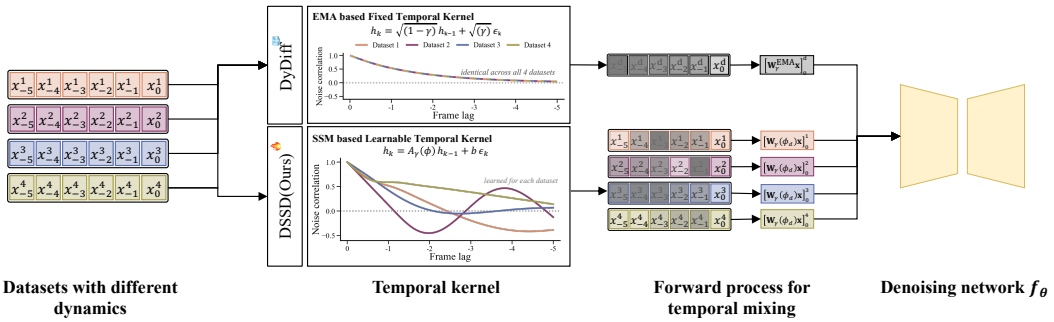


Figure 1: **DyDiff vs. DSSD.** DyDiff mixes frames with a fixed EMA kernel shared by all datasets, whereas DSSD learns a state-space kernel ϕ_d for each dataset d jointly with the denoiser f_θ.

adapt to the temporal structure of the data. The state-space parameters are shared across sequences and spatial locations, while the denoiser architecture remains unchanged. Figure 1 contrasts DyDiff's fixed kernel, shared across datasets, with DSSD's kernel learned for each dataset.

Applying this kernel to generate temporally correlated noise requires care at the start of a finite prediction window. Each output depends on both current and past noise inputs, including those preceding the window. Initializing the state-space model at zero omits the contribution of these earlier inputs, leaving the early frames with less noise variance than prescribed. This violates the requirement that every future frame receive the same noise variance at a given diffusion step. DSSD avoids this problem by using the Cholesky factor of the kernel's stationary finite-window covariance, yielding a causal and exactly invertible operator with unit marginal noise variance. We use corresponding factors to transform the signal and noise, train the denoiser to predict the resulting correlated noise, and adapt each sampling step to undo and reapply the temporal transform.

Our contributions are summarized as follows:

- We introduce DSSD, a dynamics-aware diffusion framework that uses a learnable diagonal state-space operator to define the temporal coordinates of the forward and reverse processes, rather than placing all temporal modeling burden on the denoiser.
- We construct a stationary-covariance Cholesky operator that avoids finite-window boundary effects, is causal and exactly invertible, and preserves unit marginal noise variance. This is equivalent to a Kalman innovations realization, with DyDiff's EMA operator recovered as a single-mode case.
- Experiments on scientific-field (SEVIR, Turbulence) and robot-video (BAIR, RoboNet) datasets show that DSSD outperforms diffusion baselines sharing the same denoiser. Analysis of mode evolution during training shows that DSSD's temporal operator adapts to each dataset.

2 Related Work

Diffusion models for temporal prediction. Conditional diffusion models have been applied to video prediction (Voleti et al., 2022; Ho et al., 2022), precipitation nowcasting (Gao et al., 2023), and probabilistic time-series forecasting (Rasul et al., 2021; Tashiro et al., 2021; Lopez Alcaraz & Strodthoff, 2023). These methods usually model temporal dependence through the conditioning scheme and the denoiser, for example by masking past frames, generating autoregressively, or adding temporal attention and convolution layers. The forward process remains standard, where each future time step receives independent noise. DSSD instead changes the temporal geometry of the forward process and leaves the denoiser unchanged, so it applies to existing diffusion algorithms.

Structured and learned forward processes. For video generation, temporally correlated noise priors share noise components across frames (Ge et al., 2023) or warp noise along optical flow (Chang et al., 2024); these methods structure the noise but do not apply a matched temporal transform to the clean signal. Blurring and inverse heat dissipation apply a fixed linear operator to the

signal, but along the spatial frequency axis rather than the prediction horizon (Hoogeboom & Salimans, 2023; Rissanen et al., 2023). Other works learn part of the forward process, such as the noise schedule (Kingma et al., 2021) or a data-dependent transformation of the corruption (Bartosh et al., 2024). DYffusion (Rühling Cachay et al., 2023) replaces diffusion steps with a learned temporal interpolator, so its reverse process advances physical time, whereas DSSD keeps Gaussian diffusion and changes only its covariance. DyDiff (Guo et al., 2025) applies a fixed EMA operator to both signal and noise, limiting its ability to represent multi-scale or oscillatory temporal correlations. DSSD learns this operator as a state-space model jointly with the denoiser, keeps the forward process causal, invertible and variance preserving, and recovers DyDiff as an exact special case.

State-space models. Structured state-space models parameterize linear recurrences with stable, structured transition matrices and scale to long sequences (Gu et al., 2022a;b), and selective variants make these dynamics input dependent (Gu & Dao, 2024). In diffusion models, SSMs have mainly served as network layers, either as attention-free backbones (Yan et al., 2024) or as temporal layers of time-series denoisers (Lopez Alcaraz & Strodthoff, 2023). Linear-Gaussian state-space models also underlie probabilistic forecasting, where they describe how the data are generated (Rangapuram et al., 2018). In contrast, DSSD uses an SSM to define the temporal covariance of the forward process, whose stationary innovations factor gives the diffusion forward operator.

3 Preliminaries

3.1 Conditional diffusion for temporal prediction

Given N observed frames of a spatiotemporal signal, we predict the next M. Frames are encoded by a pretrained autoencoder (Rombach et al., 2022) into observed latents $\boldsymbol{o} \in \mathbb{R}^{N\times d}$ and future latents $\boldsymbol{x} \in \mathbb{R}^{M\times d}$, where d is the dimension of a flattened latent frame. We model $p(\boldsymbol{x} \mid \boldsymbol{o})$ by diffusing only the future while conditioning the denoiser on the clean observations.

Diffusion models (Ho et al., 2020) corrupt $\boldsymbol{x}$ with Gaussian noise. For a variance-preserving schedule $\alpha_t \in (0,1)$, $t = 1,\dots,T$, whose product $\bar\alpha_t = \prod_{s=1}^{t} \alpha_s$ decreases from nearly 1 to 0,

$$\boldsymbol{z}_t = \sqrt{\bar\alpha_t}\,\boldsymbol{x} + \sqrt{1-\bar\alpha_t}\,\boldsymbol{\epsilon}, \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}). \tag{1}$$

A denoiser f_θ is trained to predict the added noise by minimizing $\mathbb{E}_{\boldsymbol{x},\boldsymbol{o},t,\boldsymbol{\epsilon}}\|f_\theta(\boldsymbol{z}_t, \boldsymbol{o}, t) - \boldsymbol{\epsilon}\|_2^2$, with t sampled uniformly. The added noise is independent across future frames, while the denoising network learns temporal dependencies from the data.

Futures are generated from Gaussian noise by repeated denoising, for example with the deterministic DDIM step (Song et al., 2021) $\boldsymbol{z}_{t'} = \sqrt{\bar\alpha_{t'}}\,\hat{\boldsymbol{x}} + \sqrt{1-\bar\alpha_{t'}}\,\hat{\boldsymbol{\epsilon}}$ for $t' < t$, where $\hat{\boldsymbol{\epsilon}} = f_\theta(\boldsymbol{z}_t, \boldsymbol{o}, t)$ and $\hat{\boldsymbol{x}} = (\boldsymbol{z}_t - \sqrt{1-\bar\alpha_t}\,\hat{\boldsymbol{\epsilon}})/\sqrt{\bar\alpha_t}$. Throughout, temporal matrices act along the frame axis and apply the same transformation independently to each latent coordinate.

3.2 State-space models

Structured state-space models (SSMs) (Gu et al., 2022a;b) describe a sequence through a hidden state that summarizes its past. For the continuous-time linear system $\dot{\boldsymbol{h}}(\tau) = \boldsymbol{A}\boldsymbol{h}(\tau) + \boldsymbol{b}\,e(\tau)$, $y(\tau) = \boldsymbol{c}^\top \boldsymbol{h}(\tau)$, zero-order-hold discretization with step size $\Delta > 0$ gives the recurrence

$$\boldsymbol{h}_k = \bar{\boldsymbol{A}}\boldsymbol{h}_{k-1} + \bar{\boldsymbol{b}}\,e_k, \qquad y_k = \boldsymbol{c}^\top \boldsymbol{h}_k, \qquad \bar{\boldsymbol{A}} = e^{\Delta \boldsymbol{A}}, \tag{2}$$

with $\bar{\boldsymbol{b}}$ the discretized input vector. From a zero initial state, unrolling gives a causal convolution $y_k = \sum_{\ell\ge 0} \bar K_\ell\, e_{k-\ell}$ whose kernel $\bar K_\ell = \boldsymbol{c}^\top \bar{\boldsymbol{A}}^\ell \bar{\boldsymbol{b}}$ weights the input ℓ steps in the past.

S4D (Gu et al., 2022b) makes $\boldsymbol{A}$ diagonal. Taking P eigenvalues together with their complex conjugates, $\boldsymbol{A} = \mathrm{diag}(\lambda_1,\dots,\lambda_P,\lambda_1^*,\dots,\lambda_P^*)$ with $\mathrm{Re}\,\lambda_n < 0$, and conjugate coefficients for conjugate states, the kernel becomes a real-valued sum of damped oscillatory modes,

$$\bar K_\ell = \sum_{n=1}^{P} r_n\, e^{\ell\Delta\,\mathrm{Re}\,\lambda_n} \cos\big(\ell\Delta\,\mathrm{Im}\,\lambda_n + \psi_n\big), \tag{3}$$

where the amplitude $r_n \geq 0$ and phase ψ_n are determined by $\bar{\boldsymbol{b}}$ and $\boldsymbol{c}$. Each mode is a damped oscillation with decay rate $-\operatorname{Re}\lambda_n$ and frequency $\operatorname{Im}\lambda_n$: one zero-frequency mode gives a geometric decay, and several modes combine decay rates and oscillations in one kernel.

When driven by i.i.d. zero-mean, unit-variance inputs from the infinite past, a stable SSM has a stationary output whose covariance depends only on the lag $\delta \geq 0$,

$$\operatorname{Cov}(y_k, y_{k+\delta}) = \sum_{\ell \geq 0} \bar{K}_\ell \, \bar{K}_{\ell+\delta}. \tag{4}$$

4 Dynamical State-Space Diffusion

We first define how a temporal operator enters the forward process, then build it from a learned SSM kernel through its stationary covariance and Cholesky factor, and finally describe training, sampling and the scope of the guarantees.

4.1 Dynamics-aware forward process

Following Guo et al. (2025), we couple the strength of temporal mixing to the noise level through a dynamics schedule $\bar{\gamma}_t = 1 - \eta\,(1 - \bar{\alpha}_t) \in [1-\eta, 1]$ with $\eta \in [0,1]$, so that $\bar{\gamma}_t \to 1$ (no mixing) as the data become clean and mixing is strongest at high noise. Let $\boldsymbol{W}_\gamma^{(L)} \in \mathbb{R}^{L\times L}$ denote a temporal operator on L frames at dynamics level $\gamma \in (0,1]$, and partition the operator on the full window as

$$\boldsymbol{W}_\gamma^{(N+M)} = \begin{bmatrix} \boldsymbol{W}_\gamma^{oo} & \boldsymbol{0} \\ \boldsymbol{W}_\gamma^{fo} & \boldsymbol{W}_\gamma^{ff} \end{bmatrix}. \tag{5}$$

The blocks act on the N observed and M future frames of Section 3.1: $\boldsymbol{W}_\gamma^{fo}$ carries the observed history into the future window, $\boldsymbol{W}_\gamma^{ff}$ mixes future frames with one another, and the zero block keeps the observations free of any influence from the future. DSSD replaces equation 1 by

$$\boldsymbol{z}_t = \sqrt{\bar{\alpha}_t}\left(\boldsymbol{W}_{\bar{\gamma}_t}^{fo}\boldsymbol{o} + \boldsymbol{W}_{\bar{\gamma}_t}^{ff}\boldsymbol{x}\right) + \sqrt{1-\bar{\alpha}_t}\,\boldsymbol{W}_{\bar{\gamma}_t}^{(M)}\boldsymbol{\epsilon}, \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}). \tag{6}$$

The clean future is mixed together with the observed history, so the first future frame inherits the temporal state of the observations, whereas the noise is mixed within the future window alone, without an observed prefix. For fixed dynamics parameters, $q(\boldsymbol{z}_t \mid \boldsymbol{o}, \boldsymbol{x})$ is Gaussian with mean $\sqrt{\bar{\alpha}_t}(\boldsymbol{W}^{fo}\boldsymbol{o} + \boldsymbol{W}^{ff}\boldsymbol{x})$ and covariance $(1-\bar{\alpha}_t)\,\boldsymbol{W}^{(M)}\boldsymbol{W}^{(M)\top}$, so training inputs are still sampled in closed form, and $\boldsymbol{W} = \boldsymbol{I}$ recovers equation 1. As in DDIM (Song et al., 2021), the process is specified by these marginals and a coupling in which all diffusion steps share the same $(\boldsymbol{x}, \boldsymbol{\epsilon})$; a Markov forward chain is not required.

We require the operator to be (i) *causal*, mixing a frame only with itself and preceding frames, i.e., lower triangular; (ii) *invertible*, so that clean frames and noise can be recovered during sampling; (iii) of *unit per-frame variance*, $\operatorname{diag}(\boldsymbol{W}\boldsymbol{W}^\top) = \boldsymbol{1}$, so that every frame receives noise of variance $1-\bar{\alpha}_t$ exactly as in equation 1 and the single noise level t given to the denoiser describes every frame; and (iv) the *identity* at $\gamma = 1$, so that the clean end of the process is unchanged. These are design requirements for DSSD, rather than necessary conditions for diffusion models in general.

4.2 State-space temporal kernel

DSSD describes the temporal correlation of the mixed frames with a diagonal SSM (Section 3.2) whose discretization step is set by the dynamics level. For P modes with learnable decay rates $a_n > 0$ and frequencies $\omega_n \geq 0$, we set

$$\lambda_n = -\tfrac{a_n}{2} + i\,\omega_n, \qquad \Delta_\gamma = -\log(1-\gamma). \tag{7}$$

For $0 < \gamma < 1$, each discrete pole $p_n = e^{\Delta_\gamma \lambda_n}$ has modulus $(1-\gamma)^{a_n/2} < 1$. Figure 2(c) illustrates these stable poles and their conjugate pairs. As $\gamma \to 1$ the step Δ_γ grows without bound and every mode forgets immediately, whereas at high noise the step is smallest and memory is longest. The step is determined by the dynamics schedule, while the learned rates and frequencies control individual modes; $a_n = 1$ reproduces the EMA decay $\sqrt{1-\gamma}$ of DyDiff. We use amplitudes $r = \operatorname{softmax}(\rho)$

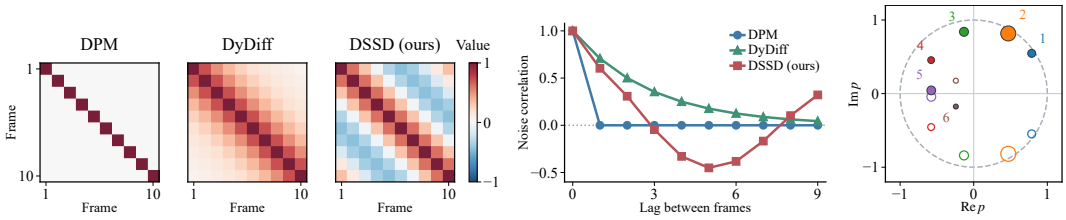


Figure 2: **Noise correlations and state-space modes in DSSD.** (a) Illustrative noise correlation matrices for DPM, DyDiff, and DSSD. (b) Corresponding noise correlations by lag: DyDiff's EMA decays exponentially, whereas DSSD can express oscillatory correlations. (c) An illustrative six-mode DSSD configuration shows discrete poles $p_n = e^{\Delta_\gamma \lambda_n}$ and their conjugates inside the unit circle. Filled and hollow markers denote conjugate partners.

and set every phase ψ_n in equation 3 to zero, giving positive real modal coefficients, and normalize the resulting kernel to unit ℓ_2 norm,

$$\tilde{u}_\ell = \sum_{n=1}^{P} r_n \, e^{\ell \Delta_\gamma \operatorname{Re} \lambda_n} \cos\big(\ell \Delta_\gamma \operatorname{Im} \lambda_n\big), \qquad u_\ell = \tilde{u}_\ell \Big/ \Big(\textstyle\sum_{\ell' \geq 0} \tilde{u}_{\ell'}^2\Big)^{1/2}. \tag{8}$$

This zero-phase choice fixes $\tilde{u}_0 = \sum_n r_n = 1$, which keeps the normalization bounded away from zero. We use this kernel for memory of past inputs and add a separate current-input term. The resulting impulse response is

$$w_0 = \sqrt{\gamma}, \qquad w_k = \sqrt{1-\gamma}\, u_{k-1} \quad (k \geq 1), \qquad \textstyle\sum_{k \geq 0} w_k^2 = 1. \tag{9}$$

Thus the SSM kernel is delayed by one frame: w_0 weights the current input and w_k weights an input k frames in the past. For independent unit-variance inputs, γ is the variance contributed by the current input, while the kernel distributes the remaining variance $1-\gamma$ over the past. At $\gamma = 1$, we define $w = (1, 0, \dots)$ by its limit. The dynamics parameters $\phi = \{\log a_n, \omega_n, \rho_n\}_{n=1}^{P}$ comprise only $3P$ scalars, shared across frames, pixels, samples and diffusion steps. Learning these parameters jointly with the denoiser allows the noise correlations to adapt to the data's temporal structure beyond a fixed exponential decay.

4.3 Causal innovations operator

Applying the impulse response equation 9 as a convolution over a finite window requires inputs preceding that window, or their accumulated state. Starting from a zero state discards the memory of earlier inputs, so the early frames have variance below one and (iii) fails: the first frame has variance $w_0^2 = \gamma$. For example, at $\gamma = 0.5$ it receives only half of the prescribed noise variance. DSSD instead assumes that the SSM has reached its stationary regime. By equation 4, driving the impulse response equation 9 with i.i.d. unit-variance inputs yields a stationary unit-variance process, and DSSD uses the causal factor of the covariance of L consecutive outputs,

$$\kappa_\delta(\gamma) = \sum_{k \geq 0} w_k \, w_{k+\delta}, \qquad \big[\boldsymbol{J}_\gamma^{(L)}\big]_{ij} = \kappa_{|i-j|}(\gamma), \qquad \boldsymbol{W}_\gamma^{(L)} = \operatorname{chol}\big(\boldsymbol{J}_\gamma^{(L)}\big), \tag{10}$$

where chol returns the unique lower-triangular factor with positive diagonal satisfying $\boldsymbol{W}\boldsymbol{W}^\top = \boldsymbol{J}$, and all infinite sums admit closed forms (Appendix A.2). For Gaussian inputs, row j of $\boldsymbol{W}$ expresses output j as the part predictable from preceding outputs plus an independent innovation, whose scale is $[\boldsymbol{W}]_{jj}$. Thus u defines the memory kernel, w includes the current-input term, $\boldsymbol{J}$ records stationary correlations, and $\boldsymbol{W}$ realizes them on a finite window. Figure 2(a–b) compares these noise correlations with those of standard diffusion and DyDiff.

Proposition 1 (Admissibility)**.** *For every $\gamma \in (0, 1]$, window length L and parameters ϕ: (a) $\boldsymbol{J}_\gamma^{(L)}$ is positive definite, so $\boldsymbol{W}_\gamma^{(L)}$ exists and is causal and invertible; (b) $\operatorname{diag}\big(\boldsymbol{W}_\gamma^{(L)} \boldsymbol{W}_\gamma^{(L)\top}\big) = \mathbf{1}$; (c) $[\boldsymbol{W}_\gamma^{(L)}]_{jj} \geq \sqrt{\gamma}$ for all j, hence $[\boldsymbol{W}_{\bar{\gamma}_t}^{(L)}]_{jj} \geq \sqrt{1-\eta}$ at every diffusion step; (d) the leading $N \times N$ block of $\boldsymbol{W}_\gamma^{(N+M)}$ equals $\boldsymbol{W}_\gamma^{(N)}$, and $\boldsymbol{W}_\gamma^{fo} \boldsymbol{W}_\gamma^{fo\top} + \boldsymbol{W}_\gamma^{ff} \boldsymbol{W}_\gamma^{ff\top} = \boldsymbol{W}_\gamma^{(M)} \boldsymbol{W}_\gamma^{(M)\top}$; and (e) $\boldsymbol{W}_\gamma^{(L)} \to \boldsymbol{W}_1^{(L)} = \boldsymbol{I}$ as $\gamma \to 1$.*

Properties (a), (b) and (e) establish requirements (i)–(iv), and (c) bounds from below every divisor in the triangular solves used during sampling. The schedule-wide bound is strictly positive when $\eta < 1$; for $\eta = 1$, invertibility requires $\bar{\alpha}_t > 0$. These bounds do not guarantee a well-conditioned inverse, which we check numerically (Appendix A.10). The first part of (d) makes the operator consistent with the observed history, so the signal branch transforms $\boldsymbol{o}$ and continues into the future; the second part is used in Section 4.5. All proofs are given in Appendix A.

The Cholesky factor also has a state-space realization. It is an innovations representation of the stationary process defined by w, rather than the original S4D convolution matrix.

Proposition 2 (State-space recurrence). *For $\gamma \in (0, 1)$, $\boldsymbol{W}_\gamma^{(L)}$ is the finite-window Kalman innovations realization (Kailath et al., 2000) of the SSM of Section 4.2, started at its stationary state covariance and zero prior state mean: $\boldsymbol{y} = \boldsymbol{W}_\gamma^{(L)}\boldsymbol{e}$ is computed by a linear recurrence with a $2P$-dimensional state (Appendix A.4) whose innovation scales satisfy $\sigma_j = [\boldsymbol{W}_\gamma^{(L)}]_{jj}$, $\sigma_0 = 1$ and $\sigma_{j+1} \le \sigma_j$.*

The state is the best estimate of the SSM state given the frames read so far, and the innovation scale shrinks as the state becomes better determined. In this view the two branches of equation 6 use different histories: the signal branch runs the recurrence with the observed latents as inputs and continues with the future latents, whereas the noise branch restarts from the stationary initialization. For long windows, this recurrence reduces the cost of applying the operator to linear in L (Appendix A.4).

Proposition 3 (EMA and richer correlations). *Let $\gamma \in (0, 1)$. With a single mode, $a_1 = 1$ and $\omega_1 = 0$, $\boldsymbol{W}_\gamma^{(L)}$ is the EMA operator of DyDiff (Guo et al., 2025), $y_0 = e_0$ and $y_j = \sqrt{1-\gamma}\, y_{j-1} + \sqrt{\gamma}\, e_j$, whose autocorrelation $\kappa_\delta(\gamma) = (1-\gamma)^{\delta/2}$ is positive at every lag δ. In contrast, the DSSD family admits a single oscillating mode with $\kappa_1(\gamma) < 0$.*

At each fixed dynamics level, the unrestricted DSSD covariance family therefore includes the EMA covariance (Corollary 1) and correlations beyond it: an oscillating mode can make neighboring frames anticorrelated, a pattern that a monotonically decaying memory cannot produce (Appendix A.5).

4.4 Training and sampling

Training. The denoiser parameters θ and the dynamics parameters ϕ are trained jointly by regressing the mixed noise,

$$\mathcal{L}(\theta, \phi) = \mathbb{E}_{\boldsymbol{x},\boldsymbol{o},t,\boldsymbol{\epsilon}} \big\| f_\theta(\boldsymbol{z}_t, \boldsymbol{o}, t) - \boldsymbol{W}_{\bar{\gamma}_t}^{(M)} \boldsymbol{\epsilon} \big\|_2^2, \tag{11}$$

with $\boldsymbol{z}_t$ given by equation 6 and t sampled uniformly. For a fixed operator, the minimizer is the posterior mean of the mixed noise. Because the noise enters $\boldsymbol{z}_t$ only through $\boldsymbol{W}_{\bar{\gamma}_t}^{(M)}\boldsymbol{\epsilon}$, the score of $q(\boldsymbol{z}_t \mid \boldsymbol{o})$ is a fixed linear transform of this posterior mean (Appendix A.8), so the correlated target requires no whitening. Since every coordinate of the target has unit variance for every ϕ, learning ϕ cannot lower the loss by shrinking the noise. Because ϕ enters both the corrupted input and the target, learning ϕ selects the temporal coordinates in which the denoiser operates rather than identifying the physical dynamics of the data. Initialization and numerical checks are given in Appendices A.9 and A.10. Algorithm 1 summarizes joint training of the denoiser and dynamics parameters.

Sampling. We use deterministic DDIM steps that undo the mixing of the current step and reapply that of the next one. Given the prediction $\hat{\boldsymbol{\epsilon}}_c = f_\theta(\boldsymbol{z}_t, \boldsymbol{o}, t)$ at step t and the next step $t' < t$,

$$\hat{\boldsymbol{x}} = \big(\boldsymbol{W}_{\bar{\gamma}_t}^{ff}\big)^{-1}\Big(\tfrac{1}{\sqrt{\bar{\alpha}_t}}\big(\boldsymbol{z}_t - \sqrt{1-\bar{\alpha}_t}\,\hat{\boldsymbol{\epsilon}}_c\big) - \boldsymbol{W}_{\bar{\gamma}_t}^{fo}\boldsymbol{o}\Big), \qquad \hat{\boldsymbol{\epsilon}} = \big(\boldsymbol{W}_{\bar{\gamma}_t}^{(M)}\big)^{-1}\hat{\boldsymbol{\epsilon}}_c,$$
$$\boldsymbol{z}_{t'} = \sqrt{\bar{\alpha}_{t'}}\,\big(\boldsymbol{W}_{\bar{\gamma}_{t'}}^{fo}\boldsymbol{o} + \boldsymbol{W}_{\bar{\gamma}_{t'}}^{ff}\hat{\boldsymbol{x}}\big) + \sqrt{1-\bar{\alpha}_{t'}}\,\boldsymbol{W}_{\bar{\gamma}_{t'}}^{(M)}\hat{\boldsymbol{\epsilon}}. \tag{12}$$

Sampling starts from $\boldsymbol{z}_T = \boldsymbol{W}_{\bar{\gamma}_T}^{(M)}\boldsymbol{\xi}$ with $\boldsymbol{\xi} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$, and the last step uses $\bar{\alpha} = \bar{\gamma} = 1$, where the operator is the identity and $\hat{\boldsymbol{x}}$ is returned. Both inverses are triangular solves whose positive divisors satisfy Proposition 1(c), and with $\boldsymbol{W} = \boldsymbol{I}$ the update reduces to the standard DDIM step.

Algorithm 2 gives the complete sampling procedure, including initialization and the clean endpoint. The following result justifies this update.

Proposition 4 (Sampler consistency). *Fix the dynamics parameters ϕ, assume finite second moments of the data, and let f_θ be the unrestricted minimizer of equation 11. For $0 < \bar{\alpha}_t < 1$, $\hat{\boldsymbol{x}} = \mathbb{E}[\boldsymbol{x} \mid \boldsymbol{z}_t, \boldsymbol{o}]$, $\hat{\boldsymbol{\epsilon}} = \mathbb{E}[\boldsymbol{\epsilon} \mid \boldsymbol{z}_t, \boldsymbol{o}]$, and equation 12 returns $\mathbb{E}[\boldsymbol{z}_{t'} \mid \boldsymbol{z}_t, \boldsymbol{o}]$ under the shared $(\boldsymbol{x}, \boldsymbol{\epsilon})$ coupling. On an interior time interval where $\bar{\alpha}$ is twice continuously differentiable and $0 < \bar{\alpha} < 1$, $\boldsymbol{z}_{t'} = \boldsymbol{z}_t + (t' - t)\, \boldsymbol{v}_t(\boldsymbol{z}_t) + O\big((t' - t)^2\big)$ for a velocity field $\boldsymbol{v}_t$ whose continuity equation is solved by the marginals $q(\boldsymbol{z}_t \mid \boldsymbol{o})$.*

Under the flow regularity conditions in Appendix A.6, equation 12 is a first-order discretization of a flow that transports the marginals of the forward process (Albergo et al., 2025), as DDIM is for standard diffusion. With the ideal denoiser, its last step returns the posterior mean of the future.

4.5 What the temporal operator changes

Since every operator is invertible, one may ask whether DSSD merely reparameterizes standard diffusion. It does not: the mixing varies with the diffusion step and differs between the signal and noise branches of equation 6. Let $\boldsymbol{F}_t = \big(\boldsymbol{W}^{(M)}_{\bar{\gamma}_t}\big)^{-1} \boldsymbol{W}^{ff}_{\bar{\gamma}_t}$, and let $\boldsymbol{z}_t^{\mathrm{std}} = \sqrt{\bar{\alpha}_t}\, \boldsymbol{x} + \sqrt{1 - \bar{\alpha}_t}\, \boldsymbol{\epsilon}$ be the standard corruption equation 1 at the same noise level.

Proposition 5 (Information ordering). *For every t, $\boldsymbol{F}_t \boldsymbol{F}_t^\top \preceq \boldsymbol{I}$. Consequently, given $\boldsymbol{o}$, a variable with the same joint distribution with $\boldsymbol{x}$ as $\boldsymbol{z}_t$ can be generated from $\boldsymbol{z}_t^{\mathrm{std}}$ and independent noise without access to $\boldsymbol{x}$, and, for data with finite second moments,*

$$\min_g \mathbb{E}\big\|\boldsymbol{x} - g(\boldsymbol{z}_t^{\mathrm{std}}, \boldsymbol{o})\big\|^2 \;\le\; \min_g \mathbb{E}\big\|\boldsymbol{x} - g(\boldsymbol{z}_t, \boldsymbol{o})\big\|^2. \tag{13}$$

Temporal mixing therefore creates no information about the future at any single noise level; for the EMA operator, $\boldsymbol{F}_t = \mathrm{diag}(\sqrt{\bar{\gamma}_t}, 1, \ldots, 1)$ attenuates only the direction of the first future frame (Appendix A.7). DSSD instead changes the geometry of the regression problem that a finite denoiser solves across noise levels, and the path along which samples are generated. These changes may help a finite denoiser and sampler, but the proposition does not establish such a benefit; whether they help is tested empirically in Section 5.

5 Experiments

Proposition 5 rules out an information advantage from temporal mixing at a fixed noise level. We therefore test whether learning the temporal operator improves finite-step forecasting over standard diffusion DPM and DyDiff. We evaluate scientific fields from SEVIR (Veillette et al., 2020) and Turbulence (Wang et al., 2020), and robot videos from BAIR (Ebert et al., 2017) and RoboNet (Dasari et al., 2019).

5.1 Experimental setup

Within each dataset, all methods use the same frozen framewise VAE and Stable Video Diffusion-inspired video U-Net (Blattmann et al., 2023). We train for 50,000 optimizer updates and use a shared 50-step DDIM sampling grid (Song et al., 2021). Task windows and training settings are detailed in Appendix C, and example forecasts for all four datasets in Appendix D.1. Metrics include the continuous ranked probability score (CRPS), critical success index (CSI) and Fréchet video distance (FVD); all are defined in Appendix C.3. In Tables 1 and 2, bold and underline mark the best and second-best scores for each dataset and metric.

5.2 Scientific-field prediction

DSSD has the lowest scores on all three CRPS variants, temporal variogram, energy score, and RMSE, and the highest CSI on both datasets. Its raw CRPS is 3.975 versus 4.174 for the strongest SEVIR baseline (DPM), and 0.00756 versus 0.01100 for the strongest Turbulence baseline (DyDiff).

Table 1: Scientific-field prediction results.

Dataset	Method	CRPS ↓			CSI w5 ↑	Temporal variogram ↓	Energy score ↓	RMSE ↓
		Raw	$w8_{avg}$	$w8_{max}$				
SEVIR	DPM	4.174	2.737	6.599	0.5518	2.339	10.862	15.420
	DyDiff	4.642	3.169	7.955	0.3911	4.492	13.024	25.540
	DSSD (Ours)	**3.975**	**2.421**	**5.995**	**0.5707**	**2.129**	**10.263**	**14.585**
Turbulence	DPM	0.01148	0.00751	0.01065	0.9641	0.00262	0.01920	0.02600
	DyDiff	0.01100	0.00752	0.01123	0.9674	0.00260	0.01791	0.02427
	DSSD (Ours)	**0.00756**	**0.00440**	**0.00757**	**0.9779**	**0.00147**	**0.01178**	**0.01582**

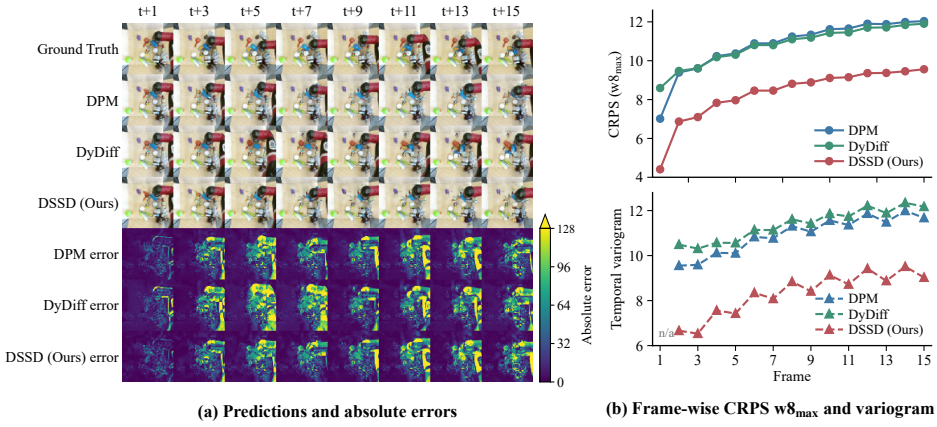
Table 2: Robot-video prediction results.

Dataset	Method	Forecast metrics				Video metrics			
		CRPS ↓	RMSE ↓	Energy ↓	Vario. ↓	PSNR ↑	SSIM ↑	LPIPS ↓	FVD ↓
BAIR	DPM	17.788	42.907	30.176	11.078	15.036	0.5938	0.2246	1329.2
	DyDiff	17.691	42.815	30.092	11.412	15.149	0.5999	0.2220	1417.4
	DSSD (Ours)	**16.825**	**42.367**	**29.661**	**8.548**	**15.717**	**0.6205**	**0.2007**	**766.0**
RoboNet	DPM	8.782	23.523	15.130	5.939	22.680	**0.7889**	**0.1370**	659.2
	DyDiff	8.383	23.735	14.705	6.504	22.630	0.7865	0.1377	770.6
	DSSD (Ours)	**8.108**	**23.485**	**14.548**	**4.409**	**22.693**	0.7858	0.1379	**430.2**

5.3 Robot-video prediction

On BAIR, DSSD outperforms both DPM and DyDiff across all eight reported metrics. On RoboNet, DSSD leads six of eight metrics, while DPM achieves the best SSIM and LPIPS. On both datasets, DSSD shows substantial improvements in FVD and temporal variogram scores, whereas RMSE gains are modest.

5.4 Prediction quality across horizons



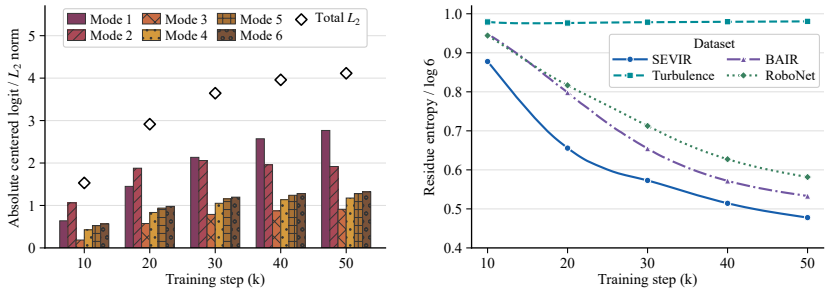
(a) Predictions and absolute errors **(b) Frame-wise CRPS $w8_{max}$ and variogram**

Figure 3: BAIR forecast quality across future frames. (a) Example predictions and absolute-error maps. (b) Test-set frame-wise pooled CRPS and temporal variogram scores; lower is better.

Figure 3 compares prediction quality across horizons on BAIR. Panel (a) shows an example predicted sequence and absolute-error maps for each method. In this example, DSSD produces more accurate predictions throughout the sequence, with particularly smaller absolute errors than DyDiff at intermediate time steps. Panel (b) reports frame-wise CRPS($w8_{max}$) and temporal variogram scores, averaged over all BAIR test windows at each prediction horizon. DSSD maintains lower scores than both baselines throughout the evaluated horizons, with its advantage persisting into the

later frames. These results indicate sustained improvements in probabilistic prediction and closer agreement with ground-truth changes between consecutive frames.

5.5 Analysis of learned temporal modes



(a) Centered residue logits and norm (SEVIR) **(b) Normalized residue entropy (four datasets)**

Figure 4: DSSD mode-parameter diagnostics. (a) SEVIR centered residue logits and their norm. (b) Normalized residue entropy on four datasets.

The learned operator is compact, with only $3P = 18$ parameters (Section 4.2). Figure 4 tracks these modes from 10,000 to 50,000 updates through the centered residue logits $|\rho_n - \bar{\rho}|$, their L_2 norm (the residue-logit norm), and the normalized entropy of $r = \mathrm{softmax}(\rho)$ (Appendix C.4). The residue-logit norm measures logit spread around the mean, whereas lower entropy indicates more concentrated residue weights across modes. On SEVIR, the residue-logit norm increases from 1.53 to 4.11 while entropy decreases. Entropy also decreases on BAIR and RoboNet: SEVIR and RoboNet concentrate residue weight on the first mode, and BAIR on the second (Appendix D.3). On Turbulence, the residue shares remain nearly uniform, yet the modes continue to evolve: at $\gamma = 0.5$, the first mode's pole radius decreases from 0.937 to 0.889, indicating faster decay and shorter temporal memory. These observations indicate dataset-dependent adaptation of both mode weights and timescales.

6 Conclusion

We introduce DSSD, the novel diffusion framework for spatiotemporal prediction to make the coupling between spatial denoising and temporal dynamics jointly learnable. Its forward process adds noise while mixing frames through a learnable state-space operator, and its reverse process combines learned denoising with inversion of the temporal mixing. Joint training of the state-space dynamics and denoiser allows this coupled process to adapt to the data without changing the denoiser architecture. We establish a theoretical foundation through a stationary-covariance construction that ensures causal, invertible temporal mixing with the prescribed noise variance, together with a consistency analysis of the sampling update. Experiments on scientific fields and robot videos show improved probabilistic forecasts and temporal consistency, while analysis of the learned modes demonstrates dataset-dependent adaptation. We expect this framework to help accelerate the development of dynamics-aware video foundation models at higher resolutions and larger scales. Its learnable temporal dynamics may also support online adaptation to changing temporal patterns in streaming data, with potential applications in robotics.

AI use statement

We used OpenAI Codex and Anthropic Claude to assist with implementing training, sampling, evaluation, and experiment-management code based on our specifications. These tools also assisted with typesetting and editing proofs, writing numerical verification scripts, providing feedback on experimental design, drafting and editing the manuscript, searching and summarizing literature, and formatting references.

The theoretical model, operator construction, mathematical claims, and proofs were developed by the authors. AI assistance with proofs was limited to presentation and numerical checks; it did not

supply proof arguments. AI tools were not used to develop the conceptual framework, propose or refine hypotheses, generate synthetic datasets, or conduct qualitative and thematic data analysis. Translation was not applicable.

We reviewed all AI-assisted work, checked the proofs against our derivations, and verified the algebra through independent numerical implementations. We tested the code and checked citations against primary sources. The authors take responsibility for the final text, claims, code, and other artifacts.

Ethics statement

This work involves no human subjects, crowdsourced annotation, or personally identifiable information; no institutional review was required.

Reproducibility statement

Section 4 specifies the method and theoretical assumptions; Appendix A provides complete proofs, and Appendix A.10 details initialization and numerical implementation. Appendix C specifies the data processing, model and optimization settings, evaluation protocols, and compute resources used in our experiments.

We use publicly accessible datasets—SEVIR, BAIR, and RoboNet—and a simulation-based benchmark, Turbulence. Third-party datasets are used in accordance with their terms and are not redistributed.

We will release model weights, training and sampling code, and configurations for the models and datasets used in this paper. The release will include environment specifications and instructions for data preparation and reproducing the experiments. Run records and numerical tests support verification of the experimental settings and theoretical identities.

References

Michael Albergo, Nicholas M. Boffi, and Eric Vanden-Eijnden. Stochastic interpolants: A unifying framework for flows and diffusions. *Journal of Machine Learning Research*, 26(209):1–80, 2025.

Alexander Alexandrov, Konstantinos Benidis, Michael Bohlke-Schneider, Valentin Flunkert, Jan Gasthaus, Tim Januschowski, Danielle C. Maddix, Syama Rangapuram, David Salinas, Jasper Schulz, Lorenzo Stella, Ali Caner Türkmen, and Yuyang Wang. GluonTS: Probabilistic and neural time series modeling in Python. *Journal of Machine Learning Research*, 21(116):1–6, 2020.

Grigory Bartosh, Dmitry Vetrov, and Christian A. Naesseth. Neural Flow Diffusion Models: Learnable forward process for improved diffusion modelling. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, Varun Jampani, and Robin Rombach. Stable Video Diffusion: Scaling latent video diffusion models to large datasets. *arXiv preprint arXiv:2311.15127*, 2023.

João Carreira and Andrew Zisserman. Quo Vadis, action recognition? A new model and the Kinetics dataset. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

Pascal Chang, Jingwei Tang, Markus Gross, and Vinicius C. Azevedo. How I warped your noise: a temporally-correlated noise prior for diffusion models. In *International Conference on Learning Representations (ICLR)*, 2024.

Kaiyuan Chen, Xingzhuo Guo, Yu Zhang, Jianmin Wang, and Mingsheng Long. CogDPM: Diffusion probabilistic models via cognitive predictive coding. In *International Conference on Machine Learning (ICML)*, 2024.

Sudeep Dasari, Frederik Ebert, Stephen Tian, Suraj Nair, Bernadette Bucher, Karl Schmeckpeper, Siddharth Singh, Sergey Levine, and Chelsea Finn. RoboNet: Large-scale multi-robot learning. In *Conference on Robot Learning (CoRL)*, volume 100 of *PMLR*, 2019.

Frederik Ebert, Chelsea Finn, Alex X. Lee, and Sergey Levine. Self-supervised visual planning with temporal skip connections. In *Conference on Robot Learning (CoRL)*, volume 78 of *PMLR*, 2017.

Zhihan Gao, Xingjian Shi, Boran Han, Hao Wang, Xiaoyong Jin, Danielle Maddix, Yi Zhu, Mu Li, and Yuyang Wang. PreDiff: Precipitation nowcasting with latent diffusion models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Songwei Ge, Seungjun Nah, Guilin Liu, Tyler Poon, Andrew Tao, Bryan Catanzaro, David Jacobs, Jia-Bin Huang, Ming-Yu Liu, and Yogesh Balaji. Preserve your own correlation: A noise prior for video diffusion models. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

Tilmann Gneiting and Adrian E. Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. In *Conference on Language Modeling (COLM)*, 2024.

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

Albert Gu, Ankit Gupta, Karan Goel, and Christopher Ré. On the parameterization and initialization of diagonal state space models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022b.

Xingzhuo Guo, Yu Zhang, Baixu Chen, Haoran Xu, Jianmin Wang, and Mingsheng Long. Dynamical Diffusion: Learning temporal dynamics with diffusion models. In *International Conference on Learning Representations (ICLR)*, 2025.

Agrim Gupta, Stephen Tian, Yunzhi Zhang, Jiajun Wu, Roberto Martín-Martín, and Li Fei-Fei. MaskViT: Masked visual pre-training for video prediction. In *International Conference on Learning Representations (ICLR)*, 2023.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J. Fleet. Video diffusion models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Emiel Hoogeboom and Tim Salimans. Blurring diffusion models. In *International Conference on Learning Representations (ICLR)*, 2023.

Thomas Kailath, Ali H. Sayed, and Babak Hassibi. *Linear Estimation*. Prentice Hall, 2000.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

Diederik P. Kingma and Max Welling. Auto-encoding variational Bayes. In *International Conference on Learning Representations (ICLR)*, 2014.

Diederik P. Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Juan Miguel Lopez Alcaraz and Nils Strodthoff. Diffusion-based time series imputation and forecasting with structured state space models. *Transactions on Machine Learning Research (TMLR)*, 2023.

Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning (ICML)*, 2021.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

Syama Sundar Rangapuram, Matthias W. Seeger, Jan Gasthaus, Lorenzo Stella, Yuyang Wang, and Tim Januschowski. Deep state space models for time series forecasting. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

Kashif Rasul, Calvin Seward, Ingmar Schuster, and Roland Vollgraf. Autoregressive denoising diffusion models for multivariate probabilistic time series forecasting. In *International Conference on Machine Learning (ICML)*, 2021.

Severi Rissanen, Markus Heinonen, and Arno Solin. Generative modelling with inverse heat dissipation. In *International Conference on Learning Representations (ICLR)*, 2023.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 2015.

Salva Rühling Cachay, Bo Zhao, Hailey Joren, and Rose Yu. DYffusion: A dynamics-informed diffusion model for spatiotemporal forecasting. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

David Salinas, Michael Bohlke-Schneider, Laurent Callot, Roberto Medico, and Jan Gasthaus. High-dimensional multivariate forecasting with low-rank Gaussian copula processes. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Michael Scheuerer and Thomas M. Hamill. Variogram-based proper scoring rules for probabilistic forecasts of multivariate quantities. *Monthly Weather Review*, 143(4):1321–1334, 2015.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations (ICLR)*, 2021.

Yusuke Tashiro, Jiaming Song, Yang Song, and Stefano Ermon. CSDI: Conditional score-based diffusion models for probabilistic time series imputation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Thomas Unterthiner, Sjoerd van Steenkiste, Karol Kurach, Raphaël Marinier, Marcin Michalski, and Sylvain Gelly. Towards accurate generative models of video: A new metric & challenges. *arXiv preprint arXiv:1812.01717*, 2018.

Mark Veillette, Siddharth Samsi, and Chris Mattioli. SEVIR: A storm event imagery dataset for deep learning applications in radar and satellite meteorology. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Vikram Voleti, Alexia Jolicoeur-Martineau, and Christopher Pal. MCVD: Masked conditional video diffusion for prediction, generation, and interpolation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Rui Wang, Karthik Kashinath, Mustafa Mustafa, Adrian Albert, and Rose Yu. Towards physics-informed deep learning for turbulent flow prediction. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2020.

Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4): 600–612, 2004.

Jialong Wu, Shaofeng Yin, Ningya Feng, Xu He, Dong Li, Jianye Hao, and Mingsheng Long. iVideoGPT: Interactive VideoGPTs are scalable world models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Jing Nathan Yan, Jiatao Gu, and Alexander M. Rush. Diffusion models without attention. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

Michaël Zamo and Philippe Naveau. Estimation of the continuous ranked probability score with limited information and applications to ensemble weather forecasts. *Mathematical Geosciences*, 50(2):209–234, 2018.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

A Mathematical Details and Proofs

We first derive the stationary covariance, then prove the operator properties and its state-space realization, and finally analyze regression and sampling. Throughout, the dynamics parameters ϕ are fixed unless joint optimization is explicitly discussed; $a_n > 0$, $\omega_n \geq 0$ and $r_n > 0$ with $\sum_n r_n = 1$. For $\gamma \in (0, 1)$, we omit γ and ϕ when no confusion arises and write $\nu = \sqrt{1-\gamma}$, $\Delta = \Delta_\gamma$, $R_n = |e^{\Delta\lambda_n}| = \nu^{a_n}$ and $\vartheta_n = \Delta\omega_n$, so that $\tilde{u}_\ell = \sum_n r_n R_n^\ell \cos(\ell\vartheta_n)$ in equation 8. As in the main text, matrices act on the time axis and are shared by all coordinates of each frame. The endpoint $\gamma = 1$ is defined by $w = (1, 0, \dots)$ and $\boldsymbol{W} = \boldsymbol{I}$; expressions involving Δ are evaluated only for $\gamma < 1$. We distinguish the SSM memory kernel u_ℓ, the full impulse response w_k, its stationary covariance $\boldsymbol{J}$, and the finite-window innovations factor $\boldsymbol{W}$.

A.1 Notation

Table 3 collects the symbols used in Sections 3 and 4 and in this appendix, grouped in the order in which they appear. Bold lowercase letters denote vectors or stacked latent frames, bold uppercase letters denote matrices, and all temporal matrices act along the frame axis, identically for every latent coordinate. A few auxiliary symbols local to a single proof step, such as y_j^{pre} in Appendix A.3, are defined where they appear.

Table 3: Notation.

Symbol	Meaning	Defined in
Data and diffusion		
N, M	Numbers of observed and future frames	§3.1
L	Length of the window on which a temporal operator acts; $L \in \{N, M, N+M\}$	§4.1
d	Dimension of a flattened latent frame	§3.1
$\boldsymbol{o} \in \mathbb{R}^{N\times d}$	Observed latents (clean conditioning history)	§3.1
$\boldsymbol{x} \in \mathbb{R}^{M\times d}$	Future latents to be predicted	§3.1
T, t	Number of diffusion steps; current step $t \in \{1, \dots, T\}$	Eq. (1)
t'	Next, less noisy step of a sampling update, $t' < t$	§3.1
α_t, $\bar{\alpha}_t$	Noise schedule and signal level $\bar{\alpha}_t = \prod_{s\leq t} \alpha_s$	Eq. (1)
$\boldsymbol{\epsilon}$	White Gaussian noise, $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$, before temporal mixing	Eq. (1)
$\boldsymbol{z}_t$	Noisy future latents at step t	Eq. (1), Eq. (6)
f_θ, θ	Denoiser and its parameters	§3.1
$p(\boldsymbol{x} \mid \boldsymbol{o})$	Conditional distribution of the future given the history	§3.1
$q(\boldsymbol{z}_t \mid \boldsymbol{o})$	Marginal of the forward process at step t	§4.4
State-space models (generic)		
τ	Continuous time	§3.2
$\boldsymbol{h}$, e, y	State, scalar input and scalar output of an SSM	Eq. (2)
$\boldsymbol{A}$, $\boldsymbol{b}$, $\boldsymbol{c}$	State, input and output parameters	§3.2
Δ, $\bar{\boldsymbol{A}}$, $\bar{\boldsymbol{b}}$	Step size and zero-order-hold discretization, $\bar{\boldsymbol{A}} = e^{\Delta \boldsymbol{A}}$	Eq. (2)
$\bar{K}_\ell$	Convolution kernel of the discretized SSM; weight of the input ℓ steps in the past	Eq. (3)

Symbol	Meaning	Defined in
P	Number of complex-conjugate mode pairs	§3.2
λ_n	Continuous-time eigenvalue of mode n, $\mathrm{Re}\,\lambda_n < 0$; in DSSD, $-a_n/2 + i\,\omega_n$	§3.2, Eq. (7)
r_n, ψ_n	Amplitude and phase of mode n	Eq. (3)
Dynamics schedule and learned modes		
η	Mixing strength, $\eta \in [0, 1]$	§4.1
$\bar{\gamma}_t$	Dynamics schedule, $1 - \eta(1 - \bar{\alpha}_t) \in [1 - \eta, 1]$; 1 means no mixing	§4.1
γ	Generic dynamics level, $\gamma \in (0, 1]$	§4.1
a_n, ω_n	Learned decay rate (> 0) and frequency (≥ 0) of mode n	Eq. (7)
ρ_n	Learned residue logit of mode n; $r = \mathrm{softmax}(\rho)$	§4.2
ϕ	Dynamics parameters $\{\log a_n, \omega_n, \rho_n\}_{n=1}^{P}$, $3P$ scalars	§4.2
Δ_γ	Discretization step tied to the dynamics level, $-\log(1 - \gamma)$	Eq. (7)
p_n	Discrete pole of mode n, $e^{\Delta_\gamma \lambda_n}$, with modulus $(1 - \gamma)^{a_n/2}$	§4.2
Kernel, covariance and operator		
$\tilde{u}_\ell$, u_ℓ	Memory kernel before and after normalization to unit ℓ_2 norm	Eq. (8)
w_k	Impulse response: $w_0 = \sqrt{\gamma}$ weights the current input, $w_k = \sqrt{1 - \gamma}\, u_{k-1}$ the input k frames earlier	Eq. (9)
$\kappa_\delta(\gamma)$	Stationary autocorrelation at lag δ, $\sum_k w_k w_{k+\delta}$; $\kappa_0 = 1$	Eq. (10)
$\boldsymbol{J}_\gamma^{(L)}$	Stationary covariance of L consecutive outputs, $[\boldsymbol{J}]_{ij} = \kappa_{\lvert i-j \rvert}$	Eq. (10)
chol	Lower-triangular Cholesky factor with positive diagonal	Eq. (10)
$\boldsymbol{W}_\gamma^{(L)}$	Temporal operator, $\mathrm{chol}(\boldsymbol{J}_\gamma^{(L)})$	Eq. (10)
$\boldsymbol{W}^{oo}$, $\boldsymbol{W}^{fo}$, $\boldsymbol{W}^{ff}$	Blocks of $\boldsymbol{W}_\gamma^{(N+M)}$: history on history, history into future, future on future	Eq. (5)
$\boldsymbol{W}_\gamma^{(M)}$	Operator applied to the noise, on the future window alone	Eq. (6)
Innovations recurrence (Proposition 2)		
$\boldsymbol{e}$, e_j	Input sequence of the recurrence, $\boldsymbol{y} = \boldsymbol{W}_\gamma^{(L)} \boldsymbol{e}$	Prop. 2
$\boldsymbol{y}$, y_j	Output sequence of the operator	Prop. 2
σ_j	Innovation scale, $[\boldsymbol{W}_\gamma^{(L)}]_{jj}$	Prop. 2
$\boldsymbol{s}_j$	Pre-input SSM state, containing only inputs before frame j	Eq. (20)
$\hat{\boldsymbol{s}}_j$	State estimate given $y_0, \ldots, y_{j-1}$	Eq. (22)
$\boldsymbol{\Sigma}$	Stationary state covariance	App. A.4
$\boldsymbol{S}_j$	Conditional state covariance given $y_0, \ldots, y_{j-1}$	Eq. (21)
$\boldsymbol{k}_j$	Kalman gain	Eq. (21)
D	Direct current-input term of the realization, $\sqrt{\gamma}$	Eq. (20)
Training and sampling		
$\mathcal{L}(\theta, \phi)$	Mixed-noise regression loss	Eq. (11)
$\hat{\boldsymbol{\epsilon}}_c$	Denoiser prediction of the mixed noise, $f_\theta(\boldsymbol{z}_t, \boldsymbol{o}, t)$	Eq. (12)
$\hat{\boldsymbol{x}}$, $\hat{\boldsymbol{\epsilon}}$	Estimates of the clean future and of the white noise at step t	Eq. (12)
$\boldsymbol{\xi}$	White Gaussian noise that initializes sampling, $\boldsymbol{z}_T = \boldsymbol{W}_{\bar{\gamma}_T}^{(M)} \boldsymbol{\xi}$	§4.4
$\boldsymbol{v}_t$	Velocity field transported by the sampler	Prop. 4
Analysis (Proposition 5)		
$\boldsymbol{z}_t^{\mathrm{std}}$	Standard corruption Eq. (1) at the same noise level	§4.5
$\boldsymbol{F}_t$	$(\boldsymbol{W}_{\bar{\gamma}_t}^{(M)})^{-1} \boldsymbol{W}_{\bar{\gamma}_t}^{ff}$, future signal in whitened noise coordinates	§4.5
g	Estimator of $\boldsymbol{x}$	Eq. (13)
Appendix-only notation		
ν	$\sqrt{1 - \gamma}$	App. A
R_n, ϑ_n	Pole modulus ν^{a_n} and angle $\Delta_\gamma \omega_n$ of mode n	App. A
Z	Normalizer of the kernel, $(\sum_\ell \tilde{u}_\ell^2)^{1/2} \geq 1$	App. A.2
V_δ	Unnormalized kernel autocorrelation, $\sum_\ell \tilde{u}_\ell \tilde{u}_{\ell+\delta}$	App. A.2
$\mathcal{C}(R, \vartheta, \psi)$	Closed form of $\sum_\ell R^\ell \cos(\ell\vartheta + \psi)$	Eq. (14)

Symbol	Meaning	Defined in
ε_j	Scalar i.i.d. $\mathcal{N}(0,1)$ inputs of the stationary process, $j \in \mathbb{Z}$	App. A.3
$\boldsymbol{E}$	Matrix of the EMA recurrence	Cor. 1
$\tilde{\boldsymbol{W}}$	Lower-triangular matrix of the unrolled recurrence	App. A.4
β_t	Noise variance $1-\bar{\alpha}_t$	App. A.6
$\boldsymbol{B}_t$	Noise operator $\boldsymbol{W}_{\tilde{\gamma}_t}^{(M)}$	App. A.6
$\boldsymbol{m}_t, \boldsymbol{M}_t, \boldsymbol{C}_t$	Coefficients of $\boldsymbol{z}_t = \boldsymbol{m}_t + \boldsymbol{M}_t\boldsymbol{x} + \boldsymbol{C}_t\boldsymbol{\epsilon}$	Eq. (23)
$f^\star$	Unrestricted minimizer of Eq. (11)	App. A.6
χ	Smooth, compactly supported test function	App. A.6
$\boldsymbol{U}$	$(\boldsymbol{W}_{\tilde{\gamma}_t}^{(M)})^{-1}\boldsymbol{W}_{\tilde{\gamma}_t}^{fo}$	App. A.7
$\boldsymbol{\zeta}$	Auxiliary white noise independent of all other variables	App. A.7
$\boldsymbol{\mu}(\boldsymbol{x})$	Mean of $q(\boldsymbol{z}_t \mid \boldsymbol{o}, \boldsymbol{x})$	App. A.8
Shorthand in Algorithms 1 and 2		
$\boldsymbol{H}, \boldsymbol{G}, \boldsymbol{B}$	$\boldsymbol{W}_\gamma^{fo}$, $\boldsymbol{W}_\gamma^{ff}$ and $\boldsymbol{W}_\gamma^{(M)}$ returned by $\mathrm{BuildOperators}(\gamma)$	App. B
$\mathcal{E}, \mathcal{D}$	Frozen encoder and decoder of the autoencoder	App. B
N_b	Minibatch size	Alg. 1
$\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{\epsilon}}_i$	Mixed signal and mixed noise of training example i	Alg. 1
$t_S > \cdots > t_0$	Sampling grid of S denoising steps, $t_0 = 0$	Alg. 2
Indices		
j	Frame (row) index within a window	
k	Delay of an input in the impulse response	
ℓ	Delay of an input in the memory kernel	
δ	Lag between two frames	
n, m	Mode indices	

A.2 Closed-form sums

For $0 \le R < 1$ and real ϑ, ψ, the real part of the geometric series $\sum_{\ell\ge0} R^\ell e^{i(\ell\vartheta+\psi)} = e^{i\psi}/(1 - Re^{i\vartheta})$ gives

$$\mathcal{C}(R,\vartheta,\psi) = \sum_{\ell\ge0} R^\ell \cos(\ell\vartheta+\psi) = \frac{\cos\psi - R\cos(\psi-\vartheta)}{(1-R)^2 + 4R\sin^2(\vartheta/2)}, \tag{14}$$

where the denominator equals $1 - 2R\cos\vartheta + R^2$ and is written in a form that avoids cancellation. Using $\cos x\cos y = \frac{1}{2}[\cos(x-y)+\cos(x+y)]$ with $x = \ell\vartheta_n$ and $y = (\ell+\delta)\vartheta_m$,

$$Z^2 = \sum_{\ell\ge0} \tilde{u}_\ell^2 = \frac{1}{2}\sum_{n,m} r_n r_m \Big[\mathcal{C}(R_nR_m, \vartheta_n-\vartheta_m, 0) + \mathcal{C}(R_nR_m, \vartheta_n+\vartheta_m, 0)\Big], \tag{15}$$

$$V_\delta = \sum_{\ell\ge0} \tilde{u}_\ell\,\tilde{u}_{\ell+\delta} = \frac{1}{2}\sum_{n,m} r_n r_m R_m^\delta \Big[\mathcal{C}(R_nR_m, \vartheta_n-\vartheta_m, -\delta\vartheta_m) + \mathcal{C}(R_nR_m, \vartheta_n+\vartheta_m, \delta\vartheta_m)\Big]. \tag{16}$$

For $\delta \ge 1$, the definition of κ_δ in equation 10 splits into the term $w_0 w_\delta$ and the terms with $k \ge 1$, and $\sum_{k\ge1} w_k w_{k+\delta} = (1-\gamma)\sum_{\ell\ge0} u_\ell u_{\ell+\delta}$. Since $u_\ell = \tilde{u}_\ell/Z$, this gives

$$\kappa_0 = 1, \qquad \kappa_\delta = \sqrt{\gamma(1-\gamma)}\,u_{\delta-1} + (1-\gamma)\,V_\delta/Z^2 \quad (\delta \ge 1). \tag{17}$$

Because the amplitudes are positive and sum to one, $\tilde{u}_0 = 1$ and $Z \ge 1$. As $\gamma \to 1$, every $R_n \to 0$, so $\tilde{u}_\ell \to 0$ for $\ell > 0$ while $\tilde{u}_0 = 1$, giving the endpoint $w = (1, 0, 0, \dots)$.

A.3 Admissibility (Proposition 1)

Proof of Proposition 1. Let $(\varepsilon_j)_{j\in\mathbb{Z}}$ be i.i.d. $\mathcal{N}(0,1)$ and $y_j = \sum_{k\ge0} w_k \varepsilon_{j-k}$. Since $\sum_k w_k^2 = 1$, the series converges in mean square, (y_j) is a stationary Gaussian process, and $\mathrm{Cov}(y_i, y_j) = \kappa_{|i-j|}$; hence $\boldsymbol{J}^{(L)}$ is the covariance of $(y_0, \dots, y_{L-1})$.

(a) For $0 \le j < L$, split $y_j = \sum_{k=0}^{j} w_{j-k}\varepsilon_k + y_j^{\mathrm{pre}}$, where y_j^{pre} depends only on $(\varepsilon_k)_{k<0}$. The first term equals $\boldsymbol{Q}\,(\varepsilon_0, \dots, \varepsilon_{L-1})^\top$ with $\boldsymbol{Q}$ lower triangular and diagonal entries $w_0 = \sqrt{\gamma} > 0$, and it is independent of y^{pre}. Thus $\boldsymbol{J}^{(L)} = \boldsymbol{Q}\boldsymbol{Q}^\top + \mathrm{Cov}(y^{\mathrm{pre}}) \succeq \boldsymbol{Q}\boldsymbol{Q}^\top \succ 0$. A positive definite matrix has a unique Cholesky factor, which is lower triangular with a positive diagonal and therefore causal and invertible.

(b) $\mathrm{diag}(\boldsymbol{W}\boldsymbol{W}^\top) = \mathrm{diag}(\boldsymbol{J}) = \kappa_0 = \sum_k w_k^2 = 1$.

(c) A conditional variance of a Gaussian vector depends only on its covariance, so we may compute it for $(y_0, \dots, y_{L-1})^\top = \boldsymbol{W}\boldsymbol{e}$ with $\boldsymbol{e} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$. Because $\boldsymbol{W}$ is lower triangular with a positive diagonal, $(e_0, \dots, e_{j-1})$ is an invertible linear function of $(y_0, \dots, y_{j-1})$, and e_j is independent of it. Hence $\mathrm{Var}(y_j \mid y_0, \dots, y_{j-1}) = [\boldsymbol{W}]_{jj}^2$. For the process itself, $y_0, \dots, y_{j-1}$ are functions of $(\varepsilon_k)_{k\le j-1}$, whereas $y_j = w_0\varepsilon_j + y_j^{\mathrm{past}}$ with y_j^{past} a function of $(\varepsilon_k)_{k\le j-1}$ and ε_j independent of the past. For jointly Gaussian variables, conditioning on more information cannot increase the conditional variance, so $[\boldsymbol{W}]_{jj}^2 = \mathrm{Var}(y_j \mid y_{<j}) \ge \mathrm{Var}\left(y_j \mid (\varepsilon_k)_{k\le j-1}\right) = w_0^2 = \gamma$. Finally, $\bar\gamma_t = 1 - \eta(1 - \bar\alpha_t) \ge 1 - \eta$. This gives a positive uniform lower bound if $\eta < 1$. For $\eta = 1$, each finite step with $\bar\alpha_t > 0$ remains admissible, but the zero-signal endpoint would have $\bar\gamma_T = 0$ and lies outside the construction.

(d) The leading $N \times N$ block of the Toeplitz matrix $\boldsymbol{J}^{(N+M)}$ is $\boldsymbol{J}^{(N)}$. In the block form equation 5, $\boldsymbol{W}^{oo}\boldsymbol{W}^{oo\top} = \boldsymbol{J}^{(N)}$ and $\boldsymbol{W}^{oo}$ is lower triangular with a positive diagonal, so $\boldsymbol{W}^{oo} = \boldsymbol{W}^{(N)}$ by uniqueness. The trailing $M \times M$ block of $\boldsymbol{J}^{(N+M)}$ is likewise $\boldsymbol{J}^{(M)}$; multiplying out the block form yields

$$\boldsymbol{W}^{fo}\boldsymbol{W}^{fo\top} + \boldsymbol{W}^{ff}\boldsymbol{W}^{ff\top} = \boldsymbol{J}^{(M)} = \boldsymbol{W}^{(M)}\boldsymbol{W}^{(M)\top}. \tag{18}$$

Moreover, the first row of $\boldsymbol{W}^{(M)}$ is $(1, 0, \dots, 0)$ because $\kappa_0 = 1$, so the noise branch of equation 6 leaves its first frame unchanged.

(e) At $\gamma = 1$, $w = (1, 0, \dots)$ by Appendix A.2, so $\boldsymbol{J}^{(L)} = \boldsymbol{I}$ and $\boldsymbol{W}^{(L)} = \boldsymbol{I}$. As $\gamma \to 1$, both coefficients in equation 17 tend to zero, while $|u_{\delta-1}| \le 1$ and $|V_\delta/Z^2| = |\sum_\ell u_\ell u_{\ell+\delta}| \le 1$ by the unit norm of u and the Cauchy–Schwarz inequality. Hence $\kappa_\delta \to 0$ for every $\delta \ge 1$, so $\boldsymbol{J}_\gamma^{(L)} \to \boldsymbol{I}$, and continuity of the Cholesky factor on positive definite matrices gives $\boldsymbol{W}_\gamma^{(L)} \to \boldsymbol{I}$. □

Corollary 1 (EMA special case)**.** *If $P = 1$, $a_1 = 1$ and $\omega_1 = 0$, then for every $\gamma \in (0, 1]$ the operator $\boldsymbol{W}_\gamma^{(L)}$ is the matrix of the recurrence $y_0 = e_0$, $y_j = \nu\, y_{j-1} + \sqrt{\gamma}\, e_j$, i.e., the exponential moving average operator of DyDiff (Guo et al., 2025).*

Proof. At $\gamma = 1$ both matrices are the identity. For $\gamma < 1$, $R_1 = \nu$, $\tilde u_\ell = \nu^\ell$, $Z^2 = 1/(1-\nu^2) = 1/\gamma$, $u_\ell = \sqrt{\gamma}\,\nu^\ell$ and $w_k = \sqrt{\gamma}\,\nu^k$, so $\kappa_\delta = \gamma \sum_k \nu^{2k+\delta} = \nu^\delta$. Let $\boldsymbol{E}$ be the matrix of the recurrence, with $E_{j0} = \nu^j$ and $E_{jk} = \sqrt{\gamma}\,\nu^{j-k}$ for $1 \le k \le j$. For $i \ge j$,

$$(\boldsymbol{E}\boldsymbol{E}^\top)_{ij} = \nu^{i+j} + \gamma \sum_{k=1}^{j} \nu^{i+j-2k} = \nu^{i+j} + \nu^{i-j}\left(1 - \nu^{2j}\right) = \nu^{i-j} = \kappa_{i-j},$$

using $\gamma = 1 - \nu^2$. Since $\boldsymbol{E}$ is lower triangular with a positive diagonal, $\boldsymbol{E} = \mathrm{chol}(\boldsymbol{J}^{(L)})$. □

A.4 State-space recurrence (Proposition 2)

State-space realization. Unlike the post-input state $\boldsymbol{h}_k$ in equation 2, we use a pre-input state $\boldsymbol{s}_j$ that contains only inputs before frame j. This convention implements the one-frame delay in equation 9, with a separate direct term for the current input. For $\gamma \in (0, 1)$, define the $2P$-dimensional real system

$$\begin{aligned} \bar{\boldsymbol{A}} &= \mathrm{blockdiag}_n\left(R_n \begin{bmatrix} \cos\vartheta_n & -\sin\vartheta_n \\ \sin\vartheta_n & \cos\vartheta_n \end{bmatrix}\right), \qquad D = \sqrt{\gamma}, \\ \boldsymbol{b} &= (1, 0, \dots, 1, 0)^\top, \qquad \boldsymbol{c} = \frac{\sqrt{1-\gamma}}{Z}\,(r_1, 0, \dots, r_P, 0)^\top. \end{aligned} \tag{19}$$

Each block rotates $(1,0)^\top$ by $\ell\vartheta_n$ and scales it by R_n^ℓ, so $\boldsymbol{c}^\top \bar{\boldsymbol{A}}^\ell \boldsymbol{b} = \sqrt{1-\gamma}\, u_\ell = w_{\ell+1}$. Hence, with $\boldsymbol{s}_j = \sum_{k\geq 1} \bar{\boldsymbol{A}}^{k-1}\boldsymbol{b}\,\varepsilon_{j-k}$, the process of Appendix A.3 satisfies

$$\boldsymbol{s}_{j+1} = \bar{\boldsymbol{A}}\boldsymbol{s}_j + \boldsymbol{b}\,\varepsilon_j, \qquad y_j = \boldsymbol{c}^\top \boldsymbol{s}_j + D\,\varepsilon_j. \tag{20}$$

Because the spectral radius $\max_n R_n$ is below one, $\boldsymbol{s}_j$ is stationary with covariance $\boldsymbol{\Sigma}$ solving $\boldsymbol{\Sigma} = \bar{\boldsymbol{A}}\boldsymbol{\Sigma}\bar{\boldsymbol{A}}^\top + \boldsymbol{b}\boldsymbol{b}^\top$, and $\boldsymbol{s}_j$ is independent of $(\varepsilon_k)_{k\geq j}$.

Innovations. Let $\hat{\boldsymbol{s}}_j = \mathbb{E}[\boldsymbol{s}_j \mid y_0,\dots,y_{j-1}]$ and $\boldsymbol{S}_j = \mathrm{Cov}(\boldsymbol{s}_j \mid y_0,\dots,y_{j-1})$, so $\hat{\boldsymbol{s}}_0 = \boldsymbol{0}$ and $\boldsymbol{S}_0 = \boldsymbol{\Sigma}$. Conditional on $y_{<j}$, the pair $(\boldsymbol{s}_{j+1}, y_j)$ is Gaussian with means $(\bar{\boldsymbol{A}}\hat{\boldsymbol{s}}_j, \boldsymbol{c}^\top\hat{\boldsymbol{s}}_j)$, variance $\sigma_j^2 = \mathrm{Var}(y_j \mid y_{<j}) = \boldsymbol{c}^\top\boldsymbol{S}_j\boldsymbol{c} + D^2$ and cross-covariance $\mathrm{Cov}(\boldsymbol{s}_{j+1}, y_j \mid y_{<j}) = \bar{\boldsymbol{A}}\boldsymbol{S}_j\boldsymbol{c} + \boldsymbol{b}D$, because ε_j is independent of $\boldsymbol{s}_j$ and of $y_{<j}$. Gaussian conditioning gives the Kalman recursion

$$\boldsymbol{k}_j = \frac{\bar{\boldsymbol{A}}\boldsymbol{S}_j\boldsymbol{c} + \boldsymbol{b}D}{\sigma_j^2}, \qquad \hat{\boldsymbol{s}}_{j+1} = \bar{\boldsymbol{A}}\hat{\boldsymbol{s}}_j + \boldsymbol{k}_j\left(y_j - \boldsymbol{c}^\top\hat{\boldsymbol{s}}_j\right), \qquad \boldsymbol{S}_{j+1} = \bar{\boldsymbol{A}}\boldsymbol{S}_j\bar{\boldsymbol{A}}^\top + \boldsymbol{b}\boldsymbol{b}^\top - \sigma_j^2\,\boldsymbol{k}_j\boldsymbol{k}_j^\top. \tag{21}$$

Let $e_j = (y_j - \boldsymbol{c}^\top\hat{\boldsymbol{s}}_j)/\sigma_j$. By the defining property of conditional expectation, e_j is uncorrelated with every function of $y_{<j}$, and each e_k with $k < j$ is such a function. The e_j are therefore uncorrelated, jointly Gaussian and of unit variance, hence i.i.d. $\mathcal{N}(0,1)$. Substituting $y_j = \boldsymbol{c}^\top\hat{\boldsymbol{s}}_j + \sigma_j\,e_j$ into equation 21 gives the recurrence of Proposition 2,

$$y_j = \boldsymbol{c}^\top\hat{\boldsymbol{s}}_j + \sigma_j\,e_j, \qquad \hat{\boldsymbol{s}}_{j+1} = \bar{\boldsymbol{A}}\hat{\boldsymbol{s}}_j + \sigma_j\,\boldsymbol{k}_j\,e_j, \qquad \hat{\boldsymbol{s}}_0 = \boldsymbol{0}, \tag{22}$$

whose input sequence is $\boldsymbol{e} = (e_0,\dots,e_{L-1})$. Both $\boldsymbol{s}_j$ and its estimate $\hat{\boldsymbol{s}}_j$ precede the current output; the latter is initialized at zero mean with uncertainty $\boldsymbol{S}_0 = \boldsymbol{\Sigma}$, not at a known zero physical state.

Proof of Proposition 2. Unrolling equation 22 gives $(y_0,\dots,y_{L-1})^\top = \tilde{\boldsymbol{W}}\,(e_0,\dots,e_{L-1})^\top$ for a lower-triangular $\tilde{\boldsymbol{W}}$ with diagonal entries $\sigma_j \geq D = \sqrt{\gamma} > 0$. Then $\boldsymbol{J}^{(L)} = \mathrm{Cov}(y) = \tilde{\boldsymbol{W}}\tilde{\boldsymbol{W}}^\top$, and uniqueness of the Cholesky factor gives $\tilde{\boldsymbol{W}} = \boldsymbol{W}^{(L)}$, in particular $\sigma_j = [\boldsymbol{W}^{(L)}]_{jj}$. Since the recurrence is linear and its coefficients do not depend on the input, it computes $\boldsymbol{W}^{(L)}\boldsymbol{e}$ for any input sequence $\boldsymbol{e}$. Furthermore, $\sigma_0^2 = \boldsymbol{c}^\top\boldsymbol{\Sigma}\boldsymbol{c} + D^2 = \mathrm{Var}(y_0) = 1$, and by stationarity $\sigma_j^2 = \mathrm{Var}(y_0 \mid y_{-j},\dots,y_{-1})$, which cannot increase as the conditioning set grows. □

In the setting of Corollary 1, y_0 determines the state, so $\boldsymbol{S}_1 = \boldsymbol{0}$ and the recurrence closes on the previous output. The gains depend only on (γ,ϕ,j) and cost $O(LP^3)$ per dynamics level, after which applying the recurrence to d-dimensional frames costs $O(LPd)$, compared with $O(L^3 + L^2 d)$ for equation 10.

A.5 Strictly richer correlations (Proposition 3)

Proof of Proposition 3. The EMA special case follows from Corollary 1, whose proof gives $\kappa_\delta = \nu^\delta = (1-\gamma)^{\delta/2} > 0$. To exhibit a correlation outside that family, fix $\gamma \in (0,1)$, take $P = 1$ and $\omega_1 = \pi/\Delta$, so that $\vartheta_1 = \pi$, and write $R = R_1 = \nu^{a_1} \in (0,1)$. Then $\tilde{u}_\ell = R^\ell\cos(\ell\pi) = (-R)^\ell$, $Z^2 = \sum_\ell R^{2\ell} = 1/(1-R^2)$ and $u_\ell = \sqrt{1-R^2}\,(-R)^\ell$, so

$$\sum_{\ell\geq 0} u_\ell u_{\ell+1} = (1-R^2)\sum_{\ell\geq 0}(-R)^{2\ell+1} = -R\,(1-R^2)\sum_{\ell\geq 0} R^{2\ell} = -R.$$

By equation 17 with $V_1/Z^2 = \sum_\ell u_\ell u_{\ell+1}$ and $u_0 = \sqrt{1-R^2}$,

$$\kappa_1 = \sqrt{\gamma(1-\gamma)}\,\sqrt{1-R^2} - (1-\gamma)\,R = \sqrt{1-\gamma}\left(\sqrt{\gamma}\sqrt{1-R^2} - \sqrt{1-\gamma}\,R\right),$$

which is negative if and only if $(1-\gamma)R^2 > \gamma(1-R^2)$, that is, $R^2 > \gamma$. Since $R^2 = (1-\gamma)^{a_1}$ and $\log(1-\gamma) < 0$, this holds exactly when $a_1 < \log\gamma/\log(1-\gamma)$, and such $a_1 > 0$ exists because the right-hand side is positive. □

For example, $\gamma = 0.5$ and $a_1 = 0.5$ give $\kappa_1 \approx -0.150$. The required frequency $\omega_1 = \pi/\Delta_\gamma$ lies within the clipping range $[0, 2\pi]$ of Appendix A.10 whenever $\Delta_\gamma \geq \frac{1}{2}$, i.e., $\gamma \geq 1 - e^{-1/2} \approx 0.39$, which covers every $\bar{\gamma}_t$ when $\eta \leq e^{-1/2}$, including $\eta = 0.5$.

A.6 SAMPLER CONSISTENCY (PROPOSITION 4)

Write $\beta_t = 1 - \bar{\alpha}_t$, $\boldsymbol{B}_t = \boldsymbol{W}^{(M)}_{\bar{\gamma}_t}$, $\boldsymbol{m}_t = \sqrt{\bar{\alpha}_t}\,\boldsymbol{W}^{fo}_{\bar{\gamma}_t}\boldsymbol{o}$, $\boldsymbol{M}_t = \sqrt{\bar{\alpha}_t}\,\boldsymbol{W}^{ff}_{\bar{\gamma}_t}$ and $\boldsymbol{C}_t = \sqrt{\beta_t}\,\boldsymbol{B}_t$, so that equation 6 reads

$$\boldsymbol{z}_t = \boldsymbol{m}_t + \boldsymbol{M}_t\boldsymbol{x} + \boldsymbol{C}_t\boldsymbol{\epsilon}, \tag{23}$$

with the same $(\boldsymbol{x}, \boldsymbol{\epsilon})$ for every t. For $0 < \bar{\alpha}_t < 1$, $\boldsymbol{M}_t$ and $\boldsymbol{C}_t$ are invertible by Proposition 1(a).

Proof of Proposition 4. Posterior means. The minimizer of equation 11 at step t is $f^\star = \mathbb{E}[\boldsymbol{B}_t\boldsymbol{\epsilon} \mid \boldsymbol{z}_t, \boldsymbol{o}]$. Taking the conditional expectation of equation 23 given $(\boldsymbol{z}_t, \boldsymbol{o})$, and using $\mathbb{E}[\boldsymbol{z}_t \mid \boldsymbol{z}_t, \boldsymbol{o}] = \boldsymbol{z}_t$, yields $\boldsymbol{z}_t = \boldsymbol{m}_t + \boldsymbol{M}_t\mathbb{E}[\boldsymbol{x} \mid \boldsymbol{z}_t, \boldsymbol{o}] + \sqrt{\beta_t}\,f^\star$. Solving for the first expectation gives exactly $\hat{\boldsymbol{x}}$ in equation 12 with $\hat{\boldsymbol{\epsilon}}_c = f^\star$, and $\mathbb{E}[\boldsymbol{\epsilon} \mid \boldsymbol{z}_t, \boldsymbol{o}] = \boldsymbol{B}_t^{-1}f^\star = \hat{\boldsymbol{\epsilon}}$. By linearity of equation 23 at step t', $\mathbb{E}[\boldsymbol{z}_{t'} \mid \boldsymbol{z}_t, \boldsymbol{o}] = \boldsymbol{m}_{t'} + \boldsymbol{M}_{t'}\hat{\boldsymbol{x}} + \boldsymbol{C}_{t'}\hat{\boldsymbol{\epsilon}}$, which is the update equation 12.

Transport. The entries of $\boldsymbol{J}^{(L)}_\gamma$ are smooth in $\gamma \in (0, 1)$ by Appendix A.2, the Cholesky factor is smooth on positive definite matrices, and $\bar{\gamma}_t$ is affine in $\bar{\alpha}_t$; hence $\boldsymbol{m}_t$, $\boldsymbol{M}_t$ and $\boldsymbol{C}_t$ are twice continuously differentiable on interior time intervals for $\eta > 0$. For $\eta = 0$, the operator is identically $\boldsymbol{I}$ and the same conclusion follows directly. Define $\boldsymbol{v}_t(\boldsymbol{z}) = \dot{\boldsymbol{m}}_t + \dot{\boldsymbol{M}}_t\mathbb{E}[\boldsymbol{x} \mid \boldsymbol{z}_t = \boldsymbol{z}, \boldsymbol{o}] + \dot{\boldsymbol{C}}_t\mathbb{E}[\boldsymbol{\epsilon} \mid \boldsymbol{z}_t = \boldsymbol{z}, \boldsymbol{o}]$. For a smooth, compactly supported test function χ and data with finite second moments,

$$\frac{d}{dt}\mathbb{E}\big[\chi(\boldsymbol{z}_t) \mid \boldsymbol{o}\big] = \mathbb{E}\big[\nabla\chi(\boldsymbol{z}_t)^\top(\dot{\boldsymbol{m}}_t + \dot{\boldsymbol{M}}_t\boldsymbol{x} + \dot{\boldsymbol{C}}_t\boldsymbol{\epsilon}) \,\big|\, \boldsymbol{o}\big] = \mathbb{E}\big[\nabla\chi(\boldsymbol{z}_t)^\top\boldsymbol{v}_t(\boldsymbol{z}_t) \,\big|\, \boldsymbol{o}\big],$$

by the tower property, which is the weak form of the continuity equation $\partial_t q_t + \nabla\cdot(q_t\boldsymbol{v}_t) = 0$ for $q_t = q(\boldsymbol{z}_t \mid \boldsymbol{o})$.

Consistency. By the first part, $\boldsymbol{z}_t = \boldsymbol{m}_t + \boldsymbol{M}_t\hat{\boldsymbol{x}} + \boldsymbol{C}_t\hat{\boldsymbol{\epsilon}}$, so $\boldsymbol{z}_{t'} - \boldsymbol{z}_t = (\boldsymbol{m}_{t'} - \boldsymbol{m}_t) + (\boldsymbol{M}_{t'} - \boldsymbol{M}_t)\hat{\boldsymbol{x}} + (\boldsymbol{C}_{t'} - \boldsymbol{C}_t)\hat{\boldsymbol{\epsilon}} = (t' - t)\,\boldsymbol{v}_t(\boldsymbol{z}_t) + O\big((t' - t)^2\big)$ by Taylor's theorem, where the remainder is uniform wherever the posterior means are bounded. □

Assuming the velocity admits a unique flow and the continuity equation has a unique solution in the considered class, integrating $\dot{\boldsymbol{z}} = \boldsymbol{v}_t(\boldsymbol{z})$ backward from the exact terminal marginal reproduces every marginal q_t on the interior interval, and equation 12 is its first-order discretization. Extension to a degenerate endpoint requires separate control of the flow limit. At the last step, $\bar{\alpha}_{t'} = \bar{\gamma}_{t'} = 1$ gives $\boldsymbol{m}_{t'} = \boldsymbol{0}$, $\boldsymbol{M}_{t'} = \boldsymbol{I}$ and $\boldsymbol{C}_{t'} = \boldsymbol{0}$, so the update returns $\hat{\boldsymbol{x}} = \mathbb{E}[\boldsymbol{x} \mid \boldsymbol{z}_t, \boldsymbol{o}]$. Two approximations remain, and both are shared with DDIM. First, by the law of total covariance, $\mathrm{Cov}(\boldsymbol{z}_{t'} \mid \boldsymbol{o})$ exceeds the covariance of $\mathbb{E}[\boldsymbol{z}_{t'} \mid \boldsymbol{z}_t, \boldsymbol{o}]$ by $\mathbb{E}[\mathrm{Cov}(\boldsymbol{z}_{t'} \mid \boldsymbol{z}_t, \boldsymbol{o}) \mid \boldsymbol{o}] \succeq 0$, so a large step omits conditional variance. Second, the initialization $\boldsymbol{B}_T\boldsymbol{\xi}$ matches q_T for arbitrary data distributions at $\bar{\alpha}_T = 0$, provided $\bar{\gamma}_T > 0$, since then $\boldsymbol{m}_T = \boldsymbol{0}$ and $\boldsymbol{M}_T = \boldsymbol{0}$. At $\bar{\alpha}_T > 0$ it is generally approximate; special data distributions can still yield equality. These statements concern the ideal denoiser under the forward marginals, not errors along a learned sampler's trajectories.

A.7 INFORMATION ORDERING (PROPOSITION 5)

Proof of Proposition 5. Fix t, write $\boldsymbol{B} = \boldsymbol{W}^{(M)}_{\bar{\gamma}_t}$, $\boldsymbol{F} = \boldsymbol{B}^{-1}\boldsymbol{W}^{ff}$ and $\boldsymbol{U} = \boldsymbol{B}^{-1}\boldsymbol{W}^{fo}$, and let $\beta = 1 - \bar{\alpha}_t$. Multiplying equation 18 by $\boldsymbol{B}^{-1}$ on the left and by $\boldsymbol{B}^{-\top}$ on the right gives $\boldsymbol{F}\boldsymbol{F}^\top + \boldsymbol{U}\boldsymbol{U}^\top = \boldsymbol{I}$, hence $\boldsymbol{F}\boldsymbol{F}^\top = \boldsymbol{I} - \boldsymbol{U}\boldsymbol{U}^\top \preceq \boldsymbol{I}$.

Given $\boldsymbol{o}$, the map $\boldsymbol{z}_t \mapsto \tilde{\boldsymbol{z}}_t = \boldsymbol{B}^{-1}(\boldsymbol{z}_t - \sqrt{\bar{\alpha}_t}\,\boldsymbol{W}^{fo}\boldsymbol{o}) = \sqrt{\bar{\alpha}_t}\,\boldsymbol{F}\boldsymbol{x} + \sqrt{\beta}\,\boldsymbol{\epsilon}$ is invertible, so estimators based on $(\boldsymbol{z}_t, \boldsymbol{o})$ and on $(\tilde{\boldsymbol{z}}_t, \boldsymbol{o})$ achieve the same errors. Let $\boldsymbol{\zeta} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ be independent of everything else and define $\tilde{\boldsymbol{z}}'_t = \boldsymbol{F}\boldsymbol{z}^{\mathrm{std}}_t + \sqrt{\beta}\,(\boldsymbol{I} - \boldsymbol{F}\boldsymbol{F}^\top)^{1/2}\boldsymbol{\zeta}$. Conditional on $(\boldsymbol{x}, \boldsymbol{o})$, $\tilde{\boldsymbol{z}}'_t$ is Gaussian with mean $\sqrt{\bar{\alpha}_t}\,\boldsymbol{F}\boldsymbol{x}$ and covariance $\beta\,\boldsymbol{F}\boldsymbol{F}^\top + \beta\,(\boldsymbol{I} - \boldsymbol{F}\boldsymbol{F}^\top) = \beta\boldsymbol{I}$, which is the conditional law of $\tilde{\boldsymbol{z}}_t$. Hence $(\boldsymbol{x}, \boldsymbol{o}, \tilde{\boldsymbol{z}}'_t)$ and $(\boldsymbol{x}, \boldsymbol{o}, \tilde{\boldsymbol{z}}_t)$ have the same joint distribution, and $\tilde{\boldsymbol{z}}'_t$ is computed from $\boldsymbol{z}^{\mathrm{std}}_t$ and $\boldsymbol{\zeta}$ without $\boldsymbol{x}$. For any estimator g with finite risk, $g(\tilde{\boldsymbol{z}}'_t, \boldsymbol{o})$ is a function of $(\boldsymbol{z}^{\mathrm{std}}_t, \boldsymbol{\zeta}, \boldsymbol{o})$, so $\mathbb{E}\|\boldsymbol{x} - g(\tilde{\boldsymbol{z}}_t, \boldsymbol{o})\|^2 = \mathbb{E}\|\boldsymbol{x} - g(\tilde{\boldsymbol{z}}'_t, \boldsymbol{o})\|^2 \geq \mathbb{E}\|\boldsymbol{x} - \mathbb{E}[\boldsymbol{x} \mid \boldsymbol{z}^{\mathrm{std}}_t, \boldsymbol{\zeta}, \boldsymbol{o}]\|^2 = \mathbb{E}\|\boldsymbol{x} - \mathbb{E}[\boldsymbol{x} \mid \boldsymbol{z}^{\mathrm{std}}_t, \boldsymbol{o}]\|^2$, where the inequality holds because the conditional mean minimizes the squared error for data with finite second moments, and the last equality uses the independence of $\boldsymbol{\zeta}$. Minimizing over g yields equation 13. □

For the EMA operator of Corollary 1 with $N \geq 1$, every future column of $\boldsymbol{W}^{(N+M)}$ has global index at least one, so its entries are $\sqrt{\gamma}\,\nu^{j-k}$, whereas $\boldsymbol{W}^{(M)}$ has first column $(\nu^j)_j$ and the same entries elsewhere. Hence $\boldsymbol{W}^{ff} = \boldsymbol{W}^{(M)}\,\mathrm{diag}(\sqrt{\gamma}, 1, \ldots, 1)$ and $\boldsymbol{F} = \mathrm{diag}(\sqrt{\gamma}, 1, \ldots, 1)$: only the first future frame is attenuated. For a general DSSD operator, $\boldsymbol{I} - \boldsymbol{F}\boldsymbol{F}^\top = \boldsymbol{U}\boldsymbol{U}^\top$ has rank at most $\min(N, M)$.

A.8 Colored-noise regression and the score

Fix t and ϕ, and write $\beta = 1 - \bar{\alpha}_t$, $\boldsymbol{B} = \boldsymbol{W}^{(M)}_{\tilde{\gamma}_t}$ and $\boldsymbol{\mu}(\boldsymbol{x}) = \sqrt{\bar{\alpha}_t}\,(\boldsymbol{W}^{fo}\boldsymbol{o} + \boldsymbol{W}^{ff}\boldsymbol{x})$. For each t, the minimizer of equation 11 over unrestricted functions is $f^\star(\boldsymbol{z}, \boldsymbol{o}, t) = \mathbb{E}[\boldsymbol{B}\boldsymbol{\epsilon} \mid \boldsymbol{z}_t = \boldsymbol{z}, \boldsymbol{o}]$. The marginal is the mixture $q(\boldsymbol{z} \mid \boldsymbol{o}) = \int p(\boldsymbol{x} \mid \boldsymbol{o})\,\mathcal{N}\big(\boldsymbol{z}; \boldsymbol{\mu}(\boldsymbol{x}), \beta\boldsymbol{B}\boldsymbol{B}^\top\big)\,d\boldsymbol{x}$. Differentiating under the integral, which is valid under standard integrability conditions,

$$\nabla_{\boldsymbol{z}} \log q(\boldsymbol{z} \mid \boldsymbol{o}) = \mathbb{E}\big[-(\beta\boldsymbol{B}\boldsymbol{B}^\top)^{-1}\big(\boldsymbol{z} - \boldsymbol{\mu}(\boldsymbol{x})\big) \,\big|\, \boldsymbol{z}_t = \boldsymbol{z}, \boldsymbol{o}\big] = -\beta^{-1/2}\,(\boldsymbol{B}\boldsymbol{B}^\top)^{-1} f^\star(\boldsymbol{z}, \boldsymbol{o}, t), \tag{24}$$

since $\boldsymbol{z}_t - \boldsymbol{\mu}(\boldsymbol{x}) = \sqrt{\beta}\,\boldsymbol{B}\boldsymbol{\epsilon}$. For any fixed operator, the colored target therefore determines the score, and the correlation between target coordinates requires no whitening of equation 11. Whitening would regress on $\boldsymbol{B}^{-1}$ times the target, that is, on $\boldsymbol{\epsilon}$ itself; since conditional expectation commutes with the fixed matrix $\boldsymbol{B}$, the unrestricted minimizers of the two losses are related by $\mathbb{E}[\boldsymbol{\epsilon} \mid \boldsymbol{z}_t, \boldsymbol{o}] = \boldsymbol{B}^{-1} f^\star$, so they carry the same information, and the unwhitened minimizer is the one that enters the sampling update equation 12 directly. The two losses do weight the coordinates of the residual differently, which can matter for a finite denoiser; this is part of the regression geometry discussed in Section 4.5. This statement concerns a fixed ϕ; the value of ϕ that minimizes equation 11 need not minimize a forecasting error.

A.9 Nonzero-frequency initialization

For $\ell \geq 0$, $\partial \tilde{u}_\ell / \partial \omega_n = -r_n R_n^\ell\, \ell\Delta\, \sin(\ell\Delta\omega_n)$, which vanishes for every ℓ and γ when $\omega_n = 0$. The normalized kernel, the autocorrelation, the operator equation 10 and the loss equation 11 depend on ω_n only through $(\tilde{u}_\ell)_{\ell \geq 0}$, so $\partial\mathcal{L}/\partial\omega_n = 0$ at $\omega_n = 0$. With a zero gradient, first-order optimizers such as Adam leave both the parameter and its moment estimates unchanged, so a mode initialized at zero frequency remains non-oscillatory throughout training. This motivates the nonzero initial frequencies of Appendix A.10.

A.10 Stable parameterization and optimization

Bounded modes. Decay rates and frequencies are clipped to $\log a_n \in [-9, 3]$ and $\omega_n \in [0, 2\pi]$, and amplitudes are kept positive by the softmax. The closed-form sums and the Cholesky factorization are computed in double precision.

Initialization. Decay rates are log-spaced in $[0.12, 3]$, frequencies are set to $\omega_n = 2\pi(n - \frac{1}{2})/P$, and amplitudes are uniform. Nonzero initial frequencies are necessary by the argument of Appendix A.9.

Operator certificates. Proposition 1 holds exactly, but finite-precision inverses can still amplify errors when a learned operator is ill conditioned. Before training and at every evaluation, we therefore check the operator on a fixed grid of 33 dynamics levels taken from the schedule, at window lengths M and $N + M$: the reconstruction error of the full and history-conditioned inverses, unit variance, causality, the identity at $\gamma = 1$, and a condition number below 10^3. The checks are run in single and double precision, and training stops if any of them fails. These checks certify only the tested dynamics levels and checkpoints, not every possible parameter value or every point between grid levels.

B TRAINING AND SAMPLING ALGORITHMS

Algorithms 1 and 2 summarize the mixed-noise training objective and deterministic sampler. The pretrained encoder $\mathcal{E}$ and decoder $\mathcal{D}$ are frozen. All temporal matrices act independently on each latent coordinate.

Shared operator construction. Both algorithms use $\mathrm{BuildOperators}(\gamma)$, with the current dynamics parameters ϕ and window lengths N, M understood. It returns three matrices: $\boldsymbol{H}$, which maps the observed history into the future; $\boldsymbol{G}$, which mixes future signal frames; and $\boldsymbol{B}$, which mixes future noise. The routine constructs the normalized SSM kernel and impulse response in equation 8 and equation 9, computes their stationary covariance matrices for lengths $N+M$ and M using Appendix A.2, and takes their lower-triangular Cholesky factors as in equation 10. The blocks $\boldsymbol{H} = \boldsymbol{W}_\gamma^{fo}$ and $\boldsymbol{G} = \boldsymbol{W}_\gamma^{ff}$ come from the length-$N+M$ factor, while $\boldsymbol{B} = \boldsymbol{W}_\gamma^{(M)}$. In general, $\boldsymbol{G} \neq \boldsymbol{B}$: signal mixing includes the observed history, whereas noise mixing uses a separate future window. At $\gamma = 1$, the routine returns $(\boldsymbol{0}, \boldsymbol{I}_M, \boldsymbol{I}_M)$. Parameter bounds and double-precision covariance calculations follow Appendix A.10. The construction is differentiable in ϕ during training.

Algorithm 1 Training of DSSD

Require: Training sequences; frozen encoder $\mathcal{E}$; window lengths N, M; batch size N_b; schedule $\{\bar\alpha_t\}_{t=1}^T$; mixing strength η; optimizer and update budget
Ensure: Trained parameters (θ, ϕ)

Initialize (θ, ϕ) as in Appendix A.10
while update budget remains **do**
  Sample a minibatch $\{(O_i, X_i)\}_{i=1}^{N_\mathrm{b}}$
  $\boldsymbol{o}_i \leftarrow \mathcal{E}(O_i)$; $\boldsymbol{x}_i \leftarrow \mathcal{E}(X_i)$
  for $i = 1, \dots, N_\mathrm{b}$ **do**
    Sample $t_i \sim \mathrm{Uniform}\{1, \dots, T\}$
    Sample $\boldsymbol{\epsilon}_i \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$
    $\gamma_i \leftarrow 1 - \eta(1 - \bar\alpha_{t_i})$
    $(\boldsymbol{H}_i, \boldsymbol{G}_i, \boldsymbol{B}_i) \leftarrow \mathrm{BuildOperators}(\gamma_i)$
    $\tilde{\boldsymbol{x}}_i \leftarrow \boldsymbol{H}_i \boldsymbol{o}_i + \boldsymbol{G}_i \boldsymbol{x}_i$
    $\tilde{\boldsymbol{\epsilon}}_i \leftarrow \boldsymbol{B}_i \boldsymbol{\epsilon}_i$
    $\boldsymbol{z}_i \leftarrow \sqrt{\bar\alpha_{t_i}}\, \tilde{\boldsymbol{x}}_i + \sqrt{1 - \bar\alpha_{t_i}}\, \tilde{\boldsymbol{\epsilon}}_i$
    $\hat{\boldsymbol{\epsilon}}_{c,i} \leftarrow f_\theta(\boldsymbol{z}_i, \boldsymbol{o}_i, t_i)$
  end for
  $\mathcal{L} \leftarrow N_\mathrm{b}^{-1} \sum_{i=1}^{N_\mathrm{b}} \|\hat{\boldsymbol{\epsilon}}_{c,i} - \tilde{\boldsymbol{\epsilon}}_i\|_2^2$
  Compute $\nabla_{\theta,\phi} \mathcal{L}$ through $\boldsymbol{z}_i$ and $\tilde{\boldsymbol{\epsilon}}_i$
  Update (θ, ϕ)
end while
return (θ, ϕ)

Algorithm 2 Sampling of DSSD

Require: Observed frames O; frozen encoder $\mathcal{E}$ and decoder $\mathcal{D}$; trained (θ, ϕ); prediction length M; schedule $\bar\alpha$ and mixing strength η; grid $T \geq t_S > \cdots > t_1 > t_0 = 0$
Ensure: Predicted future frames $\hat{X}$

$\boldsymbol{o} \leftarrow \mathcal{E}(O)$; $N \leftarrow$ number of frames in $\boldsymbol{o}$
Set $\bar\alpha_0 \leftarrow 1$ and $\bar\gamma_0 \leftarrow 1$
for $j = 0, \dots, S$ **do**
  $\bar\gamma_{t_j} \leftarrow 1 - \eta(1 - \bar\alpha_{t_j})$
  $(\boldsymbol{H}_{t_j}, \boldsymbol{G}_{t_j}, \boldsymbol{B}_{t_j}) \leftarrow \mathrm{BuildOperators}(\bar\gamma_{t_j})$
end for
Draw $\boldsymbol{\xi} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ of shape $M \times d$
$\boldsymbol{z}_{t_S} \leftarrow \boldsymbol{B}_{t_S} \boldsymbol{\xi}$
for $j = S, \dots, 1$ **do**
  $t \leftarrow t_j$; $t' \leftarrow t_{j-1}$
  $\hat{\boldsymbol{\epsilon}}_c \leftarrow f_\theta(\boldsymbol{z}_t, \boldsymbol{o}, t)$
  Solve $\boldsymbol{G}_t \hat{\boldsymbol{x}} = (\boldsymbol{z}_t - \sqrt{1 - \bar\alpha_t}\, \hat{\boldsymbol{\epsilon}}_c)/\sqrt{\bar\alpha_t} - \boldsymbol{H}_t \boldsymbol{o}$ for $\hat{\boldsymbol{x}}$
  Solve $\boldsymbol{B}_t \hat{\boldsymbol{\epsilon}} = \hat{\boldsymbol{\epsilon}}_c$ for $\hat{\boldsymbol{\epsilon}}$
  $\boldsymbol{z}_{t'} \leftarrow \sqrt{\bar\alpha_{t'}}(\boldsymbol{H}_{t'} \boldsymbol{o} + \boldsymbol{G}_{t'} \hat{\boldsymbol{x}})$
  $\boldsymbol{z}_{t'} \leftarrow \boldsymbol{z}_{t'} + \sqrt{1 - \bar\alpha_{t'}}\, \boldsymbol{B}_{t'} \hat{\boldsymbol{\epsilon}}$
end for
return $\hat{X} \leftarrow \mathcal{D}(\boldsymbol{z}_0)$

Training details. The noise target is not detached from the dynamics parameters. The loss is written as a batch average of squared norms; the implementation additionally averages over latent coordinates. Operator checks are performed before training and at evaluation checkpoints as described in Appendix A.10.

Sampling details. All model-evaluation steps satisfy $0 < \bar\alpha_{t_j} < 1$ and $\bar\gamma_{t_j} > 0$. The parameters remain fixed, so the operators can be precomputed; no gradients or new noise draws are needed inside the sampling loop. At $t' = 0$, $(\boldsymbol{H}_0, \boldsymbol{G}_0, \boldsymbol{B}_0) = (\boldsymbol{0}, \boldsymbol{I}_M, \boldsymbol{I}_M)$, and the update returns $\boldsymbol{z}_0 = \hat{\boldsymbol{x}}$. The initial colored Gaussian is generally an approximation to the terminal forward marginal at nonzero $\bar\alpha_{t_S}$, as discussed in Appendix A.6. We use steps $1, \dots, T$ and the clean endpoint 0; the implementation uses array indices $0, \dots, T-1$ and a clean-endpoint sentinel -1.

C Experimental Details

C.1 Data and forecasting tasks

Table 4 specifies the input and prediction windows. All models forecast future frames from observed frames. We map SEVIR and RGB inputs from $[0, 255]$ to $[-1, 1]$ and divide both Turbulence velocity components by four, without clipping.

Table 4: Forecasting tasks. N and M are the numbers of observed and future frames.

Dataset	Input	Resolution	$N \to M$	Input transform
SEVIR	VIL, 1 channel	128^2	$7 \to 6$	$x \mapsto x/127.5 - 1$
Turbulence	Velocity, 2 channels	64^2	$4 \to 11$	$x \mapsto x/4$
BAIR	RGB video	64^2	$1 \to 15$	$x \mapsto x/127.5 - 1$
RoboNet	RGB video	64^2	$2 \to 10$	$x \mapsto x/127.5 - 1$

C.2 Architecture, optimization, and sampling

Within each dataset, all methods use the same pretrained, frozen, framewise VAE (Kingma & Welling, 2014), including its EMA weights. The VAE reduces each spatial dimension by a factor of four, outputs three latent channels, and uses a latent scale factor of 0.18215. During training, latent frames are sampled from the VAE posterior; evaluation encodes observations using its posterior mode. The U-Net takes clean observed latents and noisy future latents as conditioning and prediction inputs, respectively. We use a video U-Net (Ronneberger et al., 2015) inspired by Stable Video Diffusion (Blattmann et al., 2023), configured identically for all three methods.

Table 5: Shared model, optimization, and sampling settings.

Component	Setting
Video U-Net	64 base channels; channel multipliers $(1, 2, 4, 4)$; two residual blocks per level.
Attention and time mixing	Attention at downsampling factors $(4, 2, 1)$; four heads; transformer depth 1; dropout 0; temporal convolution kernel $(3, 1, 1)$ and learned spatial/temporal mixing.
Diffusion	1,000 timesteps; cosine schedule (Nichol & Dhariwal, 2021) with $s = 0.008$; ϵ prediction.
Optimization	Adam (Kingma & Ba, 2015), $(\beta_1, \beta_2) = (0.9, 0.999)$, $\epsilon_{\text{Adam}} = 10^{-8}$, weight decay 0; effective batch size 16; FP32. No denoiser-weight EMA.
Sampling	50 deterministic DDIM steps on the same `released_uniform` timestep grid.

Table 6: Method-specific temporal operator and learning-rate schedule.

Method	Temporal operator	Learning rate
DPM	Identity (no temporal mixing).	10^{-4}.
DyDiff	Fixed EMA, $\eta = 0.5$.	10^{-4}.
DSSD	Learned stationary SSM operator, state size 12 (six conjugate pairs), $\eta = 0.5$	10^{-4}.

The DSSD operator parameters form a separate optimizer group; the learning-rate schedule applies to both groups. DSSD's stationary covariance and Cholesky factor are computed in double precision, with the parameter bounds and numerical operator checks detailed in Appendix A.10.

DiT backbone. For the SEVIR backbone ablation (Appendix D.2), we replace the U-Net with a DiT-S/4 (Peebles & Xie, 2023): 12 blocks of width 384 with six attention heads, MLP ratio 4 and no dropout. Each latent frame is split spatially into 4×4 patches, giving 64 tokens per frame and 832 tokens for the 13 frames, and attention runs jointly over all frames with fixed sine–cosine spatial and temporal position embeddings. The inputs are the same as for the U-Net, the timestep enters through adaLN-Zero, and only the outputs at future-frame positions are used. The DiT has

32,529,840 parameters; all other settings, including the method-specific learning rates, match the U-Net runs on SEVIR.

Compute. Each model is trained on a single NVIDIA A100 80GB GPU, using the PCIe variant for SEVIR and Turbulence and the SXM4 variant for BAIR and RoboNet. Within each dataset, the three methods share the same GPU type and batch configuration. DSSD adds 18 trainable parameters for its six conjugate mode pairs to the 60,982,185-parameter denoiser shared by all methods, whereas DyDiff's fixed EMA operator adds none. Table 7 reports the training cost of the first 50,000 updates. Relative to the faster baseline, DSSD's median time per update is 1.4% higher on SEVIR and 2.0% higher on BAIR, is the lowest on Turbulence, and is 4.9% higher on RoboNet, computed from the values in Table 7. All three methods reserve the same GPU memory on every dataset. Learning the temporal operator therefore adds little training cost.

Table 7: Training cost over the first 50,000 updates on one A100 80GB GPU. s/upd.: median time per update (s); Total: wall-clock time (h); Mem.: reserved GPU memory (GiB).

Method	SEVIR			Turbulence			BAIR			RoboNet		
	s/upd.	Total	Mem.	s/upd.	Total	Mem.	s/upd.	Total	Mem.	s/upd.	Total	Mem.
DPM	2.22	31.8	48.8	1.06	14.8	18.4	0.98	21.5	11.5	0.82	17.4	9.2
DyDiff	2.19	31.3	48.8	1.07	15.0	18.4	0.98	21.9	11.5	0.83	17.8	9.2
DSSD (Ours)	2.22	31.9	48.8	1.02	14.8	18.4	1.00	22.7	11.5	0.86	17.7	9.2

C.3 Evaluation metrics

Evaluation protocol. The number K of generated futures per input follows the evaluation protocol of DyDiff (Guo et al., 2025), which adopts the prevailing convention of each domain. For SEVIR and Turbulence, the $K = 8$ generated futures form a probabilistic ensemble, as in Chen et al. (2024). The finite-ensemble CRPS estimator below accounts for ensemble size, and $K = 8$ keeps sampling tractable on the 128×128 SEVIR test set. For BAIR and RoboNet, we follow the video-prediction protocol of Gupta et al. (2023) and Wu et al. (2024), which selects among $K = 100$ generated videos per input. Because this selection improves with K, we retain $K = 100$ for comparability. Within each dataset, all methods share the same K and 50-step DDIM sampler, so method rankings are comparable. Scores are not comparable between the two dataset groups.

CRPS. For target y and generated futures $x^{(1)}, \dots, x^{(K)}$, the finite-ensemble probability-weighted-moment CRPS (Gneiting & Raftery, 2007; Zamo & Naveau, 2018) at a pixel is

$$\widehat{\mathrm{CRPS}}(y, x^{(1:K)}) = \frac{1}{K} \sum_{k=1}^{K} |x^{(k)} - y| - \frac{1}{K(K-1)} \sum_{1 \le i < j \le K} |x^{(i)} - x^{(j)}|.$$

Unpooled CRPS averages this score over pixels, channels, and forecast times, and is reported on all four datasets. On SEVIR and Turbulence, we also report pooled variants: we pool the target and each generated future spatially, using either 8×8 average or maximum pooling with stride 2, then compute the same CRPS. The subscripts in $\mathrm{w8}_{\mathrm{avg}}$ and $\mathrm{w8}_{\max}$ identify these two pooling operations. To retain compatibility with the reference evaluation, the pooled Turbulence variants include only pooling windows whose target velocities are all nonnegative; unpooled CRPS uses the full field. The per-horizon BAIR curves in Figure 3 use frame-wise CRPS after 8×8 spatial max pooling, averaged over the full BAIR test set.

CSI. CSI is reported on SEVIR and Turbulence. We threshold the fields before applying a 5×5 maximum neighborhood operator with stride 2. SEVIR uses raw VIL thresholds $16, 74, 133, 160, 181, 219$; Turbulence uses threshold zero for each velocity component. Each CSI is $\mathrm{TP}/(\mathrm{TP} + \mathrm{FP} + \mathrm{FN})$, and the reported score is macro-averaged over defined threshold–horizon–channel cells; cells without an observed or predicted event are excluded. We follow the implemented maximum-neighborhood convention, which differs from an average-pooling description of CSI in the DyDiff paper (Guo et al., 2025).

Temporal variogram score. The temporal variogram score (Scheuerer & Hamill, 2015) averages, over lags 1–4, the squared discrepancies between the target's absolute temporal increments raised to power $1/2$ and the ensemble mean of the corresponding powered absolute increments. The per-horizon BAIR curves use only consecutive future-frame pairs and therefore start at the second future frame, because the first has no preceding future frame.

Energy score. Let $\mathrm{dist}(a, b) = \|a - b\|_2/\sqrt{MCHW}$ denote the root-mean-square distance over a complete $M \times C \times H \times W$ future video. We report the energy score (Gneiting & Raftery, 2007)

$$\mathrm{ES}(y, x^{(1:K)}) = \frac{1}{K}\sum_{k=1}^{K} \mathrm{dist}(x^{(k)}, y) - \frac{1}{2K^2}\sum_{i=1}^{K}\sum_{j=1}^{K} \mathrm{dist}(x^{(i)}, x^{(j)}).$$

RMSE. RMSE is defined differently for the two dataset groups. On SEVIR and Turbulence, it is the RMSE of the ensemble mean of the K generated futures. On BAIR and RoboNet, it is the square root of the mean per-input minimum full-video MSE, so a generated future is selected for each input; the BAIR entries are computed as square roots of the corresponding MSE scores.

PSNR, SSIM, and LPIPS. These metrics are reported on BAIR and RoboNet. For each of the $K = 100$ generated videos, we average framewise PSNR, SSIM (Wang et al., 2004), and VGG-based LPIPS (Zhang et al., 2018) over future time. We then report the maximum PSNR and SSIM and the minimum LPIPS for each input, selecting independently for each metric, so the selected video can differ across the three metrics. PSNR and SSIM are evaluated on the $[0, 255]$ RGB scale; LPIPS uses RGB in $[-1, 1]$.

FVD. Unlike the selection-based metrics above, FVD (Unterthiner et al., 2018) compares the distribution of all 100 generated videos per input with the real videos. We prepend the observed frames to every generated future and to its real counterpart, then compute the Fréchet distance between I3D (Carreira & Zisserman, 2017) feature distributions. LPIPS and FVD are computed offline from saved predictions. Their implementation follows the stated aggregation protocol, but exact numerical identity with an unspecified upstream feature extractor is not assumed.

C.4 Learned-mode diagnostics

Each temporal mode corresponds to one complex-conjugate pole pair, so the six residue weights index six modes. Figure 4(a) reports the absolute centered residue logits $|\rho_n - \bar{\rho}|$ and their L_2 norm, the residue-logit norm, on SEVIR. Figure 7 additionally shows the residue shares $r_n = \mathrm{softmax}(\rho)_n$ for all four datasets. These quantities are computed from checkpoints saved every 10,000 updates from 10,000 to 50,000 updates. Centering removes the common logit shift that leaves the shares unchanged. The residue-logit norm measures the spread of the logits around their mean; absolute centered logits alone do not indicate whether a mode gains or loses weight. The shares directly show how relative weight is redistributed.

Figure 4(b) reports normalized residue entropy, $-\sum_{n=1}^{6} r_n \log r_n / \log 6$, computed from logged training values over the same update range. Values near one indicate nearly uniform shares, while lower values indicate greater concentration. These shares are mixture weights, not independent mode energies; their entropy does not measure pole decay or frequency. As a separate check on the near-constant Turbulence entropy, we reconstruct the discrete pole of its first mode at fixed $\gamma = 0.5$: its radius is 0.937 at 10,000 updates and 0.889 at 50,000 updates, indicating faster decay at that dynamics level despite little redistribution of residue weight. These parameter changes do not establish physical mode identification or a causal contribution to forecasting quality.

D Additional Results

D.1 Qualitative forecasts

Figures 5 and 6 show one test sequence per dataset. We select the sequence on which DSSD most improves on the better baseline in per-sequence 8×8 average-pooled CRPS. For SEVIR, we take

Table 8: Backbone ablation on SEVIR. Results with a U-Net and a DiT denoiser. **Bold**: best; underline: second best within each backbone.

Backbone	Method	CRPS ↓			CSI w5 ↑	Temporal variogram ↓	Energy score ↓	RMSE ↓
		Raw	$w8_{avg}$	$w8_{max}$				
U-Net	DPM	4.174	2.737	6.599	0.5518	2.339	10.862	15.420
	DyDiff	4.642	3.169	7.955	0.3911	4.492	13.024	25.540
	DSSD (Ours)	**3.975**	**2.421**	**5.995**	**0.5707**	**2.129**	**10.263**	**14.585**
DiT	DPM	5.425	3.862	7.453	0.4314	3.559	10.437	23.689
	DyDiff	5.260	3.789	7.327	0.4383	3.326	10.777	24.905
	DSSD (Ours)	**5.211**	**3.706**	**7.188**	**0.5118**	**2.784**	**10.302**	**23.375**

the highest-ranked sequence whose last two observations and target each have at most 5% saturated pixels and a standard deviation of at least 30 VIL units, because nearly saturated fields are hard to read. Each method row shows the generated future with the lowest full-video MAE among its K samples, and the error rows show that sample's absolute error.

In all four examples, errors concentrate in moving structures: storm cells in SEVIR, shear regions in Turbulence, and the robot arm in BAIR and RoboNet. The baselines' errors spread more with lead time. This is most visible from $t{+}5$ to $t{+}9$ in Turbulence, and against DyDiff in the robot videos. DSSD's errors stay more localized, although they also grow at the final RoboNet frame.

D.2 Backbone ablation on SEVIR

Table 8 compares the three methods with either a video U-Net or a Diffusion Transformer (DiT) denoiser (Peebles & Xie, 2023), described in Appendix C.2. Within each backbone and metric, bold and underlined values indicate the best and second-best methods, respectively.

DSSD obtains the best score on all seven reported metrics within both backbone groups. With DiT, it achieves a higher CSI w5 than DyDiff, the strongest baseline on this metric (0.5118 vs. 0.4383), and a lower temporal variogram score (2.784 vs. 3.326). Its raw CRPS is also lower (5.211 vs. 5.260). Thus, the favorable within-backbone ranking observed with the U-Net also holds for the transformer denoiser on SEVIR.

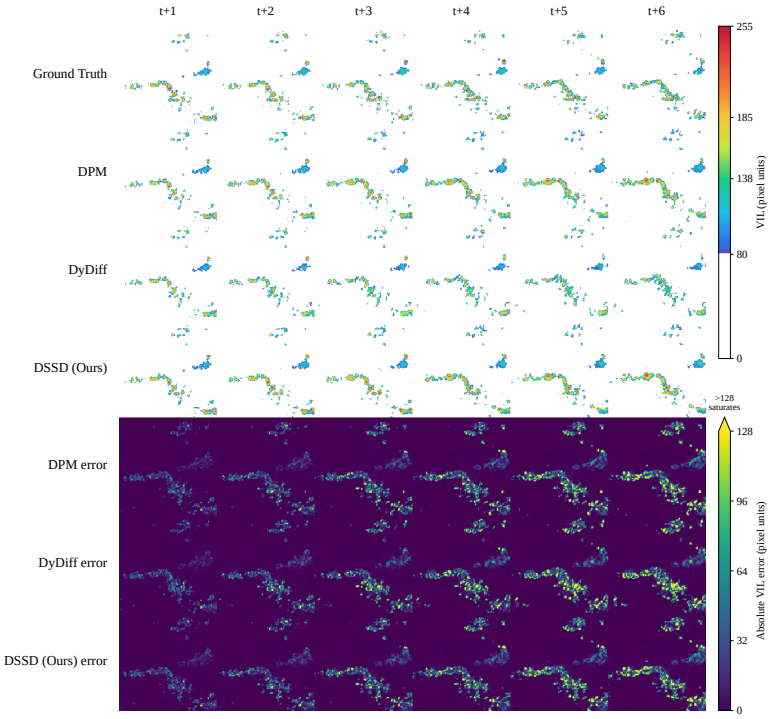
D.3 Mode-resolved temporal adaptation

Each temporal mode corresponds to one complex-conjugate pair of poles, so the six residue weights index six modes. Figure 7 tracks how the size and share of each mode change during training, revealing which modes the model comes to rely on and which it suppresses. Concretely, it shows the softmax residue shares $r_n = \mathrm{softmax}(\rho)_n$ at the saved 10,000–50,000-update checkpoints, together with $|\rho_n - \bar{\rho}|$ for each mode and the L_2 norm of the centered six-vector. Centering removes the common logit shift, which leaves the residue shares unchanged.

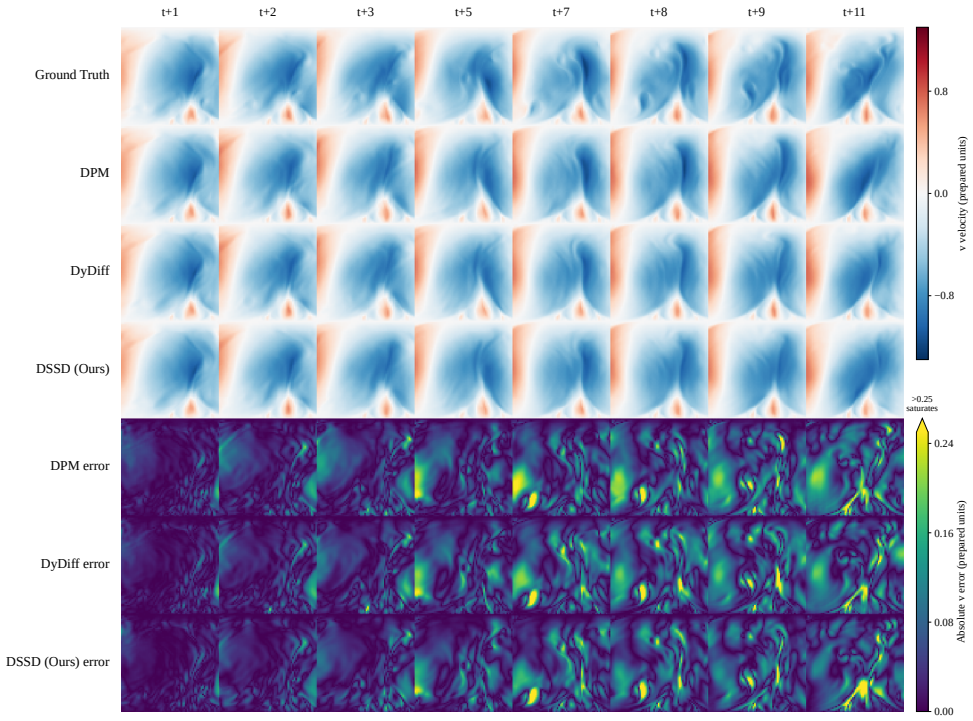
The trajectories differ across datasets. SEVIR's first-mode share increases from 0.255 to 0.664, BAIR's second-mode share from 0.329 to 0.744, and RoboNet's first-mode share from 0.234 to 0.667. Turbulence retains similar shares across the five checkpoints, and its residue-logit norm remains near 0.7. These observations explain the differing entropy trends in Figure 4; they measure residue reweighting, not independent mode energies. In particular, nearly fixed Turbulence residue shares do not imply fixed poles, as the pole-radius comparison in Section 5.5 illustrates. Mode indices are tracked within each run and do not identify matching physical modes across datasets.

D.4 Multivariate time-series forecasting

We evaluate multivariate forecasting on six datasets distributed through GluonTS (Alexandrov et al., 2020), following TimeGrad's dataset and prediction length setting (Salinas et al., 2019; Rasul et al., 2021). The datasets are Exchange, Solar, Electricity, Traffic, Taxi, and Wikipedia. To hold the representation and denoiser fixed across methods, DPM, DyDiff, and DSSD use the same temporal 1D dilated-convolution network: a pointwise channel projection followed by eight residual blocks with 128 hidden channels, which jointly processes observed values and noisy future values.

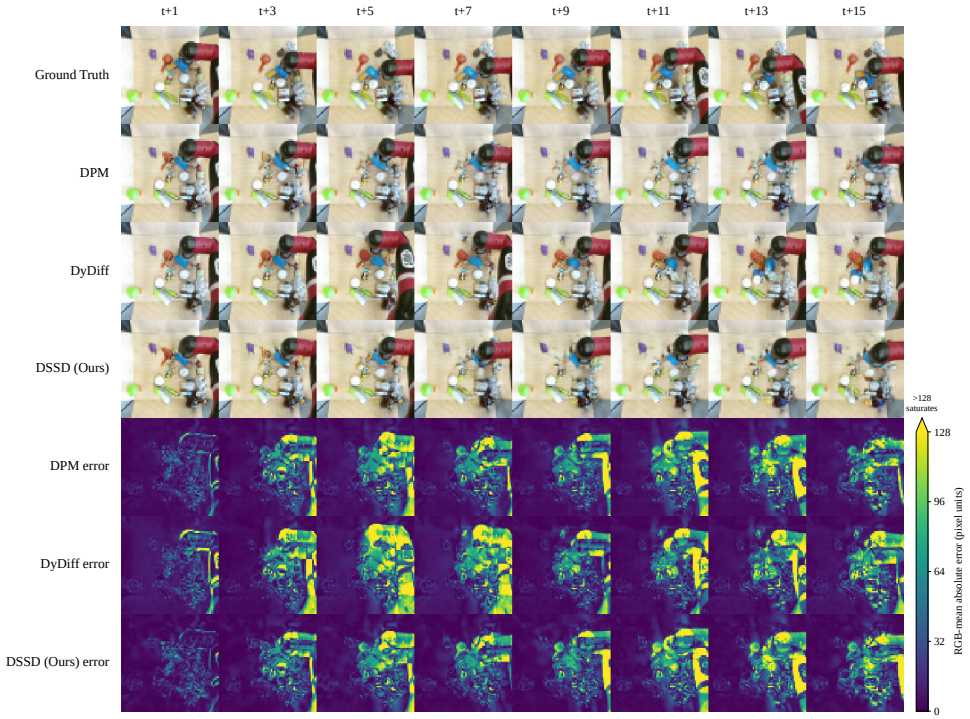


(a) SEVIR

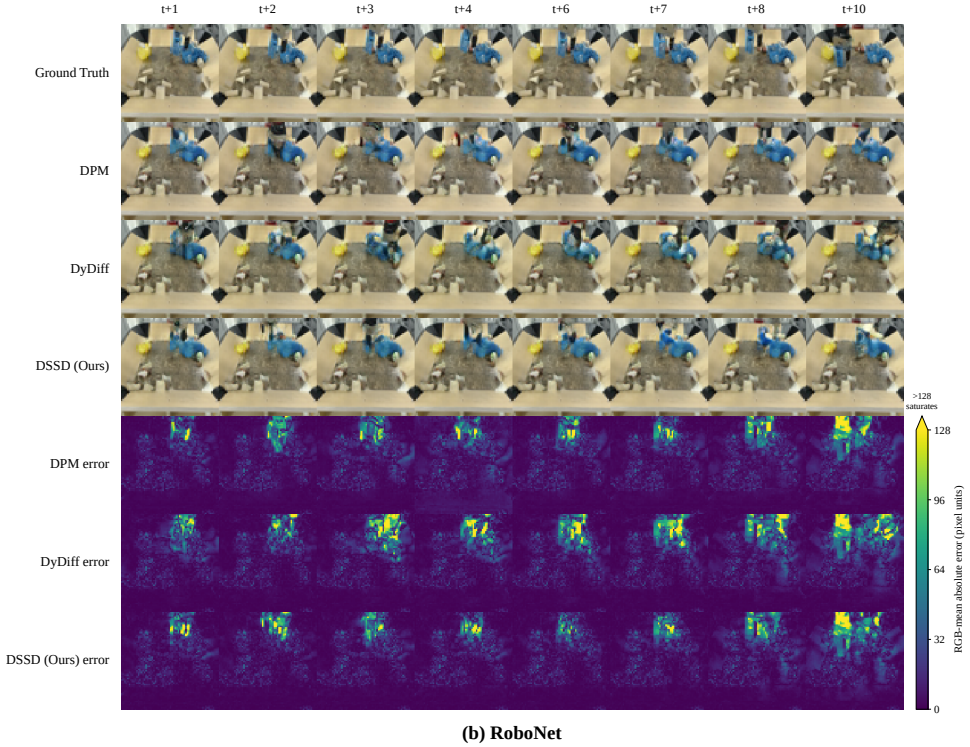


(b) Turbulence

Figure 5: Example scientific-field forecasts and absolute errors on (a) SEVIR and (b) Turbulence. Turbulence shows the v velocity component. Errors above the color-bar maximum saturate.



(a) BAIR



(b) RoboNet

Figure 6: Example robot-video forecasts and RGB-mean absolute errors on (a) BAIR and (b) RoboNet. Errors above the color-bar maximum saturate.

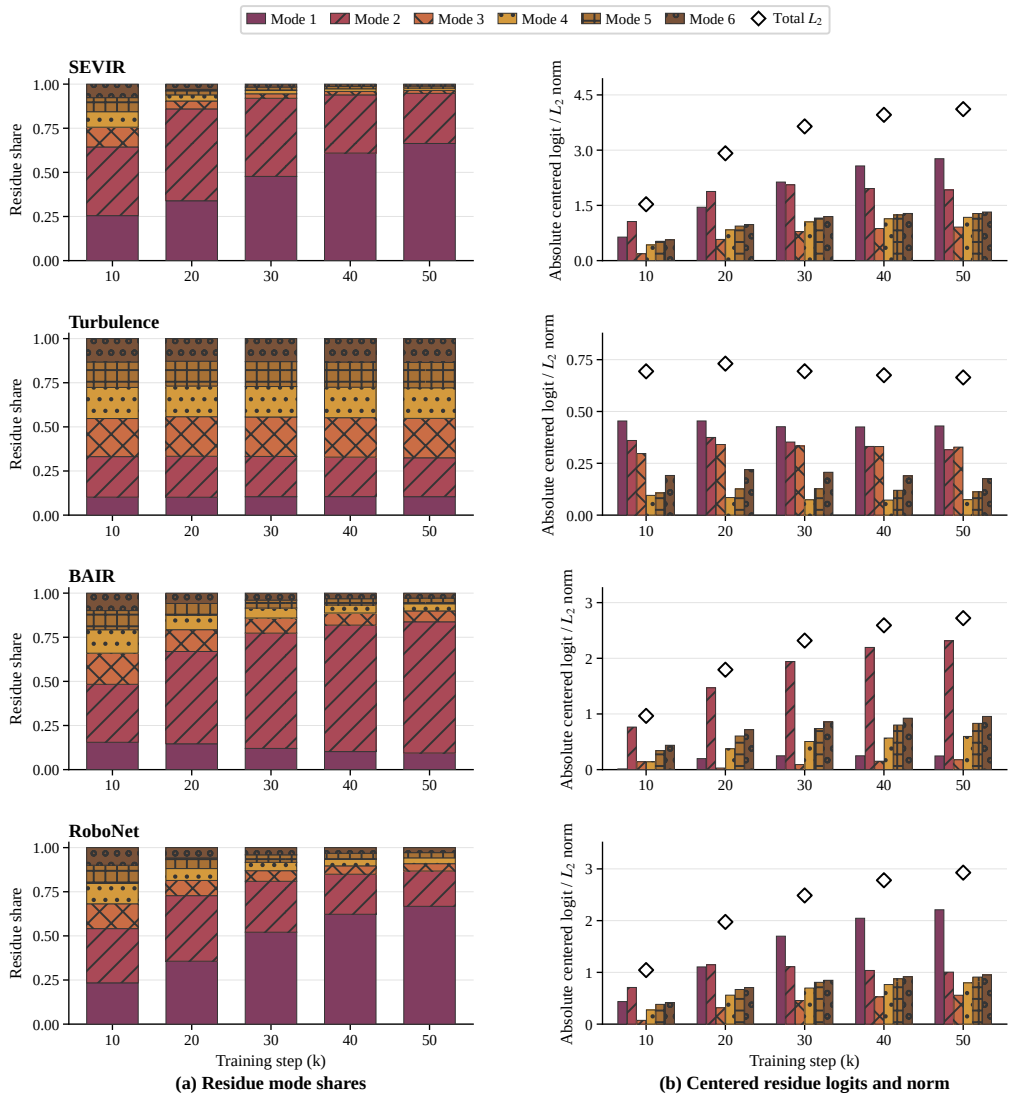


Figure 7: Mode-resolved residue diagnostics on the four datasets, measured every 10k training steps (10k–50k) from saved DSSD checkpoints. (a) Softmax residue shares of the six temporal modes, each corresponding to a conjugate pair. (b) Absolute centered residue logits of each mode (bars) and their total L_2 norm (diamonds).

Each dataset is one multivariate sequence with D channels. We compute a mean and standard deviation for each channel from the training sequence and apply the same normalization to test data. For each GluonTS test window, we forecast its future M points from the preceding N points. Table 9 gives these dimensions and window lengths.

All methods are trained from scratch for 50,000 optimizer updates with batch size 16, Adam at a constant learning rate of 10^{-5}, a 1,000-step cosine diffusion schedule, and seed 42. DyDiff uses the fixed EMA operator with $\eta = 0.5$. For DSSD, we use three conjugate SSM mode pairs (state size six), compared with six pairs in the video experiments. The time-series representation has no spatial latent feature maps, so we use fewer temporal modes as a modest capacity choice. All methods share the same denoiser. We use 50-step deterministic DDIM with a linearly spaced timestep grid.

Table 9 reports channel- and time-averaged finite-ensemble PWM CRPS, ensemble-mean RMSE and MAE, energy score, and temporal variogram score. Metric definitions follow Appendix C.3.

At 50,000 updates, DSSD has the lowest CRPS, RMSE, and energy score on five of six datasets, with DPM leading CRPS on Wikipedia and DyDiff leading its RMSE and energy score. The temporal

Table 9: Multivariate time-series test results at 50,000 updates. Bold and underlined values are best and second best within each dataset. All scores use training-set normalization; lower is better. D is the number of jointly modeled series; N and M are observed and predicted time steps.

Dataset	D	N	M	Method	CRPS ↓	RMSE ↓	MAE ↓	Energy ↓	Variogram ↓
Exchange	8	30	30	DPM	0.37573	0.68591	0.56123	0.59404	0.78559
				DyDiff	0.37640	0.76033	0.64038	0.54593	0.60261
				DSSD (Ours)	**0.29896**	**0.61315**	**0.51080**	**0.46610**	**0.44125**
Solar	137	24	24	DPM	1.43163	2.31897	**1.95040**	1.89547	**2.25310**
				DyDiff	1.48259	2.34964	1.97462	1.92896	2.37242
				DSSD (Ours)	**1.41653**	**2.31678**	1.95240	**1.88563**	2.26805
Electricity	370	24	24	DPM	7.12656	7.08183	5.72460	6.72375	**8.66073**
				DyDiff	7.17127	7.10774	5.74584	6.74831	8.67150
				DSSD (Ours)	**7.04436**	**7.05699**	**5.70272**	**6.69563**	8.87129
Traffic	963	24	24	DPM	7.12199	10.90963	9.67041	8.47188	6.88153
				DyDiff	7.14840	10.91133	**9.66698**	8.47718	**6.85649**
				DSSD (Ours)	**7.08377**	**10.90212**	9.66832	**8.46117**	7.08459
Taxi	1,214	24	24	DPM	4.40064	5.75505	4.91774	4.65397	3.57764
				DyDiff	4.41067	5.75627	4.91771	4.65638	**3.57039**
				DSSD (Ours)	**4.37568**	**5.75023**	**4.91415**	**4.64667**	3.68216
Wikipedia	9,535	30	30	DPM	**10.35604**	16.95278	11.20462	12.66044	**9.40709**
				DyDiff	10.41862	**16.70496**	**11.03162**	**12.54705**	9.47266
				DSSD (Ours)	10.41440	16.93813	11.19473	12.65975	9.52240

variogram score is mixed: DSSD is best on Exchange and second on Solar, but has the highest score of the three methods on Electricity, Traffic, Taxi, and Wikipedia.