Variational Trajectory Flow Matching with a Non-Linear Interpolant

Anonymous authors
Paper under double-blind review

Abstract

Simulation-free methods such as Trajectory Flow Matching learn dynamics by regressing the time derivative of a local interpolant placed between consecutive observations. The interpolant defines the regression target, and under sparse, irregular sampling, it underrepresents the dynamics and bounds what can be learned. We show that no fixed interpolant dominates: the best of linear, quadratic, spline and quintic changes with the system and with the sampling density. We propose to learn the interpolant instead. An amortized implicit neural representation (INR) is fitted to the training trajectories and frozen as a teacher supplying the target velocity to a learnable velocity field. We further frame the velocity field within a variational framework, yielding a probabilistic forecaster that, at inference, requires only the velocity field and no SDE solver. We evaluate our method on a range of dynamical systems: it consistently achieves high accuracy and more stable long-horizon rollouts under sparser observations, at matched velocity-field capacity. *Code will be also released upon acceptance.*

1 Introduction

Modelling dynamical systems is challenging across scientific domains, particularly when states evolve in high-dimensional spaces, as in fluid dynamics, climate science, biology, neuroscience, and engineering. Flow Matching-based methods (Lipman et al., 2023; Albergo et al., 2023; Liu et al., 2023; Tong et al., 2024) are continuous-time generative models with the ability to capture complex data distributions. They bypass the need for a numerical solver during training and avoid differentiating through it, which improves memory scalability and makes high-dimensional settings more tractable, whereas simulation-based approaches (Chen et al., 2018; Kidger et al., 2020; Dupont et al., 2019) can be more constrained. Recent simulation-free extensions include Trajectory Flow Matching (Zhang et al., 2024, TFM), proposed for dynamics learning and time-series forecasting.

Unlike standard Flow Matching, TFM learns *state-to-state* dynamics rather than a noise-to-state or noise-to-trajectory transformation (Fu et al., 2025; Huang et al., 2026). State-to-state dynamics preserves temporal causality: the velocity at a query time describes the physics of the real system, not an unrelated velocity artifact induced by a Gaussian probability path from noise to data. This makes TFM a natural framework for non-linear, high-dimensional continuous-time dynamics.

However, real-world observations are often *sparse* and *irregular* in time: a sensor collects measurements at various time points and under diverse conditions when possible, and numerical solvers are run at different step sizes for compute cost. In this regime, trajectory-matching methods rely on an often implicit modelling decision: they construct a probability path between observations and regress its analytical time derivative. The chosen interpolant is not merely a numerical detail: it determines the target velocity used for training and constrains which dynamics can be represented. In particular, the linear interpolant used in TFM provides a first-order approximation of the trajectory between consecutive observations. It is accurate only when the interval is short relative to the curvature of the dynamics. Such an assumption is often violated by real systems that are curved and their

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.

velocity may change between samples. The resulting velocity of TFM is then constant over the interval, whereas the true velocity is not.

This limitation has motivated several variants of TFM for sparse or irregularly sampled data. In this work, we review these methods within a unified framework and investigate how their interpolant assumptions affect the learned dynamics. TFM relies on linear interpolation; later methods replace it with quadratic (Islam et al., 2026, LFM), cubic spline (Rohbeck et al., 2025; Lee et al., 2025, MMFM), or quintic spline (Hou et al., 2026, CFO). These methods can provide more flexible local approximations, but they keep the same underlying assumption: the interpolant is fixed in advance and fitted locally from only a few neighbouring observations. As a result, the learning targets vary with the sampling times, and this dependence can be significant when observations are sparse. Figure 1 illustrates this failure mode.

A complementary line of work incorporates known physical expressions. Physics-informed generative models impose differential-equation constraints as soft regularizers (Sholokhov et al., 2023; Huang et al., 2024; Baldan et al., 2026) for learning forward models, while hybrid models embed known dynamics into the model architecture for forecasting tasks (Takeishi & Kalousis, 2021; Qian et al., 2021; Yin et al., 2021; Takeishi & Kalousis, 2023; Wehenkel et al., 2023; Singh et al., 2026). These approaches operate in a distinct setting: they assume access to, or exploit, explicit physical information. Other relevant work focuses on large-scale PDEs with varying geometries using spatial mesh-free methods, such as Neural Operators (Lu et al., 2021; Li et al., 2021; Xie et al., 2022; Serrano et al., 2023; Li et al., 2023), which learn mappings between functional spaces rather than on discrete fixed grids. In this paper, we study the limitations introduced by fixed interpolation targets in simulation-free methods and propose a solution for learning dynamics from irregularly sampled trajectories.

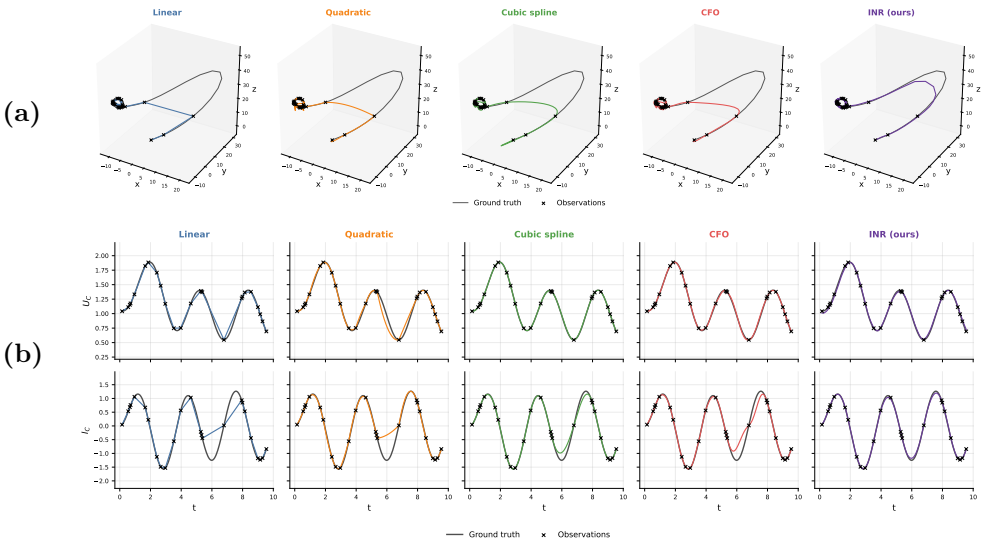

Figure 1: Effect of the interpolant choice at 25% observation density on Lorenz (a) and RLC (b) test sample trajectories. Crosses (x) in the figures are the observations every method receives. A linear interpolant draws straight-line chords across large gaps; other fixed interpolants do better, but still fit the data poorly. Our t-INR approach learns both global and local structure and tracks the trajectory under a sparse regime.

Our Contributions.

- We identify fixed interpolation as the core assumption in simulation-free trajectory matching under sparse, irregular sampling, and show experimentally that no fixed interpolant dominates across systems or sampling regimes.

- We propose to learn the interpolant by an amortized t-INR that imposes no assumption on the spacing between observations. We further show how the learned t-INR can be used as a frozen teacher to provide target velocity.
- We derive a variational velocity field trained from these teacher targets, giving a probabilistic forecaster without an SDE formulation or an SDE solver at inference.
- Across five dynamical systems and one real-world experiment, we show that our method improves the fixed-interpolant baselines, with the largest gains in the sparse regime.

2 Preliminaries

2.1 Problem setting

We are given a collection of trajectories $\mathcal{D} = \{\mathbf{X}^{(i)}\}_{i=1}^{N}$. Each sample $\mathbf{X}^{(i)}$ corresponds to a sequence of observed states $\mathbf{X}^{(i)} = \left(\mathbf{x}_{t_0}^{(i)}, \ldots, \mathbf{x}_{t_T}^{(i)}\right)$, where $\mathbf{x}_{t_k}^{(i)} \in \mathcal{X} \subseteq \mathbb{R}^d$ denotes the state at each discretized time instant t_k. We also denote a segment of a trajectory as $\mathbf{X}_{t_{k-h:k+1}} = \left(\mathbf{x}_{t_{k-h}}, \ldots, \mathbf{x}_{t_k}, \mathbf{x}_{t_{k+1}}\right)$, i.e., a sequence of consecutive points. To reproduce some real scenario case of measurements, we assume points of the trajectory are measured at sparse random (non-equidistant) time points $t_0 < t_1 < ... < t_T$ where $t_k \in \mathbb{R}$, and $T \in \mathbb{N}$. These trajectories are assumed to be independent realisations from an unknown *data* distribution $\pi(\mathbf{X}) \in \mathcal{P}(\mathcal{X}^{T+1})$, where $\mathcal{P}(\mathcal{X}^{T+1})$ is the set of all probability measures over the trajectory space $\mathcal{X}^{T+1}$. The objective is to learn the dynamical system for forecasting tasks in which, given a current state $\mathbf{x}_{t_k}$ and an arbitrary history of the past sparse observations $\mathbf{X}_{t_{k-h:k}}$, generate plausible future trajectories that approximate the true conditional distribution $\pi(\mathbf{X}_{>t_k} \mid \mathbf{X}_{t_{k-h:k}})$ for any future time $t > k$.

2.2 Trajectory Flow Matching

Trajectory Flow Matching (TFM) methods (Zhang et al., 2024; Islam et al., 2026; Rohbeck et al., 2025), extend training simulation-free methods like Conditional Flow Matching (Lipman et al., 2023, CFM) and variants (Albergo et al., 2023; Liu et al., 2023) to dynamical systems where data come as time-series trajectories $\mathbf{X} = (\mathbf{x}_{t_0}, \mathbf{x}_{t_1}, \ldots, \mathbf{x}_{t_T})$. The modelled dynamics is the following Stochastic Differential Equation (SDE):

$$d\mathbf{x}_t = v(\mathbf{x}_t, t)\, dt + g(\mathbf{x}_t, t)\, dW_t, \tag{1}$$

where $v : \mathcal{X} \times [0, T] \mapsto \mathbb{R}^d$ is the drift term, $g : \mathcal{X} \times [0, T] \mapsto \mathbb{R}^{d^2}$ is the diffusion coefficient, and W_t is the d-dimensional Wiener-Brownian process. Under such SDE formulation, when evolving an initial density $p_0(\mathbf{x}_0)$ produces a collection of marginal density $p_t(\mathbf{x_t})$ for $t \in [0, T]$, induced by the Fokker–Planck equation:

$$\partial_t p_t = -\nabla \cdot (p_t\, v_t) + \frac{g^2}{2} \Delta p_t. \tag{2}$$

Rather than mapping from noise-to-data as CFM (where the source distribution p_0 is modelled as a Standard Gaussian distribution), TFM learns state-to-state dynamics, where the velocity field describes the system's temporal evolution. To learn the dynamics in a training simulation-free approach, they first construct a Gaussian probability path between consecutive trajectory states $\mathbf{x}_{t_k}$ and $\mathbf{x}_{t_{k+1}}$ at observed time points (t_k, t_{k+1}). In the linear case, within the work of TFM (Zhang et al., 2024), the probability path is constructed as:

$$p_\tau(\mathbf{x}_\tau \mid \mathbf{X}_{t_{k:k+1}}) = \mathcal{N}\big(\mu_\tau,\, \sigma_\tau^2 \mathbf{1} \;\big|\; \mathbf{X}_{t_{k:k+1}}\big)\,, \tag{3}$$

$$\mu_\tau = \frac{(t_{k+1} - \tau)\, \mathbf{x}_k + (\tau - t_k)\, \mathbf{x}_{k+1}}{t_{k+1} - t_k}, \quad u_\tau = \frac{\mathbf{x}_{k+1} - \mathbf{x}_k}{t_{k+1} - t_k}, \quad \sigma_\tau = \sigma_0\, (\tau - t_k)\, \frac{t_{k+1} - \tau}{t_{k+1} - t_k}.$$

Therefore, learning the velocity field leads to a simple regression problem:

$$\mathcal{L}_{\text{TFM-v}}(\phi) = \mathbb{E}_{\substack{\pi(\mathbf{X}) \\ \pi(\mathbf{X}_{t_{k-h:k+1}} | \mathbf{X}) \\ \tau \sim \mathcal{U}(t_k, t_{k+1})}} \left\| v_\phi(\mathbf{x}_\tau, \tau, \mathbf{X}_{t_{k-h:k}}) - u(\tau; \mathbf{X}_{t_{k:k+1}}) \right\|^2, \tag{4}$$

where $\mathbf{x}_\tau \sim p_\tau(\mathbf{x}_\tau|\mathbf{X}_{t_{k:k+1}})$ and $u(\tau; \mathbf{X}_{t_{k:k+1}})$ are from equation 3, and $\pi(\mathbf{X}_{t_{k-h:k+1}}|\mathbf{X})$ is the empirical distribution of consecutive points, which includes the next, current, and history points. However, the above loss yields a deterministic model. To learn the uncertainty of the dynamics, Zhang et al. (2024) and its variants methods (Islam et al., 2026; Rohbeck et al., 2025; Lee et al., 2025) propose to learn jointly v_ϕ and the diffusion term g_ϕ, optimizing:

$$\mathcal{L}_{\text{TFM}}(\phi) = \mathbb{E}_{\substack{\pi(\mathbf{X}) \\ \pi(\mathbf{X}_{t_{k-h:k+1}}|\mathbf{X}) \\ \tau\sim\mathcal{U}(t_k,t_{k+1})}} \left\| g_\phi(\mathbf{x}_\tau, \tau, \mathbf{X}_{t_{k-h:k}}) - \left\| v_\phi(\mathbf{x}_\tau, \tau, \mathbf{X}_{t_{k-h:k}}) - u(\tau; \mathbf{X}_{t_{k:k+1}}) \right\|^2 \right\|^2 . \tag{5}$$

Based on this formulation, all these variants (Islam et al., 2026; Rohbeck et al., 2025; Lee et al., 2025; Hou et al., 2026) differ mainly in μ_τ with a fixed non-linear interpolation (quadratic or spline, quintic-spline). Instead, we make μ_τ learnable.

3 Method

We formalize our method in the three main building blocks. We first addressed the fixed-interpolant assumption by learning it with a temporal INR (§3.1). We then freeze it and use it as a teacher that supplies the velocity field (§3.2). Finally we make the velocity field variational (§3.3), which gives a probabilistic forecaster that needs no SDE formulation. We also provide the pseudocode of training and inference methods in the App. B.

3.1 Learning the Interpolant via INR

To address the fixed interpolant assumption of TFM and variants, we introduce a learnable functional, capable of interpolating trajectory points under sparse, irregular time-step settings. We achieve this by amortizing a Implicit Neural Representation (Sitzmann et al., 2020, INR) on the temporal axis, which learns a non-linear, smooth interpolation between sparse observations, by optimizing the following simple state $\mathbf{x}_{t_k}$ reconstruction loss:

$$L_{\text{t-INR}}(\xi, \eta) = \mathbb{E}_{\pi(\mathbf{X})} \frac{1}{T} \sum_{k=0}^{T} ||\mathbf{x}_{t_k} - I_\xi(t_k; E_\eta(\mathbf{X}))|| \,, \tag{6}$$

where $\mathbf{X}$ is the observed sequence at discretised sparse points t_k with state $\mathbf{x}_{t_k}$, and the temporal INR (t-INR) model compromises of:

- Encoder $E_\eta : \mathbf{X} \mapsto \mathbf{Z}$. It maps any discretized sequence points to a fixed dimensional latent representation of the trajectory $\mathbf{Z} \in \mathbb{R}^q$ (with $q < d$).
- Temporal Decoder $I_\xi : t \times \mathbf{Z} \mapsto \hat{\mathbf{x}}_t$. It maps any time query point $t \in [0, T]$ to a predicted state $\hat{\mathbf{x}}_t$, using the encoded representation from E_η. The decoder model is inspired by the SiREN architecture (Sitzmann et al., 2020), where the temporal axis is mapped in high-dimensional space using the sine as a periodic activation function capturing temporally varying signals, including high-frequency details.

The t-INR is amortized through an encoder that processes the sparse training data, and smoothness is enforced by the siren-architecture. Thanks to amortization across trajectories and over time states (enabled by the encoder–decoder architecture), the model captures both *local* and *global* patterns in the training trajectories and can generalize better to unobserved time points along a trajectory something that local interpolants cannot achieve.

The t-INR is trained as a pre-training stage before solving the forecasting problem. It will be used to query at any time inside the training time range, so it is not restricted to the observed discrete grid.

3.2 The Interpolant as a Teacher

After training the t-INR, we freeze it and **never** update it again, we denote the frozen parameters with $(\xi^\star, \eta^\star)$ to improve clarity in the text. The t-INR plays the role of a teacher that supplies three quantities: **query state**, **target velocity**, and **history buffer**.

Query times are drawn inside the observed span of each trajectory, $\tau \sim \mathcal{U}(t_0, t_T)$, so the teacher is always evaluated in interpolation. For any such τ it provides the pair of query state and target velocity $(\hat{\mathbf{x}}_\tau, \hat{\mathbf{u}}_\tau)$:

$$\hat{\mathbf{x}}_\tau \leftarrow I_{\xi^\star}(\tau; E_{\eta^\star}(\mathbf{X})), \quad \text{and} \quad \hat{\mathbf{u}}_\tau \leftarrow \partial_\tau I_{\xi^\star}\big(\tau; E_{\eta^\star}(\mathbf{X})\big). \tag{7}$$

We compute $\hat{\mathbf{u}}_\tau$ by efficient automatic differentiation (see App. 5 for more practical details). We remark that our target velocity is not *constant* on an interval and does not depend only on local points, whereas the linear interpolant yields a constant velocity (equation 3).

Because the teacher can be queried anywhere inside the observed span, we do not restrict conditioning to the observed grid: we place the history buffer at h times before τ whose *gaps* are drawn log-uniformly, and read the corresponding states from the teacher,

$$\hat{\mathbf{H}}_\tau = \big(\hat{\mathbf{x}}_{s_1}, \dots, \hat{\mathbf{x}}_{s_h}\big), \;\; \hat{\mathbf{x}}_{s_j} \leftarrow I_{\xi^\star}\big(s_j; E_{\eta^\star}(\mathbf{X})\big), \;\; s_j = \tau - \sum_{i=j}^{h} \delta_i, \;\; \delta_i \sim \log\text{-}\mathcal{U}(\delta_{\min}, \delta_{\max}), \tag{8}$$

so that $s_1 < \cdots < s_h < \tau$ and the buffer is ordered from the oldest to the most recent point. The gaps δ_i are drawn independently, and the two bounds are set from the inter-observation gaps of the training set: $\delta_{\min}$ is the smallest gap and $\delta_{\max}$ three times the median gap. We use the median rather than the largest gap to avoid extreme outliers, which would place buffer points outside the sampling statistics of the data. The log-Uniform between these two bounds spans several orders of magnitude, the model sees both tightly and widely spaced histories, whereas on-grid histories only contain the gaps of the training data. This can be seen as a data augmentation along the time axis, the model sees more varieties of states for the same trajectory. This feature remains comparable to the other baselines methods as the model takes always in input sparse points with the same fixed h-length history. The velocity field receives exactly as many conditioning points as every method we compare against. We ablate this choice against sampling the buffer on the observed grid in App. E.

An important remark should be highlighted: once training is finished, the teacher (t-INR), it is never used, inference time consists only rolling out a single velocity field.

3.3 Variational Velocity Field

Our current velocity field returns only a deterministic output, which is a poor description of a system especially when observations are sparse. Inspired by Guo & Schwing (2025), we introduce a latent variable $\mathbf{z} \in \mathbb{R}^m$ with $m \ll d$, inferred from the history segment using a temporal encoder, then we condition the velocity field on such latent. Such a modelling buys three buys three things at once: predictive uncertainty without committing to an SDE, re-use of the t-INR encoder, the encoder shares its backbone and is warm-started from it, and at inference no SDE solver is involved, since sampling $\mathbf{z}$ is what makes the forecast stochastic.

We model the distribution of the vector field variationally by introducing a variational posterior conditioned on the history segment $q_\psi(\mathbf{z} \mid \mathbf{X}_{t_{k-h:k}})$.

$$\log p_\tau^\phi(\dot{\mathbf{x}}_\tau \mid \mathbf{x}_{t_k}) = \log \int p_\tau^\phi(\dot{\mathbf{x}}_\tau, \mathbf{z} \mid \mathbf{x}_{t_k})\, dz \;\; = \log \int q_\psi(\mathbf{z}|\mathbf{X}_{t_{k-h:k}}) \frac{p_\tau^\phi(\dot{\mathbf{x}}_\tau, \mathbf{z} \mid \mathbf{x}_{t_k})}{q_\psi(\mathbf{z}|\mathbf{X}_{t_{k-h:k}})}\, dz \tag{9}$$

Applying Jensen's Inequality to equation 9, we derive a variational simulation-free matching objective similarly to Guo & Schwing (2025):

$$\log p_\tau^\phi(\dot{\mathbf{x}}_\tau \mid \mathbf{x}_{t_k}) \geq \mathbb{E}_{q_\psi(\mathbf{z}|\mathbf{X}_{t_{k-h:k}})}[\log p_\tau^\phi(\dot{\mathbf{x}}_\tau \mid \mathbf{z}, \mathbf{x}_{t_k})] - \mathrm{KL}\left[q_\psi(\mathbf{z} \mid \mathbf{X}_{t_{k-h:k}}) \| p(\mathbf{z})\right], \tag{10}$$

where $p(\mathbf{z}) = \mathcal{N}(\mathbf{0}, \mathbf{I})$, we report the full derivation in App. A.1.

Taking a Gaussian likelihood for the velocity match, and substituting the teacher's target equation 7, the bound in equation 10 gives the following training objective:

$$\mathcal{L}_{\text{VF-INR}}(\phi, \psi) = \mathbb{E}_{\substack{\pi(\mathbf{X}) \\ \tau \sim \mathcal{U}(t_0, t_T) \\ \hat{\mathbf{H}}_\tau \sim \pi_{\text{buf}}(\cdot|\tau)}} \left[\mathbb{E}_{q_\psi(\mathbf{z}|\hat{\mathbf{H}}_\tau)} \left[\|v_\phi(\hat{\mathbf{x}}_\tau, \mathbf{z}) - \hat{\mathbf{u}}_\tau\|^2\right] + \beta \mathrm{KL}\big(q_\psi(\mathbf{z} \mid \hat{\mathbf{H}}_\tau) \,\|\, p(\mathbf{z})\big)\right], \tag{11}$$

where π_{buf} is the buffer distribution of equation 8, and the query state $\hat{\mathbf{x}}_\tau$, the target velocity $\hat{\mathbf{u}}_\tau$ and the sparse history buffer $\hat{\mathbf{H}}_\tau$ are given from the frozen t-INR as described in §3.2. The KL term is annealed over training.

Inference. The INR teacher is never used (discarded) and the forecast is an ODE rollout of v_ϕ from the test observed h history states, where repeated draws of $\mathbf{z} \sim q_\psi$ give the predictive ensemble, so no SDE solver is involved. See App. B for inference details.

4 Experiments

We evaluate five systems and a real experiment on weather forecasting. Two systems are low-dimensional and non-linear: the Lorenz attractor, chaotic with strong curvature, and the RLC circuit, a 2-D oscillatory dynamic used in biology (Hodgkin & Huxley, 1952) and in electrical engineering. Three systems are spatio-temporal PDEs on a 128×128 grid: Reaction-Diffusion and Shallow Water from PDEBench (Takamoto et al., 2022), and Gray-Scott, a chaotic and non-linear system on 64×64 grid, which we additionally observe on an irregular spatial grid. We subsample observations at irregular intervals with different density ratio, 12.5% (very sparse), 25% (sparse), 50% (dense) and 100% (very dense). At 50% observation density, the sampling frequency is already 4–13× the Nyquist rate across systems (see App. C.3), corresponding to a dense-observation regime. Full details about each experiment, dataset, models architecture and hyperparameters are in App. C-D.

Baselines. All baselines share the formulation of equation 3 and differ in the interpolant: **TFM** (Zhang et al., 2024) (linear), **LFM** (Islam et al., 2026) (quadratic), **MMFM** (Rohbeck et al., 2025; Lee et al., 2025) (cubic spline) and **CFO** (Hou et al., 2026) (quintic spline). Crucially, every method uses the same velocity-field network with the same width, depth and activation, so differences are attributable to the target rather than to capacity; our method carries the additional encoder, which is warm-started from the t-INR.

Evaluation. During test time, the initial condition is set after h irregularly timed historical observations. The forecast is evaluated at 100% density ratio, over a long rollout horizon to assess the error accumulation. We report the relative L_2 error of the rollout against the dense ground truth, cumulated from the start of the forecast over the first third (Early), two thirds (Mid) and the whole horizon (Full). We also report the Continuous Ranked Probability Score (CRPS) of the predictive ensemble, except for CFO which is a deterministic model. More details are reported in App. C.2

Sparse observation forecasting Table 1 reports the main comparison in the sparse regime, at 25% density. In this setting, our approach is best performing across the systems, and the results gap increases with the curvature of the dynamics relative to the sampling. The advantage is greatest when curvature effects are strongest, for example on Lorenz (0.25 vs 0.89), SWE (0.02 vs 0.06) and RD (0.08 vs 0.16). It is smallest for RLC (0.38, similar to CFO), a damped, driven oscillator whose path between two observations is well-approximated by a low-order polynomial. Same ranking is observed on CRPS. We also note that the baseline methods change their relative ranking depending on the system.

Ablating observation density, when does it help? Table 2 varies the retained fraction of observations. At 12.5% and 25% we lead across the systems, particularly on Lorenz, whose dynamics exhibits the strongest curvature. At higher density (50%), on SWE and RD the gap narrows, we perform on par either with MMFM or CFO. This behaviour is expected between the approaches: as intervals shrink, every reasonable interpolant converges to the same path; the regression target stops being the bottleneck. However, we notice that our method is consistent among different densities and dynamics, performing better or on par than others.

Interpolation vs. forecasting. Figure 2 measures each interpolant directly, as MAE against held-out dense frames. No fixed interpolant dominates: the ordering among linear, quadratic, spline and CFO changes with the system and with the retained fraction. We also highlight that although our interpolant on Lorenz at 50% and Reaction-Diffusion do worse, the fixed interpolants interpolate better yet forecast worse. The forecast models regress the time derivative of the interpolant path, not its distance to the observations (reconstruction error), and the interpolant may interpolate all observed points while exhibiting an incorrect slope between points.

Table 1: Forecasting performance at **sparse** observation density (25%). We report full horizon CRPS and relative L_2 error over the full, middle, and early rollout horizons. Relative L_2 is normalized per-window; values are therefore not directly comparable across horizons.

Method	CRPS (Full) ↓	Rel. L_2 (Full) ↓	Rel. L_2 (Mid) ↓	Rel. L_2 (Early) ↓
Lorenz				
TFM (Linear)	1.41 ± 0.20	0.89 ± 0.04	0.91 ± 0.04	0.96 ± 0.08
LFM (Quad)	1.61 ± 1.07	0.92 ± 0.26	1.09 ± 0.35	1.26 ± 0.35
MMFM (Spline)	1.27 ± 0.28	0.89 ± 0.11	0.94 ± 0.19	1.04 ± 0.08
CFO	–	0.89 ± 0.22	1.02 ± 0.28	1.28 ± 0.51
Ours	$\mathbf{0.40} \pm 0.03$	$\mathbf{0.25} \pm 0.09$	$\mathbf{0.22} \pm 0.10$	$\mathbf{0.20} \pm 0.05$
RLC				
TFM (Linear)	0.41 ± 0.02	0.39 ± 0.02	0.54 ± 0.08	0.46 ± 0.04
LFM (Quad)	0.46 ± 0.02	0.39 ± 0.03	0.53 ± 0.04	$\mathbf{0.43} \pm 0.03$
MMFM (Spline)	0.41 ± 0.02	0.42 ± 0.02	0.62 ± 0.06	0.50 ± 0.03
CFO	–	$\mathbf{0.38} \pm 0.01$	0.58 ± 0.05	0.45 ± 0.02
Ours	$\mathbf{0.39} \pm 0.02$	$\mathbf{0.38} \pm 0.03$	$\mathbf{0.49} \pm 0.02$	$\mathbf{0.43} \pm 0.01$
SWE				
TFM (Linear)	0.37 ± 0.04	0.08 ± 0.01	0.06 ± 0.01	0.04 ± 0.01
LFM (Quad)	0.34 ± 0.05	0.08 ± 0.01	0.07 ± 0.01	0.04 ± 0.01
MMFM (Spline)	0.24 ± 0.05	0.06 ± 0.01	0.05 ± 0.01	$0.03 \pm 1e{-3}$
CFO	–	0.07 ± 0.01	0.04 ± 0.01	0.02 ± 0.01
Ours	$\mathbf{0.10} \pm 0.03$	$\mathbf{0.02} \pm 0.01$	$\mathbf{0.01} \pm 1e{-3}$	$\mathbf{0.01} \pm 1e{-3}$
RD				
TFM (Linear)	0.17 ± 0.03	0.16 ± 0.03	0.12 ± 0.01	$0.06 \pm 1e{-3}$
LFM (Quad)	0.22 ± 0.07	0.22 ± 0.06	0.17 ± 0.05	0.08 ± 0.03
MMFM (Spline)	0.24 ± 0.01	0.24 ± 0.01	0.20 ± 0.02	0.10 ± 0.03
CFO	–	0.18 ± 0.05	0.13 ± 0.04	0.06 ± 0.02
Ours	$\mathbf{0.06} \pm 0.01$	$\mathbf{0.08} \pm 1e{-3}$	$\mathbf{0.07} \pm 1e{-3}$	$\mathbf{0.05} \pm 1e{-3}$

Out-of-distribution in time We evaluate each method out of the time training range at 25% density; the results are shown in Figure 3 and in the last row of each block of Table 2. We perform best on all the systems and ranking is stable out of distribution, while the baselines change order relative to one another among the experiments, highlighting that no one dominates among them. Figure 3 resolves the same evaluation by lead time on the non-linear systems of Lorenz, SWE and Gray-Scott: our error grows more slowly and remains stable as the lead-time increases, whereas on Lorenz, TFM degrades sharply and drastically because in a chaotic system with strong curvature the linear interpolation is a poor choice (see visual examples in Figure 1).

Sparsity in space: Gray-Scott We additionally include an experiment on irregular spatial grid for the Gray-Scott dynamical system. In this experiment, we evaluate both

Table 2: Forecasting error across observation densities and OOD: we report the relative L_2 over the full horizon, lower is better.

Dataset (density)	TFM (Linear)	LFM (Quad)	MMFM (Spline)	CFO	Ours
Lorenz (12.5%)	0.99 ± 0.24	1.00 ± 0.35	1.33 ± 0.20	1.06 ± 0.21	$\mathbf{0.31} \pm 0.09$
Lorenz (50%)	0.72 ± 0.02	0.59 ± 0.12	0.59 ± 0.01	0.60 ± 0.10	$\mathbf{0.29} \pm 0.14$
Lorenz (OOD 25%)	1.75 ± 0.82	0.71 ± 0.15	0.76 ± 0.27	0.77 ± 0.17	$\mathbf{0.25} \pm 0.02$
RLC (12.5%)	0.89 ± 0.14	0.86 ± 0.16	0.75 ± 0.09	0.80 ± 0.07	$\mathbf{0.52} \pm 0.07$
RLC (50%)	0.24 ± 0.03	0.26 ± 0.01	0.24 ± 0.01	0.22 ± 0.02	$\mathbf{0.14} \pm 0.01$
RLC (OOD 25%)	0.96 ± 0.23	0.72 ± 0.04	0.73 ± 0.09	0.74 ± 0.06	$\mathbf{0.55} \pm 0.05$
SWE (12.5%)	$0.09 \pm 1e{-3}$	0.09 ± 0.02	$\mathbf{0.07} \pm 0.02$	0.10 ± 0.02	$\mathbf{0.07} \pm 0.01$
SWE (50%)	0.06 ± 0.01	0.06 ± 0.01	0.05 ± 0.01	$\mathbf{0.02} \pm 0.01$	$0.03 \pm 1e{-3}$
SWE (OOD 25%)	0.11 ± 0.01	0.14 ± 0.02	0.10 ± 0.01	0.10 ± 0.04	$\mathbf{0.06} \pm 1e{-3}$
RD (12.5%)	0.25 ± 0.05	0.26 ± 0.04	0.26 ± 0.05	0.15 ± 0.05	$\mathbf{0.13} \pm 0.03$
RD (50%)	0.20 ± 0.05	0.20 ± 0.04	0.19 ± 0.05	$\mathbf{0.13} \pm 0.07$	$\mathbf{0.13} \pm 0.03$
RD (OOD 25%)	0.10 ± 0.02	0.15 ± 0.05	$0.17 \pm 1e{-3}$	0.11 ± 0.04	$\mathbf{0.07} \pm 0.01$

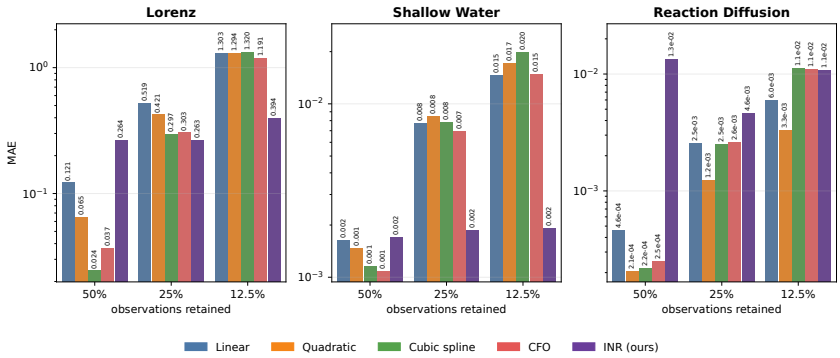

Figure 2: Interpolant reconstruction errors, measured with MAE on held-out dense frames. No fixed interpolant dominates: the ordering between linear, quadratic, spline and CFO changes with the system and with the retained fraction. On Lorenz 50% and Reaction Diffusion systems the fixed interpolants reconstruct better than ours, yet forecast worse, see Table 1-2, what the velocity field regresses is the derivative of the path, not its distance to the observations.

Table 3: Forecast performance Gray-Scott at 25% density ratio (temporal and spatial).

Metric ↓	TFM (Linear)	LFM (Quad)	MMFM (Spline)	CFO	Ours
CRPS (Full)	1.86 ± 0.02	$1.85 \pm 1e{-3}$	$1.86 \pm 1e{-3}$	–	$\mathbf{0.33} \pm 0.08$
Rel. L_2 (Full)	0.38 ± 0.02	0.36 ± 0.01	$0.36 \pm 1e{-3}$	0.16 ± 0.04	$\mathbf{0.12} \pm 0.01$
Rel. L_2 (Mid)	0.29 ± 0.02	0.28 ± 0.01	$0.28 \pm 1e{-3}$	0.09 ± 0.03	$\mathbf{0.07} \pm 1e{-3}$
Rel. L_2 (Early)	0.19 ± 0.01	$0.18 \pm 1e{-3}$	$0.18 \pm 1e{-3}$	0.04 ± 0.01	$\mathbf{0.03} \pm 1e{-3}$

temporal and spatial grids under the sparse setting, 25% density ratio. We motivate this experiment as a robustness stress-test of the methods for evaluating their capabilities in working under adaptive resolution and grid efficiency. Although our main objective is the temporal axis (main target of the paper), our method can extend to the spatial grid by including the $\zeta \in \Omega \subset \mathbb{R}^s$ spatial coordinate in the t-INR, i.e $I_\xi\big(\tau, \zeta; E_\eta(\mathbf{X})\big)$. Table 3 reports similar trends of Table 1, our method surpass all of them, while CFO is best among the baselines. The gap is largest on CRPS, where we improve by more than a factor of five with SDE baselines.

Real data: medium-range weather on ERA5. Every system was so far synthetic. We include an experiment on climate data, ERA5 (Rasp et al., 2020). We choose this setup because ERA5 is archival rather than observational: reanalyses are dense, climate archives are often stored at different temporal resolutions for cost reasons. A common regime for earth observation and remote sensing data. Our aim is not to compete with state-of-the-art weather models, which are trained on dense archives with far larger, domain-specific architectures; our question is narrower: with the velocity field held fixed, we test whether lifting the linear-interpolant assumption helps. We use ERA5 at 50% of a 15-day window

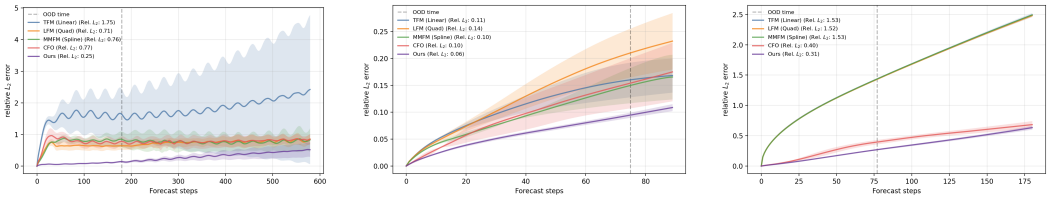

Figure 3: Out-of-distribution Evaluation: rollout forecast error on long-range prediction for Lorenz, Gray-Scott and SWE. The grey line marks the end of the training time horizon.

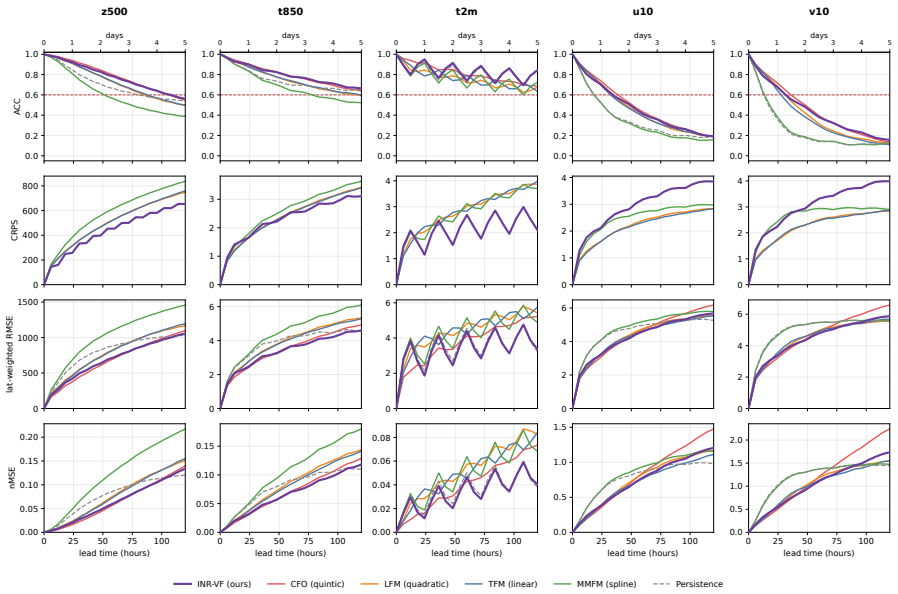

Figure 4: ERA5 skill against lead time of 120 h (5 days). Rows: ACC, CRPS, cos(lat)-weighted RMSE, nMSE; columns: the five variables. The dashed red line marks ACC = 0.6, the conventional limit of a useful forecast. Persistence is the last observation carried forward.

(30 of 60 frames, mean gap 12 h) at random irregular times, on the fixed 5.625° grid with five variables (z500, t850, t2m, u10, v10).

From a buffer of 10 history observations, all models generate a full rollout forecast, scored every 6 h with cos(lat)-weighted RMSE, Anomaly Correlation Coefficient (ACC), CRPS, and nMSE; Figure 4 reports error against lead time up to 120 h (5-day). The linear, quadratic, and cubic interpolants lose ACC quickly when going beyond the 0.6 threshold. This threshold is usually employed to indicate a useful forecast, while our method is considerably more robust when looking at a long time horizon. Wind components are the hardest to forecast: all methods, including ours, drop below around 36 h. CFO is the strongest competing baseline, matching our method closely on this metric, but ours is more consistent across all metrics, including nMSE, where it performs better on wind and t2m. We give in App. G the complete protocol and more results at 234 h and 48 h lead. (Figures 9- 8).

5 Conclusion

Simulation-free trajectory matching is not a universal solution; its efficacy is constrained by the interpolant assumption. In this work, we studied this limitation under sparse observations, where fixed local interpolants do not accurately represent the dynamics. Our experiments highlight that no fixed interpolant best dominates across systems or sampling densities. This indicates that such interpolants remain local approximations based on only a few observations. We address this issue by learning a non-linear interpolant with an amortized temporal INR (t-INR), which captures both local and global structure of the observations. We incorporate the learned interpolant into a simulation-free trajectory-matching method and introduce a variational velocity field for probabilistic forecasting. In our experiments, this approach improves long-horizon rollout accuracy and stability at matched velocity-field capacity, without requiring an SDE solver. Main limitations relate to pre-train the INR, unlike fixed-interpolant baselines, and it is most beneficial for sparse observations and non-linear, curved dynamics. For dense data, linear interpolation may be adequate, at the cost of acquiring measurements at high resolution that capture velocity changes. More generally, interpolants should consider the quality of the derivatives they induce, and not only their state-reconstruction quality.

AI use statement

In this work, we used generative AI tools to assist with implementing parts of the pipeline and baselines in the code, also to provide feedback on and discuss the experimental design, and visualize results. We did **not** use generative AI tools to generate synthetic datasets, develop the theoretical framework, propose or refine hypotheses, formulate mathematical claims, contribute to any part of the derivations, or assist in writing any proofs. Data translation, dataset cleaning and reformatting, and qualitative or thematic data analysis were not applicable to this work. We additionally used generative AI tools to edit and polish the manuscript for readability and conciseness, check the consistency of notation and cross-references, write and refactor analysis and plotting code, generate figures and tables from logged experimental runs. All AI-assisted work was reviewed by the authors. Every sentence was reviewed and revised by the authors; every citation retained in the manuscript was verified against its original source; and all AI-assisted code was verified by the authors. All reported results and numerical values come from experiments executed by the authors using the code provided in the supplementary material. The authors take full responsibility for the final content of this work, including all text, claims, analyses, and other artifacts produced with the assistance of generative AI.

Ethics statement

Time-series tasks, including forecasting and simulating dynamics, are widely used across many scientific and engineering applications. In this work, we propose a new approach for handling observations in which measurements are sparse and irregular, reflecting more realistic real-world conditions. We evaluated the robustness of our method against several baseline approaches on a simulated dataset, following established practices. In addition, we extended our evaluation to a more practical real-world setting in weather forecasting, where temporal measurements are missing. Importantly, this approach can be applied in a broad range of domains, including weather, medicine, military applications, aerodynamics, and industry.

Reproducibility statement

We have taken care to make every result in this paper reproducible. The method section and appendix specify the architecture, the objective, pseudocode training and evaluation steps. We report the complete set of hyper parameters in the appendix for our method and every baseline. The code will be released upon acceptance.

References

Michael S. Albergo, Nicholas M. Boffi, and Eric Vanden-Eijnden. Stochastic interpolants: A unifying framework for flows and diffusions. *arXiv preprint arXiv:2303.08797*, 2023.

Giacomo Baldan, Qiang Liu, Alberto Guardone, and Nils Thuerey. Physics vs distributions: Pareto optimal flow matching with physics constraints. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 101215–101243, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/a57483b394a3654f4317051e4ce3b2b8-Paper-Conference.pdf.

Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems*, 2018.

Emilien Dupont, Arnaud Doucet, and Yee Whye Teh. Augmented neural odes. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/21be9a4bd4f81549a9d1d241981cec3c-Paper.pdf.

Yuxiang Fu, Qi Yan, Lele Wang, Ke Li, and Renjie Liao. Moflow: One-step flow matching for human trajectory forecasting via implicit maximum likelihood estimation based distillation. 2025.

Pengsheng Guo and Alex Schwing. Variational rectified flow matching. In *Forty-second International Conference on Machine Learning*, 2025.

A. L. Hodgkin and A. F. Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of Physiology*, 117(4): 500–544, Aug 1952.

Xianglong Hou, Xinquan Huang, and Paris Perdikaris. CFO: Learning continuous-time pde dynamics via flow-matched neural operators. In *International Conference on Learning Representations*, volume 2026, 2026.

Jiahe Huang, Guandao Yang, Zichen Wang, and Jeong Joon Park. DiffusionPDE: Generative PDE-solving under partial observation. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

Jiahe Huang, Sihan Xu, Sharvaree Vadgama, and Rose Yu. Recursive flow matching. *arXiv preprint arXiv:2605.26535*, 2026.

Mohammad Mohaiminul Islam, Thijs P. Kuipers, Sharvaree Vadgama, Coen de Vente, Afsana Khan, Clara I. Sánchez, and Erik J Bekkers. Longitudinal flow matching for trajectory modeling. In *The 29th International Conference on Artificial Intelligence and Statistics*, 2026.

Patrick Kidger, James Morrill, James Foster, and Terry Lyons. Neural controlled differential equations for irregular time series. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

Justin Lee, Behnaz Moradijamei, and Heman Shakeri. Multi-marginal stochastic flow matching for high-dimensional snapshot data at irregular time points. In *Forty-second International Conference on Machine Learning*, volume 2025, 2025.

Zongyi Li, Nikola Borislavov Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew M. Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. In *International Conference on Learning Representations*. OpenReview.net, 2021.

Zongyi Li, Nikola Borislavov Kovachki, Chris Choy, Boyi Li, Jean Kossaifi, Shourya Prakash Otta, Mohammad Amin Nabian, Maximilian Stadler, Christian Hundt, Kamyar Azizzadenesheli, and Anima Anandkumar. Geometry-informed neural operator for large-scale 3d PDEs. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *International Conference on Learning Representations*, 2023.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *International Conference on Learning Representations*, 2023.

Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3(3):218–229, mar 2021. ISSN 2522-5839.

Zhaozhi Qian, William R. Zame, Lucas M. Fleuren, Paul Elbers, and Mihaela van der Schaar. Integrating expert ODEs into neural ODEs: Pharmacology and disease progression. In *Advances in Neural Information Processing Systems*, 2021.

Stephan Rasp, Peter D. Dueben, Sebastian Scher, Jonathan A. Weyn, Soukayna Mouatadid, and Nils Thuerey. Weatherbench: A benchmark data set for data-driven weather forecasting. *Journal of Advances in Modeling Earth Systems*, 12(11):e2020MS002203, 2020.

Martin Rohbeck, Edward De Brouwer, Charlotte Bunne, Jan-Christian Huetter, Anne Biton, Kelvin Chen, Aviv Regev, and Romain Lopez. Modeling complex system dynamics with flow matching across time and conditions. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 12164–12188, 2025.

Louis Serrano, Lise Le Boudec, Armand Kassaï Koupaï, Thomas X Wang, Yuan Yin, Jean-Noël Vittaut, and patrick gallinari. Operator learning with neural fields: Tackling PDEs on general geometries. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Aleksei Sholokhov, Yuying Liu, Hassan Mansour, and Saleh Nabi. Physics-informed neural ode (pinode): embedding physics into models using collocation points. *Scientific Reports*, 13(1):10166, Jun 2023. ISSN 2045-2322.

Gurjeet Sangra Singh, Frantzeska Lavda, Giangiacomo Mercatali, and Alexandros Kalousis. Variational grey-box dynamics matching. In *The 29th International Conference on Artificial Intelligence and Statistics*, 2026.

Vincent Sitzmann, Julien N.P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In *Proc. NeurIPS*, 2020.

Makoto Takamoto, Timothy Praditia, Raphael Leiteritz, Dan MacKinlay, Francesco Alesiani, Dirk Pflüger, and Mathias Niepert. PDEBench: An Extensive Benchmark for Scientific Machine Learning. In *36th Conference on Neural Information Processing Systems (NeurIPS 2022) Track on Datasets and Benchmarks*, 2022.

Naoya Takeishi and Alexandros Kalousis. Physics-integrated variational autoencoders for robust and interpretable generative modeling. In *Advances in Neural Information Processing Systems*, volume 34, pp. 14809–14821, 2021.

Naoya Takeishi and Alexandros Kalousis. Deep grey-box modeling with adaptive data-driven models toward trustworthy estimation of theory-driven models. In *International Conference on Artificial Intelligence and Statistics*, volume 206, pp. 4089–4100, 2023.

Alexander Tong, Kilian FATRAS, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Guy Wolf, and Yoshua Bengio. Improving and generalizing flow-based generative models with minibatch optimal transport. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856.

Antoine Wehenkel, Jens Behrmann, Hsiang Hsu, Guillermo Sapiro, Gilles Louppe, and Joern-Henrik Jacobsen. Robust hybrid learning with expert augmentation. *Transactions on Machine Learning Research*, 2023.

Yiheng Xie, Towaki Takikawa, Shunsuke Saito, Or Litany, Shiqin Yan, Numair Khan, Federico Tombari, James Tompkin, Vincent sitzmann, and Srinath Sridhar. Neural fields in visual computing and beyond. *Computer Graphics Forum*, 41(2):641–676, 2022.

Yuan Yin, Vincent Le Guen, Jérémie Dona, Emmanuel de Bézenac, Ibrahim Ayed, Nicolas Thome, and Patrick Gallinari. Augmenting physical models with deep networks for complex dynamics forecasting. *Journal of Statistical Mechanics: Theory and Experiment*, 2021(12):124012, 2021.

Xi Zhang, Yuan Pu, Yuki Kawamura, Andrew Loza, Yoshua Bengio, Dennis L. Shung, and Alexander Tong. Trajectory flow matching with applications to clinical time series modelling. In *Advances in Neural Information Processing Systems*, volume 37, pp. 107198–107224, 2024.

A Derivations

A.1 ELBO Derivation

$$\log p_\tau^\phi(\dot{\mathbf{x}}_\tau \mid \mathbf{x}_{t_k}) = \log \int p_\tau^\phi(\dot{\mathbf{x}}_\tau, \mathbf{z} \mid \mathbf{x}_{t_k})\, d\mathbf{z} \tag{12}$$

$$= \log \int q_\psi(\mathbf{z} \mid \mathbf{X}_{t_{k-h:k}}) \frac{p_\tau^\phi(\dot{\mathbf{x}}_\tau, \mathbf{z} \mid \mathbf{x}_{t_k})}{q_\psi(\mathbf{z} \mid \mathbf{X}_{t_{k-h:k}})}\, d\mathbf{z} \tag{13}$$

$$\geq \int q_\psi(\mathbf{z} \mid \mathbf{X}_{t_{k-h:k}}) \log \frac{p_\tau^\phi(\dot{\mathbf{x}}_\tau, \mathbf{z} \mid \mathbf{x}_{t_k})}{q_\psi(\mathbf{z} \mid \mathbf{X}_{t_{k-h:k}})}\, d\mathbf{z} \quad \text{(Jensen's inequality)} \tag{14}$$

$$= \int q_\psi(\mathbf{z} \mid \mathbf{X}_{t_{k-h:k}}) \log \frac{p_\tau^\phi(\dot{\mathbf{x}}_\tau \mid \mathbf{z}, \mathbf{x}_{t_k}) p_\tau^\phi(\mathbf{z} \mid \mathbf{x}_{t_k})}{q_\psi(\mathbf{z} \mid \mathbf{X}_{t_{k-h:k}})}\, d\mathbf{z} \tag{15}$$

$$= \mathbb{E}_{q_\psi(\mathbf{z} \mid \mathbf{X}_{t_{k-h:k}})} \left[\log p_\tau^\phi(\dot{\mathbf{x}}_\tau \mid \mathbf{z}, \mathbf{x}_{t_k}) - \log \frac{q_\psi(\mathbf{z} \mid \mathbf{X}_{t_{k-h:k}})}{p_\tau^\phi(\mathbf{z} \mid \mathbf{x}_{t_k})} \right] \tag{16}$$

$$= \mathbb{E}_{q_\psi(\mathbf{z} \mid \mathbf{X}_{t_{k-h:k}})} [\log p_\tau^\phi(\dot{\mathbf{x}}_\tau \mid \mathbf{z}, \mathbf{x}_{t_k})] - \mathrm{KL}\left[q_\psi(\mathbf{z} \mid \mathbf{X}_{t_{k-h:k}}) \| p_\tau^\phi(\mathbf{z} \mid \mathbf{x}_{t_k}) \right] \tag{17}$$

Assuming independent and data and time independent priors:

$$p_\tau^\phi(\mathbf{z} \mid \mathbf{x}_{t_k}) = p(\mathbf{z}) \tag{18}$$

The objective becomes:

$$\log p_\tau^\phi(\dot{\mathbf{x}}_\tau \mid \mathbf{x}_{t_k}) \geq \mathbb{E}_{q_\psi(\mathbf{z} \mid \mathbf{X}_{t_{k-h:k}})} [\log p_\tau^\phi(\dot{\mathbf{x}}_\tau \mid \mathbf{z}, \mathbf{x}_{t_k})] - \mathrm{KL}\left[q_\psi(\mathbf{z} \mid \boldsymbol{X}_{k-h:k}) \| p(z) \right], \tag{19}$$

B Method Details

This section summarises the training and the inference procedures of our method. Algorithm 1 describes the training of the variational velocity field, and assumes that the t-INR of §3.1 is already trained and frozen, since the pre-training of the t-INR only involves a simple reconstruction loss of §3.1. Algorithm 2 describes the forecast at inference time, where no t-INR is involved: the rollout only uses the encoder and the velocity field.

Algorithm 1: Training the variational velocity field from the frozen t-INR teacher

Input: $\mathcal{D}$; frozen t-INR $(\xi^\star, \eta^\star)$; h; $\delta_{\min}, \delta_{\max}$; β
Output: v_ϕ, q_ψ

warm-start ψ from $E_{\eta^\star}$
repeat
  $\mathbf{X} \sim \mathcal{D}$,
  $\mathbf{Z} \leftarrow E_{\eta^\star}(\mathbf{X})$,
  $\tau \sim \mathcal{U}(t_0, t_T)$
  $\hat{\mathbf{x}}_\tau, \hat{\mathbf{u}}_\tau \leftarrow I_{\xi^\star}(\tau; \mathbf{Z}), \partial_\tau I_{\xi^\star}(\tau; \mathbf{Z})$ `// equation 7`
  $\delta_{1:h} \sim \log\text{-}\mathcal{U}(\delta_{\min}, \delta_{\max})$,
  $s_j \leftarrow \tau - \sum_{i=j}^{h} \delta_i$
  $\hat{\mathbf{H}}_\tau \leftarrow I_{\xi^\star}(s_{1:h}; \mathbf{Z})$ `// equation 8`
  $\mathbf{z} \sim q_\psi(\cdot \mid \hat{\mathbf{H}}_\tau)$ `// reparametrised`
  $\mathcal{L} \leftarrow \|v_\phi(\hat{\mathbf{x}}_\tau \mid \mathbf{z}) - \hat{\mathbf{u}}_\tau\|^2 + \beta\, \mathrm{KL}(q_\psi \| p)$
  $(\phi, \psi) \leftarrow$ gradient step on $\mathcal{L}$
until *converged*
return v_ϕ, q_ψ `// discard the t-INR`

Algorithm 2: Forecasting at inference time: rolling encoder, no t-INR

Input: observed buffer $\mathbf{H} = (\mathbf{x}_{t_{k-h}}, \dots, \mathbf{x}_{t_{k-1}})$,
$\mathbf{x}_{r_0} = \mathbf{x}_{t_k}$,
forecast times $r_0 < r_1 < \cdots < r_M$,
trained v_ϕ, q_ψ
Output: forecast $\mathbf{x}_{r_1:r_M}$

for $i = 1$ **to** M **do**
  $\mathbf{z} \sim q_\psi(\cdot \mid \mathbf{H})$ `// re-encode the current buffer`
  $\mathbf{x}_{r_i} \leftarrow \mathbf{x}_{r_{i-1}} + \int_{r_{i-1}}^{r_i} v_\phi(\cdot \mid \mathbf{z})\, dt$ `// ODE step, no SDE solver`
  $\mathbf{H} \leftarrow (\mathbf{H}_{2:h}, \mathbf{x}_{r_{i-1}})$ `// FIFO: drop the oldest, append`
return $\mathbf{x}_{r_1:r_M}$ `// fresh z per ensemble member`

Buffer update at inference. For readability, Algorithm 2 updates the history buffer after every integration step. In practice, the buffer is updated on a *macro step* Δ_{macro}, set to the median inter-observation gap of the training set, and the encoder is re-evaluated after each update. The forecast is evaluated on the dense grid, and refreshing the buffer at every step of the ODE solver would present the velocity field with histories far denser than the ones seen during training. The macro step is used for our method and for its on-grid ablation (App. E). In the ablations, all the baselines are also rolled out with the same macro-step update, so that every method receives its history buffer at the same cadence. The macro step is most relevant in the sparse and very sparse settings, where the gap between the training observations and the dense forecast grid is the largest.

Computing the target velocity. The target velocity $\hat{\mathbf{u}}_\tau = \partial_\tau I_{\xi^\star}(\tau; E_{\eta^\star}(\mathbf{X}))$ of equation 7 is the derivative of the decoder output with respect to a single scalar input, the query time τ. The encoder output $E_{\eta^\star}(\mathbf{X})$ does not depend on τ, so it is computed once per trajectory and reused for all the query times, and only the decoder is differentiated. Since the t-INR is frozen, no computational graph is kept: the target is evaluated without tracking gradients with respect to the parameters, and enters the loss as a constant. We use two implementations, depending on the dimension of the state. For the low-dimensional systems, we use reverse-mode automatic differentiation: for each state dimension, we back-propagate the sum of that dimension over all the query times to τ. Since each query only depends on its own time, this gives the exact derivative of every query in one backward pass per state dimension, i.e., two or three passes. For the spatio-temporal systems and ERA5, the state has $C \times H \times W$ dimensions, and reverse mode would require one backward pass per dimension. We instead use forward-mode automatic differentiation, i.e., a Jacobian-vector product with a unit tangent on τ, which returns the state and its exact time derivative on the whole field in a single forward evaluation, at a cost of about two decoder passes. In both cases, the t-INR operates on min-max normalised time and states, and the target is expressed in physical units by the chain rule through the two normalisations, $\hat{\mathbf{u}}_\tau = \frac{x_{\max} - x_{\min}}{t_{\max} - t_{\min}}\, \partial_{\tilde{\tau}}\, \tilde{I}_{\xi^\star}(\tilde{\tau})$, where the tilde denotes normalised quantities. In reverse mode, this factor is included automatically by differentiating through the normalisations; in forward mode, it is applied explicitly to the output of the Jacobian-vector product.

C Experimental Setup

We evaluate five synthetic systems and a real-world weather forecasting task, which is described separately in App. G. Two systems are low-dimensional and non-linear: the Lorenz attractor, chaotic with strong curvature, and the RLC circuit, a 2-D oscillatory system. Three systems are spatio-temporal PDEs: Reaction-Diffusion and Shallow Water, following the work of Hou et al. (2026) and downloading them from PDEBench (Takamoto et al., 2022) on a 128×128 grid, and Gray-Scott on a 64×64 grid, which we additionally observe on an irregular spatial grid.

C.1 Datasets

For every system, the trajectories are first simulated on a dense and regular time grid, which defines the 100% density ratio. The training observations are then obtained by retaining 12.5%, 25% or 50% of the dense grid at irregular times, drawn independently for each trajectory. The validation and test trajectories uses the full resolution, and are subsampled only at evaluation time (App. C.2).

RLC. The RLC series circuit, consisting of a resistor (R), capacitor (C), and inductor (L), models a broad class of transfer functions. The system state at time t is defined as $\mathbf{x}_t = [I_t, U_t]^\top$, where U_t denotes the capacitor voltage and I_t the circuit current. The state evolution follows:

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix} I_t \\ U_t \end{bmatrix} = \begin{bmatrix} \frac{1}{L}(V_t - U_t) \\ \frac{I_t}{C} \end{bmatrix} + \begin{bmatrix} -\frac{R}{L} I_t \\ 0 \end{bmatrix}. \tag{20}$$

The dataset is generated by sampling L and C uniformly from $[1, 3] \times [0.5, 1.5]$, and $R \sim \mathcal{U}(1, 3)$. The input voltage V_t is a fixed, known signal. Initial conditions are given by $U_0 \sim \mathcal{N}(0, 1)$ and $I_0 = 0$. Each trajectory is simulated from $t_0 = 0$ to $t_{\max} = 10$ seconds with a time resolution of $\Delta t = 0.05\,\mathrm{s}$, yielding 200 time points at 100% density ratio per trajectory. The training, validation, and test sets contain respectively 500, 125 and 125 trajectories.

Lorenz. We analyse the Lorenz system for varying initial conditions and parameters. The system dynamics are governed by:

$$\begin{aligned} \frac{\mathrm{d}u}{\mathrm{d}t} &= \sigma(v - u), \\ \frac{\mathrm{d}v}{\mathrm{d}t} &= u(\rho - w) - v, \\ \frac{\mathrm{d}w}{\mathrm{d}t} &= uv - \beta w. \end{aligned} \tag{21}$$

We generate the trajectories with $u(0), v(0), w(0)$ sampled from a Gaussian distribution, and the parameters are also randomly sampled with $\sigma \sim \mathcal{U}(9.5, 10.5)$, $\rho \sim \mathcal{U}(27.0, 29.0)$ and $\beta \sim \mathcal{U}(2.6, 2.8)$. Each trajectory spans the time interval $t_0 = 0$ to $t_{\max} = 4$ with a time resolution of $\Delta t = 0.02$, which gives 200 states per trajectory at 100% density ratio. The training set is made of 500 trajectories, while the validation and test sets consist of 100 trajectories each.

Shallow Water (PDEBench). We use the 2-D radial dam-break problem of PDEBench (Takamoto et al., 2022), whose state is the water height on a 128×128 grid. Each trajectory is stored with 100 frames, of which we keep the first 90 as the dense grid at 100% density ratio. The 1000 trajectories are shuffled and split into 800, 100 and 100 trajectories for training, validation and test.

Reaction-Diffusion (PDEBench). We use the 2-D diffusion-reaction problem of PDEBench (Takamoto et al., 2022), which couples two non-linear fields, an activator and an inhibitor, on a 128×128 grid. As for Shallow Water, we keep the first 90 of the 100 stored frames, and split the 1000 trajectories into 800, 100 and 100 trajectories.

Gray-Scott. The Gray-Scott model describes two species u and v that react and diffuse,

$$\partial_t u = \nu_u \Delta u - uv^2 + F(1 - u), \qquad \partial_t v = \nu_v \Delta v + uv^2 - (F + k)\, v. \tag{22}$$

We simulate it with a spectral solver on a 64×64 periodic grid, with $\nu_u = 2 \cdot 10^{-5}$ and $\nu_v = 10^{-5}$, and initial conditions given by random Gaussian blobs. The feed and kill rates are sampled for each trajectory as $F \sim \mathcal{U}(0.030, 0.045)$ and $k \sim \mathcal{U}(0.060, 0.064)$. Each trajectory has 101 frames at $\Delta t = 1$. The training, validation and test sets contain 500, 125 and 125 trajectories. In addition to the temporal subsampling, each observed frame retains 25% of its pixels, at random positions that change from frame to frame.

Out-of-distribution test sets. To evaluate the forecast beyond the time range seen during training, we build for each system an additional test set whose trajectories extend past the training horizon, while the physical parameters and the initial conditions follow the same distributions as in training. For Lorenz, we regenerate the 100 test trajectories with the same random seed, and integrate them up to $t_{\max} = 12$, three times the training span (600 dense points at $\Delta t = 0.02$), retaining 150 of them (25%). The in-distribution test trajectories are thus the first third of the out-of-distribution ones. For RLC, the 125 test circuits are simulated in the same way up to $t_{\max} = 30\,\mathrm{s}$, three times the training span (598 dense points), retaining 150 of them (25%). For Shallow Water and Reaction-Diffusion, the PDEBench trajectories contain 100 frames, of which only the first 90 are used for training. The out-of-distribution test set uses the same 100 test trajectories over all their 100 frames, so the final frames lie beyond the training time range, and 22 frames are retained at irregular times. For Gray-Scott, the 125 test trajectories are simulated with the same seed for twice as long (201 frames over $[0, 3000]$), retaining 25 observations. In all cases, the history and the initial condition are taken as in the in-distribution evaluation (Appendix C.2), and the forecast is rolled out until the end of the extended trajectory, so that its later part lies outside the time range of the training data.

C.2 Evaluation protocol

All methods are evaluated on the same held-out test trajectories, which are available at the full (100%) temporal resolution. For each trajectory, the history is formed by h **sparse** observations taken from a stride of the dense grid, chosen to match the sparsity of the training data. We use $h = 10$ in the sparse and dense settings and $h = 5$ in the very sparse setting, identically for every method, including the spatio-temporal experiments. The initial condition is the observation that immediately follows the history, and the forecast is then evaluated on every remaining point of the dense grid.

We report the relative L_2 error of the rollout against the dense ground truth, accumulated from the start of the forecast over the first third (Early), the first two thirds (Mid) and the whole horizon (Full) of the scored window. The probabilistic methods are additionally evaluated with the Continuous Ranked Probability Score (CRPS), computed from an ensemble of 20 forecasts: the SDE baselines draw fresh diffusion noise for each member, while our method draws a new latent $\mathbf{z} \sim q_\psi$. CFO is a deterministic model, and its CRPS is therefore not reported. For the out-of-distribution evaluation, the forecast starts from initial times that lie outside the time range seen during training. All results are reported as mean and standard deviation over four training seeds for the small dimension problems (RLC, and Lorenz) and three training seeds for the high-dimensional problems.

C.3 Temporal bandwidth of the synthetic systems – Nyquist rate

The density labels used in the experiments (100%, 50%, etc.) are fractions of the simulation's time grid. To make these labels meaningful, we measure the temporal bandwidth of each system and re-express the densities in the unit that the Nyquist–Shannon theorem uses: samples per cycle of the fastest-varying component of the trajectory.

Evaluation. For each system, we take the dense trajectories and, for every state variable, look at how its value fluctuates over time (subtracting its average so we isolate the fluctuations). We compute the frequency content of these fluctuations and average the result across trajectories, and for the PDE systems, across 256 randomly chosen spatial locations, since each point in space evolves as its own time series. From this, we extract f_q: the frequency below which 99% of the signal's variation is contained, i.e., the fastest frequency that meaningfully matters. We convert f_q into samples per cycle,

$$\text{samples per cycle}(d) \;=\; \frac{d}{f_q\,\Delta t_{\text{dense}}}, \tag{23}$$

where d is the retained fraction of the grid. By the Nyquist-Shannon theorem, at least 2 samples per cycle are needed to resolve a signal; below that, the corresponding dynamics can no longer be recovered from the samples.

Table 4: Samples per cycle of the fastest relevant frequency (f_q), for each retained fraction of the time grid. At 50%, every system is still oversampled 4–13×; at 100%, 8–25× – which is why we call these two budgets dense and very dense.

System	100%	50%	25%	12.5%
Lorenz	15.4	7.7	3.9	1.9
RLC	50.5	25.2	12.6	6.3
SWE	20.6	10.3	5.2	2.6
Reaction-Diffusion	47.6	23.8	11.9	6.0
Gray-Scott	21.7	10.8	5.4	2.7

Results. Table 4 makes one point clearly: our whole sweep is already in the dense regime. At 50%, every system keeps 4–13× more samples than needed to resolve its dynamics; at 100%, 8–25× more. So going from 50% to 100% adds no new information about the dynamics, only redundant samples, which in practice are exactly the samples that are expensive or hard to obtain.

Second, the same percentage does not mean the same thing in every system. At 25% Lorenz keeps 3.9 samples per cycle while RLC keeps 12.6, so the two systems are in rather different regimes despite carrying the same label. It is useful to note that this ordering follows the ordering of our margins in Table 1: we gain the most where the sampling is closest to the floor, and the least where the grid is still comfortable.

Finally, only Lorenz at 12.5% actually drops below the floor, at 1.9 samples per cycle. We flag this as a possible undersampled case, but only as a warning sign, not as proof of aliasing. Our actual point is the opposite: fixed interpolants already lose accuracy at 50%, where by this measure sampling is more than sufficient.

D Models and Hyperparameters

D.1 t-INR

The t-INR is pre-trained once per system, on the training observations only, with the reconstruction loss of §3.1. We use AdamW with a learning rate of 10^{-4}, and select the checkpoint with the lowest reconstruction error on the validation set. We use two architectures, depending on the dimension of the state.

Low-dimensional systems (Lorenz, RLC). The encoder E_η is a lightweight Transformer (3 layers, 8 heads, width 256) over the observed pairs $(t_k, \mathbf{x}_{t_k})$, which returns a latent sequence of dimension 64. The decoder I_ξ embeds the query time with a SIREN (4 layers, width 128, $\omega_0 = 10$) and attends to the latent sequence with cross-attention to predict the state.

Spatio-temporal systems (Shallow Water, Reaction-Diffusion, Gray-Scott). The encoder combines factorised (2+1)-D convolutional blocks, which downsample each frame, with a Transformer (4 layers, 8 heads, width 256) along time. The decoder queries this sequence with the same SIREN cross-attention ($\omega_0 = 10$), across all the coordinates, and maps the resulting context vector to the full field with a convolutional decoder that mirrors the encoder, with circular padding. Reconstruction loss is computed on the observed pixels.

Segments. A spatio-temporal trajectory can be too large to be encoded at once within the memory of a single GPU. As is common practice, the t-INR is therefore trained on segments of consecutive observations: the encoder takes one segment as input, and the decoder interpolates within the time span of that segment. We use segments of 22 observations for Shallow Water and Reaction-Diffusion, 20 for Gray-Scott and 30 for ERA5; a trajectory with more observations is split into consecutive segments. The history buffer of the velocity field ($h = 10$ observations) remains shorter than a segment.

D.2 Velocity field

Low-dimensional systems. The velocity field v_ϕ is an MLP with four hidden layers of width 128 and SELU activations, and the query time is embedded with 10 Fourier frequencies.

Spatio-temporal systems. The velocity field is a residual convolutional network (32 base channels, 2 residual blocks per stage) that maps the state field to the velocity field. The time embedding and the conditioning vector are combined by an MLP of width 256, and injected at every stage by feature-wise linear modulation (FiLM).

For all systems, the posterior q_ψ shares the backbone of the t-INR encoder, from which it is warm-started, and outputs the mean and the log-variance of $\mathbf{z}$.

D.3 Baselines

All baselines share the formulation of equation 3 and differ only in the interpolant: **TFM** (Zhang et al., 2024) (linear), **LFM** (Islam et al., 2026) (quadratic), **MMFM** (Rohbeck et al., 2025; Lee et al., 2025) (cubic spline) and **CFO** (Hou et al., 2026) (quintic spline). They use the same velocity-field network, the same history length h and the same optimisation settings as our method (Table 5), so that differences are attributable to the regression target rather than to capacity. TFM, LFM and MMFM additionally learn the diffusion term g_ϕ of equation 5 with a small separate network: an MLP with two hidden layers of width 64 for the low-dimensional systems, and a convolutional network with 16 channels for the spatio-temporal ones. For these three baselines, we sweep the diffusion coefficient $\sigma_0 \in \{0, 0.01, 0.05\}$ and keep the best value for each experiment and ablation.

For the Gray-Scott experiment, we adapted the interpolants, which are defined pointwise in time and since they do not have mechanisms to handle a pixel that is missing in the frames they interpolate between. We therefore fill the missing pixels of each observed frame with the mean of the observed pixels in a 3×3 neighbourhood, before the interpolation. The same fill is applied to all four interpolants, so it favours none of them.

D.4 Hyperparameters

Table 5 reports the main hyperparameters of our method for each system. All models are trained with AdamW, with a weight decay of $5 \cdot 10^{-7}$ for the low-dimensional systems and 10^{-6} for the spatio-temporal ones. The KL weight β is annealed linearly from 0 over the first 1000 steps. For every trajectory, we draw 10 query times per training step. The bounds of the log-uniform buffer gaps are set from the training data (§3.2): $\delta_{\min}$ is the smallest inter-observation gap and $\delta_{\max}$ three times the median gap. For the spatio-temporal systems, we additionally clip the gradient norm to 1 and use a free-bits threshold of 0.05 on the KL term. We use early stopping on the validation error over the Early horizon; for ERA5, the model is trained for a fixed number of epochs.

Table 5: Main hyperparameters of our method. The baselines use the same velocity-field network, history length and optimisation settings. MLP: 4 hidden layers of width 128; CNN: residual convolutional network, with base channels / residual blocks.

	Lorenz	RLC	SWE	RD	Gray-Scott	ERA5
t-INR encoder	Transformer	Transformer	(2+1)-D CNN	(2+1)-D CNN	(2+1)-D CNN	(2+1)-D CNN
Velocity field	MLP	MLP	CNN 32/2	CNN 32/2	CNN 32/2	CNN 64/3
Latent dim. m	32	8	64	64	64	128
History length h	10	10	10	10	10	10
VF learning rate	10^{-3}	$5 \cdot 10^{-4}$	10^{-4}	10^{-4}	10^{-4}	10^{-4}
Encoder learning rate	10^{-4}	$3 \cdot 10^{-5}$	10^{-5}	10^{-5}	10^{-5}	10^{-5}
KL weight β	0.1	0.1	0.1	0.1	0.1	0.1
Batch size	25	25	10	10	10	16
Max. epochs	2000	2000	2000	2000	2000	2000
ODE solver	RK4	Dopri5	RK4	RK4	RK4	RK4

Table 6: Ablation on the history-buffer sampling strategy for the INR: log-uniform gaps before τ vs. sampling from the on-grid discrete buffer, across training history densities.

Dataset	History sampling	Training observations density		
		12.5% ↓	25% ↓	50% ↓
RLC	Log-uniform	**0.52** ± 0.07	**0.38** ± 0.03	**0.14** ± 0.01
	On-grid buffer	0.85 ± 0.04	1.03 ± 0.02	**0.13** ± 0.01
Lorenz	Log-uniform	**0.31** ± 0.09	**0.25** ± 0.09	**0.29** ± 0.14
	On-grid buffer	0.33 ± 0.08	0.30 ± 0.08	0.53 ± 0.03
Shallow Water	Log-uniform	**0.07** ± 0.01	**0.02** ± 0.01	0.03 ± 1e−3
	On-grid buffer	0.08 ± 1e−3	0.06 ± 1e−3	**0.03** ± 1e−3
Reaction-Diffusion	Log-uniform	**0.13** ± 0.03	**0.08** ± 1e−3	0.13 ± 0.03
	On-grid buffer	0.15 ± 1e−3	0.10 ± 1e−3	**0.12** ± 0.01

D.5 Compute resources

Each model is trained on a single GPU. For the low-dimensional systems (Lorenz, RLC), training one model for one seed takes at most about 4 hours on a GPU with 12 GB of memory. For the spatio-temporal systems and ERA5, training one model for one seed takes about 12 hours on a GPU with at least 32 GB of memory, and up to about 50 GB for the largest configurations. These costs are approximate and apply to each seed.

E Ablation: how the history buffer is sampled

Because the frozen teacher can be queried anywhere, we are free to choose where the history buffer is placed during training, and the choice matters more than we expected. Sampling buffer times with log-uniform gaps is compared in Table 6 against taking the buffer on the observed grid. Log-uniform gaps expose the velocity field to buffer points spanning several orders of magnitude, including close points and widely spaced buffers, whereas an on-grid buffer is always as dense as the training data and leaves the model unprepared for them. Notably, the ordering can reverse at high density, where the on-grid buffer can be a better choice.

F Forecast visualisations

We show exmaples of test trajectories forecast at 25% density by our method and by the strongest baseline of each spatio-temporal system in Table 1. Steps are counted from the start of the forecast.

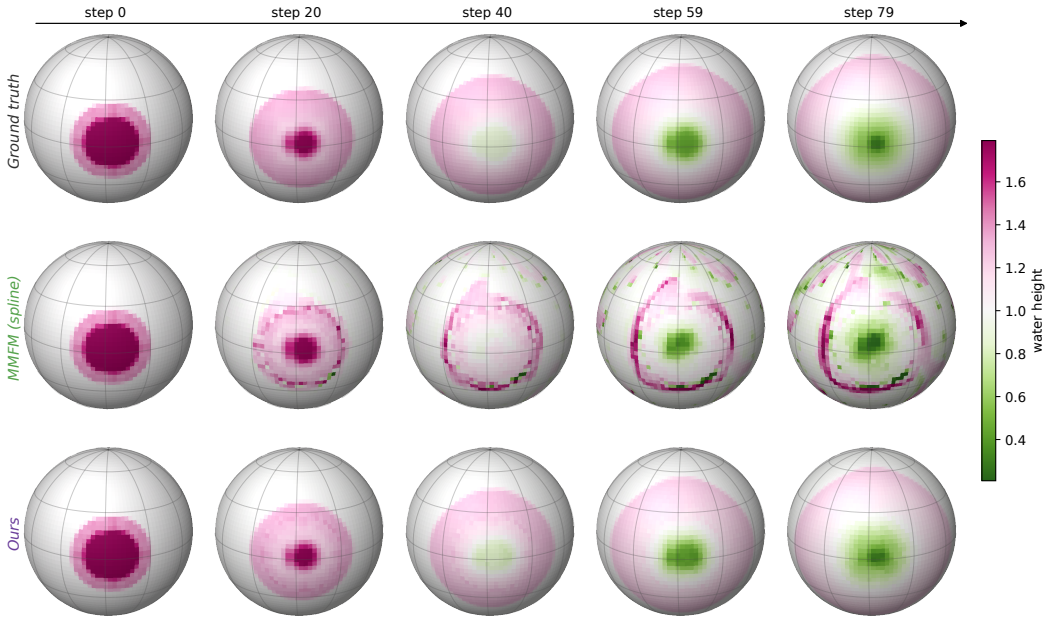

Figure 5: Shallow Water at 25% density: ground truth, MMFM (the strongest baseline) and our method on a test trajectory. For readability the square domain is drawn on a sphere, and the colours show the water height around its rest level.

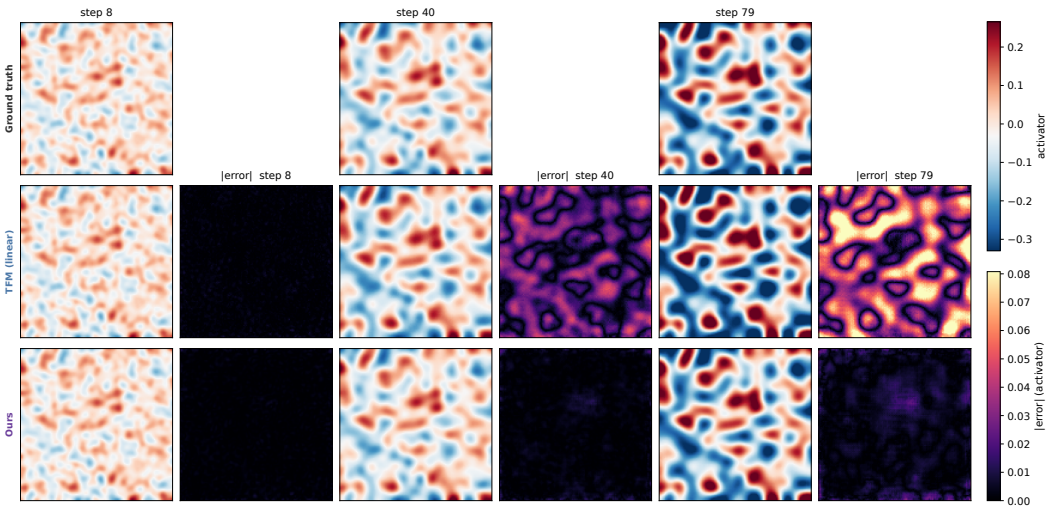

Figure 6: Reaction-Diffusion at 25% density: activator field of one test trajectory at steps 8, 40 and 79, for the ground truth, TFM (the strongest baseline) and our method, with the absolute error next to each forecast. The error of TFM grows over the whole domain as the pattern sharpens, whereas the error of our method remains close to zero over the horizon. Instead our method is able to capture those high frequency details.

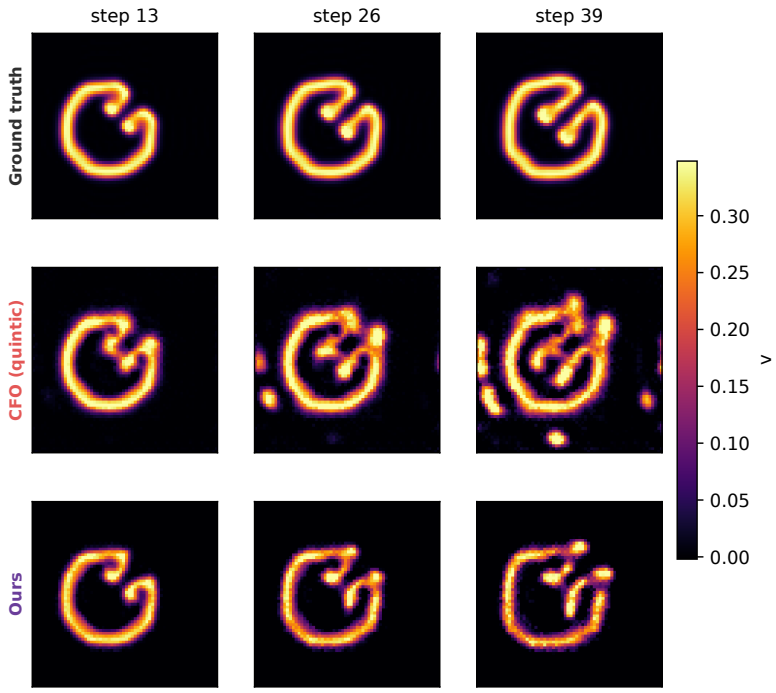

Figure 7: Gray-Scott at 25% density in time and space: concentration v of one test trajectory at steps different steps, for the ground truth, CFO (the strongest baseline) and our method.

G Weather forecasting on ERA5

Dataset. We use ERA5 reanalysis on the WeatherBench (Rasp et al., 2020) 5.625° grid (32×64), with five variables: geopotential at 500 hPa (z500), temperature at 850 hPa (t850), 2 m temperature (t2m), and the two 10 m wind components (u10, v10). The data are taken at a 6 h cadence and cut into non-overlapping windows of 15 days (60 frames). The years 1979–2015 are used for training, 2016 for validation and 2017–2018 for test, which gives 900, 24 and 48 windows. In each training window we keep 50% of the frames (30 of 60, mean gap 12 h) at irregular times. Fields are standardised per channel, and all metrics are reported back in physical units.

Models. The t-INR follows the spatio-temporal architecture of App. D.1, with 5 input channels. The velocity field is the convolutional network of App. D.2, enlarged to 64 base channels and 3 residual blocks, with a latent of dimension 128 (Table 5). The baselines use the same velocity-field network. All models are trained for 2000 epochs over three seeds.

Evaluation. From a history of $h = 10$ observations, each model produces a single rollout over the remaining 40 frames of the window, from 0 to 234 h lead, scored every 6 h against the reanalysis. We report the cos(lat)-weighted RMSE, the Anomaly Correlation Coefficient (ACC), the CRPS and the nMSE. The climatology is computed, for each variable and grid point, as the mean of the field over all the time steps of the training years (1979–2015). It serves both as a reference forecast, which predicts this mean at every lead time, and as the reference for the ACC, which is computed on the anomalies with respect to it. The SDE baselines are scored on the mean of an ensemble of 20 forecasts, and our method on the forecast obtained with the posterior mean of $\mathbf{z}$. The CRPS uses the same ensembles of 20 members; for our method, the spread of the latent ensemble is calibrated on the validation year. As references, we report persistence, which carries the last observed state forward, and climatology.

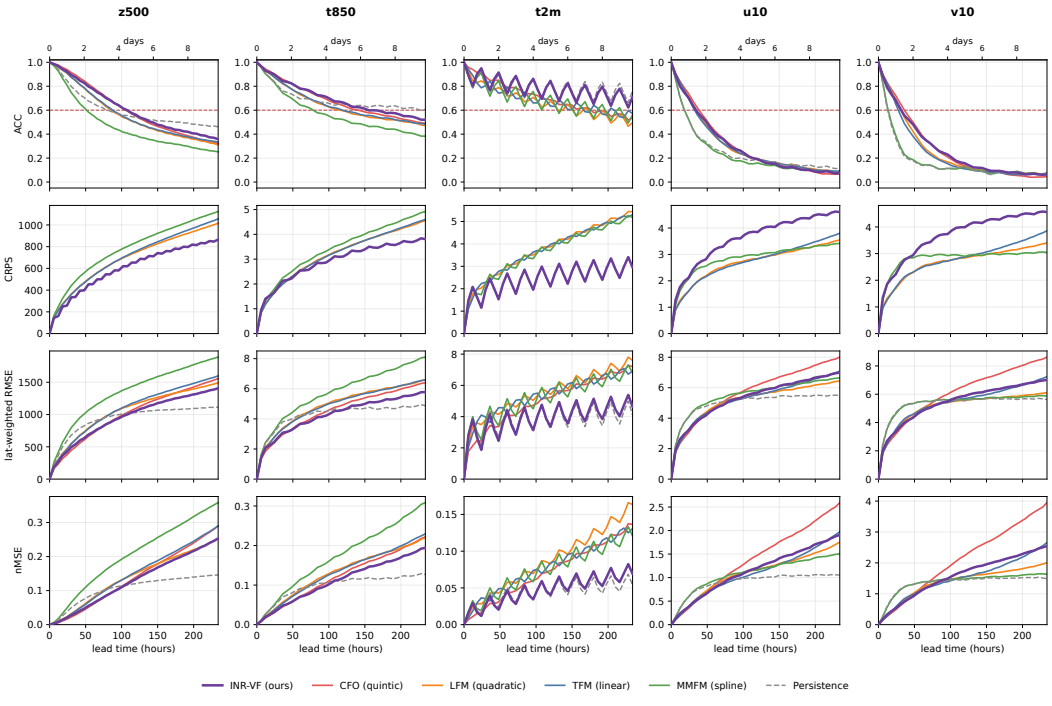

Figure 8: ERA5 skill against lead time, until 234 h.

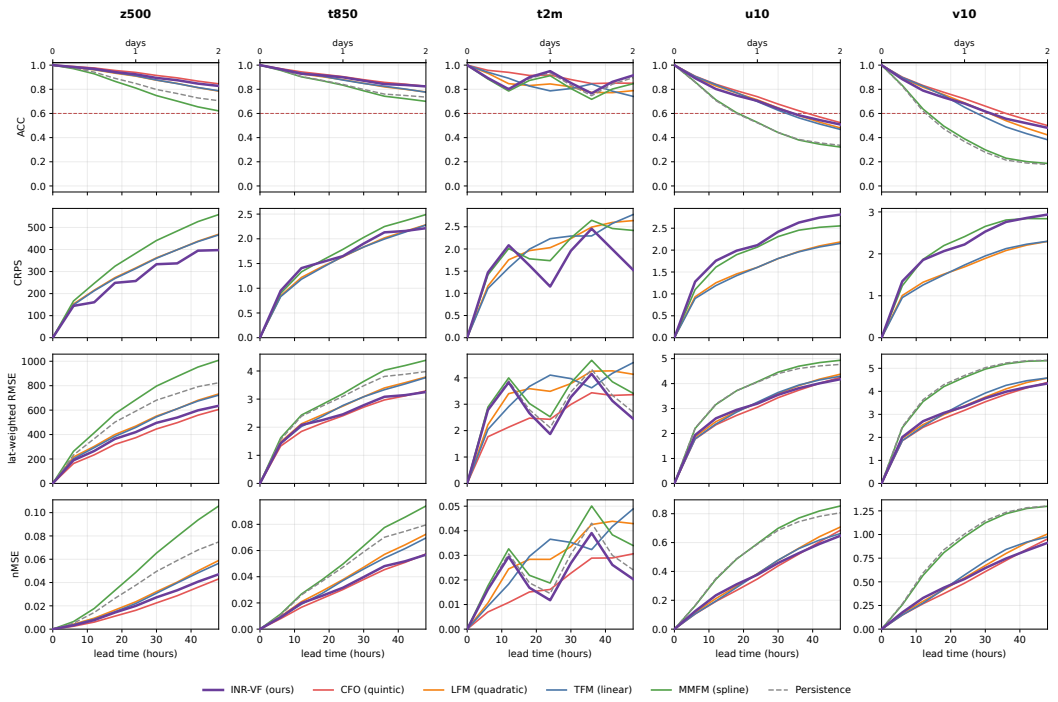

Figure 9: ERA5 skill against lead time, until 48 h.

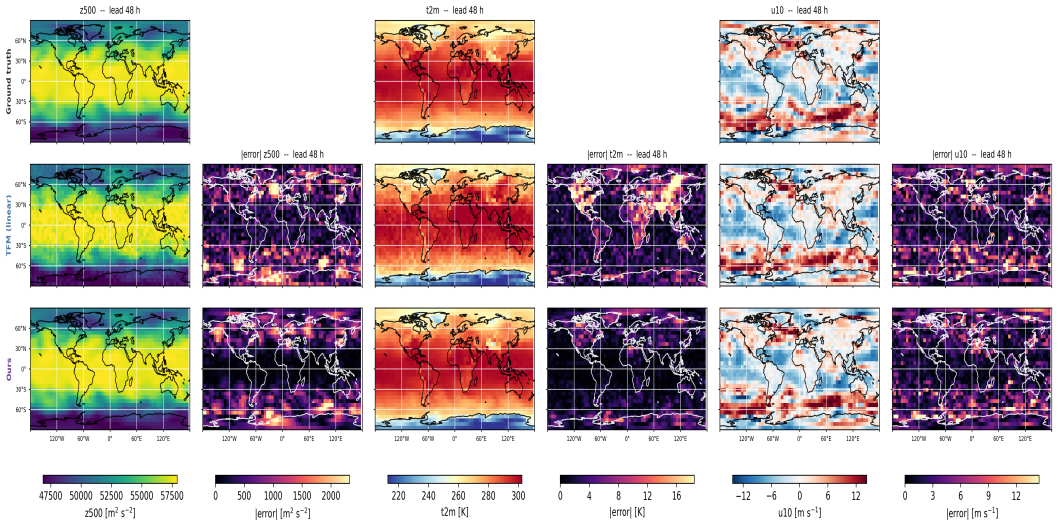

Figure 10: ERA5 ground truth test sample and forecasts from TFM and our method at a lead time of 48 h, with the corresponding prediction errors.