VARIATIONAL STREAMING FLOW: PROBABILISTIC FORECASTING IN PHYSICAL TIME

Anonymous authors
Paper under double-blind review

ABSTRACT

Probabilistic forecasting is important for predicting complex dynamical systems because intrinsic randomness and incomplete observations can cause the same observed state to evolve into multiple plausible futures. While flow matching is a flexible approach for probabilistic forecasting, it is computationally expensive. Streaming flow (SF) reformulates this approach to model temporal evolution efficiently by learning a continuous velocity field directly in physical time. However, SF learns a deterministic velocity field. Thus, it provides only a single future trajectory for a given fixed initial state and observation history. To overcome this limitation, we introduce Variational Streaming Flow (VSF). Our approach learns a latent distribution that is conditioned on the dynamics of interest. In turn, this enables probabilistic forecasting. Importantly, we retain the computational efficiency of SF by generating in physical time. Across deterministic and stochastic dynamical systems, VSF demonstrates superior predictive accuracy and distributional fidelity. We demonstrate the advantage for both long-horizon rollouts exceeding 1,000 steps, and settings with bifurcating dynamics. Moreover, VSF can be integrated into existing Joint-Embedding Predictive Architecture (JEPA)-based world models as a plug-and-play predictor to improve temporal dynamics and goal-directed success rate in navigation, motion planning, and manipulation.

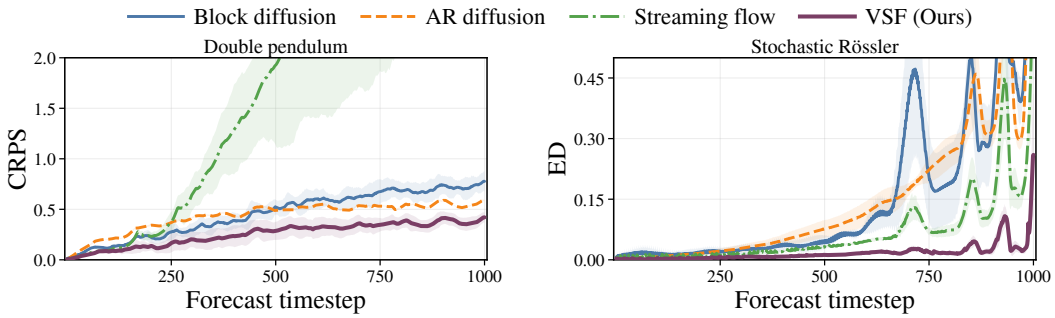

Figure 1: Our variational streaming flow approach obtains lower forecasting error over long rollouts than standard streaming flow. Left: continuous ranked probability score (CRPS) on the deterministic double pendulum. Right: energy distance (ED) on the partially observed Rössler system.

1 INTRODUCTION

Forecasting dynamical systems over long horizons remains challenging. Even when the dynamics are deterministic, small measurement or prediction errors can compound over a long rollout (Zhang & Gilpin, 2025). Predicting the entire future trajectory at once can avoid this recursive error accumulation, but this becomes an increasingly difficult learning task as the forecast horizon grows. Many real-world systems are also stochastic, which means that there can be many different future trajectories for the same sequence of observations (Pfister et al., 2026). For example, in weather forecasting (Feng & Fu, 2013; Sandu et al., 2019; Price et al., 2025), unresolved atmospheric processes can lead to different storm tracks from similar observations. In these settings, predicting just a single trajectory is insufficient for forecasting or planning. This motivates probabilistic forecasts that represent a distribution over possible futures (Chua et al., 2018; Porcher et al., 2026).

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.

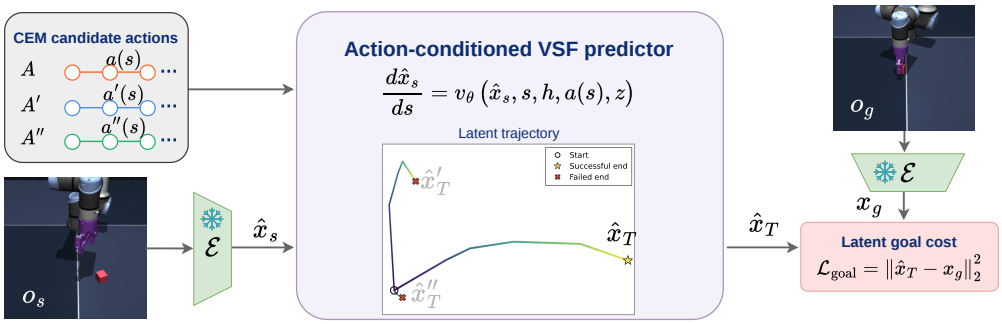

Figure 2: VSF can serve as a predictor in a JEPA-based world model. We replace the deterministic autoregressive predictor with an action-conditioned VSF predictor while keeping the JEPA encoder and CEM planner fixed. The encoder maps the current and goal observations into latent space. For each action sequence, VSF samples multiple latents and rolls out future trajectories. CEM compares the predicted terminal states with the goal representation and uses these costs to select a plan.

Flow matching provides a flexible way to learn probabilistic forecasts over entire future trajectories (Lipman et al., 2024; Kollovieh et al., 2025); see App. A for related work. However, generating a forecast requires integrating a learned vector field along an auxiliary flow time that is separate from the physical evolution of the system. Streaming and physical-time formulations instead evolve the predicted state directly through physical time, making long-horizon forecasting more efficient (Jiang et al., 2025; Liu et al., 2026). However, a standard non-latent physical-time model with a deterministic velocity field produces only one future from a fixed observed state.

To address this limitation, we introduce *Variational Streaming Flow* (VSF), a probabilistic extension of physical-time flow forecasting. VSF augments the physical-time velocity field with a latent variable that is sampled once for each forecast and kept fixed throughout the rollout. This preserves the computational efficiency of streaming flow (SF) while improving long-horizon forecasting accuracy, as shown in Fig. 1. Different latent samples can also generate different future trajectories from the same observed state, allowing VSF to represent multiple possible futures. During training, we use an encoder to infer the latent variable from the observed future transition and regularize its distribution toward a fixed prior (Guo & Schwing, 2025; Eijkelboom et al., 2024). At inference, we only need to sample a latent from the prior and keep it fixed throughout the rollout.

We evaluate VSF on deterministic and partially observed stochastic dynamical systems, including chaotic systems and settings with branching futures, testing both predictive accuracy and distributional fidelity. On deterministic systems, VSF improves long-horizon predictive accuracy over DeepAR (Salinas et al., 2020), diffusion- and flow-based forecasters, SF (Jiang et al., 2025), and Chronos-2 (Ansari et al., 2025). On stochastic Rössler and Lorenz systems, VSF more accurately recovers the conditional distribution of long-horizon futures. We further use VSF as the predictor in a JEPA-based world model (Maes et al., 2026); see Fig. 2. Across four robotics tasks, VSF improves average goal-directed planning success by 7.34% over LeWM's deterministic predictor while retaining one velocity-field evaluation per imagined transition.

Our main contributions are as follows.

- We introduce *Variational Streaming Flow*, a variational latent variable formulation of physical-time flow modeling in which one latent draw conditions a full forecast rollout. It is motivated by a limitation of non-latent physical-time flows near bifurcations, where small differences in state can make the learned dynamics highly sensitive to perturbations (see App. B).
- We develop a variational learning framework for latent-conditioned physical-time velocities (see section 3). We derive a conditional-velocity ELBO and introduce an energy-based reconstruction over multiple latent samples. Under posterior–prior alignment, we show that the resulting reconstruction controls the energy distance between predicted and true velocity distributions.
- We evaluate VSF on deterministic and stochastic forecasting and in an action-conditioned world model. VSF improves long-horizon predictive accuracy and distributional fidelity, while its predictions improve goal-directed planning and retain the efficiency of physical-time forecasting.

2 Problem Formulation and Preliminaries

This section introduces the forecasting problem and reviews the physical-time formulation of SF.

Forecasting problem. We consider forecasting from an observed history $\boldsymbol{x}_{a:m} = (\boldsymbol{x}_a, \ldots, \boldsymbol{x}_m)$, where $\boldsymbol{x}_k \in \mathcal{X} \subseteq \mathbb{R}^d$ denotes the system state at the kth observation time. Our goal is to learn a conditional distribution over future trajectories,

$$\widehat{\boldsymbol{x}}_{m+1:T} \sim Q_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{x}_{a:m}), \tag{2.1}$$

where $Q_{\boldsymbol{\theta}}$ is the learned forecasting model. When the dynamics are fully observed and deterministic, this distribution contains a single future. With partial observations or stochastic dynamics, the same observed history may instead be consistent with different future trajectories.

Streaming Flow in physical time. Our starting point is SF (Jiang et al., 2025). SF predicts the future by starting from the most recent observation $\boldsymbol{x}_m$ and learning how the system state should move forward in physical time. The state produced during the ODE rollout is therefore the forecast itself: as the integration time advances, the predicted state advances along the future trajectory.

Let $\boldsymbol{h} = \boldsymbol{x}_{a:m}$ denote the observed history and let $s \in [0, 1]$ denote normalized physical forecast time. Conditioned on $\boldsymbol{h}$, we write the future trajectory as $\mathbf{X}^{\mathrm{f}} = (\mathbf{X}^{\mathrm{f}}_s)_{s\in[0,1]}$, with $\mathbf{X}^{\mathrm{f}}_0 = \boldsymbol{x}_m$. During training, we observe a realization $\boldsymbol{\xi} : [0, 1] \to \mathcal{X}$ of this future trajectory. At any time s, the trajectory gives us its state $\boldsymbol{\xi}(s)$ and physical velocity $\dot{\boldsymbol{\xi}}(s)$. SF uses these quantities to construct a velocity target for the observed trajectory and nearby states:

$$\mathbf{g}_{\boldsymbol{\xi}}(\boldsymbol{x}, s) = \dot{\boldsymbol{\xi}}(s) - \kappa\big(\boldsymbol{x} - \boldsymbol{\xi}(s)\big), \qquad \kappa > 0. \tag{2.2}$$

The second term pulls nearby states back toward the trajectory, creating a contracting tube around it. On the trajectory itself, this correction vanishes, so the target reduces to $\dot{\boldsymbol{\xi}}(s)$.

This off-trajectory behavior is what gives SF its flow matching interpretation. Rather than specifying only the velocity along the observed trajectory, the target field also determines how nearby states move. To see this, consider a small cloud of initial states around the beginning of the trajectory, $\mathbf{Y}_0 \sim \mathcal{N}(\boldsymbol{\xi}(0), \sigma_0^2 \mathbf{I}_d)$, and evolve these states according to $\dot{\mathbf{Y}}_s = \mathbf{g}_{\boldsymbol{\xi}}(\mathbf{Y}_s, s)$. The center of the cloud follows the observed future $\boldsymbol{\xi}(s)$, while deviations from that center are contracted by the correction term. This evolution has the closed-form solution $\mathbf{Y}_s = \boldsymbol{\xi}(s) + e^{-\kappa s}\big(\mathbf{Y}_0 - \boldsymbol{\xi}(0)\big)$. The displacement from the observed trajectory therefore decays exponentially at rate κ. Since the initial displacement is Gaussian, the state at physical time s remains Gaussian

$$\mathbf{Y}_s \mid \boldsymbol{\xi} \sim \mathcal{N}\left(\boldsymbol{\xi}(s), \sigma_0^2 e^{-2\kappa s} \mathbf{I}_d\right). \tag{2.3}$$

Thus, each observed future trajectory defines a probability path around the trajectory itself: a Gaussian tube whose mean follows $\boldsymbol{\xi}(s)$ and whose width contracts over physical time. The field $\mathbf{g}_{\boldsymbol{\xi}}$ is the velocity field associated with this trajectory-conditioned path.

SF then learns a single velocity field $\boldsymbol{v}_{\boldsymbol{\theta}}(\boldsymbol{x}, s, \boldsymbol{h})$, represented by a neural network, from many such future trajectories. Given the current state $\boldsymbol{x}$, physical time s, and observed history $\boldsymbol{h}$, the network predicts how the state should move next. Training matches this prediction to the velocity target associated with the observed future. At the population level

$$\mathcal{L}_{\mathrm{SF}}(\boldsymbol{\theta}) = \mathbb{E}\left[\left\|\boldsymbol{v}_{\boldsymbol{\theta}}(\mathbf{Y}_s, s, \boldsymbol{h}) - \mathbf{g}_{\mathbf{X}^{\mathrm{f}}}(\mathbf{Y}_s, s)\right\|_2^2\right], \tag{2.4}$$

where $\mathbf{Y}_s$ denotes a state in the contracting tube around the sampled future trajectory. In practice, training can be performed directly on the observed trajectory. In that case, $\mathbf{Y}_s = \boldsymbol{\xi}(s)$ and the target reduces to its physical-time velocity $\dot{\boldsymbol{\xi}}(s)$.

Discrete observations. The above formulation is continuous in time. In practice, however, we only observe the system state at discrete time points. Consider a training window $\boldsymbol{x}_{a:T}$ with forecast origin m, and let $P = T - m$ be the number of future transitions. We normalize the observation times as $s_k = k/P$. Between two consecutive observations, we use the piecewise-linear path

$$\boldsymbol{\xi}^{\Delta}(s) = (1 - \alpha_s)\boldsymbol{x}_{m+k} + \alpha_s \boldsymbol{x}_{m+k+1}, \qquad \alpha_s = P(s - s_k), \tag{2.5}$$

for $s \in [s_k, s_{k+1})$. Its physical-time velocity is constant within each interval,

$$\dot{\boldsymbol{\xi}}^{\Delta}(s) = P\big(\boldsymbol{x}_{m+k+1} - \boldsymbol{x}_{m+k}\big) =: \boldsymbol{u}_k. \tag{2.6}$$

Thus, a discrete transition provides both a local state along the interpolated trajectory and its physical-time velocity. We use these quantities to train VSF in section 3.

3 VARIATIONAL STREAMING FLOW

Standard non-latent SF assigns a single physical-time velocity to a given state and observed history, and therefore produces a single future from a fixed initial condition. This is sufficient when the future is unique, but not when the same observed history is compatible with different trajectories. Figure 3 illustrates this setting. The difficulty becomes clear near a bifurcation. Different future trajectories can *share the same physical trajectory* before separating, so SF (Jiang et al., 2025) must encode the information needed to select a future branch through small differences in the physical state. As these differences shrink, the velocity field must distinguish similar states, which can make the rollout more sensitive to perturbations. App. B studies this fidelity–sensitivity tradeoff in detail.

VSF addresses this limitation by allowing the physical-time velocity field to depend on an additional latent variable. Let $\boldsymbol{h} = \boldsymbol{x}_{a:m}$ denote the observed history and let $\boldsymbol{z} \in \mathbb{R}^{d_z}$ denote the latent variable. We condition the velocity field on $\boldsymbol{z}$ and define the forecast dynamics as

$$\frac{d\widehat{\boldsymbol{x}}_s}{ds} = \boldsymbol{v}_{\boldsymbol{\theta}}\left(\widehat{\boldsymbol{x}}_s, s, \boldsymbol{h}, \boldsymbol{z}\right). \tag{3.1}$$

Here, $s \in [0, 1]$ is normalized physical forecast time and $\widehat{\boldsymbol{x}}_0 = \boldsymbol{x}_m$. For a fixed $\boldsymbol{z}$, the velocity field defines one physical-time trajectory starting from the most recent observation. Changing $\boldsymbol{z}$ changes the velocity field along the rollout and can therefore produce a different future from the same observed history. We call this model *Variational Streaming Flow* (VSF).

3.1 TRAINING AND INFERENCE

VSF uses the latent variable differently during training and inference. During training, the future trajectory is observed, so we can use its local motion to infer which latent values explain the dynamics. At inference, the future is unknown, so the latent must instead be sampled before the rollout. We use the fixed prior

$$p(\boldsymbol{z}) = \mathcal{N}\left(\mathbf{0}, \mathbf{I}_{d_z}\right). \tag{3.2}$$

We first describe training. As in section 2, we sample a physical time s from an observed future trajectory and construct the local state $\boldsymbol{x}_s$ together with its physical-time velocity $\boldsymbol{u}_s$. The pair $(\boldsymbol{x}_s, \boldsymbol{u}_s)$ tells us both where the trajectory is and how it is moving at that time. We use an encoder, parameterized by $\boldsymbol{\phi}$, to map this local future information together with s and the observed history $\boldsymbol{h}$ to the parameters of a Gaussian distribution over $\boldsymbol{z}$. Specifically, the encoder outputs a mean $\boldsymbol{\mu}_{\boldsymbol{\phi}}(\boldsymbol{u}_s, \boldsymbol{x}_s, s, \boldsymbol{h}) \in \mathbb{R}^{d_z}$ and a vector of positive standard deviations $\boldsymbol{\sigma}_{\boldsymbol{\phi}}(\boldsymbol{u}_s, \boldsymbol{x}_s, s, \boldsymbol{h}) \in \mathbb{R}^{d_z}_{>0}$. These outputs define the variational posterior

$$q_{\boldsymbol{\phi}}\left(\boldsymbol{z} \mid \boldsymbol{u}_s, \boldsymbol{x}_s, s, \boldsymbol{h}\right) = \mathcal{N}\left(\boldsymbol{\mu}_{\boldsymbol{\phi}}(\boldsymbol{u}_s, \boldsymbol{x}_s, s, \boldsymbol{h}), \operatorname{diag}\left[\boldsymbol{\sigma}^2_{\boldsymbol{\phi}}(\boldsymbol{u}_s, \boldsymbol{x}_s, s, \boldsymbol{h})\right]\right). \tag{3.3}$$

We draw M latent samples from this posterior. Each sample is passed to the velocity field together with the same local state, time, and history, producing $\widehat{\boldsymbol{u}}_s^{(i)} = \boldsymbol{v}_{\boldsymbol{\theta}}(\boldsymbol{x}_s, s, \boldsymbol{h}, \boldsymbol{z}^{(i)})$. These predictions represent the physical-time velocities associated with different latent values at the observed transition. Training encourages this set of predicted velocities to match the observed local dynamics through the distributional reconstruction loss $\mathcal{L}_{\mathrm{ES}}$. A KL term simultaneously keeps the inferred

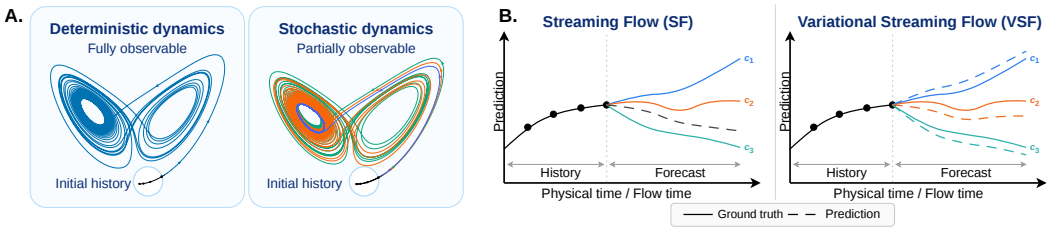

Figure 3: Deterministic and stochastic systems pose different forecasting problems. (A) When deterministic dynamics are fully observed, the future is unique. With partial observations and stochastic dynamics, the same observed history can lead to different future trajectories. (B) SF produces a single forecast from the observed state, while VSF can generate multiple plausible futures.

posterior close to the prior in Eq. (3.2), so that samples from this prior can be used when the future is no longer available. We derive both terms in the following subsections.

At inference, $\boldsymbol{u}_s$ is unknown. Thus, we sample one latent $\boldsymbol{z} \sim p(\boldsymbol{z})$ at the beginning of the forecast and use it throughout the rollout. Starting from $\widehat{\boldsymbol{x}}_0 = \boldsymbol{x}_m$, we discretize the physical-time dynamics in Eq. (3.1). With a forward-Euler step of size Δt, the update is

$$\widehat{\boldsymbol{x}}_{k+1} = \widehat{\boldsymbol{x}}_k + \Delta t\, \boldsymbol{v}_{\boldsymbol{\theta}}\left(\widehat{\boldsymbol{x}}_k, s_k, \boldsymbol{h}, \boldsymbol{z}\right), \tag{3.4}$$

where s_k is the normalized physical time at the current forecast step. The generated state $\widehat{\boldsymbol{x}}_{k+1}$ is then added to the rolling history used by the predictor. When the history representation $\boldsymbol{h}$ is refreshed from these generated states, the latent $\boldsymbol{z}$ remains unchanged. Thus, the history evolves with the generated trajectory while one latent draw continues to condition the complete forecast.

The two procedures are summarized below. The complete algorithms, including batching, history-window updates, and segment boundaries, are given in App. E.1.

Training

Construct a local state and velocity $(\boldsymbol{x}_s, \boldsymbol{u}_s)$.

$$\boldsymbol{z}^{(1:M)} \overset{\text{iid}}{\sim} q_{\boldsymbol{\phi}}\big(\cdot \mid \boldsymbol{u}_s, \boldsymbol{x}_s, s, \boldsymbol{h}\big)$$

Evaluate the physical-time velocities

$$\widehat{\boldsymbol{u}}_s^{(1:M)} = \boldsymbol{v}_{\boldsymbol{\theta}}\big(\boldsymbol{x}_s, s, \boldsymbol{h}, \boldsymbol{z}^{(1:M)}\big)$$

Train $\boldsymbol{v}_{\boldsymbol{\theta}}$ and the encoder using $\mathcal{L}_{\text{ES}}$ together with KL regularization toward $p(\boldsymbol{z})$.

Inference

Sample one latent from the prior

$$\boldsymbol{z} \sim p(\boldsymbol{z})$$

and keep it fixed for the complete forecast. Advance the state in physical time

$$\widehat{\boldsymbol{x}}_{k+1} = \widehat{\boldsymbol{x}}_k + \Delta s\, \boldsymbol{v}_{\boldsymbol{\theta}}\big(\widehat{\boldsymbol{x}}_k, s_k, \boldsymbol{h}, \boldsymbol{z}\big)$$

Add generated states to the rolling history and refresh $\boldsymbol{h}$ while retaining the same $\boldsymbol{z}$.

3.2 Variational Training Objective

We now derive a variational objective for the training procedure above. At a sampled physical time s, the model observes the local state $\boldsymbol{x}_s$, the forecast time s, and the history $\boldsymbol{h}$, and predicts the corresponding physical-time velocity $\boldsymbol{u}_s$. To keep the notation compact, define the local context

$$\boldsymbol{c}_s := (\boldsymbol{x}_s, s, \boldsymbol{h}). \tag{3.5}$$

The latent variable $\boldsymbol{z}$ provides additional information about which local dynamics are being realized.

We begin with an auxiliary probabilistic model for the local velocity. Given the context $\boldsymbol{c}_s$ and latent $\boldsymbol{z}$, we place an isotropic Gaussian around the velocity predicted by the VSF field

$$p_{\boldsymbol{\theta}}\big(\boldsymbol{u}_s \mid \boldsymbol{c}_s, \boldsymbol{z}\big) = \mathcal{N}\left(\boldsymbol{u}_s; \boldsymbol{v}_{\boldsymbol{\theta}}\big(\boldsymbol{x}_s, s, \boldsymbol{h}, \boldsymbol{z}\big), \sigma_u^2 \mathbf{I}_d\right), \qquad \sigma_u > 0. \tag{3.6}$$

Here, σ_u^2 controls the variance of the Gaussian likelihood and $\mathbf{I}_d$ is the identity matrix in the velocity dimension. This likelihood is used only to formulate learning at a sampled physical time. It models the local velocity $\boldsymbol{u}_s$ and should not be interpreted as a Gaussian model for the full forecast trajectory generated by Eq. (3.1). Because $\boldsymbol{z}$ is latent, the conditional velocity distribution is obtained by integrating over the prior

$$p_{\boldsymbol{\theta}}\big(\boldsymbol{u}_s \mid \boldsymbol{c}_s\big) = \int p_{\boldsymbol{\theta}}\big(\boldsymbol{u}_s \mid \boldsymbol{c}_s, \boldsymbol{z}\big) p(\boldsymbol{z})\, d\boldsymbol{z}. \tag{3.7}$$

Directly optimizing this marginal likelihood requires integrating over the latent variable. During training, we instead use the encoder introduced in section 3.1 to approximate the posterior over $\boldsymbol{z}$ from the observed local dynamics. Applying the standard variational bound gives

$$\log p_{\boldsymbol{\theta}}\big(\boldsymbol{u}_s \mid \boldsymbol{c}_s\big) \geq \mathbb{E}_{\mathbf{Z} \sim q_{\boldsymbol{\phi}}(\cdot \mid \boldsymbol{u}_s, \boldsymbol{c}_s)}\left[\log p_{\boldsymbol{\theta}}\big(\boldsymbol{u}_s \mid \boldsymbol{c}_s, \mathbf{Z}\big)\right] - \mathrm{KL}\left(q_{\boldsymbol{\phi}}\big(\boldsymbol{z} \mid \boldsymbol{u}_s, \boldsymbol{c}_s\big) \,\|\, p(\boldsymbol{z})\right).$$

The two terms have the same roles as in the training procedure described above. The first rewards latent samples for which the velocity field explains the observed physical-time velocity. The second keeps the encoder distribution close to the prior that will be used once the future velocity is unavailable at inference. For the Gaussian likelihood in Eq. (3.6), the negative expected log-likelihood has

a simple form. Up to an additive constant

$$-\mathbb{E}_{\mathbf{Z}\sim q_{\boldsymbol{\phi}}(\cdot|\boldsymbol{u}_s,\boldsymbol{c}_s)}\left[\log p_{\boldsymbol{\theta}}\big(\boldsymbol{u}_s \mid \boldsymbol{c}_s,\mathbf{Z}\big)\right] = \frac{1}{2\sigma_u^2}\mathbb{E}_{\mathbf{Z}\sim q_{\boldsymbol{\phi}}(\cdot|\boldsymbol{u}_s,\boldsymbol{c}_s)}\left[\left\|\boldsymbol{v}_{\boldsymbol{\theta}}\big(\boldsymbol{x}_s,s,\boldsymbol{h},\mathbf{Z}\big)-\boldsymbol{u}_s\right\|_2^2\right] + \text{const.}$$

Thus, the local ELBO leads to a latent-conditioned MSE reconstruction together with KL regularization toward the prior. This provides a natural variational starting point for VSF.

3.3 Distributional Velocity Reconstruction

The variational objective above leads naturally to an MSE reconstruction of the local physical-time velocity. This is appropriate when the latent variable fully separates the different future dynamics. In practice, however, different future continuations can still remain compatible with the same context and latent value. To make this precise, let $(\mathbf{C},\mathbf{U})$ denote the random local context and velocity obtained by sampling a training trajectory and a physical time, and let

$$\mathbf{Z}\sim q_{\boldsymbol{\phi}}\big(\cdot \mid \mathbf{U},\mathbf{C}\big).$$

For a fixed encoder $q_{\boldsymbol{\phi}}$, the population minimizer of the MSE reconstruction is

$$\boldsymbol{v}^*(\boldsymbol{c},\boldsymbol{z}) = \mathbb{E}\left[\mathbf{U} \mid \mathbf{C}=\boldsymbol{c},\mathbf{Z}=\boldsymbol{z}\right]. \tag{3.8}$$

Thus, the learned velocity is the conditional mean of all local velocities that remain compatible with the same $(\boldsymbol{c},\boldsymbol{z})$. If the encoder assigns distinct future continuations to well-separated latent values, this conditional distribution can be narrow. If different continuations still overlap in latent space, their remaining velocity variation is averaged. Such overlap can arise from imperfect inference, overlapping posterior distributions, or posterior collapse.

We therefore train the velocity field using several latent samples jointly. For the observed local transition $(\boldsymbol{x}_s,\boldsymbol{u}_s)$, draw $\boldsymbol{z}^{(1)},\dots,\boldsymbol{z}^{(M)}$ independently from the encoder and evaluate

$$\widehat{\boldsymbol{u}}_s^{(i)} = \boldsymbol{v}_{\boldsymbol{\theta}}\left(\boldsymbol{x}_s,s,\boldsymbol{h},\boldsymbol{z}^{(i)}\right), \qquad \boldsymbol{z}^{(i)} \overset{\text{iid}}{\sim} q_{\boldsymbol{\phi}}\big(\cdot \mid \boldsymbol{u}_s,\boldsymbol{x}_s,s,\boldsymbol{h}\big). \tag{3.9}$$

These samples give a set of latent-conditioned velocity predictions for the same observed transition. Rather than matching each prediction independently with MSE, we compare this set with the observed velocity using a generalized energy reconstruction,

$$\mathcal{L}_{\mathrm{ES}}(s) = \frac{1}{M}\sum_{i=1}^{M}\left\|\widehat{\boldsymbol{u}}_s^{(i)}-\boldsymbol{u}_s\right\|_2^{\alpha} - \frac{1}{2M^2}\sum_{i,j=1}^{M}\left\|\widehat{\boldsymbol{u}}_s^{(i)}-\widehat{\boldsymbol{u}}_s^{(j)}\right\|_2^{\alpha}, \qquad 0<\alpha<2. \tag{3.10}$$

The first term keeps the predicted velocities close to the observed physical velocity. The second term accounts for the spread among the predictions, so the loss does not favor collapsing all latent samples to the same velocity. For $\alpha=1$, Eq. (3.10) has the standard energy-score form and reduces to the CRPS in one dimension (Gneiting & Raftery, 2007; Lang et al., 2026).

This reconstruction also connects the posterior samples used during training to the prior samples used at inference. Under aggregate posterior–prior alignment, App. D shows that the expected population energy reconstruction upper-bounds one half of the energy distance between the prior-predictive velocity distribution and the true conditional velocity distribution. Combining the energy reconstruction with the KL regularization gives the objective that we use to train VSF

$$\mathcal{L}_{\mathrm{VSF}} = \mathbb{E}\left[\mathcal{L}_{\mathrm{ES}}(s) + \beta_{\mathrm{KL}}\,\mathrm{KL}\left(q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{u}_s,\boldsymbol{x}_s,s,\boldsymbol{h}) \,\|\, p(\boldsymbol{z})\right)\right]. \tag{3.11}$$

The expectation is over training trajectories and sampled physical times. Replacing the Gaussian ELBO reconstruction with Eq. (3.10) means that Eq. (3.11) is no longer an ELBO, but a KL-regularized distributional reconstruction under the same local latent model.

4 Experimental Results

In this section, we evaluate VSF on fully observed deterministic systems (Section 4.1), partially observed stochastic systems (Section 4.2), and as a predictor for model-based planning (Section 4.3). Additional experimental details and ablations are provided in App. G–M.

Table 1: VSF achieves the lowest forecasting error across all deterministic and stochastic systems over a 1,000-step rollout. We report CRPS, RMSE, and MAE for deterministic systems and CME, CSE, W_1, and ED for stochastic systems. Runtime (RT), in seconds, is averaged across tasks.

Deterministic dynamics										
	Double pendulum			Rössler			Lorenz			
Model	CRPS ↓	RMSE ↓	MAE ↓	CRPS ↓	RMSE ↓	MAE ↓	CRPS ↓	RMSE ↓	MAE ↓	RT
Block diffusion	0.46 ± 0.07	0.91 ± 0.14	0.47 ± 0.07	0.29 ± 0.13	0.54 ± 0.26	0.29 ± 0.13	0.31 ± 0.12	0.42 ± 0.18	0.33 ± 0.12	30.2
AR diffusion	0.43 ± 0.00	0.96 ± 0.01	0.59 ± 0.01	0.35 ± 0.06	1.00 ± 0.13	0.53 ± 0.08	0.86 ± 0.04	1.96 ± 0.06	1.50 ± 0.06	78.2
Rectified flow	0.53 ± 0.07	1.08 ± 0.11	0.67 ± 0.07	0.40 ± 0.07	0.85 ± 0.20	0.48 ± 0.08	1.46 ± 0.89	2.69 ± 1.46	1.94 ± 1.14	41.1
DeepAR	0.58 ± 0.00	1.07 ± 0.00	0.68 ± 0.00	0.17 ± 0.04	0.54 ± 0.05	0.21 ± 0.05	10.30 ± 0.22	12.42 ± 0.13	10.88 ± 0.13	3.9
SF	2.12 ± 0.87	9.18 ± 5.17	2.12 ± 0.87	0.16 ± 0.07	0.34 ± 0.18	0.16 ± 0.07	1.42 ± 0.14	2.75 ± 0.31	1.42 ± 0.14	6.4
SF (perturb.)	1.83 ± 0.86	8.38 ± 5.18	1.96 ± 0.89	0.11 ± 0.05	0.34 ± 0.18	0.16 ± 0.07	1.39 ± 0.14	2.75 ± 0.31	1.42 ± 0.14	6.4
Chronos-2	0.78 ± 0.00	1.50 ± 0.02	0.96 ± 0.01	1.98 ± 0.00	4.11 ± 0.02	2.62 ± 0.01	13.50 ± 0.06	28.19 ± 1.14	25.22 ± 0.85	0.8
Chronos-2 (calib.)	0.62 ± 0.00	1.16 ± 0.00	0.74 ± 0.00	1.91 ± 0.00	3.91 ± 0.02	2.52 ± 0.01	2.39 ± 0.02	6.01 ± 0.27	4.54 ± 0.27	0.4
VSF (Ours)	$\mathbf{0.25 \pm 0.07}$	$\mathbf{0.71 \pm 0.11}$	$\mathbf{0.33 \pm 0.08}$	$\mathbf{0.03 \pm 0.00}$	$\mathbf{0.13 \pm 0.01}$	$\mathbf{0.05 \pm 0.00}$	$\mathbf{0.16 \pm 0.02}$	$\mathbf{0.25 \pm 0.02}$	$\mathbf{0.19 \pm 0.01}$	7.2

Stochastic dynamics									
	Rössler				Lorenz				
Model	CME ↓	CSE ↓	W_1 ↓	ED ↓	CME ↓	CSE ↓	W_1 ↓	ED ↓	RT
Block diffusion	0.16 ± 0.02	0.72 ± 0.08	8.88 ± 1.05	12.67 ± 1.71	0.20 ± 0.01	0.93 ± 0.01	19.88 ± 0.28	13.73 ± 0.45	21.8
AR diffusion	0.20 ± 0.03	4.63 ± 0.40	16.21 ± 1.12	11.46 ± 2.06	0.22 ± 0.05	0.95 ± 0.14	25.64 ± 3.10	5.60 ± 1.54	42.4
Rectified flow	0.22 ± 0.09	0.89 ± 0.30	12.60 ± 4.44	16.72 ± 8.72	0.26 ± 0.03	0.60 ± 0.12	19.74 ± 1.63	13.24 ± 2.69	19.4
DeepAR	0.20 ± 0.02	0.61 ± 0.02	11.81 ± 0.80	15.62 ± 1.57	2.14 ± 0.49	2.61 ± 0.27	131.84 ± 15.99	138.30 ± 45.71	1.2
SF	0.06 ± 0.02	1.00 ± 0.00	4.53 ± 0.67	5.00 ± 1.34	1.04 ± 0.56	1.00 ± 0.00	58.12 ± 24.23	89.84 ± 48.45	7.5
SF (perturb.)	0.06 ± 0.02	1.12 ± 0.23	4.92 ± 0.84	2.28 ± 0.82	0.68 ± 0.33	1.33 ± 0.75	53.62 ± 25.69	35.78 ± 19.55	6.6
Chronos-2	1.28 ± 0.06	47.87 ± 1.63	90.54 ± 3.86	94.84 ± 0.79	6.84 ± 0.17	37.49 ± 1.66	411.32 ± 9.43	342.98 ± 1.87	0.3
Chronos-2 (calib.)	1.20 ± 0.06	61.83 ± 2.09	101.69 ± 4.61	93.23 ± 0.20	1.44 ± 0.06	10.53 ± 0.49	111.81 ± 3.11	64.97 ± 0.79	0.1
VSF (Ours)	$\mathbf{0.04 \pm 0.01}$	$\mathbf{0.36 \pm 0.14}$	$\mathbf{2.62 \pm 0.38}$	$\mathbf{1.28 \pm 0.42}$	$\mathbf{0.19 \pm 0.03}$	$\mathbf{0.50 \pm 0.21}$	$\mathbf{15.95 \pm 5.64}$	$\mathbf{5.19 \pm 0.93}$	6.7

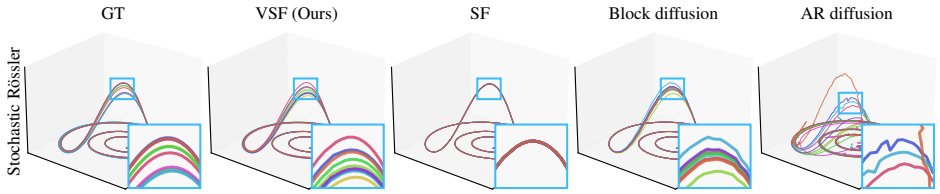

Figure 4: VSF preserves the Rössler geometry while capturing variability across plausible futures. We show 10 trajectories from each method. Block diffusion produces less diverse forecasts, while AR diffusion is more prone to rollout drift.

Baselines. For the dynamical-system forecasting experiments, we compare VSF with eight baselines: (1) Block diffusion, a conditional diffusion model that generates the forecast horizon jointly; (2) AR diffusion, an autoregressive diffusion model that generates one future state at a time; (3) Rectified flow, which autoregressively generates one future state at a time by integrating a rectified-flow velocity field (Liu et al., 2022); (4) DeepAR (Salinas et al., 2020), an autoregressive recurrent probabilistic forecaster; (5) SF (Jiang et al., 2025), a deterministic physical-time flow; (6) SF (perturbation), which introduces forecast diversity through Gaussian perturbations of the initial state; (7) Chronos-2 (Ansari et al., 2025), a pretrained universal time-series forecaster; and (8) Chronos-2 (calibrated), which applies post-hoc calibration to the Chronos-2 predictions.

4.1 Deterministic Forecasting

We evaluate fully observed deterministic forecasting on the double pendulum, Rössler, and Lorenz systems using 1,000-step rollouts. We report CRPS for the forecast ensemble and RMSE and MAE for its mean; detailed definitions are provided in App. G.

VSF produces accurate forecasts with samples concentrated around the ground-truth trajectory. As shown in Table 1, VSF achieves the lowest CRPS, RMSE, and MAE across all three systems, while providing up to a 7× speedup over AR diffusion. The qualitative forecasts in The qualitative forecasts in the Appendix (Figure 15) further show that its sampled trajectories remain close to the reference trajectory throughout the rollout. Chronos-2 is less accurate in these experiments. One possible factor is the short forecasting context of only 10 time steps. This is consistent with Zhang & Gilpin (2025), who use 512 context points in their experiments on chaotic systems and report improved zero-shot forecasting performance for Chronos as the context length increases.

VSF reduces long-horizon drift relative to SF. This difference is most apparent on the double pendulum in Figure 1. Adding noise to the initial state of SF reduces the error somewhat, but its forecasts still drift progressively from the reference trajectory. VSF maintains lower error over the same rollout. In our implementation, the previous-frame representation is combined with the latent variable z through a weighted sum at the network bottleneck (see section I). The ablation in section J shows that this latent conditioning is important for the improved long-horizon accuracy.

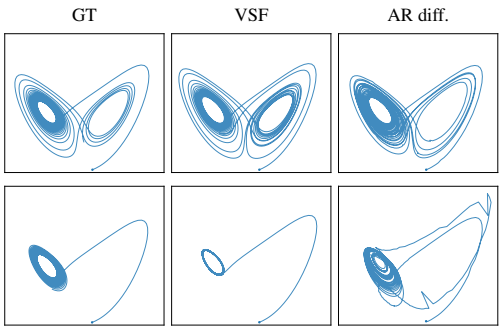

Figure 5: VSF samples remain consistent with the stochastic Lorenz dynamics, while AR diffusion drifts from the attractor. Two futures are shown.

4.2 Stochastic Forecasting

We evaluate partially observed stochastic forecasting on the Rössler and Lorenz systems. For each initial state, we generate multiple trajectories by independently sampling an unobserved dynamical parameter. Each trajectory is deterministic once this parameter is fixed, but the model does not observe it. The same observed history can therefore lead to different future trajectories, including trajectories that separate into different dynamical regimes. We evaluate these forecast distributions using conditional mean error (CME), conditional standard-deviation error (CSE), Wasserstein distance (W_1), and energy distance (ED). For the values reported in our tables, W_1 and ED treat each complete rollout as a single vector, so they compare distributions over whole trajectories and account for temporal structure. In the rollout-error plots, W_1 and ED are instead computed pointwise in time. Detailed metric definitions are provided in App. G.

VSF best matches the conditional distribution of future trajectories. As shown in Table 1, VSF achieves the lowest value for every metric on both stochastic systems. The trajectory ensembles in Figures 4 and 16 show that VSF captures both the spread of the reference futures and the geometry of the underlying dynamics. Block diffusion and DeepAR cover a narrower part of the distribution, while AR diffusion produces diverse futures but is more prone to rollout drift. Perturbing the initial state of SF introduces some diversity, but this inference-time perturbation is not learned by its training objective. Unperturbed SF produces only a single trajectory, yielding a CSE of 1.

VSF preserves trajectory structure over very long horizons. Table 2 shows 2,500-step Lorenz rollout, VSF achieves the lowest CME, CSE, and ED. AR diffusion also achieves a low ED, but its sampled trajectories drift from the system dynamics, as shown in Fig. 5. Block diffusion reaches a lower W_1, but its higher CSE and ED are consistent with the narrower mode coverage seen in Figure 16. In contrast, VSF samples traverse both lobes of the attractor while preserving its characteristic geometry.

Table 2: VSF achieves the lowest CME, CSE, and ED on stochastic Lorenz over a 2,500-step rollout. AR diffusion produces diverse samples but can drift from the system dynamics, as shown in Fig. 5.

	Stochastic dynamics			
	Lorenz			
Model	CME ↓	CSE ↓	W_1 ↓	ED ↓
Block diffusion	0.79 ± 0.09	0.66 ± 0.11	$\mathbf{76.48 \pm 3.20}$	34.79 ± 11.90
AR diffusion	0.62 ± 0.03	0.53 ± 0.03	96.60 ± 2.48	12.43 ± 0.52
Rectified flow	0.77 ± 0.16	0.59 ± 0.21	90.41 ± 1.75	39.60 ± 16.90
SF (perturb.)	1.13 ± 0.33	0.70 ± 0.21	111.95 ± 22.66	39.81 ± 14.86
VSF	$\mathbf{0.56 \pm 0.07}$	$\mathbf{0.39 \pm 0.12}$	85.67 ± 7.29	$\mathbf{12.00 \pm 2.67}$

4.3 World Model Predictor

We next evaluate VSF as an action-conditioned predictor in a JEPA-based world model. The goal is to test whether its physical-time forecasting formulation can improve prediction and planning in a learned representation space. We freeze LeWM's pretrained JEPA representation (Maes et al., 2026) and train only the predictor on offline observation–action trajectories, following the action-conditioned training setup of V-JEPA 2 (Assran et al., 2025). Future latent states are used as teacher-forced training targets. We compare LeWM's AR predictor with SF and VSF predictors while keeping the JEPA encoder, CEM planner, and planning budget fixed. We report latent-trajectory RMSE along the action sequences selected by CEM and planning success on PushT, Cube, TwoRoom, and

Table 3: As a world-model predictor, VSF achieves the highest planning success on all four tasks and the lowest predictor RMSE on three of four tasks. RMSE is measured between predicted and ground-truth latent trajectories along the CEM-selected action sequence.

	PushT		Cube		TwoRoom		Reacher	
Method	Success	RMSE ↓	Success	RMSE ↓	Success	RMSE ↓	Success	RMSE ↓
AR predictor	84.7 ± 4.1	0.28 ± 0.01	69.3 ± 2.5	0.80 ± 0.05	78.7 ± 9.0	0.67 ± 0.03	61.3 ± 8.2	1.06 ± 0.01
SF predictor	88.0 ± 2.8	0.30 ± 0.04	76.7 ± 0.9	$\mathbf{0.64 \pm 0.02}$	76.7 ± 8.2	0.59 ± 0.05	49.3 ± 5.0	1.13 ± 0.03
VSF predictor	$\mathbf{92.7 \pm 2.5}$	$\mathbf{0.27 \pm 0.03}$	$\mathbf{80.0 \pm 1.6}$	0.65 ± 0.08	$\mathbf{82.7 \pm 5.0}$	$\mathbf{0.58 \pm 0.01}$	$\mathbf{68.0 \pm 0.0}$	$\mathbf{1.03 \pm 0.02}$

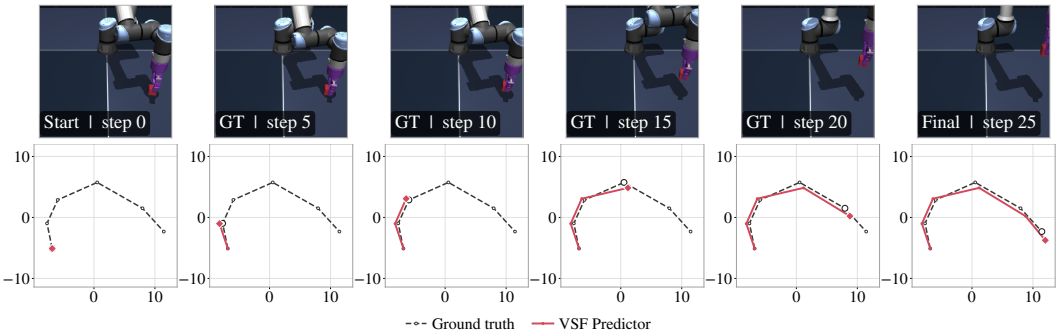

Figure 6: VSF serves as an effective world-model predictor on Cube. The top row shows ground-truth rollout frames, and the bottom row compares the VSF latent trajectory (red) with the ground truth (dashed). The prediction closely follows the ground truth and leads to a successful plan.

Reacher. The learned JEPA representation also evolves smoothly over time, as illustrated in Fig. 6, making physical-time prediction a natural fit for this latent space. Details of the action-conditioned VSF extension are provided in App. F.

VSF improves both latent prediction and planning success. As shown in Table 3, VSF reduces latent-trajectory RMSE relative to the AR predictor on all four tasks and achieves the lowest RMSE on PushT, TwoRoom, and Reacher. It also achieves the highest planning success on every task, improving average success by 7.34% over the AR predictor. The successful Cube rollout in Fig. 6 provides a qualitative example, with the predicted latent trajectory closely following the ground truth. VSF retains LeWM's budget of one velocity-field evaluation per imagined transition.

5 Conclusion

We introduced VSF, a probabilistic extension of SF for forecasting directly in physical time, where one latent variable conditions an entire rollout. This allows VSF to model different possible futures while preserving the direct physical-time generation of SF. We train the latent-conditioned velocity field with variational inference and an energy-based reconstruction over multiple latent samples. We also provide a population-level guarantee connecting this objective to how well the model captures the distribution of possible local dynamics under prior sampling. Empirically, VSF improves long-horizon forecasting on deterministic and partially observed stochastic systems, including the quality of the predicted future distribution, and improves planning success as an action-conditioned world-model predictor, while retaining one velocity-field evaluation per imagined transition.

Limitations and future work. VSF relies on adjacent states providing a meaningful local direction of motionfor learning the physical-time velocity. This assumption is most reliable when the dynamics vary smoothly in physical time. Abrupt impacts, discontinuities, or strong observation noise can make the local velocity poorly resolved. VSF also uses explicit forward Euler integration, which can accumulate discretization error for coarse step sizes or rapidly varying learned fields (Hairer et al., 1993); solver choice can also affect the effective dynamics learned by neural ODEs (Zhu et al., 2022). Future work could therefore explore hybrid or jump-aware latent dynamics (Jia & Benson, 2019), controlled or rough differential equations (Kidger et al., 2020; Morrill et al., 2021), and higher-order or adaptive Runge–Kutta solvers (Queiruga et al., 2020).

AI USE STATEMENT

In this work, we conceived and carried out the original research questions, core ideas, proposed methodology, theoretical contributions, experimental design, and overall execution; generative AI tools were used only in an assistive capacity.

We used generative AI tools to assist with editing and polishing the text, checking mathematical arguments and derivations, and assisting with the implementation and verification of computational methods. We additionally used generative AI tools for improving clarity and presentation, brainstorming and refining research ideas, searching for and identifying potentially relevant literature, assisting with coding and research workflow, and reviewing the manuscript for consistency, clarity, and completeness.

We have reviewed all AI-assisted work. We have independently checked all mathematical arguments, derivations, and claims. We have also verified suggested references against the original sources. The code was inspected and tested, and all reported experimental results were verified. All substantive research decisions, including the formulation of the method, theoretical claims, experimental protocols, interpretation of results, and conclusions, were made by us.

We take full responsibility for the final content of this work, including text, mathematical claims, experimental results, code, and other artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work advances fundamental methodology for generative modeling and does not involve human subjects, sensitive personal data, or deployment in high-stakes applications. We do not identify any specific societal or ethical concerns arising from the contributions of this work.

REPRODUCIBILITY STATEMENT

We have taken several steps to facilitate reproducibility. For the theoretical results, the Appendix provides the full mathematical settings, assumptions, statements, and proofs. For the empirical results, the Appendix provides detailed descriptions of the datasets and benchmarks, experimental protocols, model and training configurations, evaluation procedures, and relevant hyperparameters. Upon acceptance, we will release the source code required to reproduce our experiments in a public GitHub repository and provide the corresponding repository link in the paper.

REFERENCES

Abdul Fatir Ansari, Lorenzo Stella, Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Sundar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, et al. Chronos: Learning the language of time series. *arXiv preprint arXiv:2403.07815*, 2024.

Abdul Fatir Ansari, Oleksandr Shchur, Jaris Küken, Andreas Auer, Boran Han, Pedro Mercado, Syama Sundar Rangapuram, Huibin Shen, Lorenzo Stella, Xiyuan Zhang, et al. Chronos-2: From univariate to universal forecasting. *arXiv preprint arXiv:2510.15821*, 2025.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pp. 214–223. Pmlr, 2017.

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, et al. V-jepa 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*, 2025.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44(10-11):1684–1704, 2025.

Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. *Advances in neural information processing systems*, 31, 2018.

Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron Courville, and Yoshua Bengio. A recurrent latent variable model for sequential data. *Advances in neural information processing systems*, 28, 2015.

Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, 26, 2013.

Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A decoder-only foundation model for time-series forecasting. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 10148–10167. PMLR, 21–27 Jul 2024.

Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Floor Eijkelboom, Grigory Bartosh, Christian A Naesseth, Max Welling, and Jan-Willem van de Meent. Variational flow matching for graph generation. *Advances in Neural Information Processing Systems*, 37:11735–11764, 2024.

Ahmed ElGazzar and Marcel van Gerven. Probabilistic forecasting via autoregressive flow matching. *arXiv preprint arXiv:2503.10375*, 2025.

N Benjamin Erichson, Vinicius Mikuni, Dongwei Lyu, Yang Gao, Omri Azencot, Soon Hoe Lim, and Michael W Mahoney. Flex: A backbone for diffusion-based modeling of spatio-temporal physical systems. *arXiv preprint arXiv:2505.17351*, 2025.

S Feng and Q Fu. Expansion of global drylands under a warming climate. *Atmospheric Chemistry and Physics*, 13(19):10081–10094, 2013.

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

Pengsheng Guo and Alexander G Schwing. Variational rectified flow matching. *arXiv preprint arXiv:2502.09616*, 2025.

David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2(3):440, 2018.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *International conference on machine learning*, pp. 2555–2565. PMLR, 2019.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse control tasks through world models. *Nature*, 640(8059):647–653, 2025.

Ernst Hairer, Gerhard Wanner, and Syvert P Nørsett. *Solving ordinary differential equations I: Nonstiff problems*. Springer, 1993.

Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. Video diffusion models. *Advances in neural information processing systems*, 35:8633–8646, 2022.

Matthew D Hoffman, Matthew J Johnson, et al. Elbo surgery: yet another way to carve up the variational evidence lower bound. In *Workshop in advances in approximate Bayesian inference, NIPS*, volume 1, 2016.

Yinan Huang, Hans Hao-Hsun Hsu, Junran Wang, Bo Dai, and Pan Li. Accelerated sequential flow matching: A bayesian filtering perspective. *arXiv preprint arXiv:2602.05319*, 2026.

Junteng Jia and Austin R Benson. Neural jump stochastic differential equations. *Advances in neural information processing systems*, 32, 2019.

Sunshine Jiang, Xiaolin Fang, Nicholas Roy, Tomás Lozano-Pérez, Leslie Pack Kaelbling, and Siddharth Ancha. Streaming flow policy: Simplifying diffusion/flow-matching policies by treating action trajectories as flow trajectories. In Joseph Lim, Shuran Song, and Hae-Won Park (eds.), *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 238–257. PMLR, 27–30 Sep 2025.

Nouha Karaouli, Denis Coquenet, Elisa Fromont, Martial Mermillod, and Marina Reyboz. How foundational are foundation models for time series forecasting? *arXiv preprint arXiv:2510.00742*, 2025.

Patrick Kidger, James Morrill, James Foster, and Terry Lyons. Neural controlled differential equations for irregular time series. *Advances in neural information processing systems*, 33:6696–6707, 2020.

Marcel Kollovieh, Marten Lienen, David Lüdke, Leo Schwinn, and Stephan Günnemann. Flow matching with gaussian process priors for probabilistic time series forecasting. In *International Conference on Learning Representations*, volume 2025, pp. 95435–95457, 2025.

Simon Lang, Mihai Alexe, Mariana CA Clare, Christopher Roberts, Rilwan Adewoyin, Zied Ben Bouallègue, Matthew Chantry, Jesper Dramsch, Peter D Dueben, Sara Hahner, et al. Aifs-crps: ensemble forecasting using a model trained with a loss function based on the continuous ranked probability score. *npj Artificial Intelligence*, 2(1):18, 2026.

Soon Hoe Lim, Yijin Wang, Annan Yu, Emma Hart, Michael W. Mahoney, Sherry Li, and N. Benjamin Erichson. Elucidating the design choice of probability paths in flow matching for forecasting. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856.

Soon Hoe Lim, Shizheng Lin, Michael W Mahoney, and N Benjamin Erichson. Is flow matching just trajectory replay for sequential data? *arXiv preprint arXiv:2602.08318*, 2026.

Yaron Lipman, Marton Havasi, Peter Holderrieth, Neta Shaul, Matt Le, Brian Karrer, Ricky TQ Chen, David Lopez-Paz, Heli Ben-Hamu, and Itai Gat. Flow matching guide and code. *arXiv preprint arXiv:2412.06264*, 2024.

Dongxiu Liu, Haoyi Niu, Peng Cheng, Yuan Gao, Xirui Kang, Sangli Teng, Koushil Sreenath, and Xianyuan Zhan. Odeworld: A continuous predictive architecture via physical-time flow. *arXiv preprint arXiv:2607.27924*, 2026.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.

Lucas Maes, Quentin Le Lidec, Damien Scieur, Yann LeCun, and Randall Balestriero. Leworld-model: Stable end-to-end joint-embedding predictive architecture from pixels. *arXiv preprint arXiv:2603.19312*, 2026.

James Morrill, Cristopher Salvi, Patrick Kidger, and James Foster. Neural rough differential equations for long time series. In *International conference on machine learning*, pp. 7829–7838. PMLR, 2021.

Kin G Olivares, Geoffrey Negiar, Ruijun Ma, O Nganba Meetei, Mengfei Cao, Michael Mahoney, Yixin Liu, Ajanthan Thalaiyasingam, Hisham Husain, and Vu Nguyen. Clover: Probabilistic forecasting with coherent learning objective reparameterization. *Sort*, 1:1–523, 2015.

Seohong Park, Kevin Frans, Benjamin Eysenbach, and Sergey Levine. Ogbench: Benchmarking offline goal-conditioned rl. In *International Conference on Learning Representations*, volume 2025, pp. 94937–94982, 2025.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4172–4182. IEEE, 2023.

Sebastian Pfister, Benjamin Holzschuh, and Nils Thuerey. Stocbench: A benchmark for generative modeling of stochastic dynamics. *arXiv preprint arXiv:2608.22309*, 2026.

Francois Porcher, Nicolas Carion, Karteek Alahari, and Shizhe Chen. Flow matching in feature space for stochastic world modeling. *arXiv preprint arXiv:2606.29059*, 2026.

Ilan Price, Alvaro Sanchez-Gonzalez, Ferran Alet, Tom R Andersson, Andrew El-Kadi, Dominic Masters, Timo Ewalds, Jacklynn Stott, Shakir Mohamed, Peter Battaglia, et al. Probabilistic weather forecasting with machine learning. *Nature*, 637(8044):84–90, 2025.

Alejandro F Queiruga, N Benjamin Erichson, Dane Taylor, and Michael W Mahoney. Continuous-in-depth neural networks. *arXiv preprint arXiv:2008.02389*, 2020.

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*, pp. 8857–8868. PMLR, 2021.

Kashif Rasul, Arjun Ashok, Andrew Robert Williams, Hena Ghonia, Rishika Bhagwatkar, Arian Khorasani, Mohammad Javad Darvishi Bayazi, George Adamopoulos, Roland Riachi, Nadhir Hassen, et al. Lag-llama: Towards foundation models for probabilistic time series forecasting. *arXiv preprint arXiv:2310.08278*, 2023.

David Ruhe, Jonathan Heek, Tim Salimans, and Emiel Hoogeboom. Rolling diffusion models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 42818–42835. PMLR, 21–27 Jul 2024.

David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. Deepar: Probabilistic forecasting with autoregressive recurrent networks. *International journal of forecasting*, 36(3):1181–1191, 2020.

Irina Sandu, Annelize van Niekerk, Theodore G Shepherd, Simon B Vosper, Ayrton Zadra, Julio Bacmeister, Anton Beljaars, Andrew R Brown, Andreas Dörnbrack, Norman McFarlane, et al. Impacts of orography on large-scale atmospheric circulation. *npj Climate and Atmospheric Science*, 2(1):10, 2019.

Vlad Sobal, Wancong Zhang, Kyunghyun Cho, Randall Balestriero, Tim GJ Rudner, and Yann LeCun. Stress-testing offline reward-free reinforcement learning: A case for planning with latent dynamics models. In *7th Robot Learning Workshop: Towards Robots with Human-Level Abilities*, 2025.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

Gábor J Székely and Maria L Rizzo. Energy statistics: A class of statistics based on distances. *Journal of statistical planning and inference*, 143(8):1249–1272, 2013.

Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, et al. Deepmind control suite. *arXiv preprint arXiv:1801.00690*, 2018.

Ilya Tolstikhin, Olivier Bousquet, Sylvain Gelly, and Bernhard Schoelkopf. Wasserstein auto-encoders. 2018.

Vikram Voleti, Alexia Jolicoeur-Martineau, and Chris Pal. Mcvd-masked conditional video diffusion for prediction, generation, and interpolation. *Advances in neural information processing systems*, 35:23371–23385, 2022.

Kartik Waghmare and Johanna Ziegel. Proper scoring rules for estimation and forecast evaluation. *Annual Review of Statistics and Its Application*, 13, 2025.

Manuel Watter, Jost Springenberg, Joschka Boedecker, and Martin Riedmiller. Embed to control: A locally linear latent dynamics model for control from raw images. *Advances in neural information processing systems*, 28, 2015.

Gerald Woo, Chenghao Liu, Akshat Kumar, Caiming Xiong, Silvio Savarese, and Doyen Sahoo. Unified training of universal time series forecasting transformers. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 53140–53164. PMLR, 21–27 Jul 2024.

Yuanzhao Zhang and William Gilpin. Zero-shot forecasting of chaotic systems. In *International Conference on Learning Representations*, volume 2025, pp. 93873–93899, 2025.

Gaoyue Zhou, Hengkai Pan, Yann Lecun, and Lerrel Pinto. DINO-WM: World models on pre-trained visual features enable zero-shot planning. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 79115–79135. PMLR, 13–19 Jul 2025.

Aiqing Zhu, Pengzhan Jin, Beibei Zhu, and Yifa Tang. On numerical integration in neural ordinary differential equations. In *International Conference on Machine Learning*, pp. 27527–27547. PMLR, 2022.

APPENDIX

A RELATED WORK

Table 4: **Positioning of VSF relative to closely related flow-based formulations.** "Physical time" means that flow progression is identified with the temporal evolution of the modeled sequence, rather than an artificial generative-time coordinate. "'History-conditioned" denotes explicit conditioning on an observed history. "Variational" denotes the use of a training-time recognition model for latent inference. "Future-informed latent" indicates that the latent representation is inferred during training from the realized target dynamics, rather than introduced solely as an auxiliary stochastic coordinate.

Method	Physical time	History-cond.	Latent	Variational	Future-informed latent
FM/Rectified Flow	×	×	×	×	×
Variational Rectified FM	×	×	✓	✓	✓
SFP	✓	✓	×	×	×
SFP + auxiliary latent	✓	✓	✓	×	×
VSF (ours)	✓	✓	✓	✓	✓

Probabilistic and generative forecasting. Probabilistic time-series forecasting has traditionally modeled future uncertainty through autoregressive conditional distributions—using either deterministic states like DeepAR (Salinas et al., 2020) or stochastic latent variables like the Variational RNN (Chung et al., 2015). CLOVER (Olivares et al., 2015) further refines traditional parametric forecasting by reparameterizing the coherent learning objective to better optimize distribution matching. TimeGrad instead replaces these parametric one-step distributions with an autoregressive diffusion model (Rasul et al., 2021). More recently, pretrained time-series models have enabled forecasting across datasets without task-specific training. Lag-Llama provides a probabilistic decoder-only model (Rasul et al., 2023), while Chronos models scaled and quantized time-series values with language-model architectures (Ansari et al., 2024). Other foundation models instead use direct multi-horizon prediction, including TimesFM (Das et al., 2024), Moirai (Woo et al., 2024), and the multivariate and covariate-conditioned Chronos-2 (Ansari et al., 2025). Such models can also provide strong zero-shot forecasts for chaotic systems (Zhang & Gilpin, 2025), but they do not explicitly parameterize the underlying evolution as a physical-time dynamical system (Karaouli et al., 2025). A complementary line of work uses diffusion and flow matching to model conditional future distributions (Erichson et al., 2025). Rolling Diffusion adapts the diffusion process across forecast lead time (Ruhe et al., 2024), TSFlow introduces Gaussian-process priors for conditional flow matching (Kollovieh et al., 2025), and flow-based forecasting has been improved through informed probability paths (Lim et al., 2025), autoregressive conditional flows (ElGazzar & van Gerven, 2025), and Bayesian-filtering-inspired transport between successive predictive distributions (Huang et al., 2026).

Streaming and physical-time flows. Another line of work removes the distinction between generative flow time and temporal evolution. Streaming Flow Policy treats an action trajectory itself as the flow trajectory, enabling actions to be generated and executed incrementally (Jiang et al., 2025). For sequential data, Lim et al. (2026) show that empirical flow matching can be interpreted as a memory-augmented continuous-time dynamical system whose velocity field is constructed from historical transitions. Most directly, ODEWorld introduces Physical-Time Flow, which learns a latent velocity field whose integration directly follows physical time (Liu et al., 2026).

Stochastic world models and probabilistic planning. Predictive uncertainty has long played a role in model-based control (Watter et al., 2015; Ha & Schmidhuber, 2018; Hafner et al., 2025). PETS combines probabilistic dynamics ensembles with trajectory sampling (Chua et al., 2018), while PlaNet learns latent dynamics with both deterministic and stochastic components and plans through imagined trajectories (Hafner et al., 2019). Recent visual world models instead often predict directly in learned feature spaces. DINO-WM predicts future pretrained visual features under candidate actions and uses these predictions for test-time planning (Zhou et al., 2025), while LeWorldModel learns an end-to-end joint-embedding predictor for efficient latent-space planning (Maes et al.,

2026). FlowWM explicitly addresses multimodal future uncertainty by applying flow matching in pretrained feature space (Porcher et al., 2026).

Positioning. Streaming Flow (SF) formulates sequence generation directly in physical time (Jiang et al., 2025), but its standard construction represents residual stochasticity through the evolving physical state; an auxiliary-latent extension is also possible. Variational flow matching provides a mechanism for representing multimodal velocity fields by inferring latent variables from observed target dynamics (Guo & Schwing, 2025; Eijkelboom et al., 2024). We bring these ideas together in *Variational Streaming Flow (VSF)*, a conditional latent variable formulation of physical-time flow modeling for probabilistic forecasting. During training, a variational encoder associates observed future dynamics with latent codes, while at inference a latent drawn from the prior conditions a complete physical-time rollout from the observed state. Thus, we view VSF not as the first latent SF model, but as a variational latent variable formulation of SF for probabilistic forecasting. Table 4 summarizes this positioning.

This combination also raises a question specific to physical-time forecasting: why is it useful to represent future-specific information outside the evolving physical state? We illustrate one such mechanism through a simple bifurcation example in the next section.

B Motivation for VSF

A limitation of standard non-latent Streaming Flow (SF) (Jiang et al., 2025) appears when multiple future trajectories share (almost) the same physical state before bifurcating. The following symmetric two-branch example isolates this issue: even at the population optimum, standard SF can exhibit a fidelity-robustness tradeoff near a bifurcation.

A bifurcation example. Fix an observed history $H = h$. Let $B \in \{-1, +1\}$ be equally likely conditional on h, and consider the two future trajectories

$$\xi_B(t) = B\, m(t), \qquad t \in [0, T], \tag{B.1}$$

where $m : [0, T] \to [0, \infty)$ is absolutely continuous on $[0, T]$, continuously differentiable on $(\tau, T]$, satisfies $m(t) = 0$ for $t \le \tau$, and $m(t) > 0$ for $t > \tau$. Thus the two futures share the same physical trajectory up to the bifurcation time τ and separate afterward.

Let $k > 0$ and $\sigma_0 > 0$, and consider the branch-conditioned fields

$$g_B(x, t) = \dot{\xi}_B(t) - k\big(x - \xi_B(t)\big). \tag{B.2}$$

Define $\sigma_t^2 = \sigma_0^2 e^{-2kt}$, $q(t) = \dot{m}(t) + km(t)$.

Proposition B.1 (Fidelity-sensitivity tradeoff) *Assume that the population optimum of the SF objective is attained. Then, we have the following results.*

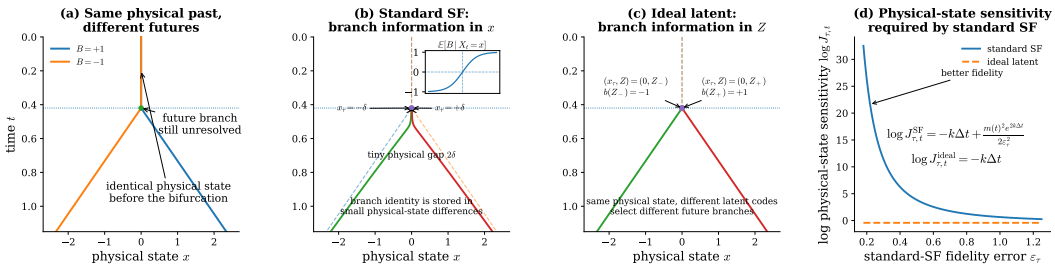

Figure 7: (a) Two futures share the same physical state up to the bifurcation time τ. (b) In non-latent SF, branch identity must be encoded by small differences in the physical state x, making the branch signal increasingly sharp as the Gaussian tube narrows. (c) An ideal latent realization instead carries branch identity in Z, allowing the same physical state $X_\tau = 0$ to generate either future. (d) Accordingly, the standard-SF physical-state sensitivity grows as $J_{\tau,t}^{\mathrm{SF}} \sim \exp(C/\varepsilon_\tau^2)$ as the fidelity error $\varepsilon_\tau \to 0$, while the ideal latent realization retains $J_{\tau,t}^{\mathrm{ideal}} = e^{-k(t-\tau)}$.

(i) The optimal SF marginal field is

$$v^{\star}_{\mathrm{SF}}(x,t) = -kx + q(t)\tanh\!\left(\frac{x\,m(t)}{\sigma_t^2}\right). \tag{B.3}$$

At the bifurcation time, $P_\tau^{\mathrm{SF}} = \mathcal{N}(0,\sigma_\tau^2)$, $P_\tau^{\mathrm{data}} = \delta_0$, *and therefore*

$$\varepsilon_\tau := W_2\big(P_\tau^{\mathrm{SF}}, P_\tau^{\mathrm{data}}\big) = \sigma_\tau. \tag{B.4}$$

(ii) $x(t) \equiv 0$ *is a trajectory of* $v^{\star}_{\mathrm{SF}}$. *If* $\Phi^{\mathrm{SF}}_{\tau,t}$ *denotes its flow map, then for every* $t > \tau$,

$$J^{\mathrm{SF}}_{\tau,t} := \left.\frac{\partial \Phi^{\mathrm{SF}}_{\tau,t}(x)}{\partial x}\right|_{x=0} = \exp\!\left[-k(t-\tau) + \frac{m(t)^2 e^{2k(t-\tau)}}{2\varepsilon_\tau^2}\right]. \tag{B.5}$$

Hence, for every fixed $t > \tau$ *with* $m(t) > 0$, *and with* k, m, *and* τ *fixed,*

$$\sigma_0 \downarrow 0 \quad\Longleftrightarrow\quad \varepsilon_\tau \downarrow 0 \quad\Longrightarrow\quad J^{\mathrm{SF}}_{\tau,t} \to \infty. \tag{B.6}$$

Thus, in this example, standard SF faces an explicit fidelity-sensitivity tradeoff: making the physical tube arbitrarily close to the common target state forces the population marginal field near the branch separator to become arbitrarily sensitive to physical-state perturbations.

Proof. For branch B, plugging $\xi_B(t) = Bm(t)$ into equation B.2 gives

$$g_B(x,t) = -kx + Bq(t), \qquad q(t) = \dot{m}(t) + km(t). \tag{B.7}$$

We first derive the branch-conditioned marginal field. Let $E_t = X_t - \xi_B(t)$. From equation B.2,

$$\dot{E}_t = \dot{X}_t - \dot{\xi}_B(t) = -kE_t. \tag{B.8}$$

Hence

$$E_t = e^{-kt}E_0. \tag{B.9}$$

Since $\xi_B(0) = 0$ and the standard SF construction uses $X_0 \sim \mathcal{N}(0,\sigma_0^2)$, we have $E_0 \sim \mathcal{N}(0,\sigma_0^2)$. Consequently,

$$X_t \mid B \sim \mathcal{N}\big(Bm(t), \sigma_t^2\big), \qquad \sigma_t^2 = \sigma_0^2 e^{-2kt}. \tag{B.10}$$

Under conditional flow matching, the population-optimal marginal field is the conditional expectation of the branch-conditioned target:

$$v^{\star}_{\mathrm{SF}}(x,t) = \mathbb{E}[g_B(X_t,t) \mid X_t = x], \tag{B.11}$$

where the corresponding marginal probability path is:

$$P_t^{\mathrm{SF}} = \frac{1}{2}\mathcal{N}(m(t),\sigma_t^2) + \frac{1}{2}\mathcal{N}(-m(t),\sigma_t^2). \tag{B.12}$$

Using equation B.7,

$$v^{\star}_{\mathrm{SF}}(x,t) = -kx + q(t)\,\mathbb{E}[B \mid X_t = x]. \tag{B.13}$$

It remains to evaluate the posterior mean of B. From equation B.10 and the equal branch probabilities,

$$\frac{\mathbb{P}(B=+1 \mid X_t = x)}{\mathbb{P}(B=-1 \mid X_t = x)} = \frac{\exp\big(-(x-m(t))^2/(2\sigma_t^2)\big)}{\exp(-(x+m(t))^2/(2\sigma_t^2))}. \tag{B.14}$$

Therefore

$$\begin{aligned}\log\frac{\mathbb{P}(B=+1 \mid X_t = x)}{\mathbb{P}(B=-1 \mid X_t = x)} &= \frac{(x+m(t))^2 - (x-m(t))^2}{2\sigma_t^2} \\ &= \frac{2x\,m(t)}{\sigma_t^2}.\end{aligned} \tag{B.15}$$

For a random variable taking values in $\{-1, +1\}$, this gives:

$$\mathbb{E}[B \mid X_t = x] = \tanh\left(\frac{x\, m(t)}{\sigma_t^2}\right). \tag{B.16}$$

Substitution into equation B.13 allows us to obtain equation B.3.

At the bifurcation time, $m(\tau) = 0$. Hence equation B.10 gives

$$P_\tau^{\mathrm{SF}} = \mathcal{N}(0, \sigma_\tau^2), \tag{B.17}$$

whereas the two noiseless target branches both satisfy $\xi_B(\tau) = 0$, so

$$P_\tau^{\mathrm{data}} = \delta_0. \tag{B.18}$$

The only coupling between a distribution and a Dirac mass sends every sample to the location of the Dirac mass. Therefore

$$W_2^2\big(P_\tau^{\mathrm{SF}}, \delta_0\big) = \mathbb{E}|X_\tau|^2 = \sigma_\tau^2. \tag{B.19}$$

Thus $W_2\big(P_\tau^{\mathrm{SF}}, P_\tau^{\mathrm{data}}\big) = \sigma_\tau$, which is precisely equation B.4.

We next compute the sensitivity of the population SF near the branch separator. By symmetry, $v_{\mathrm{SF}}^\star(0, t) = 0$, so $x(t) \equiv 0$ is a trajectory. Differentiating equation B.3 with respect to x gives

$$\partial_x v_{\mathrm{SF}}^\star(x, t) = -k + \frac{m(t)q(t)}{\sigma_t^2} \operatorname{sech}^2\left(\frac{x\, m(t)}{\sigma_t^2}\right). \tag{B.20}$$

Along the central trajectory,

$$\partial_x v_{\mathrm{SF}}^\star(0, t) = -k + \frac{m(t)[\dot{m}(t) + k m(t)]}{\sigma_t^2}. \tag{B.21}$$

Let

$$J_{\tau,t}^{\mathrm{SF}} = \left.\frac{\partial \Phi_{\tau,t}^{\mathrm{SF}}(x)}{\partial x}\right|_{x=0}. \tag{B.22}$$

The variational equation for the one-dimensional ODE flow gives

$$\frac{d}{dt} J_{\tau,t}^{\mathrm{SF}} = \partial_x v_{\mathrm{SF}}^\star(0, t) J_{\tau,t}^{\mathrm{SF}}, \qquad J_{\tau,\tau}^{\mathrm{SF}} = 1. \tag{B.23}$$

Therefore

$$J_{\tau,t}^{\mathrm{SF}} = \exp\left(\int_\tau^t \partial_x v_{\mathrm{SF}}^\star(0, s)\, ds\right). \tag{B.24}$$

Since $\sigma_s^2 = \sigma_0^2 e^{-2ks}$,

$$\frac{d}{ds}\left(\frac{m(s)^2}{2\sigma_s^2}\right) = \frac{m(s)[\dot{m}(s) + k m(s)]}{\sigma_s^2}. \tag{B.25}$$

Using $m(\tau) = 0$ in equation B.21, we obtain

$$\begin{aligned}\int_\tau^t \partial_x v_{\mathrm{SF}}^\star(0, s)\, ds &= -k(t - \tau) + \int_\tau^t \frac{m(s)[\dot{m}(s) + k m(s)]}{\sigma_s^2} ds \\ &= -k(t - \tau) + \frac{m(t)^2}{2\sigma_t^2}.\end{aligned} \tag{B.26}$$

From $\sigma_t^2 = \sigma_\tau^2 e^{-2k(t-\tau)}$ and $\varepsilon_\tau = \sigma_\tau$, we further obtain

$$\frac{m(t)^2}{2\sigma_t^2} = \frac{m(t)^2 e^{2k(t-\tau)}}{2\varepsilon_\tau^2}. \tag{B.27}$$

Using these and equation B.24, we obtain equation B.5.

Finally, for every fixed $t > \tau$ with $m(t) > 0$, the second term in the exponent diverges as $\varepsilon_\tau \to 0$, which proves equation B.6.

Corollary B.2 *Under the assumptions of Proposition B.1, suppose that for some fixed $t > \tau$ the SF sensitivity is required to satisfy $J^{\mathrm{SF}}_{\tau,t} \le J_{\max}$ for some threshold $J_{\max}$. Then, whenever this constraint is feasible, it is necessarily that $J_{\max} > e^{-k(t-\tau)}$, and*

$$\varepsilon_\tau \ge \frac{m(t)e^{k(t-\tau)}}{\sqrt{2\left[\log J_{\max} + k(t-\tau)\right]}}. \tag{B.28}$$

Thus, within this model, keeping the physical-state sensitivity bounded by a threshold imposes a strictly positive lower bound on the fidelity error.

Proposition B.1 reveals a limitation in SF: the physical state must simultaneously track the shared trajectory and encode future branch information, so near a bifurcation improved physical fidelity necessarily cause branch selection to be more sensitive to state perturbations.

Equation B.5 quantifies the resulting fidelity–sensitivity tradeoff.

Why introducing a latent model? While latent variable models are more expressive and can improve performance in general, the preceding example gives a clear motivation to separate the stochastic branch information from the physical state, motivating the use of latent SF models. In the following, we show that there exists such a latent model, without making a claim about the learned model itself. Let $Z \sim \mathcal{N}(0,1)$ be sampled once for the trajectory, define

$$b(Z) = \begin{cases} +1, & Z \ge 0, \\ -1, & Z < 0, \end{cases} \tag{B.29}$$

and use the latent-conditioned physical-time field

$$v_{\mathrm{ideal}}(x,t,Z) = -kx + b(Z)q(t) = b(Z)\dot{m}(t) - k\big(x - b(Z)m(t)\big), \tag{B.30}$$

where $q(t) = \dot{m}(t) + km(t)$ as above. Thus, conditional on the latent branch code $b(Z)$, v_{ideal} has exactly the same form as the branch-conditioned SF field in equation B.7; the difference is that the branch information is now supplied by the latent variable rather than having to be inferred from the physical state.

Starting from the observed state $X_0 = 0$, this field gives

$$X_t = b(Z)m(t), \tag{B.31}$$

and hence realizes the target conditional path law

$$\mathrm{Law}(X_\cdot \mid H = h) = \frac{1}{2}\delta_{\xi_+} + \frac{1}{2}\delta_{\xi_-}. \tag{B.32}$$

In particular, $W_2\left(P^{\mathrm{ideal}}_\tau, P^{\mathrm{data}}_\tau\right) = 0$. Furthermore,

$$\partial_x v_{\mathrm{ideal}}(x,t,Z) = -k, \tag{B.33}$$

so, for fixed Z,

$$\frac{\partial \Phi^{\mathrm{ideal}}_{\tau,t}(x,Z)}{\partial x} = e^{-k(t-\tau)}. \tag{B.34}$$

Indeed, define $E_t = X_t - b(Z)m(t)$. Using equation B.30,

$$\dot{E}_t = \dot{X}_t - b(Z)\dot{m}(t) = -k\big(X_t - b(Z)m(t)\big) = -kE_t. \tag{B.35}$$

Since $X_0 = 0$ and $m(0) = 0$, we have $E_0 = 0$, and therefore $E_t = 0$ for every t, which proves equation B.31. Since $\mathbb{P}(b(Z) = +1) = \mathbb{P}(b(Z) = -1) = 1/2$, this proves equation B.32; since $m(\tau) = 0$, we have $W_2\left(P^{\mathrm{ideal}}_\tau, P^{\mathrm{data}}_\tau\right) = 0$..

More generally, for fixed Z, solving equation B.30 from time τ with $X_\tau = x$ gives

$$\Phi^{\mathrm{ideal}}_{\tau,t}(x,Z) = e^{-k(t-\tau)}x + b(Z)m(t), \tag{B.36}$$

where we used $m(\tau) = 0$. Differentiating with respect to x gives equation B.34. This construction shows what becomes possible once branch information need not be stored in the physical coordinate: the physical state can remain on the common stem, while the latent variable distinguishes the future trajectories.

Connection to rollout robustness. The sensitivity calculation also gives a direct local mechanism to understand rollout error amplification. If a small physical-state perturbation δ is introduced at the bifurcation, then standard SF satisfies

$$\Phi^{\mathrm{SF}}_{\tau,t}(\delta) - \Phi^{\mathrm{SF}}_{\tau,t}(0) = J^{\mathrm{SF}}_{\tau,t}\delta + o(\delta), \tag{B.37}$$

whereas the ideal latent construction satisfies, for fixed Z,

$$\Phi^{\mathrm{ideal}}_{\tau,t}(x+\delta, Z) - \Phi^{\mathrm{ideal}}_{\tau,t}(x, Z) = e^{-k(t-\tau)}\delta. \tag{B.38}$$

Thus, in this example the same mechanism behind the fidelity-sensitivity tradeoff also provides a mechanism by which standard non-latent SF can amplify physical-state rollout errors near a branch separator. By contrast, once the branch information is encoded by the latent variable in the idealized construction, physical-state perturbations are governed only by the original stabilizing dynamics and are contracted at rate $e^{-k(t-\tau)}$.

This motivates VSF. The proposition and discussions above identify a bottleneck of standard non-latent SF, and the idealized latent construction shows how that bottleneck can in principle be removed by carrying future-specific information outside the physical state. This observation motivates VSF. Rather than requiring the physical state alone to carry all residual uncertainty needed for later bifurcation, VSF conditions the physical-time velocity field on an additional latent variable sampled once per forecast.

The variational framework of VSF provides a principled way to associate observed future dynamics with *learned latent codes* during training, while the learned prior provides the corresponding randomness at inference.

The goal is therefore to learn a distribution of coherent physical-time dynamics in which distinct future continuations can be represented without requiring increasingly sensitive separation in the physical state itself.

C A VARIATIONAL OBJECTIVE FOR CLEAN STREAMING TRAJECTORIES

Proposition C.1 (Conditional-velocity evidence lower bound) *Fix a training tuple $(\boldsymbol{u}_s, \boldsymbol{x}_s, s, \boldsymbol{h})$ constructed from the physical-time path in Eqs. 2.5– 2.6. Let $p(\boldsymbol{z})$ be the latent prior, $p_{\boldsymbol{\theta}}(\boldsymbol{u}_s \mid \boldsymbol{x}_s, s, \boldsymbol{h}, \boldsymbol{z})$ be a conditional velocity likelihood, and $q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{u}_s, \boldsymbol{x}_s, s, \boldsymbol{h})$ be a variational posterior. Define the conditional velocity evidence by*

$$p_{\boldsymbol{\theta}}(\boldsymbol{u}_s \mid \boldsymbol{x}_s, s, \boldsymbol{h}) := \int p_{\boldsymbol{\theta}}(\boldsymbol{u}_s \mid \boldsymbol{x}_s, s, \boldsymbol{h}, \boldsymbol{z}) p(\boldsymbol{z})\, d\boldsymbol{z}. \tag{C.1}$$

Assume that the displayed quantities are finite, that the evidence is positive, and that the variational posterior is absolutely continuous with respect to the exact model posterior. Then

$$\begin{aligned}\log p_{\boldsymbol{\theta}}(\boldsymbol{u}_s \mid \boldsymbol{x}_s, s, \boldsymbol{h}) &\geq \mathcal{E}_{\boldsymbol{\theta},\boldsymbol{\phi}}(\boldsymbol{u}_s, \boldsymbol{x}_s, s, \boldsymbol{h}) \\ &:= \mathbb{E}_{q_{\boldsymbol{\phi}}}[\log p_{\boldsymbol{\theta}}(\boldsymbol{u}_s \mid \boldsymbol{x}_s, s, \boldsymbol{h}, \boldsymbol{z})] - D_{\mathrm{KL}}(q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{u}_s, \boldsymbol{x}_s, s, \boldsymbol{h}) \,\|\, p(\boldsymbol{z}))\,.\end{aligned}$$

Proof. Inserting the variational posterior into the conditional evidence gives

$$\begin{aligned}\log p_{\boldsymbol{\theta}}(\boldsymbol{u}_s \mid \boldsymbol{x}_s, s, \boldsymbol{h}) &= \log \int q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{u}_s, \boldsymbol{x}_s, s, \boldsymbol{h}) \frac{p_{\boldsymbol{\theta}}(\boldsymbol{u}_s \mid \boldsymbol{x}_s, s, \boldsymbol{h}, \boldsymbol{z}) p(\boldsymbol{z})}{q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{u}_s, \boldsymbol{x}_s, s, \boldsymbol{h})}\, d\boldsymbol{z} \\ &= \log \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{u}_s, \boldsymbol{x}_s, s, \boldsymbol{h})}\left[\frac{p_{\boldsymbol{\theta}}(\boldsymbol{u}_s \mid \boldsymbol{x}_s, s, \boldsymbol{h}, \boldsymbol{z}) p(\boldsymbol{z})}{q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{u}_s, \boldsymbol{x}_s, s, \boldsymbol{h})}\right] \\ &\geq \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{u}_s, \boldsymbol{x}_s, s, \boldsymbol{h})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{u}_s \mid \boldsymbol{x}_s, s, \boldsymbol{h}, \boldsymbol{z}) + \log p(\boldsymbol{z}) - \log q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{u}_s, \boldsymbol{x}_s, s, \boldsymbol{h})] \\ &= \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{u}_s, \boldsymbol{x}_s, s, \boldsymbol{h})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{u}_s \mid \boldsymbol{x}_s, s, \boldsymbol{h}, \boldsymbol{z})] \\ &\quad - D_{\mathrm{KL}}(q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{u}_s, \boldsymbol{x}_s, s, \boldsymbol{h}) \,\|\, p(\boldsymbol{z})) \\ &= \mathcal{E}_{\boldsymbol{\theta},\boldsymbol{\phi}}(\boldsymbol{u}_s, \boldsymbol{x}_s, s, \boldsymbol{h}),\end{aligned}$$

where Jensen's inequality gives the lower bound.

For completeness, Bayes' rule gives the exact conditional posterior

$$p_{\boldsymbol{\theta}}(\boldsymbol{z} \mid \boldsymbol{u}_s, \boldsymbol{x}_s, s, \boldsymbol{h}) = \frac{p_{\boldsymbol{\theta}}(\boldsymbol{u}_s \mid \boldsymbol{x}_s, s, \boldsymbol{h}, \boldsymbol{z}) p(\boldsymbol{z})}{p_{\boldsymbol{\theta}}(\boldsymbol{u}_s \mid \boldsymbol{x}_s, s, \boldsymbol{h})}.$$

Therefore,

$$\begin{aligned}&D_{\mathrm{KL}}(q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{u}_s, \boldsymbol{x}_s, s, \boldsymbol{h}) \,\|\, p_{\boldsymbol{\theta}}(\boldsymbol{z} \mid \boldsymbol{u}_s, \boldsymbol{x}_s, s, \boldsymbol{h})) \\ &= \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{u}_s, \boldsymbol{x}_s, s, \boldsymbol{h})}\left[\log \frac{q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{u}_s, \boldsymbol{x}_s, s, \boldsymbol{h})}{p_{\boldsymbol{\theta}}(\boldsymbol{u}_s \mid \boldsymbol{x}_s, s, \boldsymbol{h}, \boldsymbol{z}) p(\boldsymbol{z})}\right] + \log p_{\boldsymbol{\theta}}(\boldsymbol{u}_s \mid \boldsymbol{x}_s, s, \boldsymbol{h}) \\ &= \log p_{\boldsymbol{\theta}}(\boldsymbol{u}_s \mid \boldsymbol{x}_s, s, \boldsymbol{h}) - \mathcal{E}_{\boldsymbol{\theta},\boldsymbol{\phi}}(\boldsymbol{u}_s, \boldsymbol{x}_s, s, \boldsymbol{h}),\end{aligned}$$

which proves the variational gap where $p_{\boldsymbol{\theta}}(\boldsymbol{z} \mid \boldsymbol{u}_s, \boldsymbol{x}_s, s, \boldsymbol{h})$ is the exact model posterior. Equality holds if and only if the variational and exact posteriors agree almost everywhere □

Equivalent local conditioning. For a realized training tuple, $\boldsymbol{u}_s$ is the prediction target of the conditional velocity likelihood $p_{\boldsymbol{\theta}}(\boldsymbol{u}_s \mid \boldsymbol{x}_s, s, \boldsymbol{h}, \boldsymbol{z})$, rather than one of its conditioning variables. It is provided only to the training-time variational posterior $q_{\boldsymbol{\phi}}$. For the piecewise-linear physical-time path and $s \in I_k := [s_k, s_{k+1})$,

$$\boldsymbol{u}_s = P\big(\boldsymbol{x}_{m+k+1} - \boldsymbol{x}_{m+k}\big). \tag{C.2}$$

Conversely, because s determines both k and α_s, the two endpoints can be recovered from $(\boldsymbol{u}_s, \boldsymbol{x}_s, s)$ according to

$$\boldsymbol{x}_{m+k} = \boldsymbol{x}_s - \frac{\alpha_s}{P}\boldsymbol{u}_s, \qquad \boldsymbol{x}_{m+k+1} = \boldsymbol{x}_s + \frac{1-\alpha_s}{P}\boldsymbol{u}_s. \tag{C.3}$$

Therefore, on the support of the piecewise-linear interpolation and for almost every s,

$$(\boldsymbol{u}_s, \boldsymbol{x}_s, s) \quad \Longleftrightarrow \quad (\boldsymbol{x}_{m+k}, \boldsymbol{x}_{m+k+1}, \boldsymbol{x}_s, s). \tag{C.4}$$

The local endpoint representation thus provides the same conditioning information to the variational posterior as $(\boldsymbol{u}_s, \boldsymbol{x}_s, s)$. This reparameterization does not alter the conditioning variables $(\boldsymbol{x}_s, s, \boldsymbol{h})$ of the velocity likelihood. Accordingly, the local variational posterior can equivalently be parameterized as

$$q^{\mathrm{loc}}_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x}_{m+k}, \boldsymbol{x}_{m+k+1}, \boldsymbol{x}_s, s, \boldsymbol{h})\,. \tag{C.5}$$

D Population Energy Reconstruction and Energy-Distance Control

Fix $(\boldsymbol{x}_s, s, \boldsymbol{h})$ and draw

$$\boldsymbol{u}, \boldsymbol{u}' \overset{\text{iid}}{\sim} p_{\text{data}}(\cdot \mid \boldsymbol{x}_s, s, \boldsymbol{h}), \quad \boldsymbol{z}, \boldsymbol{z}' \mid \boldsymbol{u} \overset{\text{iid}}{\sim} q_{\boldsymbol{\phi}}(\cdot \mid \boldsymbol{u}, \boldsymbol{x}_s, s, \boldsymbol{h}), \quad \widetilde{\boldsymbol{z}}, \widetilde{\boldsymbol{z}}' \overset{\text{iid}}{\sim} p(\boldsymbol{z}),$$

where all draws are independent except for the displayed conditioning. Define

$$\widehat{\boldsymbol{u}} := \boldsymbol{v}_{\boldsymbol{\theta}}(\boldsymbol{x}_s, s, \boldsymbol{h}, \boldsymbol{z}), \qquad \widehat{\boldsymbol{u}}' := \boldsymbol{v}_{\boldsymbol{\theta}}(\boldsymbol{x}_s, s, \boldsymbol{h}, \boldsymbol{z}'),$$
$$\widetilde{\boldsymbol{u}} := \boldsymbol{v}_{\boldsymbol{\theta}}(\boldsymbol{x}_s, s, \boldsymbol{h}, \widetilde{\boldsymbol{z}}), \qquad \widetilde{\boldsymbol{u}}' := \boldsymbol{v}_{\boldsymbol{\theta}}(\boldsymbol{x}_s, s, \boldsymbol{h}, \widetilde{\boldsymbol{z}}').$$

For a realized target $\boldsymbol{u}$, let

$$S_\alpha(\boldsymbol{u}) := \mathbb{E}[\|\widehat{\boldsymbol{u}} - \boldsymbol{u}\|_2^\alpha \mid \boldsymbol{u}] - \frac{1}{2}\mathbb{E}\big[\|\widehat{\boldsymbol{u}} - \widehat{\boldsymbol{u}}'\|_2^\alpha \mid \boldsymbol{u}\big]. \tag{D.1}$$

This is the population counterpart of the generalized energy reconstruction in Eq. 3.10; replacing its expectations with the M posterior samples gives the finite-ensemble V-statistic $\mathcal{L}_{\text{ES}}$.

The energy distance between the prior-predictive velocity and the true conditional velocity is

$$D_\alpha(\widetilde{\boldsymbol{u}}, \boldsymbol{u}) := 2\mathbb{E}\,\|\widetilde{\boldsymbol{u}} - \boldsymbol{u}\|_2^\alpha - \mathbb{E}\,\|\widetilde{\boldsymbol{u}} - \widetilde{\boldsymbol{u}}'\|_2^\alpha - \mathbb{E}\,\|\boldsymbol{u} - \boldsymbol{u}'\|_2^\alpha. \tag{D.2}$$

Define the aggregate posterior

$$\overline{q}_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x}_s, s, \boldsymbol{h}) := \mathbb{E}_{\boldsymbol{u} \sim p_{\text{data}}(\cdot \mid \boldsymbol{x}_s, s, \boldsymbol{h})}\left[q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{u}, \boldsymbol{x}_s, s, \boldsymbol{h})\right]. \tag{D.3}$$

Proposition D.1 (Prior-predictive energy-distance control) *If $0 < \alpha < 2$, all expectations above are finite, and $\overline{q}_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x}_s, s, \boldsymbol{h}) = p(\boldsymbol{z})$ for almost every $\boldsymbol{z}$, then*

$$\frac{1}{2}D_\alpha(\widetilde{\boldsymbol{u}}, \boldsymbol{u}) \le \mathbb{E}_{\boldsymbol{u} \sim p_{\text{data}}(\cdot \mid \boldsymbol{x}_s, s, \boldsymbol{h})}[S_\alpha(\boldsymbol{u})] \tag{D.4}$$

Proof. For $\rho_\alpha(a, b) = \|a - b\|_2^\alpha$ and $0 < \alpha < 2$, there exist a Hilbert space $\mathcal{H}_\alpha$ and a feature map ψ_α such that

$$\rho_\alpha(a, b) = \|\psi_\alpha(a) - \psi_\alpha(b)\|_{\mathcal{H}_\alpha}^2$$

(Gneiting & Raftery, 2007; Székely & Rizzo, 2013). Define

$$m_q(\boldsymbol{u}) := \mathbb{E}[\psi_\alpha(\widehat{\boldsymbol{u}}) \mid \boldsymbol{u}], \qquad m_p := \mathbb{E}[\psi_\alpha(\widetilde{\boldsymbol{u}})].$$

Aggregate posterior–prior alignment and $\overline{q}_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x}_s, s, \boldsymbol{h}) := \mathbb{E}_{\boldsymbol{u} \sim p_{\text{data}}(\cdot \mid \boldsymbol{x}_s, s, \boldsymbol{h})}[q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{u}, \boldsymbol{x}_s, s, \boldsymbol{h})]$ imply

$$m_p = \mathbb{E}_{\boldsymbol{u} \sim p_{\text{data}}(\cdot \mid \boldsymbol{x}_s, s, \boldsymbol{h})}[m_q(\boldsymbol{u})].$$

The energy-score identities and Jensen's inequality now give

$$\begin{aligned}\frac{1}{2}D_\alpha(\widetilde{\boldsymbol{u}}, \boldsymbol{u}) &= \|\mathbb{E}_{\boldsymbol{u}}[m_q(\boldsymbol{u}) - \psi_\alpha(\boldsymbol{u})]\|_{\mathcal{H}_\alpha}^2 \\ &\le \mathbb{E}_{\boldsymbol{u}}\Big[\|m_q(\boldsymbol{u}) - \psi_\alpha(\boldsymbol{u})\|_{\mathcal{H}_\alpha}^2\Big] = \mathbb{E}_{\boldsymbol{u}}[S_\alpha(\boldsymbol{u})],\end{aligned}$$

where $\boldsymbol{u} \sim p_{\text{data}}(\cdot \mid \boldsymbol{x}_s, s, \boldsymbol{h})$ in each expectation. □

The expected energy reconstruction upper-bounds half the energy distance between the decoded aggregate posterior and the true conditional velocity law. Thus, reducing the reconstruction objective controls a distribution-level discrepancy, rather than only an error in the conditional mean.

Interpretation of aggregate alignment. For each fixed context $c = (\boldsymbol{x}_s, s, \boldsymbol{h})$, the equality says that averaging the encodings of true velocities at that context gives the prior. It can hold in several circumstances:

- **Ideal inference and model fit.** If $q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{u}, c)$ is the exact model posterior and $p_{\boldsymbol{\theta}}(\boldsymbol{u} \mid c) = p_{\mathrm{data}}(\boldsymbol{u} \mid c)$, then Bayes' rule gives

$$\overline{q}_{\boldsymbol{\phi}}(\boldsymbol{z} \mid c) = \int p_{\boldsymbol{\theta}}(\boldsymbol{z} \mid \boldsymbol{u}, c) p_{\boldsymbol{\theta}}(\boldsymbol{u} \mid c)\, d\boldsymbol{u} = p(\boldsymbol{z}).$$

  This is the intended justification for the assumption (Hoffman et al., 2016).
- **Aggregate-matching training.** A penalty on the aggregate posterior can encourage approximate equality, although finite-capacity training does not guarantee exact equality (Tolstikhin et al., 2018).
- **Posterior collapse.** The encoder can output $q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{u}, c) = p(\boldsymbol{z})$ for every $\boldsymbol{u}$, which also satisfies the equality.

E TRAINING AND INFERENCE DETAILS

E.1 FORECASTING TRAINING AND INFERENCE

Algorithms 1 and 2 summarize the two procedures. The current state and rolling window are updated at every step, while $\boldsymbol{h}$ is fixed within each K-step segment. Because future observations are unavailable in forecasting, $\boldsymbol{h}$ is recomputed from the generated-state window at each segment boundary.

Algorithm 1 VSF training

Input: Training windows $\mathcal{D} = \{\boldsymbol{x}_{a:T}\}$; posterior samples M; α and β_{KL}.
Output: Trained velocity field $\boldsymbol{v}_{\boldsymbol{\theta}}$ and posterior $q_{\boldsymbol{\phi}}$.
while not converged **do**
  Sample $\boldsymbol{x}_{a:T} \sim \mathcal{D}$ and $s \sim \mathcal{U}[0,1]$.
  Set $\boldsymbol{h} \leftarrow \boldsymbol{x}_{a:m}$ and construct $(\boldsymbol{x}_s, \boldsymbol{u}_s)$ by equation 2.5–equation 2.6.
  Draw $\boldsymbol{z}^{(1:M)} \overset{\text{iid}}{\sim} q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{u}_s, \boldsymbol{x}_s, s, \boldsymbol{h})$.
  Evaluate in one batch: $\widehat{\boldsymbol{u}}_s^{(1:M)} \leftarrow \boldsymbol{v}_{\boldsymbol{\theta}}(\boldsymbol{x}_s, s, \boldsymbol{h}, \boldsymbol{z}^{(1:M)})$.
  Evaluate $\mathcal{L}_{\mathrm{VSF}}$ by equation 3.11.
  Update $(\boldsymbol{\theta}, \boldsymbol{\phi})$ using $\nabla_{\boldsymbol{\theta},\boldsymbol{\phi}} \mathcal{L}_{\mathrm{VSF}}$.
end while

Algorithm 2 VSF inference

Input: $\boldsymbol{v}_{\boldsymbol{\theta}}$, $\boldsymbol{x}_{a:m}$; integration step $\Delta t = 1/P$; segment length K; forecast endpoint T.
Output: Forecast $\widehat{\boldsymbol{x}}_{m+1:T}$.
Draw $\boldsymbol{z} \sim p(\boldsymbol{z})$ once for the complete rollout.
$\mathcal{W} \leftarrow \boldsymbol{x}_{a:m}$, $\widehat{\boldsymbol{x}} \leftarrow \boldsymbol{x}_m$, and $\boldsymbol{h} \leftarrow \mathcal{W}$.
while the forecast is incomplete **do**
  Set $s \leftarrow 0$ and keep $\boldsymbol{h}$ fixed below.
  for up to K steps while the forecast is incomplete **do**
    $\widehat{\boldsymbol{x}} \leftarrow \widehat{\boldsymbol{x}} + \Delta t\, \boldsymbol{v}_{\boldsymbol{\theta}}(\widehat{\boldsymbol{x}}, s, \boldsymbol{h}, \boldsymbol{z})$.
    $s \leftarrow s + \Delta t$; emit $\widehat{\boldsymbol{x}}$ as the next state.
    $\mathcal{W} \leftarrow \mathrm{Update}(\mathcal{W}, \widehat{\boldsymbol{x}})$.
  end for
  $\boldsymbol{h} \leftarrow \mathcal{W}$ *(refresh from generated states)*.
end while

Here Update is a fixed-length queue: it enqueues the generated state and dequeues the oldest state when the window is full. The integration step Δt advances the normalized forecast time s. The history encoding $\boldsymbol{h}$ is refreshed after each segment, while $\boldsymbol{z}$ is fixed throughout the complete rollout.

F WORLD-MODEL VSF PREDICTOR: TRAINING AND INFERENCE

The world-model implementation specializes the preceding procedures to the LeWM latent space. Let $\mathcal{E}$ denote the frozen JEPA representation map (visual encoder followed by its projection head), and let N be the causal-context length. Write $\boldsymbol{x}_k = \mathcal{E}(\boldsymbol{o}_k)$, with $\boldsymbol{a}_k$ aligned with the transition from $\boldsymbol{x}_k$ to $\boldsymbol{x}_{k+1}$. A training clip contains $N+1$ observations and N aligned actions. The causal token window plays the role of history and is refreshed after every imagined transition; no separate representation $\boldsymbol{h}$ is cached. Only the VSF predictor and its trajectory posterior are trained. Inference below uses the single-frame planning initialization evaluated in our experiments.

For each training clip, draw one $\rho \sim \mathcal{U}[0,1]$ and, for $k = 0, \dots, N-1$, define the teacher-forced tuples

$$s_k = \frac{k+\rho}{N}, \qquad \boldsymbol{x}_{s_k} = (1-\rho)\boldsymbol{x}_k + \rho \boldsymbol{x}_{k+1}, \qquad \boldsymbol{u}_{s_k} = N(\boldsymbol{x}_{k+1} - \boldsymbol{x}_k).$$

Following LeWM (Maes et al., 2026), training uses teacher forcing with temporal causal masking. Every $\boldsymbol{x}_{s_k}$ is constructed from the observed pair $(\boldsymbol{x}_k, \boldsymbol{x}_{k+1})$, and the predictor never consumes an earlier prediction. In contrast, inference is open-loop and feeds each generated latent back into the rolling causal context. Define $\widehat{\boldsymbol{u}}_{s_k}^{(j)} := [\boldsymbol{v}_{\boldsymbol{\theta}}(\boldsymbol{x}_{\boldsymbol{s}}, \boldsymbol{s}, \boldsymbol{a}_{0:N-1}, \boldsymbol{z}^{(j)})]_k$. Causal masking restricts this prediction to the prefix through k. Averaging the energy reconstruction in equation 3.10 over the N teacher-forced transitions gives

$$\begin{aligned}\mathcal{L}^{\mathrm{TF}}_{\text{WM-VSF}} = &\frac{1}{NM}\sum_{k=0}^{N-1}\sum_{j=1}^{M}\left\|\widehat{\boldsymbol{u}}_{s_k}^{(j)} - \boldsymbol{u}_{s_k}\right\|_2^{\alpha} \\ &- \frac{1}{2NM^2}\sum_{k=0}^{N-1}\sum_{i,j=1}^{M}\left\|\widehat{\boldsymbol{u}}_{s_k}^{(i)} - \widehat{\boldsymbol{u}}_{s_k}^{(j)}\right\|_2^{\alpha} \\ &+ \beta_{\mathrm{KL}} D_{\mathrm{KL}}(q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x}_{0:N}, \boldsymbol{a}_{0:N-1}, \boldsymbol{x}_{\boldsymbol{s}}, \boldsymbol{s}) \,\|\, p(\boldsymbol{z}))\,, \qquad 0 < \alpha < 2.\end{aligned} \tag{F.1}$$

Algorithm 3 World-model VSF training

Input: Offline clips $\mathcal{D}_{\mathrm{WM}} = \{(\boldsymbol{o}_{0:N}, \boldsymbol{a}_{0:N-1})\}$; frozen $\mathcal{E}$; posterior samples M; α and β_{KL}.

Output: Trained predictor $\boldsymbol{v}_{\boldsymbol{\theta}}$ and posterior $q_{\boldsymbol{\phi}}$

while not converged **do**

- Sample a minibatch from $\mathcal{D}_{\mathrm{WM}}$ and encode $\boldsymbol{x}_{0:N} \leftarrow \mathcal{E}(\boldsymbol{o}_{0:N})$.
- Construct $(\boldsymbol{s}, \boldsymbol{x}_{\boldsymbol{s}}, \boldsymbol{u}_{\boldsymbol{s}})$ from the teacher-forced tuples defined above.
- Form $q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x}_{0:N}, \boldsymbol{a}_{0:N-1}, \boldsymbol{x}_{\boldsymbol{s}}, \boldsymbol{s})$ and draw $\boldsymbol{z}^{(1:M)}$ by batched reparameterization.
- In one teacher-forced causal batched call, evaluate $\widehat{\boldsymbol{u}}_{\boldsymbol{s}}^{(1:M)} \leftarrow \boldsymbol{v}_{\boldsymbol{\theta}}(\boldsymbol{x}_{\boldsymbol{s}}, \boldsymbol{s}, \boldsymbol{a}_{0:N-1}, \boldsymbol{z}^{(1:M)})$.
- Evaluate $\mathcal{L}^{\mathrm{TF}}_{\text{WM-VSF}}$ by equation F.1 and update $(\boldsymbol{\theta}, \boldsymbol{\phi})$.

end while

Algorithm 4 World-model VSF inference

Input: Current observation $\boldsymbol{o}_m$; candidate actions $\boldsymbol{a}_{m:T-1}$; frozen $\mathcal{E}$; trained $\boldsymbol{v}_{\boldsymbol{\theta}}$; and step $\Delta t = 1/N$.

Output: Imagined latent trajectory $\widehat{\boldsymbol{x}}_{m+1:T}$.

Draw $\boldsymbol{z} \sim p(\boldsymbol{z})$ for the complete rollout.

Set $s \leftarrow 0$ and $\widehat{\boldsymbol{x}}_s \leftarrow \mathcal{E}(\boldsymbol{o}_m)$; initialize rolling sequences $\widehat{\boldsymbol{x}}_{\mathrm{win}}, \boldsymbol{a}_{\mathrm{win}} \leftarrow \emptyset$.

while the candidate plan has an unused action **do**

- Take the next action $\boldsymbol{a}$; append $\widehat{\boldsymbol{x}}_s$ to $\widehat{\boldsymbol{x}}_{\mathrm{win}}$ and $\boldsymbol{a}$ to $\boldsymbol{a}_{\mathrm{win}}$; retain their N most recent entries.
- Set $L \leftarrow |\boldsymbol{a}_{\mathrm{win}}|$ and $\boldsymbol{s} \leftarrow (0, \Delta t, \dots, (L-1)\Delta t)$.
- $\widehat{\boldsymbol{u}}_s \leftarrow [\boldsymbol{v}_{\boldsymbol{\theta}}(\widehat{\boldsymbol{x}}_{\mathrm{win}}, \boldsymbol{s}, \boldsymbol{a}_{\mathrm{win}}, \boldsymbol{z})]_{L-1}$.
- $\widehat{\boldsymbol{x}}_{s+\Delta t} \leftarrow \widehat{\boldsymbol{x}}_s + \Delta t\, \widehat{\boldsymbol{u}}_s$; emit it and set $s \leftarrow s + \Delta t$.

end while

G EVALUATION METRICS

Let c_b, $b = 1, \dots, B$, denote the test conditions. For forecast time $t = 1, \dots, L$, let $\mathbf{y}_{b,t} \in \mathbb{R}^d$ denote the state, and write $\mathbf{y}_b = (\mathbf{y}_{b,1}, \dots, \mathbf{y}_{b,L}) \in \mathbb{R}^{Ld}$ for the complete future trajectory. For condition c_b, let

$$P_b := P(\cdot \mid c_b), \qquad Q_{\theta,b} := Q_\theta(\cdot \mid c_b)$$

denote the reference and learned conditional distributions, respectively. The model produces an ensemble $\{\widehat{\mathbf{y}}_b^{(m)}\}_{m=1}^M$ from $Q_{\theta,b}$, with $\widehat{\mathbf{y}}_{b,t}^{(m)} \in \mathbb{R}^d$ denoting the state at time t in ensemble member m. All reported metrics are averaged over test conditions, and lower values are better. The definitions below give the full-horizon scores reported in the main tables; their relationship to the per-timestep rollout curves is stated at the end of this subsection.

G.1 DETERMINISTIC GROUND TRUTH

For deterministic systems, $P_b = \delta_{\mathbf{y}_b}$ is concentrated at the single ground-truth future $\mathbf{y}_b \in \mathbb{R}^{Ld}$. The metrics in this subsection are pointwise in forecast time: they evaluate time-local states and then aggregate over forecast times, without comparing the joint temporal distribution.

Continuous ranked probability score (CRPS). CRPS is a proper scoring rule for comparing a scalar predictive cumulative distribution function F with an observation y (Gneiting & Raftery, 2007; Waghmare & Ziegel, 2025). It is defined as

$$\mathrm{CRPS}(F, y) = \int_{-\infty}^{\infty} \left[F(a) - \mathbf{1}\{a \geq y\}\right]^2 \, \mathrm{d}a. \tag{G.1}$$

Equivalently, if X and X' are independent draws from F, then

$$\mathrm{CRPS}(F, y) = \mathbb{E}|X - y| - \frac{1}{2}\mathbb{E}|X - X'|. \tag{G.2}$$

We apply this score coordinatewise to the state at each forecast time. Its finite-ensemble estimate, averaged over conditions, forecast times, and state coordinates, is

$$\widehat{\mathrm{CRPS}} = \frac{1}{BLd} \sum_{b=1}^{B} \sum_{t=1}^{L} \left[\frac{1}{M} \sum_{m=1}^{M} \left\| \widehat{\mathbf{y}}_{b,t}^{(m)} - \mathbf{y}_{b,t} \right\|_1 - \frac{1}{2M^2} \sum_{m=1}^{M} \sum_{m'=1}^{M} \left\| \widehat{\mathbf{y}}_{b,t}^{(m)} - \widehat{\mathbf{y}}_{b,t}^{(m')} \right\|_1 \right]. \tag{G.3}$$

The ℓ_1 norms in Eq. G.3 are compact notation for summing scalar CRPS terms over the d state coordinates; they do not make CRPS a joint multivariate or trajectory-level score.

Root mean squared error (RMSE) and mean absolute error (MAE). Both metrics use the time-local ensemble mean $\overline{\mathbf{y}}_{b,t} = M^{-1} \sum_{m=1}^{M} \widehat{\mathbf{y}}_{b,t}^{(m)}$ as the point forecast:

$$\begin{aligned} \mathrm{RMSE} &= \left[\frac{1}{BLd} \sum_{b=1}^{B} \sum_{t=1}^{L} \left\| \overline{\mathbf{y}}_{b,t} - \mathbf{y}_{b,t} \right\|_2^2 \right]^{1/2}, \\ \mathrm{MAE} &= \frac{1}{BLd} \sum_{b=1}^{B} \sum_{t=1}^{L} \left\| \overline{\mathbf{y}}_{b,t} - \mathbf{y}_{b,t} \right\|_1. \end{aligned} \tag{G.4}$$

G.1.1 STOCHASTIC GROUND TRUTH

For stochastic systems, the simulator provides a reference ensemble $\{\mathbf{y}_b^{(r)}\}_{r=1}^R$ from P_b, which we compare with the model ensemble from $Q_{\theta,b}$. For forecast time t, let $P_{b,t}$ and $Q_{\theta,b,t}$ denote the d-dimensional marginal laws of the state at time t under P_b and $Q_{\theta,b}$, respectively. The metrics below define the full-horizon scores reported in the main tables. For the rollout plots, we evaluate each metric separately at each forecast time: CME and CSE use the time-t moments in their relative-error ratios, while W_1 and ED compare the time-t marginal ensembles.

Conditional mean error (CME) and conditional standard-deviation error (CSE). Let $\boldsymbol{\mu}_{P,b,t}$ and $\boldsymbol{\mu}_{Q,b,t}$ denote the mean vectors of $P_{b,t}$ and $Q_{\theta,b,t}$, and let $\boldsymbol{\sigma}_{P,b,t}$ and $\boldsymbol{\sigma}_{Q,b,t}$ denote their coordinatewise standard-deviation vectors. The full-horizon scores reported in the main tables aggregate the resulting marginal-moment errors over all forecast times:

$$\begin{aligned}\mathrm{CME} &= \frac{1}{B}\sum_{b=1}^{B}\frac{\left[\sum_{t=1}^{L}\left\|\boldsymbol{\mu}_{P,b,t}-\boldsymbol{\mu}_{Q,b,t}\right\|_2^2\right]^{1/2}}{\left[\sum_{t=1}^{L}\left\|\boldsymbol{\mu}_{P,b,t}\right\|_2^2\right]^{1/2}},\\ \mathrm{CSE} &= \frac{1}{B}\sum_{b=1}^{B}\frac{\left[\sum_{t=1}^{L}\left\|\boldsymbol{\sigma}_{P,b,t}-\boldsymbol{\sigma}_{Q,b,t}\right\|_2^2\right]^{1/2}}{\left[\sum_{t=1}^{L}\left\|\boldsymbol{\sigma}_{P,b,t}\right\|_2^2\right]^{1/2}}.\end{aligned} \tag{G.5}$$

The moments are estimated from the respective ensembles using population standard deviations. Thus, CME and CSE use the complete horizon in their final norm, but remain pointwise-marginal rather than trajectory-level distribution metrics.

By contrast, the W_1 and ED metrics below are trajectory-level: each ensemble member is retained as one complete vector in $\mathbb{R}^{Ld}$, so their comparisons are sensitive to cross-time dependence.

First-order Wasserstein distance (W_1). Following its standard use in generative modeling (Arjovsky et al., 2017), the population distance is

$$W_1(P_b, Q_{\theta,b}) = \inf_{\gamma\in\Pi(P_b,Q_{\theta,b})}\mathbb{E}_{(\mathbf{Y},\widehat{\mathbf{Y}})\sim\gamma}\left[\left\|\mathbf{Y}-\widehat{\mathbf{Y}}\right\|_2\right], \tag{G.6}$$

where $\Pi(P_b, Q_{\theta,b})$ is the set of couplings with marginals P_b and $Q_{\theta,b}$. For the two empirical ensembles, define the cost matrix $C_{rm}^{(b)} = \|\mathbf{y}_b^{(r)} - \widehat{\mathbf{y}}_b^{(m)}\|_2$. We compute the entropically regularized transport plan (Cuturi, 2013)

$$\gamma_{b,\varepsilon}^{\star} \in \operatorname*{arg\,min}_{\gamma\in\mathcal{U}_{R,M}}\left\langle\gamma, C^{(b)}\right\rangle - \varepsilon H(\gamma), \qquad \widehat{W}_{1,\varepsilon} = \frac{1}{B}\sum_{b=1}^{B}\left\langle\gamma_{b,\varepsilon}^{\star}, C^{(b)}\right\rangle, \tag{G.7}$$

where $\mathcal{U}_{R,M}$ contains nonnegative transport plans with uniform marginals and H is entropy. We use $\varepsilon = 0.1$ and 100 Sinkhorn iterations.

Energy distance (ED). For independent $\mathbf{Y}, \mathbf{Y}' \sim P_b$ and $\widehat{\mathbf{Y}}, \widehat{\mathbf{Y}}' \sim Q_{\theta,b}$, energy distance is defined by (Székely & Rizzo, 2013)

$$\mathrm{ED}(P_b, Q_{\theta,b}) = 2\,\mathbb{E}\left\|\mathbf{Y}-\widehat{\mathbf{Y}}\right\|_2 - \mathbb{E}\left\|\mathbf{Y}-\mathbf{Y}'\right\|_2 - \mathbb{E}\left\|\widehat{\mathbf{Y}}-\widehat{\mathbf{Y}}'\right\|_2. \tag{G.8}$$

We estimate this quantity with the unbiased finite-ensemble statistic

$$\begin{aligned}\widehat{\mathrm{ED}}_b = \frac{2}{RM}\sum_{r=1}^{R}\sum_{m=1}^{M}\left\|\mathbf{y}_b^{(r)}-\widehat{\mathbf{y}}_b^{(m)}\right\|_2 \\ - \frac{1}{R(R-1)}\sum_{\substack{r,r'=1\\ r\neq r'}}^{R}\left\|\mathbf{y}_b^{(r)}-\mathbf{y}_b^{(r')}\right\|_2 \\ - \frac{1}{M(M-1)}\sum_{\substack{m,m'=1\\ m\neq m'}}^{M}\left\|\widehat{\mathbf{y}}_b^{(m)}-\widehat{\mathbf{y}}_b^{(m')}\right\|_2,\end{aligned} \tag{G.9}$$

and report $\widehat{\mathrm{ED}} = B^{-1}\sum_{b=1}^{B}\widehat{\mathrm{ED}}_b$. Unlike the coordinatewise CRPS, both W_1 and ED use Euclidean distances between complete trajectories and therefore evaluate the joint temporal distribution.

H Dataset Statistics

We evaluate five forecasting datasets: three deterministic systems and two conditional parameter-mixture variants. Table 5 summarizes the data used in the reported experiments. We integrate every system with the adaptive DOP853 solver in 64-bit floating-point precision, using relative and absolute error tolerances of 10^{-9}. We store the resulting trajectories in 32-bit floating-point precision for training. The Δt values in Table 5 specify the uniform output-sampling intervals; the solver selects its internal step sizes adaptively.

Dataset	Regime	State dimension	Initial conditions	Trajectories per initial condition	Time steps	Δt	History length
Double pendulum	Deterministic	8	200	1	1,200	0.01	4
Rössler	Deterministic	3	200	1	1,500	0.02	10
Lorenz	Deterministic	3	200	1	3,000	0.01	5
Rössler	Parameter mixture	3	40	100	1,500	0.02	10
Lorenz	Parameter mixture	3	40	100	3,000	0.01	5

Table 5: Forecasting dataset statistics. For deterministic datasets, each sampled initial condition produces one trajectory. For parameter-mixture datasets, each initial condition is paired with 100 independently sampled dynamical parameters, producing 100 trajectories from that same initial state. We ensure all 100 trajectories share an identical path for at least the full length of the conditioning history before diverging. The integration intervals are $[0, 12]$ for double pendulum and $[0, 30]$ for Rössler and Lorenz.

H.1 Deterministic Datasets

Double pendulum. We use only the Cartesian state in the forecasting experiments. Let (x_i, y_i) denote the position of mass i. The constrained Cartesian dynamics are

$$\begin{aligned} &m_1\ddot{x}_1 = -\lambda_1 x_1 + \lambda_2(x_2 - x_1), && m_2\ddot{x}_2 = -\lambda_2(x_2 - x_1),\\ &m_1\ddot{y}_1 = -m_1 g - \lambda_1 y_1 + \lambda_2(y_2 - y_1), && m_2\ddot{y}_2 = -m_2 g - \lambda_2(y_2 - y_1),\\ &x_1^2 + y_1^2 = l_1^2, && (x_2 - x_1)^2 + (y_2 - y_1)^2 = l_2^2. \end{aligned}$$

where λ_1 and λ_2 enforce the rod-length constraints. We set $m_1 = m_2 = l_1 = l_2 = 1$ and $g = 9.81$. For each of the 200 trajectories, we independently sample the initial rod orientations from $\mathcal{U}[-\pi/2, \pi/2]$ and their angular rates from $\mathcal{U}[-0.5, 0.5]$, then convert the initial condition to the Cartesian state $\boldsymbol{x} = (x_1, y_1, x_2, y_2, v_{x_1}, v_{y_1}, v_{x_2}, v_{y_2}) \in \mathbb{R}^8$.

Rössler system. The Rössler dynamics are

$$\dot{x} = -y - z, \qquad \dot{y} = x + ay, \qquad \dot{z} = b + z(x - c). \tag{H.1}$$

We set $a = b = 0.2$ and $c = 5.7$. To generate each of the 200 initial conditions, we select a center uniformly from $\{-1, 1\}^3$; the initial-condition perturbation has standard deviation 0.01.

Lorenz system. The Lorenz dynamics are

$$\dot{x} = \sigma(y - x), \qquad \dot{y} = \rho x - y - xz, \qquad \dot{z} = xy - \beta z. \tag{H.2}$$

We use $\sigma = 10$, $\beta = 8/3$, and $\rho = 24.74$. Each of the 200 initial conditions is centered at $(1, 1, 1)$, with perturbation standard deviation 0.01.

H.2 Stochastic Datasets

Stochastic data-generation. These benchmarks contain uncertainty over a trajectory-level dynamical parameter rather than additive process noise. For each initial condition i, we sample one state $\boldsymbol{x}_0^{(i)}$ and hold it fixed while generating 100 trajectories. For each trajectory j, we independently draw a parameter η_{ij} and integrate the resulting deterministic system:

$$\eta_{ij} \sim \mathcal{U}(\eta_{\min}, \eta_{\max}), \qquad \boldsymbol{x}_{ij}(t) = \mathrm{ODESolve}\Big(f_{\eta_{ij}}, \boldsymbol{x}_0^{(i)}, t\Big). \tag{H.3}$$

The parameter is withheld from the forecaster, so a shared observed context corresponds to a distribution of possible futures. We split the 40 initial conditions, rather than the resulting 4,000 trajectories, into 70%/15%/15% train/validation/test partitions. For evaluation, we condition on each held-out initial condition and use all 100 corresponding futures as samples from its reference conditional distribution.

Stochastic Rössler system. We use Eq. H.1 with $a = b = 0.2$ and independently sample $c_{ij} \sim \mathcal{U}[5.2, 6.2]$ for every trajectory. Each initial condition is obtained by selecting a center uniformly from $\{-1, 1\}^3$, with perturbation standard deviation 0.01.

Stochastic Lorenz system. We use Eq. H.2 with $\sigma = 10$ and $\beta = 8/3$, and independently sample $\rho_{ij} \sim \mathcal{U}[23.74, 25.74]$ for every trajectory. Each initial condition is centered at $(1, 1, 1)$, with perturbation standard deviation 0.01.

H.2.1 World-Model Datasets

We use the four offline visual-control datasets in LeWM (Maes et al., 2026). Trajectories contain 224×224 RGB images, continuous actions, and auxiliary simulator state. Following LeWM, we sample images every five environment steps and form four-frame clips with a five-action block per transition. The predictor uses only frozen LeWM image embeddings and actions. Auxiliary state is used only to restore evaluation starts and goals and compute success.

PushT. In this continuous 2D task, a blue point agent pushes a T-shaped block to a demonstrated future configuration (Chi et al., 2025). Block interactions are limited to pushing. The goal specifies both pusher and block poses. Following LeWM, we reuse noisy DINO-WM expert replay (Zhou et al., 2025) and reports 20,000 episodes averaging 196 steps.

Cube. OGBench-Cube is the single-object variant of the OGBench manipulation benchmark (Park et al., 2025). A UR5e arm with a parallel-jaw gripper is controlled through end-effector displacement, yaw, and gripper commands to pick up a cube and place it at a target location. The dataset contains 10,000 200-step episodes collected with the benchmark's scripted collection heuristic. Success depends on the cube center reaching its target, regardless of cube orientation.

TwoRoom. A red point agent starts at a random position in one room and navigates to a sampled target in the other through the single doorway (Sobal et al., 2025). The dataset contains 10,000 episodes averaging 92 steps, generated by a noisy doorway-then-target heuristic.

Reacher. Reacher is based on the DeepMind Control Suite planar two-link arm (Tassa et al., 2018). We use the DINO-WM/LeWM *qpos-match* variant, the goal is a complete future joint configuration. The dataset contains 10,000 200-step episodes collected with a Soft Actor–Critic policy.

I EXPERIMENTAL SETUP

I.1 FORECASTING

Baseline models. We compare VSF with eight baselines: Block diffusion, which jointly denoises a future block; AR diffusion, which denoises one state at a time; Rectified flow (Liu et al., 2022), which denoises one state at a time; DeepAR (Salinas et al., 2020); SF (Jiang et al., 2025); SF (perturbation), which perturbs the normalized initial state at inference; frozen Chronos-2 (Ansari et al., 2025); and Chronos-2 (calibrated), our post-hoc affine calibration of its predictive quantiles.

Method-specific settings.

- **Block diffusion.** A conditional one-dimensional U-Net jointly denoises a length-20 target window comprising the anchor and 19 future states. We commit 5 predicted states as a block before reconditioning. A blockwise autoregressive strategy is also used in video generation (Voleti et al., 2022; Ho et al., 2022). We use 100 training noise levels, x_0 prediction, and DDIM (Song et al., 2020) with 20 denoising steps.
- **AR diffusion.** Following TimeGrad's autoregressive forecasting formulation (Rasul et al., 2021), we denoise one future state at a time. Our implementation uses a four-layer, four-head causal Transformer Peebles & Xie (2023) of width 128 with diffusion v-prediction. We use 100 training noise levels and DDIM with 20 denoising steps per predicted state.
- **Rectified flow** (Liu et al., 2022) uses a four-layer, four-head transformer with width 128 same as AR diffusion. It autoregressively generates one future state at a time using velocity prediction and 10 denoising steps per predicted state.
- **DeepAR** (Salinas et al., 2020) is an autoregressive probabilistic forecasting model based on recurrent neural networks.We use a two-layer LSTM with hidden width 128 and a diagonal Gaussian output distribution with minimum scale 10^{-4}.
- **SF and SF (perturbation).** SF (Jiang et al., 2025) uses the thin-tube path with $\sigma_0 = 0.4$ and $k = 10$ and uses a U-Net backbone following the original paper and similar to block diffusion, and advances ten states before reconditioning, except on stochastic Lorenz where it advances one state. The perturbation variant modifies the normalized initial state once, at the start of the rollout. We select the standard deviation of the perturbation noise between 0.1 and 0.01 depending on the performance.
- **Chronos-2.** Chronos-2 is a pretrained time-series foundation model that directly returns predictive quantiles over a requested forecast horizon (Ansari et al., 2025). We keep its parameters frozen and use its full-horizon quantile forecasts. Chronos-2 (calibrated) is our post-hoc variant, not part of the standard Chronos-2 method.
- **VSF.** VSF uses the same conditional one-dimensional U-Net as SF and adds a four-dimensional trajectory-level latent at the network bottleneck. A two-layer MLP maps this latent to the bottleneck feature width. After the middle residual blocks and before upsampling, we fuse the latent feature with the time- and history-conditioned bottleneck activation using a weighted sum. We fix the latent fusion using 90% latent features and 10% bottleneck features, while all skip connections and the original SF conditioning remain unchanged. We also fix the $\alpha = 1.999$ and $\beta_{\mathrm{KL}} = 2 \times 10^{-3}$ for every forecasting dataset. We follow SF's tube-based training construction for double pendulum and Rössler. We observe Lorenz dynamics are sensitive to injected state noise, hence we disable tube perturbations.

Chronos-2 calibration. Suppressing the context index, let $Q_{t,d}(\tau)$ be the raw Chronos-2 τ-quantile at forecast time t for state coordinate d, and let $\mu_{t,d} := \int_0^1 Q_{t,d}(u)\,du$ be its predictive mean. We apply the per-coordinate transformation

$$Q^{\mathrm{cal}}_{t,d}(\tau) = a_d + b_d\mu_{t,d} + c_d\big(Q_{t,d}(\tau) - \mu_{t,d}\big), \qquad b_d, c_d > 0. \tag{I.1}$$

Thus, a_d shifts the forecast, b_d rescales its mean, and c_d rescales its spread around the mean. We compute $\mu_{t,d}$ by trapezoidal integration over the returned quantile grid, using flat tails at 0 and 1.

Experimental details. We use 70%/15%/15% train/validation/test splits over trajectories for deterministic data and over initial conditions for parameter-mixture data. We normalize each coordinate to $[-1, 1]$ using training-set extrema and turn the 100 trajectories per initial condition into individual training examples only after splitting. Learned methods use AdamW with learning rate 10^{-4}, zero weight decay, a 200-step warmup followed by cosine decay, and exponential-moving-average weights with decay 0.75. We train for 90 epochs on deterministic datasets and 18 epochs on parameter-mixture datasets. Results are the mean and population standard deviation over three seeds. We run all training and evaluation jobs on NVIDIA A100 GPUs. At test time we generate 100 trajectories per held-out context and evaluate 1,000-step rollouts; the additional stochastic-Lorenz experiment uses 2,500 steps.

I.2 World Models

Baseline models. We train the SF and VSF predictors using the JEPA-based LeWM representation from the corresponding task-specific pretrained checkpoint (Maes et al., 2026). We freeze LeWM's visual encoder, projection head, and action encoder, and compare against its pretrained autoregressive (AR) predictor. All three predictors operate in the same 192-dimensional latent space and use the same CEM planner, evaluation tasks, and planning budget.

Method-specific settings.

- **AR predictor.** We use the pretrained LeWM predictor, which autoregressively predicts the next JEPA latent conditioned on the action sequence.
- **SF predictor.** We replace the AR predictor with a deterministic physical-time velocity model and use one velocity-field evaluation for each imagined transition.
- **VSF predictor.** We condition the physical-time velocity on a prior latent sampled once and held fixed throughout each imagined rollout. This also requires one velocity-field evaluation per transition.

Experimental details. LeWM encodes each 224×224 RGB frame with a ViT-Tiny (Dosovitskiy, 2020) backbone using 14×14 patches, 12 Transformer layers, three attention heads, and hidden dimension 192. Its final-layer `[CLS]` token is passed through LeWM's BatchNorm MLP projector to obtain the 192-dimensional JEPA state used by every predictor. Each modeled transition spans five environment steps. We use a 90%/10% training/validation split. SF and VSF are trained with AdamW using learning rate 10^{-4}, zero weight decay, batch size 4,096, and exponential-moving-average decay 0.75. We train VSF using Eq. (F.1) as loss for 60 epochs on Push-T and Cube and 90 epochs on Two-Room and Reacher, and report results over three seeds.

For planning, CEM maintains a Gaussian distribution over every action sequence. Each of 30 CEM updates evaluates 300 candidates and fits the mean and diagonal scale to the best 30, starting from unit scale. Following LeWM, we evaluate 50 goals per run, sample each goal 25 environment steps ahead, and allow a 50-step execution budget. Predictor RMSE is measured on CEM-selected action sequences. For VSF, we keep $\alpha = 1.999$. We set $\beta_{\mathrm{KL}} = 1$ for PushT, Cube, and TwoRoom and 2×10^{-3} for Reacher.

J FULL ABLATION RESULTS

igures 8– Figure 12 report the complete ablation study across deterministic double pendulum and stochastic Rössler. Across both systems, the results support the roles of the generalized energy loss, prior alignment, trajectory-level latent conditioning, and rollout-wise latent consistency.

Energy-score exponent α and MSE. Figure 8 shows that performance improves consistently as the energy-score exponent α increases, with all settings outperforming the baseline and the near-quadratic setting also surpassing MSE. This trend reflects the optimization geometry of the generalized energy score. At $\alpha = 1$, it reduces to the standard energy score, whereas $1 < \alpha < 2$ places relatively greater emphasis on large local errors, since the gradient of a residual term r^α scales as $\alpha r^{\alpha-1}$. MSE remains competitive, but it lacks the pairwise term through which the energy loss directly separates multiple forecast samples.

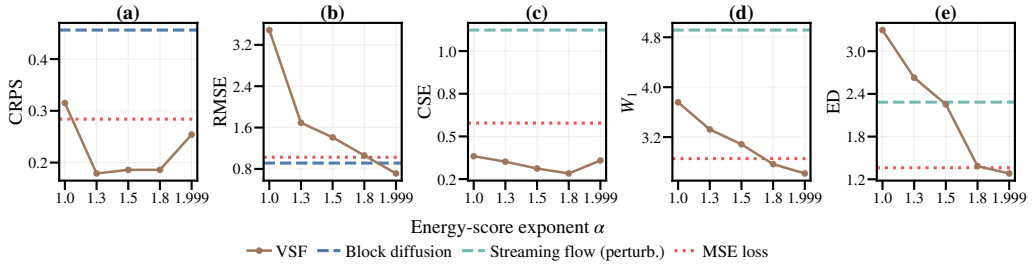

Figure 8: **Energy-score exponent.** α ablation on (a)–(b) deterministic double pendulum and (c)–(e) stochastic Rössler. Dashed lines show the best overall baseline for each system; dotted lines show MSE-trained VSF. Three-seed means.

KL regularization. Figure 9 shows that performance is relatively robust across the tested KL weights and remains better overall than SF, with an intermediate weight giving the strongest aggregate result. The KL penalty must balance retaining target-specific branch information in the training posterior against aligning that posterior with the prior used for forecasting. When regularization is too weak $\beta_{\mathrm{KL}} = 10^{-3}$, the posterior–prior gap makes prior samples less reliable at inference.

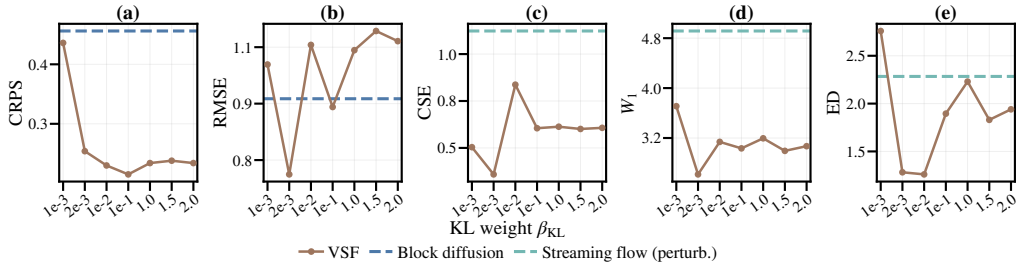

Figure 9: **KL regularization.** β_{KL} ablation on (a)–(b) deterministic double pendulum and (c)–(e) stochastic Rössler. Dashed lines show the best overall baseline for each system. Three-seed means.

Latent contribution. Figure 10 varies the latent contribution in bottleneck-sum fusion. With no latent contribution, the model reduces to SF and must rely on its evolving predicted history, so rollout errors can accumulate. Every positive latent contribution improves overall performance relative to SF by supplying a stable trajectory-level signal that is not corrupted by previous prediction errors. An intermediate contribution performs best, indicating that feedback through the recurrent bottleneck remains useful. Although performance weakens at maximal latent contribution, the model still retains state and physical-time conditioning and therefore remains robust; maximal latent contribution does not make the predictor independent of the physical state.

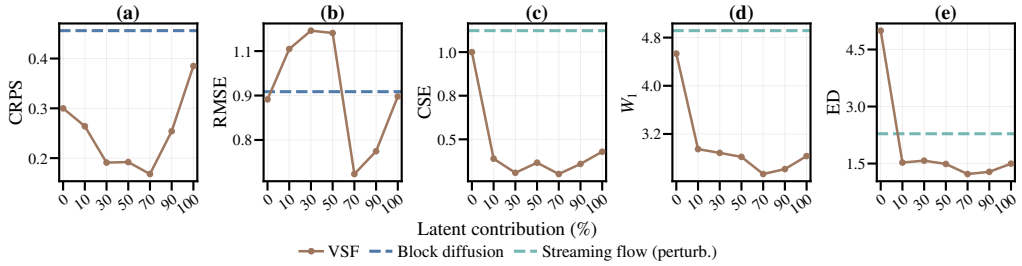

Figure 10: **Latent contribution.** Bottleneck latent-weight ablation on (a)–(b) deterministic double pendulum and (c)–(e) stochastic Rössler. Dashed lines show the best overall baseline for each system. Three-seed means.

Latent resampling. Figure 11 compares sampling one latent for the complete rollout with drawing a new latent at every forecast step. The two strategies perform similarly early in the forecast, but the gap widens with horizon in favor of a fixed latent. A rollout-wise fixed z provides a coherent trajectory-level source of uncertainty, whereas stepwise resampling introduces temporally inconsistent perturbations and primarily harms long-horizon probabilistic accuracy.

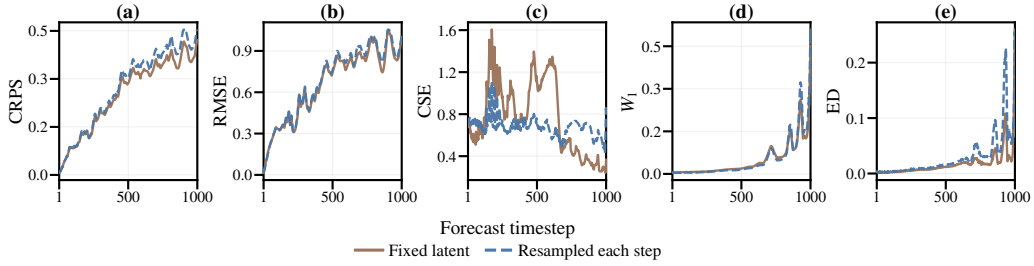

Figure 11: **Latent resampling.** Fixed rollout latent versus per-step resampling on (a)–(b) deterministic double pendulum and (c)–(e) stochastic Rössler. Three-seed means.

Latent-source and prior-alignment variants. Figure 12 separates three effects that are otherwise coupled in VSF: stochastic latent conditioning, future-conditioned variational training, and alignment of the training posterior with the inference prior. Here $p(\boldsymbol{z}) = \mathcal{N}(\mathbf{0}, \mathbf{I}_{d_z})$, and $q_{\boldsymbol{\phi}} := q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{u}_s, \boldsymbol{x}_s, s, \boldsymbol{h})$ is the future-conditioned training posterior. Table 6 defines the four variants explicitly.

SF is the non-latent, MSE-trained reference. The three latent-conditioned variants use the same decoder, M-sample generalized energy reconstruction loss, and bottleneck latent contribution of 0.9. Prior-latent SF is the direct non-variational control: it uses prior samples during both training and forecasting, with neither a posterior encoder nor a KL loss. In contrast, Posterior VSF (no KL) remains variational because it uses the future-conditioned posterior during training. Its prior samples at inference are therefore unaligned with the training latents, creating a train–test mismatch.

Table 6: Definition of the latent-source and prior-alignment variants.

Variant	Training latent	Posterior encoder	β_{KL}	Forecast latent
SF	None	No	–	None
Prior-latent SF	$\boldsymbol{z}^{(1:M)} \overset{\text{iid}}{\sim} p(\boldsymbol{z})$	No	–	$\boldsymbol{z} \sim p(\boldsymbol{z})$, fixed
Posterior VSF (no KL)	$\boldsymbol{z}^{(1:M)} \overset{\text{iid}}{\sim} q_{\boldsymbol{\phi}}$	Yes	0	$\boldsymbol{z} \sim p(\boldsymbol{z})$, fixed
Full VSF	$\boldsymbol{z}^{(1:M)} \overset{\text{iid}}{\sim} q_{\boldsymbol{\phi}}$	Yes	2×10^{-3}	$\boldsymbol{z} \sim p(\boldsymbol{z})$, fixed

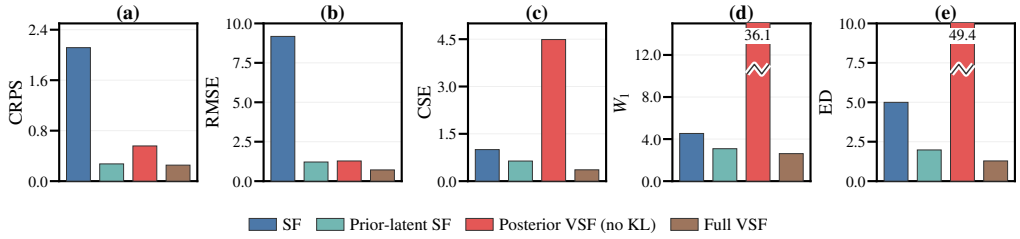

Figure 12: **Latent source and prior alignment.** SF, Prior-latent SF, Posterior no-KL, and Full VSF on (a)–(b) deterministic double pendulum and (c)–(e) stochastic Rössler. Three-seed means; breaks mark labeled out-of-range values.

Does latent conditioning help? Prior-latent SF versus SF measures the benefit of the latent-conditioned, multi-sample predictor without a learned posterior. Its improvement across all metrics shows that latent-conditioned prediction is already useful without variational training.

Does variational training help? Full VSF has lower error than Prior-latent SF on all five metrics in Figure 12. Prior-latent SF samples training latents independently of the motion it must predict, making it harder to associate latent values with different possible futures. Full VSF instead infers latents from the observed local motion during training, helping different latent values represent different continuations. This is useful near a bifurcation, where nearly identical physical states can lead to different outcomes.

Why prior alignment matters. Full VSF versus Posterior no-KL isolates the role of the KL term when training uses a future-conditioned posterior. Removing alignment degrades every metric, especially the Rössler distributional scores. This is consistent with train–test mismatch. Without KL regularization, posterior codes may occupy regions or relative masses that are not reproduced by the prior, so prior-sampled rollouts can miss or misweight learned continuations. The gains from posterior-based training therefore depend on aligning it with the forecasting prior.

K ROLLOUT ERROR PLOTS

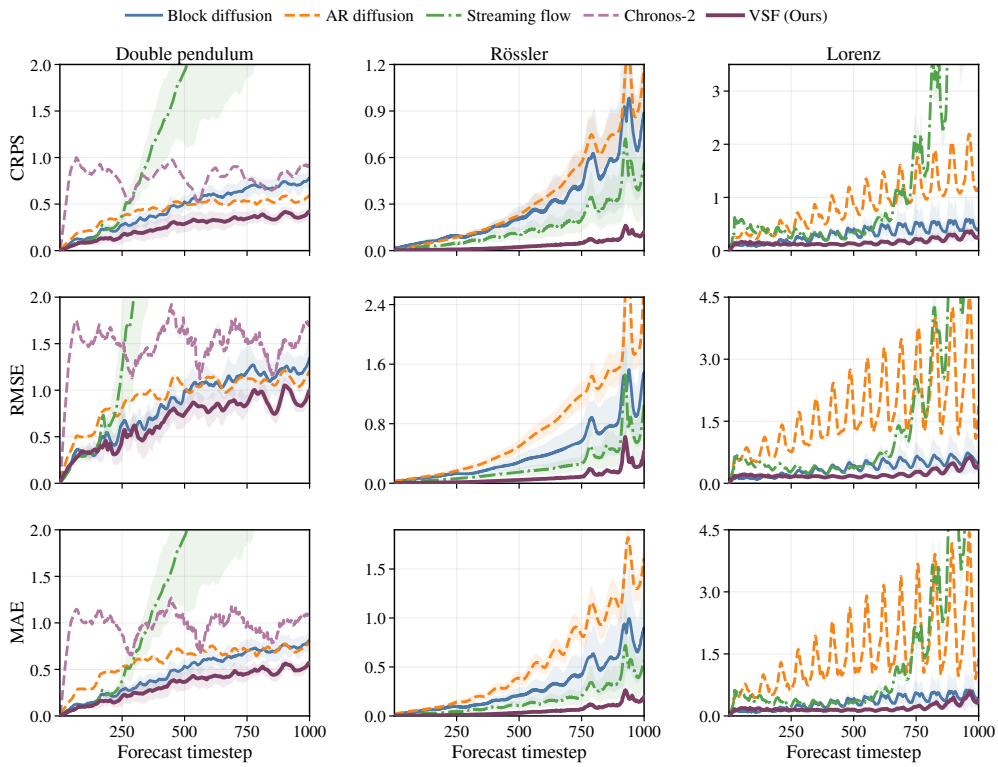

Figure 13: Per-timestep error accumulation for 1,000-step deterministic forecasts. Columns show double pendulum, Rössler, and Lorenz; rows show CRPS, RMSE, and MAE. Curves and shaded bands report the mean and one standard deviation.

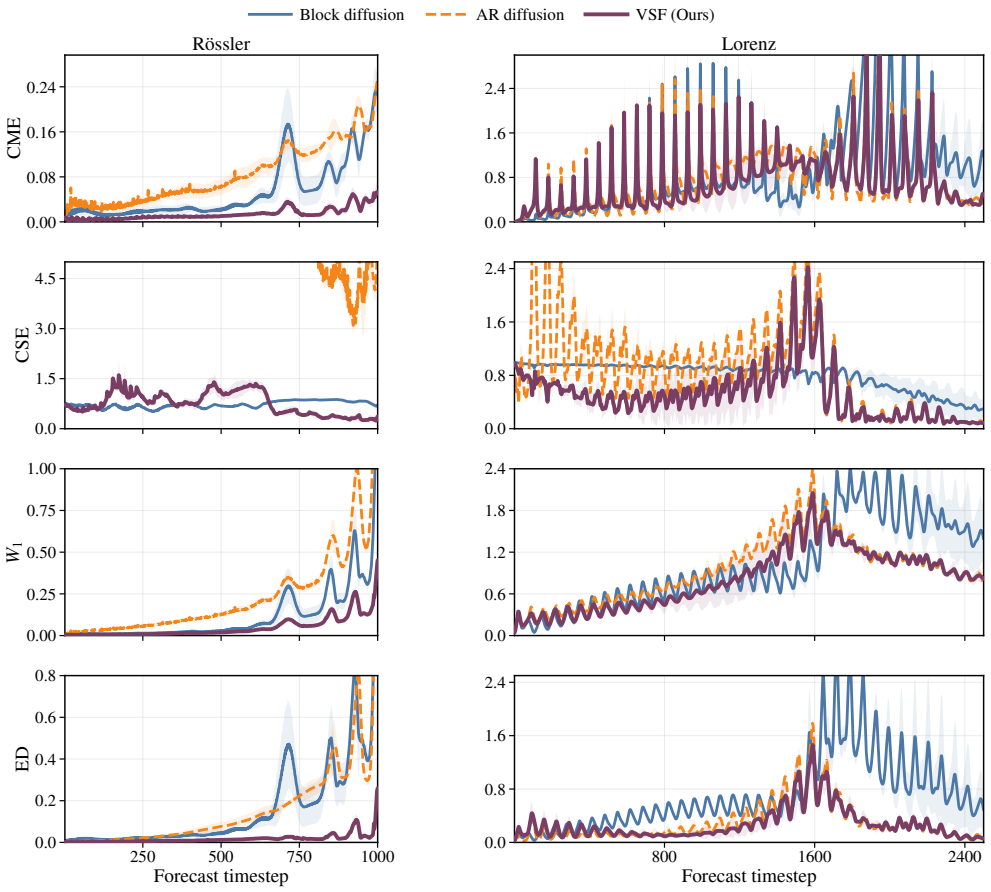

Figure 14: Per-timestep distributional error for stochastic Rössler over 1,000 steps and stochastic Lorenz over 2,500 steps. Columns show the two systems; rows show CME, CSE, marginal W_1, and marginal energy distance. Curves and shaded bands report the mean and one standard deviation. These per-timestep scores compare marginal standardized state distributions. They provide a different view from the joint whole-trajectory metrics in Table 1 and 2.

L TRAJECTORY FORECASTS VISUALIZATIONS

We compare sampled forecast trajectories across the evaluated systems and baselines. For the deterministic systems, we additionally show the VSF ensemble predictive mean.

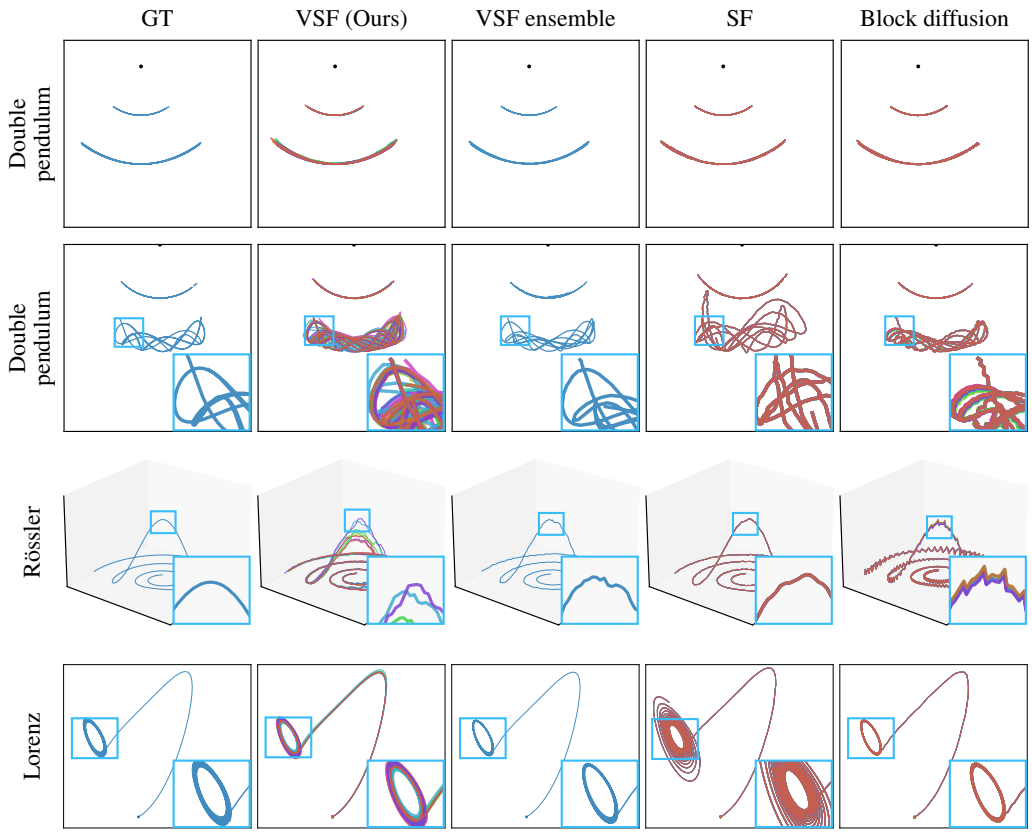

Figure 15: Qualitative forecasts on the deterministic systems. Columns show ground truth (GT), VSF samples, the VSF ensemble predictive mean, SF, and block diffusion. Sample panels contain $N = 10$ trajectories; The VSF ensemble is the mean of the samples. The last three rows magnify highlighted regions of the chaotic double-pendulum, Rössler, and Lorenz forecasts.

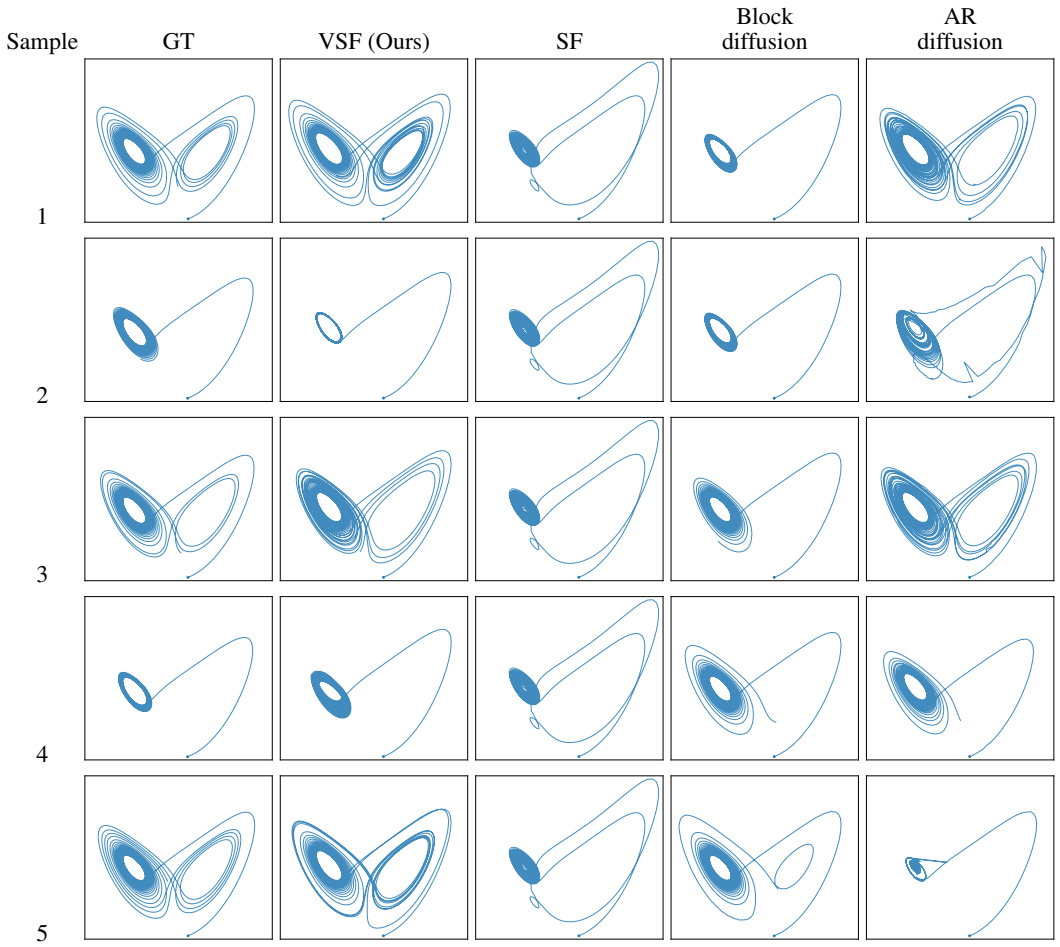

Figure 16: Visualization samples from the ground truth and generated trajectories for the partially observed stochastic Lorenz.

M WORLD-MODEL LATENT-TRAJECTORY VISUALIZATIONS

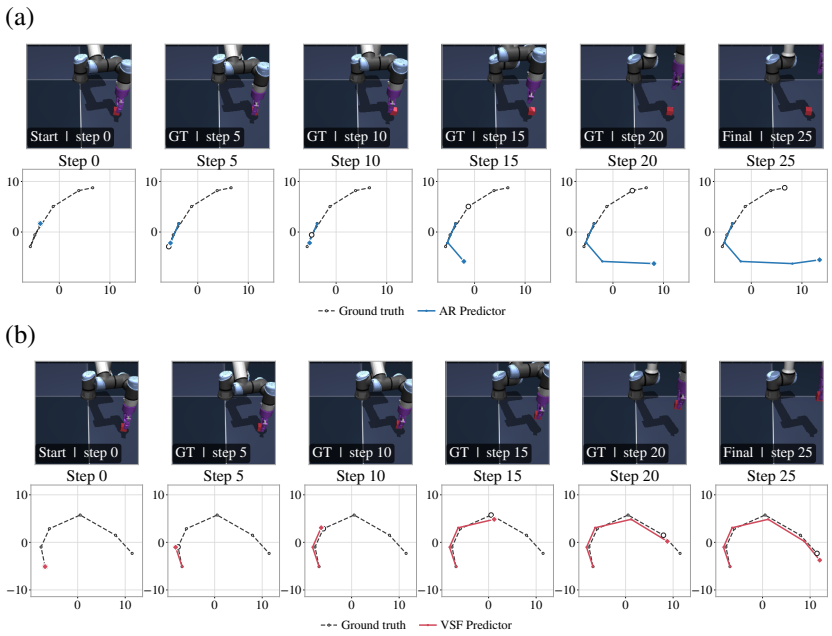

Figure 17: The top row shows the ground-truth observations during rollout. The bottom row compares the predicted and ground-truth latent trajectories after joint-PCA projections. (a) The AR Predictor trajectory diverges and the rollout fails. (b) The VSF Predictor trajectory follows the ground-truth trajectory and the rollout succeeds.

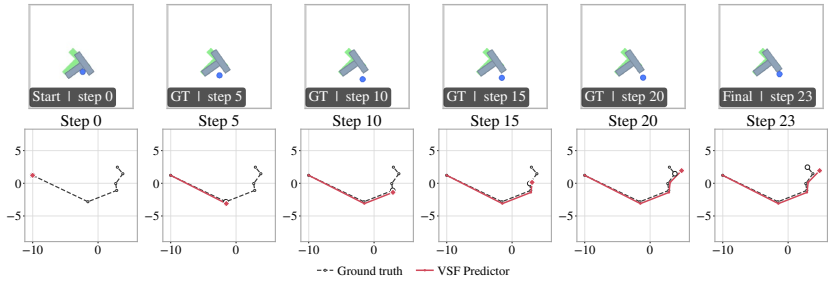

Figure 18: Qualitative PushT result. VSF action-conditioned predicted latent trajectory follows the ground-truth trajectory in a successful rollout.

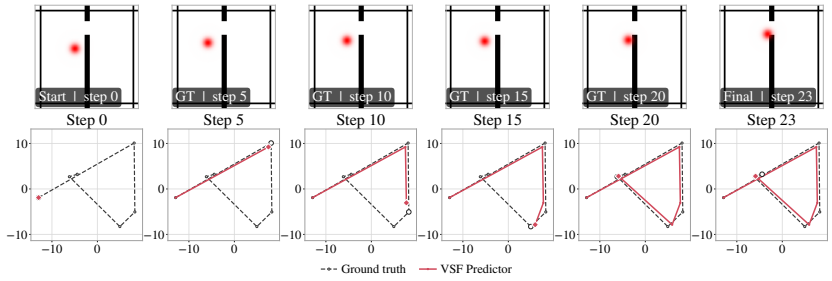

Figure 19: Qualitative TwoRoom result. VSF action-conditioned predicted latent trajectory follows the ground-truth trajectory in a successful rollout.