DISKO: DEEP KOOPMAN LEARNING IN DISTRIBUTION SPACE FROM UNPAIRED SNAPSHOTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Many complex systems are observed only through temporally unpaired distribution snapshots, making trajectory-based dynamical learning difficult without additional assumptions. We therefore formulate the problem directly in distribution space, treating the distribution itself as the dynamical state. The challenge is that distribution space is infinite-dimensional, making compact and approximately closed representations difficult to learn from finite snapshots. We introduce **DisKO**, which extends deep Koopman learning to distribution dynamics by jointly learning predictive distributional observables, a finite-dimensional Koopman representation, and a generative map back to the full distribution. Across seven diverse benchmarks, DisKO achieves **state-of-the-art** extrapolation performance, with **substantially slower error accumulation** on long-horizon prediction tasks. DisKO further recovers leading Koopman eigenvalues and eigenfunctions on systems with analytic spectra, revealing meaningful dynamical structure in the learned representation.

1 INTRODUCTION

Reconstructing the dynamics of complex systems from observations is a central problem in science (Chen et al., 2022). Many established methods rely on temporally paired state observations to estimate state transitions or temporal derivatives (Brunton et al., 2016b; Lusch et al., 2018). In many settings, however, destructive measurements, acquisition costs, or the spatial and temporal scales of the process make it infeasible to track the same entities across time (Weinreb et al., 2020; Bunne et al., 2023). Instead, observations consist of finite samples from the system's state distribution at a limited number of time points, without information linking individual observations across time. We refer to such observations as *unpaired distribution snapshots*, or simply *distribution snapshots*. Such data arise in diverse settings, including single-cell sequencing, particle ensembles in collective motion and transport systems, and halo populations in cosmological simulations (Schiebinger et al., 2019; Klein et al., 2025; Weiler et al., 2024; Reynolds, 1987; Villaescusa-Navarro et al., 2021). The problem is therefore to learn dynamics directly from the temporal evolution of distributions.

When individual correspondences across time are unavailable, many methods infer latent trajectories between snapshots using optimal transport, stochastic dynamics, or path regularization (Schiebinger et al., 2019; Tong et al., 2020; Zhang et al., 2025b; Sha et al., 2024; Zhang et al., 2025a). Yet marginal distributions do not uniquely determine individual trajectories: different couplings or stochastic processes can produce the same snapshots (Gu et al., 2025; Petrović et al., 2025). Trajectory reconstruction therefore requires additional assumptions to recover information that is not observed. If the goal is to predict population distributions, a more direct approach is to model dynamics in distribution space itself (Atanackovic et al., 2025; Haviv et al., 2025).

This better matches the form of snapshot data, but distribution space is infinite-dimensional, and directly learning nonlinear dynamics there does not naturally yield a compact, approximately closed representation for long-horizon prediction. Koopman methods offer a different perspective by lifting nonlinear state dynamics to linear evolution of observables, enabling structured propagation and spectral analysis (Brunton et al., 2022; Froyland et al., 2021). Deep Koopman learning further learns these observables from data rather than prescribing them (Li et al., 2017; Lusch et al., 2018; Mardt et al., 2018).

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.

For distributional dynamics, however, observables become functionals of entire distributions. The Distributional Koopman Operator extends Koopman theory to this setting, but existing data-driven methods estimate finite-dimensional operators over prescribed observable dictionaries (Oprea et al., 2025). They neither learn predictive distributional observables nor provide a generative map from those observables back to the full distribution.

To close this gap, we introduce **DisKO**, a deep **Dis**tributional **KO**opman learning framework that extends deep Koopman learning to distribution-valued dynamics. Rather than prescribing distributional observables, DisKO jointly learns predictive observables, a finite-dimensional representation of the corresponding Koopman operator, and a generative map back to the full probability distribution. Concretely, a permutation-invariant encoder maps empirical distribution snapshots to finite-dimensional latent coordinates, a shared affine generator advances these coordinates in continuous time, and a conditional generative decoder maps encoded or propagated coordinates back to full distributions (Figure 1). Table 1 compares method capabilities.

Our contributions are threefold: **(1)** We **extend deep Koopman learning to distribution dynamics** and introduce DisKO, an end-to-end framework that learns directly from unpaired distribution snapshots. **(2)** Across seven benchmarks spanning stochastic dynamics, collective motion, transport, cosmology, and single-cell dynamics, DisKO achieves **state-of-the-art** extrapolation while exhibiting **substantially slower error accumulation** over long prediction horizons. **(3)** On systems with analytic Koopman spectra, DisKO recovers leading eigenvalues and eigenfunctions, showing that its learned representation is not only predictive but also **captures interpretable decay and oscillatory modes**.

Method	Distribution snapshots	Learned observables	Distribution space	Full-distribution prediction	Spectral access
Deep Koopman (Lusch et al., 2018)	×	✓	×	×	✓
DeepRUOT (Zhang et al., 2025b)	✓	×	×	✓	×
WLM (Guan et al., 2026)	✓	×	✓	✓	×
Distributional DMD (Oprea et al., 2025)	✓	×	✓	×	✓
DisKO	✓	✓	✓	✓	✓

Table 1: Comparison of methods for learning distribution dynamics.

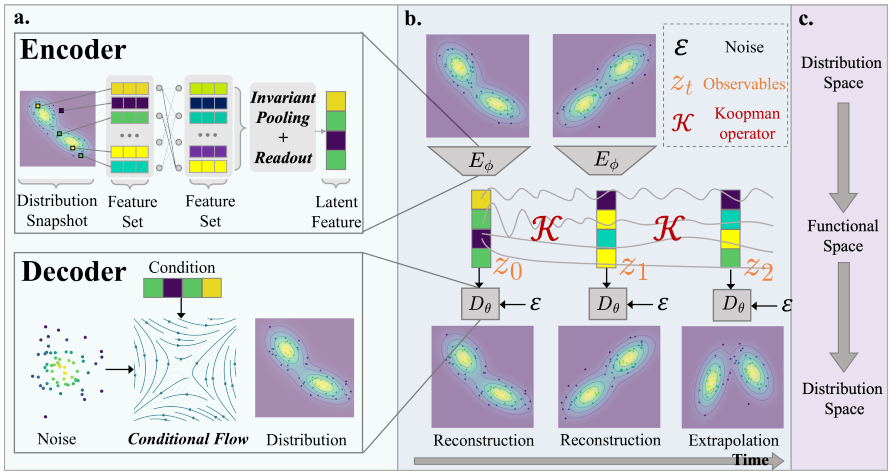

Figure 1: Overview of DisKO. Distribution snapshots are encoded into learned observables, evolved under Koopman dynamics, and decoded back to distributions for reconstruction and extrapolation.

2 Related Work

Learning distribution dynamics from snapshots. Learning population dynamics from unpaired snapshots requires assumptions about the evolution between observations (La Manno et al., 2018; Bergen et al., 2020; Gayoso et al., 2024; Rohbeck et al., 2025; Kviman et al., 2026). TrajectoryNet constrains the paths between snapshots through transport regularization (Tong et al., 2020), while

DeepRUOT and VGFM couple transport with population growth (Zhang et al., 2025b; Wang et al., 2025). This evolution can also be modeled in learned latent coordinates, as in MIOFlow and scNODE (Huguet et al., 2022; Zhang et al., 2024), or constrained by energy and mechanical principles (Persiianov et al., 2026), as in PRESCIENT, JKOnet*, and WLM (Yeo et al., 2021; Terpin et al., 2024; Guan et al., 2026). By contrast, DisKO brings a Koopman perspective to this problem, learning distribution observables whose evolution supports both prediction and spectral analysis.

Koopman theory and learning. Koopman methods obtain such spectral descriptions by approximating the linear evolution of observables (Ryu et al., 2024; Jeong et al., 2025; Colbrook et al., 2026). Dynamic mode decomposition and its extended form estimate this evolution from paired state observations using a prescribed representation (Schmid, 2010; Williams et al., 2015). Dictionary learning and neural embeddings make that representation trainable (Li et al., 2017; Lusch et al., 2018; Naiman et al., 2024; Xu et al., 2025; Selim et al., 2025). For population prediction, spectral expansions have been used to forecast probability densities (Wang & Araki, 2026), and distributional Koopman theory extends the observables themselves to functionals of distributions (Oprea et al., 2025). The latter formulation provides the basis for DisKO, which learns these functionals jointly with a continuous generator and a generative decoder.

Set representations and conditional generation. Learning distribution observables from finite snapshots requires representations that respect sample permutations (Fishman et al., 2025). Deep Sets and Set Transformer provide such encoders (Zaheer et al., 2017; Lee et al., 2019). To recover a distribution from its evolved representation, we pair these encoders with conditional generative flows trained by flow matching (Lipman et al., 2023; Tong et al., 2024).

3 Theoretical Framework

We introduce the snapshot prediction problem and Koopman representations of distribution dynamics, with details in Appendix G.

3.1 Problem Formulation

We learn from *distribution snapshots*, each consisting of an unordered set of state samples collected at a given time. Such data may be generated by deterministic or stochastic dynamics (Qiu et al., 2022; Vinyard et al., 2025). A representative model is

$$dX_t = b(X_t)\,dt + \sigma(X_t)\,dW_t, \qquad X_0 \sim \nu_0, \tag{1}$$

where $X_t \in \mathcal{X} \subseteq \mathbb{R}^p$ and W_t is a standard Brownian motion independent of X_0. We assume a unique nonexplosive solution and write $\nu_t = \mathrm{Law}(X_t)$ for its distribution. Setting $\sigma = 0$ recovers an ODE.

Sequences of distribution snapshots, indexed by r, share the dynamics and may have different initial distributions $\nu_0^{(r)}$. At time t_k, sequence r provides an unordered sample $\{y_{k,i}^{(r)}\}_{i=1}^{N_{r,k}}$ from $\nu_{t_k}^{(r)}$, represented by

$$\widehat{\nu}_{t_k}^{(r)} = \frac{1}{N_{r,k}} \sum\nolimits_{i=1}^{N_{r,k}} \delta_{y_{k,i}^{(r)}}. \tag{2}$$

Observation times and sequence membership are known, while individual samples are unpaired across time.

Our goal is to extrapolate distribution evolution beyond the training interval $[0, \tau]$. After training on snapshots within this interval, the model receives a source snapshot $\widehat{\nu}_s^{(r)}$ at $s \le \tau$ and predicts $\nu_t^{(r)}$ for $t > \tau$ without further observations. Our experiments use the first snapshot as input, with future snapshots providing ground truth distributions for evaluation (Appendix A.2).

3.2 The Distributional Koopman Operator

Let $\mathfrak{M} \subseteq \mathcal{P}(\mathcal{X})$ be a family of distributions equipped with a deterministic evolution $T_t : \mathfrak{M} \to \mathfrak{M}$ satisfying $T_0 = \mathrm{Id},\ T_{t+s} = T_t \circ T_s,\ t, s \ge 0$. This formulation describes evolution on the space of distributions independently of a particular state equation. Well-posed autonomous ODEs and

time-homogeneous Markov SDEs induce such evolutions through deterministic pushforward and Markov transition probabilities, respectively.

Let $\mathcal{H}$ be a linear space of real-valued distribution observables $h : \mathfrak{M} \to \mathbb{R}$ that is preserved under composition with T_t. Following Oprea et al. (2025), the distributional Koopman operator is

$$(\mathcal{K}_t h)(\mu) = h(T_t \mu). \tag{3}$$

We use $\mathcal{K}_t$ for this operator and reserve D for the decoder. The input to h is a distribution, and its output is a scalar statistic of that distribution. This extends the usual composition of a state observable with a state evolution map to the space of distributions. Composition is linear in the observable, so $\mathcal{K}_t(\alpha h + \beta g) = \alpha \mathcal{K}_t h + \beta \mathcal{K}_t g$, $\mathcal{K}_{t+s} = \mathcal{K}_t \mathcal{K}_s$. On a function space where this semigroup is strongly continuous, its generator $\mathcal{L}$ is defined for observables in its domain by

$$\mathcal{L}h = \lim_{\Delta t \downarrow 0} \frac{\mathcal{K}_{\Delta t} h - h}{\Delta t}. \tag{4}$$

The generator describes the instantaneous evolution of distribution observables and provides the continuous-time object approximated below.

3.3 Learning Finite-Dimensional Distributional Representations

To model distribution evolution in finite dimensions, we jointly learn observables, their dynamics, and a decoder. The components of an encoder $E : \mathcal{P}(\mathcal{X}) \to \mathbb{R}^m$ define the distribution observables, $h_j(\mu) = E_j(\mu)$, $z = E(\nu) = \mathbf{h}(\nu) \in \mathbb{R}^m$, $\mathbf{h} = (h_1, \ldots, h_m)^\top$. The h_j are potentially nonlinear functionals on $\mathfrak{M}$, and z collects their values at ν. In practice, the encoder is evaluated on empirical snapshots. To propagate these coordinates, we learn an affine system

$$\dot{z} = Az + c, \qquad A \in \mathbb{R}^{m \times m}, \quad c \in \mathbb{R}^m, \tag{5}$$

with flow F_t. For observables in the domain of $\mathcal{L}$, we seek the generator and flow approximations

$$\mathcal{L}\mathbf{h} \approx A\mathbf{h} + c, \qquad \mathbf{h}(T_t \mu) \approx F_t(\mathbf{h}(\mu)). \tag{6}$$

Adding the constant observable makes the model linear in $(z, 1)$, approximating the action of $\mathcal{L}$ on the span of $1, h_1, \ldots, h_m$. Exact propagation requires this space to be invariant under Koopman evolution (Brunton et al., 2016a; Cenedese et al., 2022; Fathi et al., 2024; Lu et al., 2026). In practice, we seek approximate closure and assess it through future predictions and spectral diagnostics. Appendix G.2 provides further analysis of observable expressiveness, dynamical closure, and spectral interpretation.

To recover distributions from these coordinates, we learn a decoder $D : \mathbb{R}^m \to \mathcal{P}(\mathcal{X})$ satisfying $D(E(\nu)) \approx \nu$. Reconstruction encourages the encoder to retain distributional information. Exact recovery on a modeled family requires E to distinguish its members and D to invert E on its image. For $t > s$, we predict $\widetilde{\nu}^{(r)}_{t|s} = D(F_{t-s}(E(\widehat{\nu}^{(r)}_s)))$ by encoding, propagating, and decoding the source snapshot. In practice, D generates samples conditioned on the propagated coordinates. Section 4 specifies the parameterizations and joint training objectives. The following OU example illustrates exact finite-dimensional propagation and distribution reconstruction.

Example 3.1 (Distribution observables for an OU system). Consider $dX_t = -aX_t\,dt + \sigma\,dW_t$ with $a > 0$, $\sigma \geq 0$, and $X_0 \sim \mathcal{N}(m_0, v_0)$ independent of the Brownian motion W_t. Its distribution remains Gaussian and is determined by the values of the mean and variance observables $h_1(\nu) = \int x\,d\nu(x)$, $h_2(\nu) = \int x^2\,d\nu(x) - h_1(\nu)^2$. Their Koopman evolution is explicit, with $v_\infty = \sigma^2/(2a)$, $(\mathcal{K}_t h_1)(\nu_0) = h_1(\nu_t) = m_t = e^{-at} m_0$, $(\mathcal{K}_t h_2)(\nu_0) = h_2(\nu_t) = v_t = e^{-2at} v_0 + v_\infty(1 - e^{-2at})$. The encoder $E(\nu) = (h_1(\nu), h_2(\nu))^\top$ therefore gives coordinates $z_t = (m_t, v_t)^\top$ with exact affine dynamics $\dot{z}_t = (-am_t, -2av_t + \sigma^2)^\top$. The span of $1, h_1, h_2$ is invariant under $\mathcal{K}_t$. Within the Gaussian family, $D(m, v) = \mathcal{N}(m, v)$ for $v \geq 0$ then gives exact reconstruction, $D(F_t(E(\nu_0))) = \mathcal{N}(e^{-at} m_0, e^{-2at} v_0 + v_\infty(1 - e^{-2at})) = \nu_t$.

4 DisKO Model and Learning Algorithm

DisKO instantiates the distribution encoder E, latent flow F_t, and distribution decoder D from Section 3 with a set network, affine dynamics, and conditional flow matching. We describe these

components and their joint training, followed by a discrete closed-form alternative to continuous generator learning.

4.1 Distribution Encoding with Set Networks

For a snapshot $S = \{y_i\}_{i=1}^N$, $E_\theta(S)$ denotes the encoding of its empirical distribution. We use permutation-invariant networks based on Deep Sets (Zaheer et al., 2017) and Set Transformer (Lee et al., 2019), combining a learned aggregate $g_\theta(S)$ with snapshot statistics, $E_\theta(S) = \rho_\theta([g_\theta(S), \bar{y}_S, s_S]) \in \mathbb{R}^m$, where $\bar{y}_S$ and s_S are the coordinatewise mean and standard deviation with normalization $1/N$, brackets denote concatenation, and ρ_θ is a multilayer perceptron. These statistics give the encoder direct access to the distribution's location and marginal spread. The pooling variant concatenates the mean and coordinatewise maximum of point features from a shared network ϕ_θ. The attention variant processes these features with self-attention blocks and pools them through multihead attention with a learned query (Lee et al., 2019). Its blocks use residual connections, feedforward layers, and layer normalization. Both variants produce a fixed-dimensional vector invariant to sample order; the choice is specified for each dataset. The interface also supports other pooling networks or Galerkin attention (Cao, 2021) with permutation-invariant pooling.

4.2 Continuous-Time Latent Dynamics

We propagate the encoded coordinates with the affine generator in Equation 5, learning A, c through an augmented matrix exponential,

$$B = \begin{pmatrix} A & c \\ 0 & 0 \end{pmatrix}, \qquad \begin{pmatrix} F_{\Delta t}(z) \\ 1 \end{pmatrix} = \exp(\Delta t B) \begin{pmatrix} z \\ 1 \end{pmatrix}. \tag{7}$$

A single parameter set governs all intervals and gives $F_{t+s} = F_t \circ F_s$, supporting irregular observation times and prediction at arbitrary future times. The affine term accommodates nonzero offsets in observable evolution, as in Example 3.1. Gradients pass through the matrix exponential to the generator and encoder. The eigenvalues of A describe latent decay, growth, and oscillation rates; their agreement with the underlying system is assessed experimentally.

4.3 Distribution Decoding via Conditional Flow Matching

To recover a distribution from the propagated coordinates, we implement D_ψ using conditional flow matching (Lipman et al., 2023; Tong et al., 2024). Given a condition z and target snapshot S, draw y uniformly from S, independent noise $\epsilon \sim \mathcal{N}(0, I_p)$, and $\alpha \sim \mathcal{U}(0,1)$, where p is the data dimension. The linear path and its velocity are $\xi_\alpha = (1-\alpha)\epsilon + \alpha y,\ u = y - \epsilon$. A multilayer perceptron v_ψ receives the concatenation of ξ_α, z, and α and minimizes

$$\ell_{\mathrm{CFM}}(z, S) = \mathbb{E}_{\alpha,\epsilon,y}\left[\frac{1}{p}\left\|v_\psi(\xi_\alpha, \alpha, z) - (y - \epsilon)\right\|_2^2\right]. \tag{8}$$

To sample from $D_\psi(z)$, we independently draw $U_0 \sim \mathcal{N}(0, I_p)$ for each sample and numerically integrate $\frac{dU_\alpha}{d\alpha} = v_\psi(U_\alpha, \alpha, z),\ \alpha \in [0,1],\ D_\psi(z) = \mathrm{Law}(U_1)$. Here, α parametrizes generation at a fixed physical time. Physical evolution enters through the shared condition $z = F_{t-s}(E_\theta(S_s))$ when predicting from time s to t. This decoder accommodates high-dimensional distributions, including the 30-dimensional Pancreas and 50-dimensional WOT iPSC representations in our experiments. For low-dimensional data with suitable structure, conditional Gaussian mixtures or other parametric families provide alternative decoders with interpretable parameters, trained with an appropriate likelihood or distribution loss.

4.4 Training Objectives and Algorithm

To couple representation learning with temporal prediction, we draw pairs (S_s, S_t) from the same snapshot sequence, with $s < t$ in the training window. Points are sampled independently from each snapshot. Define $z_s = E_\theta(S_s)$, $z_t = E_\theta(S_t)$, and $\widehat{z}_t = F_{t-s}(z_s)$. The prediction, reconstruction, and latent consistency objectives are

$$\mathcal{J}_{\mathrm{pred}} = \mathbb{E}\,\ell_{\mathrm{CFM}}(\widehat{z}_t, S_t),\ \mathcal{J}_{\mathrm{rec}} = \mathbb{E}\,\ell_{\mathrm{CFM}}(z_t, S_t),\ \mathcal{J}_{\mathrm{lat}} = \mathbb{E}\left[\frac{1}{m}\|\widehat{z}_t - \mathrm{sg}(z_t)\|_2^2\right], \tag{9}$$

where expectations are over training pairs, m is the latent dimension, and sg stops the gradient. Reconstruction encourages the encoding to retain distributional information, while prediction trains the decoder with the propagated condition used at inference. Latent consistency aligns the propagated coordinates with the target encoding, passing its encoder gradient through the source branch. We further compare a generated set $\widetilde{S}_t$ from $D_\psi(\widehat{z}_t)$ with an equally sized target subset S_t^*,

$$\mathcal{J}_{\mathrm{dist}} = \mathbb{E}\Big[\mathrm{SW}_1(\widetilde{S}_t, S_t^*) + \beta\,\mathrm{MMD}^2(\widetilde{S}_t, S_t^*) + \gamma d_{\mathrm{mom}}(\widetilde{S}_t, S_t^*)\Big]. \tag{10}$$

Here, SW_1 averages one-dimensional Wasserstein distances over random projections (Bonneel et al., 2015), MMD^2 uses a mixture of Gaussian kernels (Gretton et al., 2012), and d_{mom} sums mean squared differences between sample means and raw second-moment matrices. Gradients pass through the numerical sampler to supervise the generated distribution. We combine the four terms with nonnegative weights,

$$\mathcal{J} = \lambda_{\mathrm{pred}}\mathcal{J}_{\mathrm{pred}} + \lambda_{\mathrm{rec}}\mathcal{J}_{\mathrm{rec}} + \lambda_{\mathrm{lat}}\mathcal{J}_{\mathrm{lat}} + \lambda_{\mathrm{dist}}\mathcal{J}_{\mathrm{dist}}. \tag{11}$$

Continuous generator learning proceeds in three stages. We first pretrain E_θ, D_ψ on individual snapshots with $\ell_{\mathrm{CFM}}(E_\theta(S), S)$, then fix them and fit A, c using latent prediction error on training pairs. Finally, we jointly optimize all components with Equation 11. This initializes reconstruction and propagation before coupling them through the full objective. We use AdamW with gradient clipping; the full training procedure is detailed in Appendix B, with hyperparameter settings in Table B.2.

Discrete closed-form updates. For uniformly spaced snapshots, an alternative is to learn the action of $\mathcal{K}_{\Delta t}$ directly through $F_{\Delta t}(z) = Kz + b$. Let X contain augmented source encodings $[z_s^\top, 1]^\top$ as columns and Y the corresponding target encodings at $s + \Delta t$, using all adjacent pairs from training sequences within the training window. With $\Theta = [K\ b]$, the ridge update is

$$\widehat{\Theta} = \operatorname*{argmin}_{\Theta} \|Y - \Theta X\|_F^2 + \eta\|\Theta\|_F^2 = YX^\top(XX^\top + \eta I)^{-1}, \qquad \eta > 0. \tag{12}$$

The fit uses unstandardized latent coordinates and a sum of squared residuals over pairs, with a penalty on both K and b. We evaluate the solution by singular value decomposition. After the same pretraining, this fit initializes the discrete dynamics. Joint training alternates encoder and decoder updates under Equation 11 with re-encoding the training pairs and refitting K, b. The ridge solve is detached from the gradient computation; K, b are fixed during each gradient step, while propagation retains gradients to the source encoder. The spectral experiments refit after every step with $\eta = 10^{-3}$ (Appendix C.2). Repeated application of $F_{\Delta t}$ yields $F_{q\Delta t}$ for nonnegative integers q, supporting the same prediction losses over multiple horizons. For either variant, inference encodes the source snapshot once, propagates its coordinates to each requested time, and decodes the corresponding distribution.

5 Results

5.1 Experimental Setup

Datasets. We evaluate DisKO's ability to extrapolate distribution dynamics on the following seven datasets. OU and Duffing test long-horizon extrapolation across multiple initial distributions (*50 and 80 future steps*, respectively), while Boids tests extrapolation from a single sequence (*50 future steps*). Ocean tests prediction from a short sequence (one future step). CAMELS tests *generalization across initial distributions* in 10 dimensions. Pancreas and WOT iPSC assess prediction on real single-cell data (*30 and 50 dimensions*, respectively), with WOT iPSC spanning 39 snapshots. Circle and Torus provide analytic spectra for spectral validation. Dataset details appear in Appendix A.1.

Experimental protocol. We divide the observation times into training and extrapolation windows. For datasets with multiple initial distributions (OU, Duffing, and CAMELS), we also split the initial distributions into training and test sets. Detailed splits are provided in Table A.1 of Appendix A.1.

Evaluation. We quantify prediction accuracy using W_1 and SW_1 between predicted and ground truth distributions; W_2 and MMD^2 are additionally reported in the appendix.

Baselines. We compare DisKO with eight population dynamics models to assess its predictive advantage. TrajectoryNet learns regularized transport flows (Tong et al., 2020). DeepRUOT

models stochasticity and growth through unbalanced transport (Zhang et al., 2025b). VGFM jointly estimates velocity and growth by flow matching (Wang et al., 2025). MIOFlow learns latent differential dynamics with a geodesic autoencoder (Huguet et al., 2022). scNODE couples a variational autoencoder to a latent Neural ODE (Zhang et al., 2024). PRESCIENT models stochastic evolution through a potential landscape (Yeo et al., 2021). JKOnet* identifies energy terms using Wasserstein gradient-flow optimality (Terpin et al., 2024). WLM represents inertial population dynamics with a Wasserstein Lagrangian (Guan et al., 2026).

5.2 Distribution Extrapolation

Table 2 reports distribution extrapolation results for DisKO and eight baselines across seven datasets. DisKO achieves the lowest W_1 on all seven datasets and the lowest SW_1 on six. Relative to the strongest baseline on each dataset, DisKO reduces W_1 by 58.6% on OU, 53.9% on Duffing, and 36.2% on Ocean. The corresponding reductions on the real single-cell datasets Pancreas and WOT iPSC are 18.0% and 18.7%, respectively. Complete numerical results across time windows and initial distribution splits are provided in Tables B.3–B.6 of Appendix B.

Table 2: Results over the extrapolation window for test initial distributions in datasets with multiple initial distributions, and for single sequences. Red and blue indicate the best and second best results, respectively.

Method	Metric	OU	Duffing	Boids	Ocean	CAMELS	Pancreas	WOT iPSC
TrajectoryNet	W_1	$1.369_{\pm 0.071}$	$9.222_{\pm 0.783}$	$11.843_{\pm 0.560}$	$2.198_{\pm 0.900}$	$143.767_{\pm 2.170}$	$47.376_{\pm 2.950}$	$118.524_{\pm 2.509}$
	SW1	$0.857_{\pm 0.047}$	$5.815_{\pm 0.712}$	$7.267_{\pm 0.365}$	$1.426_{\pm 0.556}$	$35.364_{\pm 1.128}$	$6.765_{\pm 0.826}$	$14.502_{\pm 0.647}$
DeepRUOT	W_1	$0.846_{\pm 0.044}$	$1.032_{\pm 0.287}$	$5.962_{\pm 0.283}$	$0.214_{\pm 0.020}$	$79.989_{\pm 1.252}$	$17.286_{\pm 1.572}$	$23.635_{\pm 1.623}$
	SW1	$0.554_{\pm 0.030}$	$0.650_{\pm 0.186}$	$3.796_{\pm 0.172}$	$0.149_{\pm 0.014}$	$14.362_{\pm 0.466}$	$2.113_{\pm 0.221}$	$2.108_{\pm 0.220}$
VGFM	W_1	$1.290_{\pm 0.486}$	$12.834_{\pm 0.976}$	$31.008_{\pm 1.709}$	$0.195_{\pm 0.047}$	$123.071_{\pm 2.075}$	$27.497_{\pm 1.337}$	$187.154_{\pm 2.894}$
	SW1	$0.807_{\pm 0.315}$	$9.358_{\pm 0.352}$	$19.604_{\pm 1.428}$	$0.128_{\pm 0.034}$	$24.995_{\pm 17.602}$	$2.978_{\pm 2.126}$	$27.389_{\pm 1.762}$
MIOFlow	W_1	$1.514_{\pm 0.135}$	$0.692_{\pm 0.109}$	$9.642_{\pm 0.390}$	$0.187_{\pm 0.044}$	$98.173_{\pm 1.421}$	$11.111_{\pm 0.902}$	$34.854_{\pm 0.627}$
	SW1	$0.950_{\pm 0.087}$	$0.432_{\pm 0.070}$	$5.879_{\pm 1.216}$	$0.130_{\pm 0.033}$	$20.381_{\pm 1.104}$	$0.895_{\pm 0.171}$	$3.931_{\pm 0.053}$
scNODE	W_1	$0.759_{\pm 0.010}$	$0.887_{\pm 0.075}$	$7.822_{\pm 0.359}$	$0.783_{\pm 0.246}$	$143.751_{\pm 1.692}$	$43.047_{\pm 1.030}$	$94.268_{\pm 1.536}$
	SW1	$0.481_{\pm 0.008}$	$0.541_{\pm 0.070}$	$4.381_{\pm 0.233}$	$0.515_{\pm 0.154}$	$34.896_{\pm 1.137}$	$6.168_{\pm 0.831}$	$10.544_{\pm 1.150}$
PRESCIENT	W_1	$0.849_{\pm 0.034}$	$0.532_{\pm 0.040}$	$3.498_{\pm 0.136}$	$0.153_{\pm 0.009}$	$143.797_{\pm 0.232}$	$12.169_{\pm 0.325}$	$23.328_{\pm 0.086}$
	SW1	$0.525_{\pm 0.020}$	$0.280_{\pm 0.025}$	$1.951_{\pm 0.087}$	$0.097_{\pm 0.006}$	$36.244_{\pm 0.062}$	$0.965_{\pm 0.068}$	$1.883_{\pm 0.013}$
JKOnet*	W_1	$1.288_{\pm 0.067}$	$5.246_{\pm 0.447}$	$3.282_{\pm 0.187}$	$2.332_{\pm 0.876}$	$134.105_{\pm 2.073}$	$52.395_{\pm 1.119}$	$63.051_{\pm 1.367}$
	SW1	$0.797_{\pm 0.044}$	$3.290_{\pm 0.413}$	$1.770_{\pm 0.086}$	$1.561_{\pm 0.587}$	$30.058_{\pm 0.976}$	$8.221_{\pm 0.713}$	$7.076_{\pm 0.325}$
WLM	W_1	$0.602_{\pm 0.032}$	$0.318_{\pm 0.028}$	$1.595_{\pm 0.047}$	$0.941_{\pm 0.264}$	$55.019_{\pm 0.370}$	$13.345_{\pm 0.517}$	$25.805_{\pm 0.638}$
	SW1	$0.362_{\pm 0.020}$	$0.160_{\pm 0.019}$	$0.656_{\pm 0.050}$	$0.520_{\pm 0.246}$	$6.699_{\pm 0.206}$	$0.988_{\pm 0.140}$	$1.929_{\pm 0.093}$
DisKO	W_1	$0.249_{\pm 0.013}$	$0.146_{\pm 0.005}$	$1.454_{\pm 0.110}$	$0.097_{\pm 0.024}$	$54.241_{\pm 0.830}$	$9.114_{\pm 0.143}$	$18.973_{\pm 0.330}$
	SW1	$0.135_{\pm 0.010}$	$0.056_{\pm 0.004}$	$0.656_{\pm 0.087}$	$0.062_{\pm 0.019}$	$7.039_{\pm 0.500}$	$0.559_{\pm 0.007}$	$0.940_{\pm 0.039}$

The prediction error curves in Figure 2 further characterize DisKO's long-horizon extrapolation performance. On OU, prediction error **remains approximately constant** over 50 extrapolation steps. On Duffing, error grows slowly over 80 extrapolation steps and remains below that of the baselines shown in the figure. These results indicate that the learned Koopman representation supports sustained distribution prediction beyond the training window. We further examine the contribution of the shared linear evolution structure to extrapolation stability through an ablation against a nonlinear latent Neural ODE in Section 5.4. The complete error curves for all seven datasets are shown in Figure B.1 in Appendix B.

We obtain the reported performance with limited hyperparameter search, evaluating only one to three configurations per dataset. The main table reports the results of this initial search. For some datasets, further hyperparameter tuning can yield better results. These results demonstrate that DisKO achieves competitive performance across datasets with a modest tuning budget.

5.3 Koopman Spectral Approximation

Recovering distributional Koopman spectra from finite, unpaired snapshots is challenging because the eigenfunctions are defined on an infinite-dimensional space of probability measures. We assess

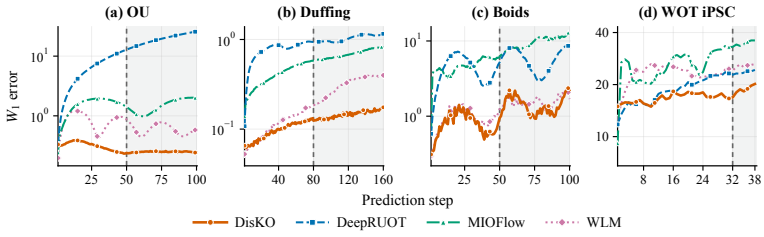

Figure 2: W_1 prediction errors on four benchmarks. Vertical dashed lines mark the end of training, with the extrapolation window shaded. All error axes use a logarithmic scale.

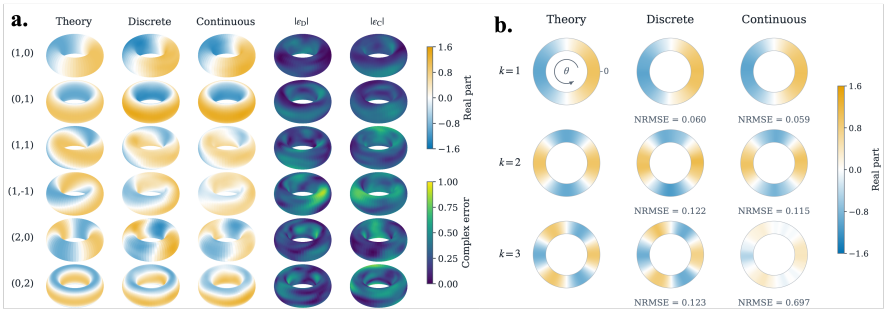

Figure 3: Distributional eigenfunctions on wrapped Gaussian probes ($\sigma = 0.15$), indexed by center. (a) Torus. (b) Circle. Plots show real components and complex errors after amplitude and phase calibration on training snapshots.

distributional Koopman recovery on Circle ($q = 1$) and Torus ($q = 2$), governed by angular drift and diffusion, $d\vartheta_{j,t} = \omega_j\, dt + \sqrt{2\kappa_j}\, dW_{j,t} \pmod{2\pi}$, $j = 1, \ldots, q$, with independent Brownian motions. Circle uses $(\omega_1, \kappa_1) = (1, 0.01)$; Torus uses $\omega = (1, 1.7)$ and $\kappa = (0.01, 0.014)$. The analytical eigenpairs for $k \in \mathbb{Z}^q$ satisfy

$$h_k(\mu) = \int e^{\mathrm{i}k\cdot\vartheta}\, d\mu(\vartheta), \quad \lambda_k = -\sum_{j=1}^{q} \kappa_j k_j^2 + \mathrm{i}\,\omega\cdot k, \qquad \mathcal{K}_t h_k = e^{t\lambda_k} h_k, \quad \rho_k = e^{\Delta t \lambda_k}. \tag{13}$$

Here h_k is a distributional Koopman eigenfunction with continuous-time generator eigenvalue λ_k, finite-time eigenvalue ρ_k, decay rate $-\mathrm{Re}(\lambda_k)$, and angular frequency $\mathrm{Im}(\lambda_k)$.

Uniformly spaced snapshots permit direct, closed-form fitting of the finite-time Koopman representation, providing a complementary route to spectral recovery. We therefore evaluate discrete DisKO (Section 4.4) alongside the continuous generator model (Section 4.2). Table 3 shows close agreement between analytical and recovered finite-time eigenvalues at $\Delta t = 0.1$ for both parameterizations. Absolute errors are below 0.006 for Circle's first three harmonics and 0.003 for the Torus fundamental modes.

	Circle			Torus	
DisKO	$k=1$	$k=2$	$k=3$	$k=(1,0)$	$k=(0,1)$
Discrete	1.315	3.191	5.928	2.404	2.892
Continuous	1.314	2.410	5.868	1.781	1.377

Table 3: Absolute eigenvalue errors $|\widehat{\rho}_k - \rho_k|$ ($\times 10^{-3}$) for $\mathcal{K}_{0.1}$. Full eigenvalues and higher Circle harmonics appear in Appendix C.3.

The nonlinear set encoder coordinates and the constant observable span a finite-dimensional observable space. Left eigenvectors of the augmented operator or generator define candidate eigenfunctions through linear combinations $u_k(\nu) = w_k^\top [E(\nu)^\top, 1]^\top$ (Appendix C.2). We evaluate their shapes on wrapped Gaussian probes with center a and covariance $\sigma^2 I$, where $h_k(\mu_{a,\sigma}) = e^{-\sigma^2\|k\|^2/2} e^{\mathrm{i}k\cdot a}$ (Figure 3). On future test snapshots, complex correlations reach 0.999, 0.994, and 0.976 for the first three Circle harmonics with discrete DisKO and exceed 0.995 for both Torus fundamental modes with either parameterization (Appendix C.4).

DisKO thus learns an observable space containing accurate approximations of analytical Fourier distribution functionals **directly from snapshots, without a prescribed Fourier dictionary.** Discrete DisKO better resolves higher Circle harmonics, while continuous DisKO gives more accurate Torus eigenvalues. Eigenvalue accuracy and eigenfunction fidelity are therefore distinct tests.

5.4 Dynamical Closure and Extrapolation Stability

On OU, we test approximate closure of the learned observables under Koopman evolution and the contribution of this structure to extrapolation stability.

Latent dynamical consistency. For a nearly closed observable subspace, propagating the initial encoding should agree with directly encoding later distributions. With the trained encoder fixed, Figure 4 compares these trajectories, $F_t(E(\widehat{\nu}_0))$ and $E(\widehat{\nu}_t)$. Their agreement in rotation and contraction beyond the training interval **supports approximate closure** along this OU trajectory in the PCA projection (Appendix D).

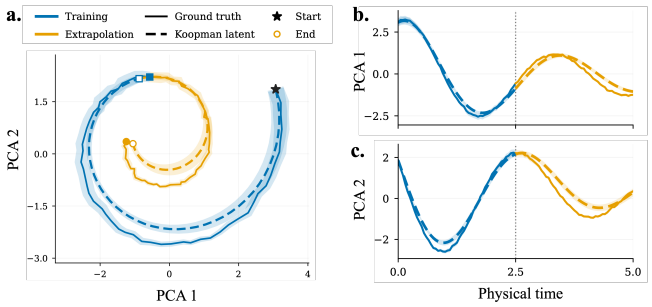

Figure 4: Reference encodings and forecasts for one OU test sequence, with one standard deviation across five seeds. PCA and rigid alignment use all times after forecasting.

Ablation of Koopman latent dynamics. To isolate Koopman latent dynamics, we replace only the affine generator with a Neural ODE, keeping encoder and decoder architectures, losses, and training budgets fixed. Within each of five seeds, both models start from identical pretrained encoder and decoder weights, with comparable dynamics parameter counts (4,160 and 4,192). Figure 5 shows lower future errors and smaller seed variation for Koopman dynamics in all four metrics. Future W_1 decreases from 0.201 ± 0.026 to 0.149 ± 0.007, and SW_1 from 0.107 ± 0.018 to 0.065 ± 0.005. The OU system has a stable stationary distribution, so long-horizon errors do not necessarily increase monotonically: as both models approach the attracting distribution, distribution discrepancies can decrease at later times. The Koopman parameterization achieves lower future errors and reduced variability across seeds, *supporting its role as an effective inductive bias for extrapolation.* (Appendix E).

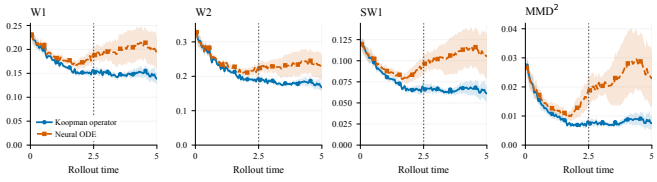

Figure 5: OU errors averaged over 64 test initial distributions and five seeds. Shading shows one seed standard deviation; the dotted line marks the training boundary. Forecasts start at $t = 0$.

6 Conclusion and Limitations

DisKO extends deep Koopman learning to population evolution, showing that a shared representation learned from unpaired snapshots can support both accurate distribution extrapolation and the recovery of interpretable decay and oscillation.

Two important theoretical questions remain open. First, under what conditions are the Koopman dynamics identifiable from unpaired distribution snapshots? Second, when identifiability holds, what guarantees can be established for convergence and stability of the proposed learning procedure toward that operator?

AI USE STATEMENT

We used generative AI tools to assist with code implementation and debugging, including support for baseline implementations, experimental visualization, and language polishing. AI tools were not used to generate research ideas, design the core methodology, formulate the theoretical framework, or produce experimental results. All AI-assisted content was reviewed and verified by the authors, who take full responsibility for the final content.

ETHICS STATEMENT

This work does not involve new human-subject data collection. Our experiments use synthetic simulations and previously released benchmark datasets, including publicly available single-cell datasets. We do not use participant identifiers or personally identifiable information. Data sources, preprocessing, experimental splits, and evaluation protocols are documented in the paper and appendices. The proposed method is intended for scientific modeling of distribution dynamics; applications in sensitive or high-stakes settings should require appropriate domain-specific validation and safeguards.

REPRODUCIBILITY STATEMENT

We facilitate reproducibility by providing anonymized code and configuration files in the supplementary materials. Dataset sources, preprocessing procedures, model architectures, training settings, evaluation protocols, and hyperparameters are documented in the paper and appendix. The accompanying code includes the random seeds and implementation settings used in our experiments. Additional ablations and supporting results are provided in the appendix. Together, these materials are intended to enable independent reproduction and verification of our results.

REFERENCES

Lazar Atanackovic, Xi (Nicole) Zhang, Brandon Amos, Mathieu Blanchette, Leo J Lee, Yoshua Bengio, Alexander Tong, and Kirill Neklyudov. Meta flow matching: Integrating vector fields on the Wasserstein manifold. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/ebdb990471f653dffb425eff03c7c980-Abstract-Conference.html.

Volker Bergen, Marius Lange, Stefan Peidli, F. Alexander Wolf, and Fabian J. Theis. Generalizing RNA velocity to transient cell states through dynamical modeling. *Nature Biotechnology*, 38(12): 1408–1414, 2020. doi: 10.1038/s41587-020-0591-3.

Nicolas Bonneel, Julien Rabin, Gabriel Peyré, and Hanspeter Pfister. Sliced and Radon Wasserstein barycenters of measures. *Journal of Mathematical Imaging and Vision*, 51:22–45, 2015. doi: 10.1007/s10851-014-0506-3.

Steven L. Brunton, Bingni W. Brunton, Joshua L. Proctor, and J. Nathan Kutz. Koopman invariant subspaces and finite linear representations of nonlinear dynamical systems for control. *PLOS ONE*, 11(2):e0150171, 2016a. doi: 10.1371/journal.pone.0150171.

Steven L. Brunton, Joshua L. Proctor, and J. Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 113(15):3932–3937, 2016b. doi: 10.1073/pnas.1517384113.

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

Charlotte Bunne, Stefan G. Stark, Gabriele Gut, Jacobo Sarabia del Castillo, Mitch Levesque, Kjong-Van Lehmann, Lucas Pelkmans, Andreas Krause, and Gunnar Rätsch. Learning single-cell perturbation responses using neural optimal transport. *Nature Methods*, 20(11):1759–1768, 2023. doi: 10.1038/s41592-023-01969-x.

Shuhao Cao. Choose a transformer: Fourier or Galerkin. In *Advances in Neural Information Processing Systems*, volume 34, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/d0921d442ee91b896ad95059d13df618-Abstract.html.

Mattia Cenedese, Joar Axås, Bastian Bäuerlein, Kerstin Avila, and George Haller. Data-driven modeling and prediction of non-linearizable dynamics via spectral submanifolds. *Nature Communications*, 13(1):872, 2022. doi: 10.1038/s41467-022-28518-y.

Boyuan Chen, Kuang Huang, Sunand Raghupathi, Ishaan Chandratreya, Qiang Du, and Hod Lipson. Automated discovery of fundamental variables hidden in experimental data. *Nature Computational Science*, 2(7):433–442, 2022. doi: 10.1038/s43588-022-00281-6.

Matthew J. Colbrook, Igor Mezić, and Alexei Stepanenko. Adversarial dynamical systems characterize when data-driven learning succeeds or fails. *Nature Communications*, 17(1):5397, 2026. doi: 10.1038/s41467-026-74220-8.

Klaus-Jochen Engel and Rainer Nagel. *One-Parameter Semigroups for Linear Evolution Equations*, volume 194 of *Graduate Texts in Mathematics*. Springer, 2000. doi: 10.1007/b97696.

Mahan Fathi, Clement Gehring, Jonathan Pilault, David Kanaa, Pierre-Luc Bacon, and Ross Goroshin. Course correcting Koopman representations. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/0429ececfb199efc93182990169e73bb-Abstract-Conference.html.

Nic Fishman, Gokul Gowri, Peng Yin, Jonathan Gootenberg, and Omar Abudayyeh. Generative distribution embeddings: Lifting autoencoders to the space of distributions for multiscale representation learning. In *Advances in Neural Information Processing Systems*, volume 38, pp. 24644–24685, 2025. doi: 10.52202/085713-0830. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/233b892889d0a18f9da39226ee5c9dc5-Abstract-Conference.html.

Gary Froyland, Dimitrios Giannakis, Benjamin R. Lintner, Maxwell Pike, and Joanna Slawinska. Spectral analysis of climate dynamics with operator-theoretic approaches. *Nature Communications*, 12(1):6570, 2021. doi: 10.1038/s41467-021-26357-x.

Adam Gayoso, Philipp Weiler, Mohammad Lotfollahi, Dominik Klein, Justin Hong, Aaron Streets, Fabian J. Theis, and Nir Yosef. Deep generative modeling of transcriptional dynamics for RNA velocity analysis in single cells. *Nature Methods*, 21(1):50–59, 2024. doi: 10.1038/s41592-023-01994-w.

Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13(25):723–773, 2012. URL https://jmlr.org/papers/v13/gretton12a.html.

Anming Gu, Edward Chien, and Kristjan Greenewald. Partially observed trajectory inference using optimal transport and a dynamics prior. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/69a076724e7228aba0272305bb98727e-Abstract-Conference.html.

Vincent Guan, Lazar Atanackovic, and Kirill Neklyudov. A call to Lagrangian action: Learning population mechanics from temporal snapshots. In *International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2605.08550.

Doron Haviv, Aram-Alexandre Pooladian, Dana Pe'er, and Brandon Amos. Wasserstein flow matching: Generative modeling over families of distributions. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 22238–22258, 2025. URL https://proceedings.mlr.press/v267/haviv25a.html.

Guillaume Huguet, Daniel Sumner Magruder, Alexander Tong, Oluwadamilola Fasina, Manik Kuchroo, Guy Wolf, and Smita Krishnaswamy. Manifold interpolating optimal-transport flows for trajectory inference. In *Advances in Neural Information Processing Systems*, volume 35, 2022. doi: 10.52202/068431-2154.

Minchan Jeong, Jongha (Jon) Ryu, Se-Young Yun, and Gregory Wornell. Efficient parametric SVD of Koopman operator for stochastic dynamical systems. In *Advances in Neural Information Processing Systems*, volume 38, pp. 25564–25600, 2025. doi: 10.52202/085713-0860. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/24826a1c623fe41706850748f3dc7370-Abstract-Conference.html.

Dominik Klein, Giovanni Palla, Marius Lange, Michal Klein, Zoe Piran, Manuel Gander, Laetitia Meng-Papaxanthos, Michael Sterr, Lama Saber, Changying Jing, Aimée Bastidas-Ponce, Perla Cota, Marta Tarquis-Medina, Shrey Parikh, Ilan Gold, Heiko Lickert, Mostafa Bakhti, Mor Nitzan, Marco Cuturi, and Fabian J. Theis. Mapping cells through time and space with moscot. *Nature*, 638(8052):1065–1075, 2025. doi: 10.1038/s41586-024-08453-2.

Stefan Klus, Feliks Nüske, Sebastian Peitz, Jan-Hendrik Niemann, Cecilia Clementi, and Christof Schütte. Data-driven approximation of the Koopman generator: Model reduction, system identification, and control. *Physica D: Nonlinear Phenomena*, 406:132416, 2020. doi: 10.1016/j.physd.2020.132416.

Oskar Kviman, Kirill Tamogashev, Nicola Branchini, Víctor Elvira, Jens Lagergren, and Nikolay Malkin. Multi-marginal flow matching with adversarially learnt interpolants. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/cb0ce861adaf6f8a93069c064733f402-Abstract-Conference.html.

Gioele La Manno, Ruslan Soldatov, Amit Zeisel, Emelie Braun, Hannah Hochgerner, Viktor Petukhov, Katja Lidschreiber, Maria E. Kastriti, Peter Lönnerberg, Alessandro Furlan, Jean Fan, Lars E. Borm, Zehua Liu, David van Bruggen, Jimin Guo, Xiaoling He, Roger Barker, Erik Sundström, Gonçalo Castelo-Branco, Patrick Cramer, Igor Adameyko, Sten Linnarsson, and Peter V. Kharchenko. RNA velocity of single cells. *Nature*, 560(7719):494–498, 2018. doi: 10.1038/s41586-018-0414-6.

Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3744–3753, 2019. URL https://proceedings.mlr.press/v97/lee19d.html.

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

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

Wanfeng Lu, He Ma, Wei Lin, and Qunxi Zhu. Interpretable functional Koopman learning with Non-Markovian closure for spatiotemporal systems. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=lquDiBCgNJ.

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

Andreas Mardt, Luca Pasquali, Hao Wu, and Frank Noé. VAMPnets for deep learning of molecular kinetics. *Nature Communications*, 9(1):5, 2018. doi: 10.1038/s41467-017-02388-1.

Ilan Naiman, N. Benjamin Erichson, Pu Ren, Michael W Mahoney, and Omri Azencot. Generative modeling of regular and irregular time series data via Koopman VAEs. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/dbeb7e621d4a554069a6a775da0f7273-Abstract-Conference.html.

Maria Oprea, Alex Townsend, and Yunan Yang. The distributional Koopman operator for random dynamical systems. *Mathematics of Control, Signals, and Systems*, 37:769–798, 2025. doi: 10.1007/s00498-025-00423-x.

Mikhail Persiianov, Jiawei Chen, Petr Mokrov, Alexander Tyurin, Evgeny Burnaev, and Alexander Korotin. Learning of population dynamics: Inverse optimization meets JKO scheme. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=tVJIKd6CLF.

Katarina Petrović, Lazar Atanackovic, Viggo Moro, Kacper Kapusniak, Ismail Ilkan Ceylan, Michael Bronstein, Joey Bose, and Alexander Tong. Curly flow matching for learning non-gradient field dynamics. In *Advances in Neural Information Processing Systems*, volume 38, pp. 121544–121573, 2025. doi: 10.52202/085713-4053. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/b00692865928625ace212e89ded2decd-Abstract-Conference.html.

Tomáš Pevný and Vojtěch Kovařík. Approximation capability of neural networks on spaces of probability measures and tree-structured domains. *arXiv preprint arXiv:1906.00764*, 2019. URL https://arxiv.org/abs/1906.00764.

Xiaojie Qiu, Yan Zhang, Jorge D. Martin-Rufino, Chen Weng, Shayan Hosseinzadeh, Dian Yang, Angela N. Pogson, Marco Y. Hein, Kyung Hoi (Joseph) Min, Li Wang, Emanuelle I. Grody, Matthew J. Shurtleff, Ruoshi Yuan, Song Xu, Yian Ma, Joseph M. Replogle, Eric S. Lander, Spyros Darmanis, Ivet Bahar, Vijay G. Sankaran, Jianhua Xing, and Jonathan S. Weissman. Mapping transcriptomic vector fields of single cells. *Cell*, 185(4):690–711.e45, 2022. doi: 10.1016/j.cell.2021.12.045.

Craig W. Reynolds. Flocks, herds and schools: A distributed behavioral model. In *Proceedings of SIGGRAPH '87*, pp. 25–34, 1987. URL https://www.red3d.com/cwr/papers/1987/boids.html.

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 *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/206b764b00195b8c896c5af8bde80726-Abstract-Conference.html.

Jongha Jon Ryu, Xiangxiang Xu, Hasan Sabri Melihcan Erol, Yuheng Bu, Lizhong Zheng, and Gregory W. Wornell. Operator SVD with neural networks via nested low-rank approximation. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 42870–42905, 2024. URL https://proceedings.mlr.press/v235/ryu24b.html.

Geoffrey Schiebinger, Jian Shu, Marcin Tabaka, Brian Cleary, Vidya Subramanian, Aryeh Solomon, Joshua Gould, Siyan Liu, Stacie Lin, Peter Berube, Lia Lee, Jenny Chen, Justin Brumbaugh, Philippe Rigollet, Konrad Hochedlinger, Rudolf Jaenisch, Aviv Regev, and Eric S. Lander. Optimal-transport analysis of single-cell gene expression identifies developmental trajectories in reprogramming. *Cell*, 176(4):928–943.e22, 2019. doi: 10.1016/j.cell.2019.01.006.

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

Mahmoud Selim, Sriharsha Bhat, and Karl H. Johansson. MetaKoopman: Bayesian meta-learning of Koopman operators for modeling structured dynamics under distribution shifts. In *Advances in Neural Information Processing Systems*, volume 38, pp. 29551–29584, 2025. doi: 10.52202/085713-0989. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/2a5dd6bad2d10966a8bfa254dfb7dc76-Abstract-Conference.html.

Yutong Sha, Yuchi Qiu, Peijie Zhou, and Qing Nie. Reconstructing growth and dynamic trajectories from single-cell transcriptomics data. *Nature Machine Intelligence*, 6(1):25–39, 2024. doi: 10.1038/s42256-023-00763-w.

Antonio Terpin, Nicolas Lanzetti, Martín Gadea, and Florian Dörfler. Learning diffusion at lightspeed. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-0218.

Alexander Tong, Jessie Huang, Guy Wolf, David Van Dijk, and Smita Krishnaswamy. TrajectoryNet: A dynamic optimal transport network for modeling cellular dynamics. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 9526–9536, 2020. URL https://proceedings.mlr.press/v119/tong20a.html.

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. URL https://openreview.net/forum?id=CD9Snc73AW.

Adrian Veres, Aubrey L. Faust, Henry L. Bushnell, Elise N. Engquist, Jennifer Hyoje-Ryu Kenty, George Harb, Yeh-Chuin Poh, Elad Sintov, Mads Gürtler, Felicia W. Pagliuca, Quinn P. Peterson, and Douglas A. Melton. Charting cellular identity during human in vitro β-cell differentiation. *Nature*, 569:368–373, 2019. doi: 10.1038/s41586-019-1168-5.

Francisco Villaescusa-Navarro, Daniel Anglés-Alcázar, Shy Genel, David N. Spergel, Rachel S. Somerville, Romeel Davé, Annalisa Pillepich, Lars Hernquist, Dylan Nelson, Paul Torrey, Desika Narayanan, Yin Li, Oliver Philcox, Valentina La Torre, Ana Maria Delgado, Shirley Ho, Sultan Hassan, Blakesley Burkhart, Digvijay Wadekar, Nicholas Battaglia, Gabriella Contardo, and Greg L. Bryan. The CAMELS project: Cosmology and astrophysics with machine-learning simulations. *The Astrophysical Journal*, 915(1):71, 2021. doi: 10.3847/1538-4357/abf7ba.

Michael E. Vinyard, Anders W. Rasmussen, Ruitong Li, Allon M. Klein, Gad Getz, and Luca Pinello. Learning cell dynamics with neural differential equations. *Nature Machine Intelligence*, 7(12):1969–1984, 2025. doi: 10.1038/s42256-025-01150-3.

Dongyi Wang, Yuanwei Jiang, Zhenyi Zhang, Xiang Gu, Peijie Zhou, and Jian Sun. Joint velocity-growth flow matching for single-cell dynamics modeling. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-5363.

Ziyue Wang and Yuko Araki. Functional time series forecasting of distributions: A Koopman-Wasserstein approach. *Behaviormetrika*, 53:571–597, 2026. doi: 10.1007/s41237-025-00278-1.

Philipp Weiler, Marius Lange, Michal Klein, Dana Pe'er, and Fabian Theis. CellRank 2: unified fate mapping in multiview single-cell data. *Nature Methods*, 21(7):1196–1205, 2024. doi: 10.1038/s41592-024-02303-9.

Caleb Weinreb, Alejo Rodriguez-Fraticelli, Fernando D. Camargo, and Allon M. Klein. Lineage tracing on transcriptional landscapes links state to fate during differentiation. *Science*, 367(6479): eaaw3381, 2020. doi: 10.1126/science.aaw3381.

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

Yuanchao Xu, Kaidi Shao, Nikos K. Logothetis, and Zhongwei Shen. ResKoopNet: Learning Koopman representations for complex dynamics with spectral residuals. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 69647–69674, 2025. URL https://proceedings.mlr.press/v267/xu25y.html.

Grace Hui Ting Yeo, Sachit D. Saksena, and David K. Gifford. Generative modeling of single-cell time series with PRESCIENT enables prediction of cell trajectories with interventions. *Nature Communications*, 12:3222, 2021. doi: 10.1038/s41467-021-23518-w.

Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabás Póczos, Ruslan Salakhutdinov, and Alexander J. Smola. Deep sets. In *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://arxiv.org/abs/1703.06114.

Jiaqi Zhang, Erica Larschan, Jeremy Bigness, and Ritambhara Singh. scNODE: Generative model for temporal single cell transcriptomic data prediction. *Bioinformatics*, 40(Supplement_2):ii146–ii154, 2024. doi: 10.1093/bioinformatics/btae393.

Stephen Zhang, Suryanarayana Maddu, Xiaojie Qiu, and Victor Chardès. Inferring stochastic dynamics with growth from cross-sectional data. In *Advances in Neural Information Processing Systems*, volume 38, pp. 126512–126545, 2025a. doi: 10.52202/085713-4219. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/b8013bbc033390cce2537ea50e2504e8-Abstract-Conference.html.

Zhenyi Zhang, Tiejun Li, and Peijie Zhou. Learning stochastic dynamics from snapshots through regularized unbalanced optimal transport. In *International Conference on Learning Representations*, 2025b. URL https://arxiv.org/abs/2410.00844.

This appendix provides theoretical foundations, experimental details, and supplementary analyses for DisKO. Appendix A describes the datasets, preprocessing, prediction protocols, and evaluation metrics. Appendix B presents model configurations, training settings, and complete distribution extrapolation results, including error curves and distribution visualizations. Appendix C details the analytical spectral references, the discrete and continuous DisKO implementations, and the evaluation of recovered eigenvalues and eigenfunctions. Appendices D, E, and F examine latent dynamical consistency, the contribution of Koopman latent dynamics through a Neural ODE comparison, and data efficiency on the OU benchmark, respectively. Appendix G develops the distributional Koopman representation, learned observable spaces and closure, and observation sufficiency and identifiability.

A Experimental Setup

A.1 Datasets and Preprocessing

Table A.1 summarizes the seven distribution prediction benchmarks. Each sequence comprises temporally ordered empirical distribution snapshots, which DisKO receives as sample sets without individual correspondences between times.

Table A.1: Distribution prediction datasets and temporal partitions. The observation dimension is p. Sequence counts are training/test counts for OU, Duffing, and CAMELS; the other datasets contain one sequence. Snapshot indices start at zero. Sample counts refer to the available data before evaluation subsampling.

Dataset	p	Sequences	Snapshots	Samples/snapshot	Training	Future
OU	2	192/64	101	1,024	0–50	51–100
Duffing	2	192/64	161	1,024	0–80	81–160
Boids	2	1	101	1,000	0–50	51–100
Ocean	2	1	11	400	0–9	10
CAMELS	10	21/6	9	4,096	0–5	6–8
Pancreas	30	1	8	3,562–12,477	0–5	6–7
WOT iPSC	50	1	39	1,956–12,416	0–32	33–38

Ornstein–Uhlenbeck (OU). The two-dimensional process satisfies

$$dX_t = FX_t\,dt + G\,dW_t, \qquad F = \begin{pmatrix} -0.25 & -2 \\ 2 & -0.25 \end{pmatrix}, \qquad G = \begin{pmatrix} 0.35 & 0 \\ 0 & 0.15 \end{pmatrix}. \tag{14}$$

Initial distributions are Gaussian mixtures with one, two, or three components. Component means are sampled uniformly from $[-2, 2]^2$, principal standard deviations from $[0.10, 0.35]$, and covariance orientations uniformly over a full rotation; mixture weights follow a symmetric Dirichlet distribution with concentration one. Each Gaussian component is propagated using its exact mean and covariance evolution, and 1,024 samples are drawn independently at each observation time. Snapshots are spaced by 0.05, with training through 2.5 and extrapolation over $[2.55, 5]$.

Duffing. The stochastic double-well oscillator is

$$dx_t = v_t\,dt, \qquad dv_t = (x_t - x_t^3 - 0.4v_t)\,dt + 0.35\,dW_t, \tag{15}$$

with observed state (x_t, v_t). We set mutilple initial laws, each corresponding to a different Gaussian distribution. For each law, 64 independent population simulations are generated with 5,000 particles using Euler–Maruyama integration at step size 0.005. Each saved snapshot contains 1,024 randomly selected particles, with sample order shuffled across times. Observations are spaced by 0.05 over $[0, 8]$; training ends at 4 and extrapolation covers 4.05 to 8. The training and test sequence split assesses predictions for independently sampled populations from these initial laws.

Boids and Ocean. Boids contains the planar positions of 1,000 interacting agents governed by cohesion, separation, and alignment rules with a confining boundary, following the flocking model of Reynolds (1987). Snapshots are spaced by 0.5, with training over $[0, 25]$ and extrapolation over $[25.5, 50]$. Ocean uses the large-vortex Gulf of Mexico benchmark distributed with WLM (Guan et al., 2026), based on a HYCOM ocean velocity field. Its time coordinate is the snapshot index, with observations 0 through 9 used for training and observation 10 for extrapolation. Both benchmarks are evaluated in planar position space.

CAMELS. We use 27 IllustrisTNG simulations from the cosmic variance set of CAMELS (Villaescusa-Navarro et al., 2021), with fixed cosmological parameters and different random initial conditions. Each sequence contains nine halo population snapshots, parameterized by the cosmological scale factor a. The observed ten-dimensional state comprises six periodic position coordinates $\{\sin(2\pi x_j/L), \cos(2\pi x_j/L)\}_{j=1}^{3}$, three velocity components, and $\log_{10} M_{200c}$, with box size $L = 25\,\mathrm{Mpc}/h$ and halo mass in $M_\odot/h$. Halos with positive mass and finite position and velocity are retained, and 4,096 are sampled without replacement per snapshot. The first six scale factors, approximately 0.143, 0.200, 0.263, 0.333, 0.423, and 0.537, form the training interval; predictions are evaluated at approximately 0.682, 0.866, and 1. The sequence split contains 21 training simulations and six test simulations, with no validation partition.

Pancreas. This benchmark describes human pancreatic β-cell differentiation (Veres et al., 2019). We use the preprocessed 30-dimensional representation released with DeepRUOTv2.[1] The resulting dataset contains 51,274 cells across eight ordered observations, with 6,190, 6,177, 3,562, 12,477, 6,480, 6,404, 6,013, and 3,971 cells, respectively. The supplied representation is used directly, and the observation index provides the time coordinate. The first six observations are used for training, and the final two for extrapolation.

WOT iPSC. The WOT reprogramming dataset tracks population changes during the induction of pluripotent stem cells (Schiebinger et al., 2019). Our processed dataset contains 236,285 cells at 39 observation times between day 0 and day 18. The 50-dimensional representation is obtained by PCA or truncated singular value decomposition fitted to cells pooled across observation times in training set. Training uses the 33 observations through day 15, and extrapolation uses the remaining six observations from day 15.5 through day 18. The recorded observation times are used in propagation, including the irregular sampling times within the training interval.

Circle and Torus. These synthetic systems provide analytical reference spectra for dynamical validation. On the circle, the angular process follows $d\theta_t = \omega\, dt + \sqrt{2\kappa}\, dW_t$ modulo 2π, with $\omega = 1$ and $\kappa = 0.01$, and observations are $(\cos\theta_t, \sin\theta_t)$. On the torus, two independent angular processes have $(\omega_1, \omega_2) = (1, 1.7)$ and $(\kappa_1, \kappa_2) = (0.01, 0.014)$, with observations given by the sine and cosine of each angle. Initial laws are mixtures of wrapped Gaussians. Both systems are observed every 0.1 time units over $[0, 12]$, with training through time 8. For $h_k(\mu) = \int e^{\mathrm{i}k\cdot\theta}\, d\mu(\theta)$, the distributional generator eigenvalue is

$$\lambda_k = -\sum_j \kappa_j k_j^2 + \mathrm{i}\sum_j \omega_j k_j, \tag{16}$$

where k is an integer on the circle and an integer pair on the torus. These reference modes permit direct assessment of learned oscillation frequencies and decay rates.

A.2 Prediction Protocol

The initial distribution and time partitions follow Section 3.1. OU, Duffing, and CAMELS have distinct training and test sequences, allowing evaluation of all four combinations of initial distribution split and time window. Boids, Ocean, Pancreas, and WOT iPSC contain a single sequence and use a temporal partition. For these datasets, Table B.3 reports predictions within the training window, and Table B.6 reports temporal extrapolation.

Every prediction is conditioned on the earliest snapshot of the sequence being evaluated. For a target time t_k, DisKO encodes $\widehat{\nu}_{t_0}^{(r)}$ and generates samples from $D(F_{t_k - t_0}(E(\widehat{\nu}_{t_0}^{(r)})))$. Training-window

[1] https://github.com/zhenyiizhang/DeepRUOTv2

scores also measure prediction from this initial snapshot. For a test sequence, its first snapshot is available as input; subsequent observations are used only for evaluation, with no updates from intervening targets. Time increments use the dataset's stated coordinate, including the source scale factor $a \approx 0.143$ in CAMELS.

For DisKO, coordinatewise means and standard deviations are estimated using only training sequences within the training time window and then fixed for prediction. Both single-cell datasets follow this standardization procedure. Evaluation uses predictions transformed back to the input coordinates, which are the reduced representations for Pancreas and WOT iPSC. Errors are computed separately for each sequence and target time and then averaged over the relevant sequence and temporal partition.

A.3 Evaluation Metrics

W1, SW1, W2, and squared maximum mean discrepancy quantify prediction errors, with smaller values indicating closer agreement. Table A.2 gives the standard sampling budgets. Sampling uses replacement when a snapshot contains fewer points than required. Evaluation sampling and random projections use seed zero, with separate random streams for each dataset, initial distribution split, population, prediction horizon, and sampling operation, independently of training randomness.

Table A.2: Standard evaluation sample counts per distribution. SW1 uses the full evaluation sample and 128 projection directions; MMD^2 uses at most 512 samples per distribution. The DeepRUOT evaluations on Pancreas and CAMELS use the weighted configuration described below.

Dataset	Evaluation samples	W1/W2 sample limit
OU, Duffing, Ocean	512	256
Boids	1,000	256
CAMELS, Pancreas, WOT iPSC	1,024	512

Wasserstein distances. Let $\{\widetilde{y}_i\}_{i=1}^n$ and $\{y_i\}_{i=1}^n$ denote equally sized samples from the predicted and target distributions. For $q \in \{1, 2\}$, we compute

$$\widehat{W}_q = \left(\frac{1}{n} \min_{\pi \in \mathfrak{S}_n} \sum_{i=1}^{n} \|\widetilde{y}_i - y_{\pi(i)}\|_2^q \right)^{1/q}, \tag{17}$$

where $\mathfrak{S}_n$ is the set of permutations of n elements. The sample count is the minimum of the prescribed transport limit and the available counts on both sides. The two matching problems use Euclidean and squared Euclidean costs, respectively, and are solved without entropic regularization.

Sliced Wasserstein distance. Following Bonneel et al. (2015), we approximate the first-order sliced distance using $L = 128$ independent directions u_ℓ obtained by normalizing standard Gaussian vectors in $\mathbb{R}^p$. If $\widetilde{s}_{\ell,(i)}$ and $s_{\ell,(i)}$ are the sorted projections of the predicted and target samples onto u_ℓ, then

$$\widehat{\mathrm{SW}}_1 = \frac{1}{Ln} \sum_{\ell=1}^{L} \sum_{i=1}^{n} |\widetilde{s}_{\ell,(i)} - s_{\ell,(i)}|. \tag{18}$$

Here n is the smaller of the two evaluation sample counts, using the full balanced sample before the optimal transport limit is applied.

Maximum mean discrepancy. We use the Gaussian kernel $k_\sigma(y, y') = \exp(-\|y - y'\|_2^2/(2\sigma^2))$ and the biased squared estimator (Gretton et al., 2012)

$$\widehat{\mathrm{MMD}}^2 = \frac{1}{n^2} \sum_{i,j=1}^{n} k_\sigma(\widetilde{y}_i, \widetilde{y}_j) + \frac{1}{n^2} \sum_{i,j=1}^{n} k_\sigma(y_i, y_j) - \frac{2}{n^2} \sum_{i,j=1}^{n} k_\sigma(\widetilde{y}_i, y_j). \tag{19}$$

This estimator includes diagonal terms and uses the first $n = \min(N_{\mathrm{pred}}, N_{\mathrm{target}}, 512)$ points of each evaluation sample, where N_{pred} and N_{target} are the two available sample counts. The bandwidth σ is the median pairwise Euclidean distance exceeding 10^{-12} in the pooled predicted and

target sample, with $\sigma = 1$ when no such distance exists. Entries labeled MMD in the result tables report this squared quantity.

Weighted evaluation for DeepRUOT. On Pancreas and CAMELS, DeepRUOT retains particle masses estimated by its growth model, normalized to total mass one. Evaluation uses 1,024 samples per distribution, weighted optimal transport with all samples for W1/W2, and 128 directions for SW1. MMD^2 also uses all 1,024 samples, with bandwidth determined by the target sample alone. These entries therefore use a larger transport budget and a different kernel bandwidth convention than the standard configuration.

B Distribution Extrapolation Experiments

Model architecture and implementation settings. All datasets use a DeepSets encoder, continuous affine latent dynamics, and a conditional flow matching decoder. Table B.1 summarizes the state dimension, encoder depth, network width, latent dimension, and decoder depth for each dataset.

Table B.1: DisKO model configurations for the distribution extrapolation benchmarks. Enc. and Dec. denote encoder and decoder, respectively.

Dataset	State dim.	Enc. depth	Width	Latent dim.	Dec. depth
OU	2	3	128	32	4
Duffing	2	3	256	64	8
Boids	2	4	512	32	4
Ocean	2	3	128	32	4
CAMELS	10	3	256	64	8
Pancreas	30	3	256	96	8
WOT iPSC	50	3	128	64	4

Training procedure. Each training pair comprises two snapshots (S_s, S_t) from the same sequence. We sample a valid source index uniformly, then a positive snapshot lag uniformly among those whose target remains in the training window. Particle subsets are sampled independently, and propagation uses the actual time difference $t - s$. Training uses varying source times; evaluation starts from the initial snapshot (Appendix A.2). We first pretrain E_θ, D_ψ on individual training snapshots using the conditional flow matching reconstruction loss in Equation 8, with dynamics fixed. We then freeze both networks and update only A, c, minimizing the mean squared difference between $F_{t-s}(E_\theta(S_s))$ and $E_\theta(S_t)$. Finally, we jointly update all three modules using the prediction, reconstruction, latent consistency, and distribution losses in Equation 11. The prediction branch conditions the decoder on the propagated encoding; the reconstruction branch uses the directly encoded target. Target encodings are detached only in the latent consistency term, so reconstruction continues to train the encoder. Each stage uses a fresh AdamW optimizer with gradient norm clipping. Table B.2 specifies stage budgets, learning rates, batch sizes, snapshot sample counts, and loss weights. The discrete variant initializes its operator by ridge regression and alternates encoder and decoder updates with operator refitting, as detailed in Appendix C.2.

Table B.2 reports the training and inference settings. The stage order pre/dyn/joint denotes encoder and decoder pretraining, latent dynamics fitting with the encoder and decoder fixed, and joint optimization, following Section 4.4. Batch denotes the training batch size, and samples/snapshot is the number of samples used from each snapshot during training. The endpoint/inference entries give the numbers of decoder integration steps used for endpoint supervision during training and for sample generation at inference, respectively.

Loss weights are listed in the order prediction FM, oracle FM, latent consistency, and endpoint distribution loss, followed after the semicolon by the MMD and moment weights,

$$\lambda_{\text{pred}}/\lambda_{\text{rec}}/\lambda_{\text{lat}}/\lambda_{\text{dist}};\ \beta/\gamma.$$

Here oracle FM is the reconstruction objective $\mathcal{J}_{\text{rec}}$ in Equation 9, which conditions the decoder on the encoded target snapshot. The endpoint loss $\mathcal{J}_{\text{dist}}$ uses $\text{SW}_1 + \beta\,\text{MMD}^2 + \gamma d_{\text{mom}}$, as defined in

Equation 10; its contribution to the joint objective is weighted by λ_{dist}. Duffing and Pancreas use W_1 for the endpoint discrepancy, with the same outer weight λ_{dist}. The dashes in their MMD and moment entries indicate that these two terms are not used.

Table B.2: DisKO training and inference settings. Stage updates and learning rates follow the order pre/dyn/joint. The two panels report complementary settings for the same seven datasets. Loss weights follow the order specified in the text.

(a) Optimization and training samples

Dataset	Stage updates ($\times 10^3$)	Stage learning rates ($\times 10^{-4}$)	Batch	Samples/ snapshot
OU	$10/1/2.5$	$3/3/3$	4	512
Duffing	$30/3/15$	$3/3/3$	4	512
Boids	$30/30/75$	$3/1/1$	4	512
Ocean	$10/10/20$	$3/1/1$	4	400
CAMELS	$40/3/15$	$3/3/3$	4	1024
Pancreas	$60/3/24$	$3/3/3$	8	512
WOT iPSC	$15/1.5/4$	$3/3/3$	8	512

(b) Loss weights and decoder integration

Dataset	Loss weights	Endpoint loss	Endpoint/ inference steps
OU	$1/.5/.05/.2;\ .25/.5$	$\mathcal{J}_{\text{dist}}$	$8/32$
Duffing	$1/.5/.05/.2;$ –/–	W_1	$8/32$
Boids	$1/.25/.05/.5;\ .1/1$	$\mathcal{J}_{\text{dist}}$	$16/32$
Ocean	$1/.5/.2/.2;\ .25/.5$	$\mathcal{J}_{\text{dist}}$	$8/32$
CAMELS	$1/.5/.05/.2;\ .25/.5$	$\mathcal{J}_{\text{dist}}$	$8/32$
Pancreas	$1/.5/.05/.2;$ –/–	W_1	$8/32$
WOT iPSC	$1/.5/.05/.2;\ .25/.5$	$\mathcal{J}_{\text{dist}}$	$8/32$

Prediction error across the training and extrapolation intervals. Figure B.1 compares the W_1 prediction errors of DisKO, DeepRUOT, MIOFlow, and WLM across the training and extrapolation intervals for all seven datasets. Each panel uses prediction step on the horizontal axis and a logarithmic error scale on the vertical axis. Vertical dashed lines mark the final training step, and the shaded regions identify extrapolation. DisKO has the lowest displayed extrapolation errors among these four methods on OU, Duffing, Ocean, Pancreas, and WOT iPSC. Its OU error remains approximately level throughout the future interval, while its Duffing error increases gradually with the prediction horizon. For Ocean, the error increases from the final training step at step 9 to the single future target at step 10, while remaining below the three comparison methods. The Pancreas and WOT iPSC panels likewise show lower future errors for DisKO despite some increase beyond the training boundary.

CAMELS and Boids exhibit closer comparisons between DisKO and WLM. On CAMELS, errors increase with the prediction horizon for all four methods; DisKO and WLM remain close near the end of the interval and finish below DeepRUOT and MIOFlow.

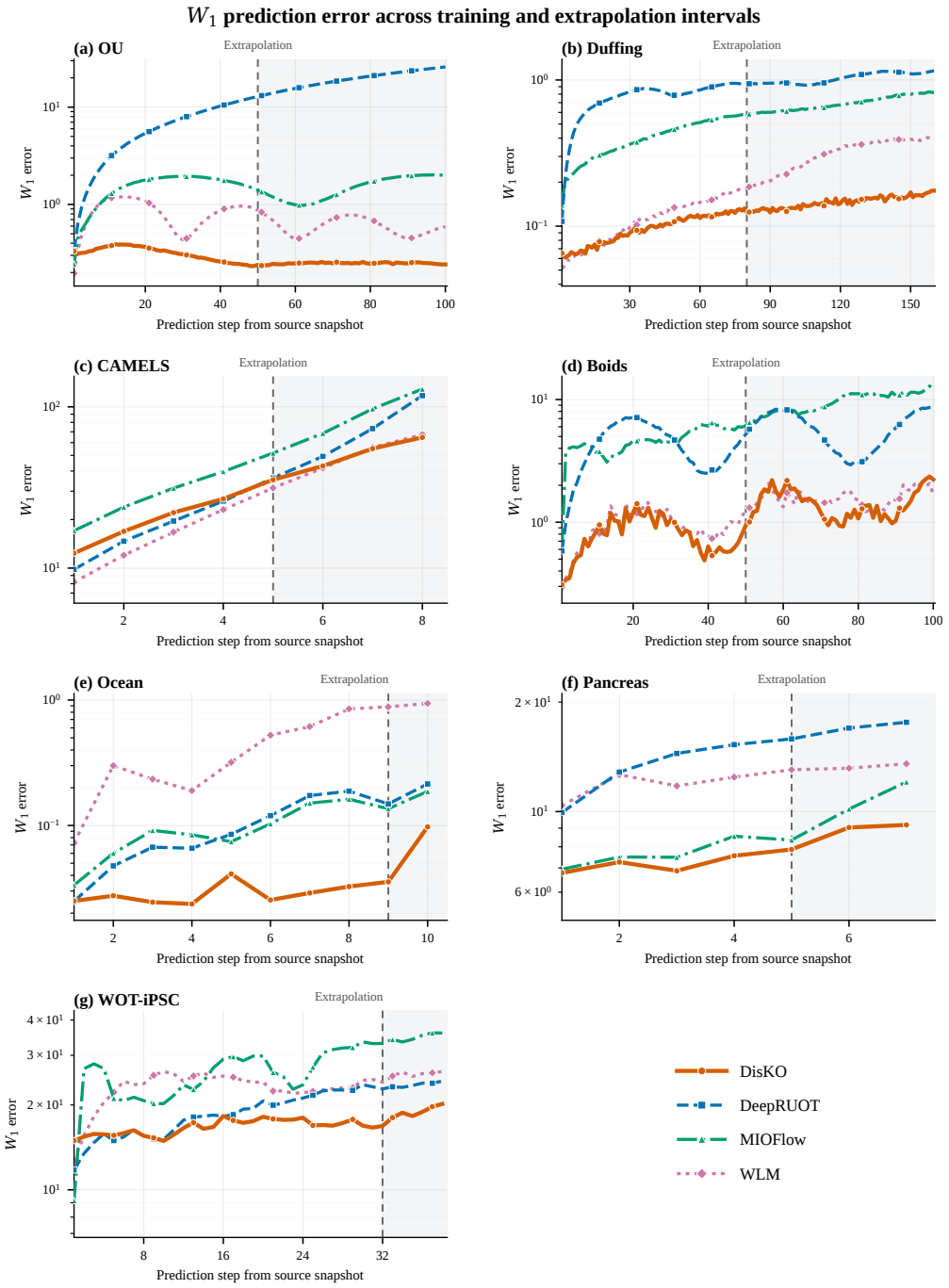

Figure B.1: W_1 prediction error across the training and extrapolation intervals for (a) OU, (b) Duffing, (c) CAMELS, (d) Boids, (e) Ocean, (f) Pancreas, and (g) WOT iPSC. Curves compare DisKO, DeepRUOT, MIOFlow, and WLM. Vertical dashed lines mark the training boundary, and shaded regions indicate extrapolation.

Tables B.3 through B.6 provide the complete aggregate numerical results for W1, SW1, W2, and MMD across the four combinations of time window and initial distribution split. These supporting tables retain all eight baselines and DisKO.

Table B.3: Prediction within the training time window for training initial distributions and single sequences. Boids, Ocean, Pancreas, and WOT iPSC each contain a single population sequence and use only a temporal partition. Means and standard deviations are rounded to three decimal places. Red and blue mark the lowest and second lowest means before rounding.

Method	Metric	OU	Duffing	Boids	Ocean	CAMELS	Pancreas	WOT iPSC
TrajectoryNet	W_1	$1.536_{\pm 0.026}$	$0.788_{\pm 0.067}$	$4.715_{\pm 0.131}$	$1.508_{\pm 0.098}$	$46.276_{\pm 0.186}$	$20.423_{\pm 0.497}$	$35.166_{\pm 1.357}$
	SW1	$0.956_{\pm 0.029}$	$0.493_{\pm 0.053}$	$2.758_{\pm 0.308}$	$0.979_{\pm 0.076}$	$11.758_{\pm 0.333}$	$2.488_{\pm 0.201}$	$2.847_{\pm 0.291}$
	W_2	$1.592_{\pm 0.027}$	$0.843_{\pm 0.057}$	$5.595_{\pm 0.135}$	$1.513_{\pm 0.097}$	$52.473_{\pm 0.268}$	$20.984_{\pm 0.539}$	$54.721_{\pm 2.050}$
	MMD	$0.813_{\pm 0.015}$	$0.621_{\pm 0.102}$	$0.264_{\pm 0.044}$	$1.002_{\pm 0.038}$	$0.175_{\pm 0.008}$	$0.329_{\pm 0.054}$	$0.666_{\pm 0.099}$
DeepRUOT	W_1	$1.221_{\pm 0.020}$	$0.773_{\pm 0.145}$	$4.342_{\pm 0.546}$	$0.102_{\pm 0.031}$	$20.784_{\pm 0.084}$	$13.669_{\pm 1.096}$	$18.648_{\pm 1.053}$
	SW1	$0.768_{\pm 0.024}$	$0.484_{\pm 0.091}$	$2.715_{\pm 0.358}$	$0.064_{\pm 0.020}$	$3.683_{\pm 0.103}$	$1.340_{\pm 0.319}$	$1.441_{\pm 0.178}$
	W_2	$1.282_{\pm 0.021}$	$0.827_{\pm 0.147}$	$4.816_{\pm 0.488}$	$0.117_{\pm 0.028}$	$25.479_{\pm 0.135}$	$14.714_{\pm 2.030}$	$18.847_{\pm 1.069}$
	MMD	$0.638_{\pm 0.013}$	$0.510_{\pm 0.137}$	$0.245_{\pm 0.047}$	$0.305_{\pm 0.118}$	$0.016_{\pm 0.001}$	$0.079_{\pm 0.043}$	$0.157_{\pm 0.041}$
VGFM	W_1	$1.184_{\pm 0.099}$	$0.431_{\pm 0.184}$	$2.736_{\pm 0.667}$	$0.055_{\pm 0.006}$	$29.751_{\pm 1.353}$	$12.183_{\pm 0.845}$	$33.836_{\pm 1.578}$
	SW1	$0.733_{\pm 0.064}$	$0.258_{\pm 0.114}$	$1.601_{\pm 0.510}$	$0.032_{\pm 0.003}$	$5.864_{\pm 0.719}$	$0.796_{\pm 0.072}$	$2.591_{\pm 0.958}$
	W_2	$1.237_{\pm 0.099}$	$0.542_{\pm 0.201}$	$3.270_{\pm 0.746}$	$0.066_{\pm 0.006}$	$35.337_{\pm 1.433}$	$14.099_{\pm 1.294}$	$190.213_{\pm 1.063}$
	MMD	$0.582_{\pm 0.042}$	$0.108_{\pm 0.044}$	$0.084_{\pm 0.047}$	$0.072_{\pm 0.012}$	$0.028_{\pm 0.020}$	$0.033_{\pm 0.014}$	$0.077_{\pm 0.029}$
MIOFlow	W_1	$1.462_{\pm 0.047}$	$0.412_{\pm 0.078}$	$4.703_{\pm 0.988}$	$0.099_{\pm 0.021}$	$32.147_{\pm 0.733}$	$7.765_{\pm 0.166}$	$25.994_{\pm 0.673}$
	SW1	$0.911_{\pm 0.030}$	$0.252_{\pm 0.051}$	$2.811_{\pm 0.404}$	$0.063_{\pm 0.014}$	$7.527_{\pm 0.215}$	$0.462_{\pm 0.038}$	$2.533_{\pm 0.108}$
	W_2	$1.522_{\pm 0.050}$	$0.467_{\pm 0.085}$	$5.723_{\pm 0.579}$	$0.116_{\pm 0.024}$	$37.471_{\pm 0.599}$	$8.321_{\pm 0.187}$	$26.271_{\pm 0.691}$
	MMD	$0.692_{\pm 0.007}$	$0.188_{\pm 0.066}$	$0.156_{\pm 0.063}$	$0.278_{\pm 0.085}$	$0.077_{\pm 0.002}$	$0.009_{\pm 0.002}$	$0.358_{\pm 0.035}$
scNODE	W_1	$1.191_{\pm 0.043}$	$0.834_{\pm 0.097}$	$4.796_{\pm 0.057}$	$0.782_{\pm 0.105}$	$53.869_{\pm 2.635}$	$20.511_{\pm 1.862}$	$43.246_{\pm 1.129}$
	SW1	$0.750_{\pm 0.028}$	$0.520_{\pm 0.072}$	$2.750_{\pm 0.033}$	$0.508_{\pm 0.063}$	$13.103_{\pm 0.891}$	$2.505_{\pm 0.509}$	$4.715_{\pm 0.759}$
	W_2	$1.253_{\pm 0.042}$	$0.888_{\pm 0.078}$	$5.577_{\pm 0.107}$	$0.787_{\pm 0.104}$	$60.330_{\pm 2.823}$	$21.064_{\pm 2.162}$	$43.377_{\pm 1.057}$
	MMD	$0.582_{\pm 0.014}$	$0.449_{\pm 0.089}$	$0.091_{\pm 0.006}$	$0.971_{\pm 0.062}$	$0.175_{\pm 0.026}$	$0.336_{\pm 0.064}$	$0.707_{\pm 0.214}$
PRESCIENT	W_1	$1.452_{\pm 0.024}$	$0.259_{\pm 0.004}$	$3.100_{\pm 0.017}$	$0.962_{\pm 0.012}$	$45.111_{\pm 0.095}$	$11.523_{\pm 0.217}$	$25.992_{\pm 0.046}$
	SW1	$0.902_{\pm 0.015}$	$0.141_{\pm 0.003}$	$1.871_{\pm 0.018}$	$0.599_{\pm 0.007}$	$11.546_{\pm 0.024}$	$0.989_{\pm 0.039}$	$2.379_{\pm 0.008}$
	W_2	$1.502_{\pm 0.025}$	$0.332_{\pm 0.003}$	$3.318_{\pm 0.022}$	$0.976_{\pm 0.012}$	$51.258_{\pm 0.090}$	$12.234_{\pm 0.137}$	$26.084_{\pm 0.046}$
	MMD	$0.683_{\pm 0.002}$	$0.074_{\pm 0.002}$	$0.123_{\pm 0.003}$	$0.894_{\pm 0.004}$	$0.169_{\pm 0.001}$	$0.079_{\pm 0.008}$	$0.373_{\pm 0.002}$
JKOnet*	W_1	$1.655_{\pm 0.027}$	$0.405_{\pm 0.035}$	$3.080_{\pm 0.086}$	$1.838_{\pm 0.305}$	$37.951_{\pm 0.151}$	$47.086_{\pm 1.003}$	$29.582_{\pm 1.154}$
	SW1	$1.028_{\pm 0.031}$	$0.221_{\pm 0.024}$	$1.844_{\pm 0.047}$	$1.166_{\pm 0.196}$	$9.282_{\pm 0.265}$	$7.248_{\pm 0.599}$	$2.561_{\pm 0.258}$
	W_2	$1.707_{\pm 0.028}$	$0.723_{\pm 0.048}$	$3.363_{\pm 0.096}$	$1.946_{\pm 0.374}$	$51.292_{\pm 0.274}$	$59.232_{\pm 1.185}$	$43.680_{\pm 1.353}$
	MMD	$0.703_{\pm 0.014}$	$0.047_{\pm 0.009}$	$0.093_{\pm 0.001}$	$0.913_{\pm 0.050}$	$0.078_{\pm 0.004}$	$0.290_{\pm 0.048}$	$0.339_{\pm 0.051}$
WLM	W_1	$0.709_{\pm 0.006}$	$0.116_{\pm 0.004}$	$0.934_{\pm 0.059}$	$0.443_{\pm 0.118}$	$16.601_{\pm 0.066}$	$12.053_{\pm 0.330}$	$23.056_{\pm 1.109}$
	SW1	$0.428_{\pm 0.003}$	$0.042_{\pm 0.002}$	$0.219_{\pm 0.064}$	$0.284_{\pm 0.262}$	$1.251_{\pm 0.023}$	$0.840_{\pm 0.057}$	$1.680_{\pm 0.129}$
	W_2	$0.762_{\pm 0.006}$	$0.165_{\pm 0.006}$	$1.300_{\pm 0.072}$	$0.824_{\pm 0.435}$	$20.246_{\pm 0.106}$	$12.852_{\pm 0.332}$	$23.401_{\pm 1.094}$
	MMD	$0.350_{\pm 0.006}$	$0.002_{\pm 0.000}$	$0.002_{\pm 0.001}$	$0.480_{\pm 0.241}$	$0.002_{\pm 0.000}$	$0.048_{\pm 0.006}$	$0.186_{\pm 0.023}$
DisKO	W_1	$0.279_{\pm 0.005}$	$0.097_{\pm 0.001}$	$0.816_{\pm 0.011}$	$0.029_{\pm 0.004}$	$16.634_{\pm 0.035}$	$7.268_{\pm 0.046}$	$16.700_{\pm 0.217}$
	SW1	$0.135_{\pm 0.004}$	$0.033_{\pm 0.001}$	$0.176_{\pm 0.016}$	$0.015_{\pm 0.003}$	$1.312_{\pm 0.072}$	$0.345_{\pm 0.011}$	$0.614_{\pm 0.073}$
	W_2	$0.337_{\pm 0.004}$	$0.139_{\pm 0.001}$	$1.261_{\pm 0.014}$	$0.034_{\pm 0.006}$	$20.622_{\pm 0.065}$	$7.737_{\pm 0.037}$	$16.874_{\pm 0.203}$
	MMD	$0.025_{\pm 0.002}$	$0.002_{\pm 0.000}$	$0.001_{\pm 0.000}$	$0.017_{\pm 0.011}$	$0.002_{\pm 0.000}$	$0.006_{\pm 0.000}$	$0.032_{\pm 0.005}$

Table B.4: Prediction within the training time window for test initial distributions. A hyphen (-) denotes the absence of a separate set of test initial distributions for a single sequence. Means and standard deviations are rounded to three decimal places. Red and blue mark the lowest and second lowest means before rounding.

Method	Metric	OU	Duffing	Boids	Ocean	CAMELS	Pancreas	WOT iPSC
TrajectoryNet	W_1	$1.585_{\pm 0.044}$	$0.751_{\pm 0.064}$	-	-	$45.835_{\pm 0.506}$	-	-
	SW1	$0.985_{\pm 0.034}$	$0.469_{\pm 0.050}$	-	-	$11.567_{\pm 0.342}$	-	-
	W_2	$1.648_{\pm 0.031}$	$0.820_{\pm 0.055}$	-	-	$53.090_{\pm 0.311}$	-	-
	MMD	$0.706_{\pm 0.022}$	$0.499_{\pm 0.098}$	-	-	$0.186_{\pm 0.015}$	-	-
DeepRUOT	W_1	$1.263_{\pm 0.034}$	$0.772_{\pm 0.146}$	-	-	$21.261_{\pm 0.725}$	-	-
	SW1	$0.797_{\pm 0.027}$	$0.484_{\pm 0.092}$	-	-	$3.622_{\pm 0.106}$	-	-
	W_2	$1.322_{\pm 0.025}$	$0.827_{\pm 0.149}$	-	-	$27.025_{\pm 0.165}$	-	-
	MMD	$0.625_{\pm 0.020}$	$0.510_{\pm 0.137}$	-	-	$0.014_{\pm 0.001}$	-	-
VGFM	W_1	$1.226_{\pm 0.127}$	$0.429_{\pm 0.183}$	-	-	$31.046_{\pm 1.957}$	-	-
	SW1	$0.758_{\pm 0.082}$	$0.256_{\pm 0.113}$	-	-	$6.050_{\pm 1.064}$	-	-
	W_2	$1.281_{\pm 0.127}$	$0.536_{\pm 0.200}$	-	-	$37.239_{\pm 2.114}$	-	-
	MMD	$0.581_{\pm 0.060}$	$0.108_{\pm 0.044}$	-	-	$0.031_{\pm 0.026}$	-	-
MIOFlow	W_1	$1.547_{\pm 0.088}$	$0.411_{\pm 0.078}$	-	-	$32.735_{\pm 0.356}$	-	-
	SW1	$0.963_{\pm 0.057}$	$0.251_{\pm 0.051}$	-	-	$7.662_{\pm 0.227}$	-	-
	W_2	$1.613_{\pm 0.099}$	$0.466_{\pm 0.084}$	-	-	$39.520_{\pm 0.231}$	-	-
	MMD	$0.707_{\pm 0.021}$	$0.188_{\pm 0.067}$	-	-	$0.081_{\pm 0.003}$	-	-
scNODE	W_1	$1.266_{\pm 0.046}$	$0.834_{\pm 0.096}$	-	-	$54.432_{\pm 2.139}$	-	-
	SW1	$0.800_{\pm 0.027}$	$0.521_{\pm 0.071}$	-	-	$13.294_{\pm 0.570}$	-	-
	W_2	$1.325_{\pm 0.046}$	$0.888_{\pm 0.077}$	-	-	$61.763_{\pm 2.481}$	-	-
	MMD	$0.589_{\pm 0.019}$	$0.450_{\pm 0.087}$	-	-	$0.190_{\pm 0.019}$	-	-
PRESCIENT	W_1	$1.542_{\pm 0.027}$	$0.258_{\pm 0.004}$	-	-	$44.631_{\pm 0.115}$	-	-
	SW1	$0.956_{\pm 0.016}$	$0.141_{\pm 0.004}$	-	-	$11.321_{\pm 0.027}$	-	-
	W_2	$1.595_{\pm 0.027}$	$0.331_{\pm 0.003}$	-	-	$51.860_{\pm 0.128}$	-	-
	MMD	$0.690_{\pm 0.003}$	$0.074_{\pm 0.003}$	-	-	$0.163_{\pm 0.001}$	-	-
JKOnet*	W_1	$1.728_{\pm 0.046}$	$0.401_{\pm 0.035}$	-	-	$39.020_{\pm 0.434}$	-	-
	SW1	$1.071_{\pm 0.037}$	$0.218_{\pm 0.023}$	-	-	$9.559_{\pm 0.282}$	-	-
	W_2	$1.783_{\pm 0.034}$	$0.703_{\pm 0.047}$	-	-	$52.598_{\pm 0.322}$	-	-
	MMD	$0.696_{\pm 0.022}$	$0.047_{\pm 0.009}$	-	-	$0.074_{\pm 0.006}$	-	-
WLM	W_1	$0.843_{\pm 0.006}$	$0.116_{\pm 0.004}$	-	-	$18.274_{\pm 0.102}$	-	-
	SW1	$0.510_{\pm 0.003}$	$0.041_{\pm 0.002}$	-	-	$1.895_{\pm 0.055}$	-	-
	W_2	$0.900_{\pm 0.006}$	$0.165_{\pm 0.006}$	-	-	$23.334_{\pm 0.140}$	-	-
	MMD	$0.397_{\pm 0.004}$	$0.002_{\pm 0.000}$	-	-	$0.004_{\pm 0.000}$	-	-
DisKO	W_1	$0.315_{\pm 0.009}$	$0.098_{\pm 0.001}$	-	-	$22.710_{\pm 0.466}$	-	-
	SW1	$0.154_{\pm 0.006}$	$0.033_{\pm 0.001}$	-	-	$4.100_{\pm 0.239}$	-	-
	W_2	$0.372_{\pm 0.007}$	$0.139_{\pm 0.001}$	-	-	$28.157_{\pm 0.463}$	-	-
	MMD	$0.031_{\pm 0.003}$	$0.002_{\pm 0.000}$	-	-	$0.024_{\pm 0.003}$	-	-

Table B.5: Temporal extrapolation from training initial distributions in datasets with multiple sequences. Hyphens (-) mark single sequences, whose temporal extrapolation results are reported in Table B.6. Means and standard deviations are rounded to three decimal places. Red and blue mark the lowest and second lowest means before rounding.

Method	Metric	OU	Duffing	Boids	Ocean	CAMELS	Pancreas	WOT iPSC
TrajectoryNet	W_1	$1.279_{\pm 0.055}$	$12.544_{\pm 1.069}$	-	-	$145.995_{\pm 1.209}$	-	-
	SW1	$0.800_{\pm 0.033}$	$7.876_{\pm 0.975}$	-	-	$36.488_{\pm 1.768}$	-	-
	W_2	$1.339_{\pm 0.052}$	$26.279_{\pm 1.501}$	-	-	$167.079_{\pm 1.926}$	-	-
	MMD	$0.656_{\pm 0.019}$	$0.523_{\pm 0.138}$	-	-	$0.212_{\pm 0.020}$	-	-
DeepRUOT	W_1	$0.765_{\pm 0.033}$	$1.033_{\pm 0.290}$	-	-	$78.561_{\pm 0.676}$	-	-
	SW1	$0.492_{\pm 0.020}$	$0.651_{\pm 0.187}$	-	-	$14.119_{\pm 0.687}$	-	-
	W_2	$0.814_{\pm 0.032}$	$1.100_{\pm 0.281}$	-	-	$93.868_{\pm 1.043}$	-	-
	MMD	$0.575_{\pm 0.017}$	$0.522_{\pm 0.140}$	-	-	$0.019_{\pm 0.002}$	-	-
VGFM	W_1	$1.247_{\pm 0.477}$	$11.542_{\pm 0.528}$	-	-	$117.983_{\pm 2.040}$	-	-
	SW1	$0.780_{\pm 0.309}$	$8.395_{\pm 1.109}$	-	-	$24.104_{\pm 1.296}$	-	-
	W_2	$1.303_{\pm 0.469}$	$50.033_{\pm 1.136}$	-	-	$138.186_{\pm 2.053}$	-	-
	MMD	$0.653_{\pm 0.234}$	$0.525_{\pm 0.107}$	-	-	$0.060_{\pm 0.057}$	-	-
MIOFlow	W_1	$1.424_{\pm 0.057}$	$0.692_{\pm 0.109}$	-	-	$98.199_{\pm 1.144}$	-	-
	SW1	$0.895_{\pm 0.034}$	$0.432_{\pm 0.071}$	-	-	$20.917_{\pm 1.353}$	-	-
	W_2	$1.507_{\pm 0.044}$	$0.790_{\pm 0.142}$	-	-	$115.136_{\pm 1.277}$	-	-
	MMD	$0.669_{\pm 0.014}$	$0.318_{\pm 0.085}$	-	-	$0.058_{\pm 0.023}$	-	-
scNODE	W_1	$0.738_{\pm 0.008}$	$0.886_{\pm 0.075}$	-	-	$145.715_{\pm 1.770}$	-	-
	SW1	$0.468_{\pm 0.006}$	$0.541_{\pm 0.071}$	-	-	$36.200_{\pm 1.752}$	-	-
	W_2	$0.806_{\pm 0.008}$	$0.952_{\pm 0.056}$	-	-	$166.598_{\pm 1.898}$	-	-
	MMD	$0.399_{\pm 0.012}$	$0.321_{\pm 0.093}$	-	-	$0.202_{\pm 0.019}$	-	-
PRESCIENT	W_1	$0.768_{\pm 0.032}$	$0.534_{\pm 0.041}$	-	-	$147.606_{\pm 0.199}$	-	-
	SW1	$0.474_{\pm 0.019}$	$0.282_{\pm 0.025}$	-	-	$38.188_{\pm 0.053}$	-	-
	W_2	$0.814_{\pm 0.031}$	$0.663_{\pm 0.053}$	-	-	$169.867_{\pm 0.200}$	-	-
	MMD	$0.446_{\pm 0.009}$	$0.093_{\pm 0.007}$	-	-	$0.282_{\pm 0.001}$	-	-
JKOnet*	W_1	$1.223_{\pm 0.051}$	$5.267_{\pm 0.452}$	-	-	$130.758_{\pm 1.101}$	-	-
	SW1	$0.761_{\pm 0.032}$	$3.300_{\pm 0.403}$	-	-	$29.383_{\pm 1.434}$	-	-
	W_2	$1.263_{\pm 0.048}$	$8.796_{\pm 0.707}$	-	-	$156.289_{\pm 1.761}$	-	-
	MMD	$0.584_{\pm 0.017}$	$0.179_{\pm 0.048}$	-	-	$0.059_{\pm 0.005}$	-	-
WLM	W_1	$0.534_{\pm 0.033}$	$0.316_{\pm 0.027}$	-	-	$52.000_{\pm 0.323}$	-	-
	SW1	$0.318_{\pm 0.021}$	$0.159_{\pm 0.019}$	-	-	$6.013_{\pm 0.212}$	-	-
	W_2	$0.599_{\pm 0.033}$	$0.411_{\pm 0.033}$	-	-	$65.203_{\pm 0.326}$	-	-
	MMD	$0.223_{\pm 0.020}$	$0.028_{\pm 0.010}$	-	-	$0.004_{\pm 0.000}$	-	-
DisKO	W_1	$0.238_{\pm 0.014}$	$0.145_{\pm 0.005}$	-	-	$47.995_{\pm 0.404}$	-	-
	SW1	$0.129_{\pm 0.011}$	$0.056_{\pm 0.004}$	-	-	$6.142_{\pm 0.401}$	-	-
	W_2	$0.279_{\pm 0.014}$	$0.195_{\pm 0.008}$	-	-	$62.773_{\pm 0.711}$	-	-
	MMD	$0.029_{\pm 0.006}$	$0.003_{\pm 0.001}$	-	-	$0.005_{\pm 0.001}$	-	-

Table B.6: Temporal extrapolation from test initial distributions and single sequences. Boids, Ocean, Pancreas, and WOT iPSC each contain a single population sequence and use only a temporal partition. Means and standard deviations are rounded to three decimal places. Red and blue mark the lowest and second lowest means before rounding.

Method	Metric	OU	Duffing	Boids	Ocean	CAMELS	Pancreas	WOT iPSC
TrajectoryNet	W_1	$1.369_{\pm 0.071}$	$9.222_{\pm 0.783}$	$11.843_{\pm 0.560}$	$2.198_{\pm 0.900}$	$143.767_{\pm 2.170}$	$47.376_{\pm 2.950}$	$118.524_{\pm 2.509}$
	SW1	$0.857_{\pm 0.047}$	$5.815_{\pm 0.712}$	$7.267_{\pm 0.365}$	$1.426_{\pm 0.556}$	$35.364_{\pm 1.128}$	$6.765_{\pm 0.826}$	$14.502_{\pm 0.647}$
	W_2	$1.443_{\pm 0.067}$	$17.953_{\pm 1.433}$	$15.612_{\pm 0.973}$	$2.204_{\pm 0.893}$	$166.789_{\pm 2.871}$	$53.487_{\pm 1.372}$	$321.936_{\pm 2.588}$
	MMD	$0.666_{\pm 0.052}$	$0.523_{\pm 0.140}$	$0.348_{\pm 0.078}$	$0.957_{\pm 0.074}$	$0.233_{\pm 0.024}$	$0.372_{\pm 0.101}$	$0.628_{\pm 0.086}$
DeepRUOT	W_1	$0.846_{\pm 0.044}$	$1.032_{\pm 0.287}$	$5.962_{\pm 0.283}$	$0.214_{\pm 0.020}$	$79.989_{\pm 1.252}$	$17.286_{\pm 1.572}$	$23.635_{\pm 1.623}$
	SW1	$0.554_{\pm 0.030}$	$0.650_{\pm 0.186}$	$3.796_{\pm 0.172}$	$0.149_{\pm 0.014}$	$14.362_{\pm 0.466}$	$2.113_{\pm 0.221}$	$2.108_{\pm 0.220}$
	W_2	$0.892_{\pm 0.042}$	$1.099_{\pm 0.278}$	$6.356_{\pm 0.349}$	$0.274_{\pm 0.020}$	$96.028_{\pm 1.647}$	$18.801_{\pm 2.136}$	$23.822_{\pm 1.575}$
	MMD	$0.593_{\pm 0.046}$	$0.523_{\pm 0.141}$	$0.331_{\pm 0.090}$	$0.598_{\pm 0.062}$	$0.022_{\pm 0.002}$	$0.122_{\pm 0.050}$	$0.227_{\pm 0.046}$
VGFM	W_1	$1.290_{\pm 0.486}$	$12.834_{\pm 0.976}$	$31.008_{\pm 1.709}$	$0.195_{\pm 0.047}$	$123.071_{\pm 2.075}$	$27.497_{\pm 1.337}$	$187.154_{\pm 2.894}$
	SW1	$0.807_{\pm 0.315}$	$9.358_{\pm 0.352}$	$19.604_{\pm 1.428}$	$0.128_{\pm 0.034}$	$24.995_{\pm 17.602}$	$2.978_{\pm 2.126}$	$27.389_{\pm 1.762}$
	W_2	$1.352_{\pm 0.478}$	$62.176_{\pm 1.946}$	$32.798_{\pm 1.546}$	$0.210_{\pm 0.047}$	$145.189_{\pm 2.565}$	$49.009_{\pm 1.408}$	$370.921_{\pm 2.025}$
	MMD	$0.651_{\pm 0.242}$	$0.526_{\pm 0.109}$	$0.440_{\pm 0.251}$	$0.248_{\pm 0.100}$	$0.065_{\pm 0.066}$	$0.091_{\pm 0.048}$	$0.166_{\pm 0.032}$
MIOFlow	W_1	$1.514_{\pm 0.135}$	$0.692_{\pm 0.109}$	$9.642_{\pm 0.390}$	$0.187_{\pm 0.044}$	$98.173_{\pm 1.421}$	$11.111_{\pm 0.902}$	$34.854_{\pm 0.627}$
	SW1	$0.950_{\pm 0.087}$	$0.432_{\pm 0.070}$	$5.879_{\pm 1.216}$	$0.130_{\pm 0.033}$	$20.381_{\pm 1.104}$	$0.895_{\pm 0.171}$	$3.931_{\pm 0.053}$
	W_2	$1.612_{\pm 0.161}$	$0.804_{\pm 0.166}$	$13.322_{\pm 1.066}$	$0.233_{\pm 0.048}$	$115.750_{\pm 2.881}$	$12.305_{\pm 1.198}$	$34.916_{\pm 0.620}$
	MMD	$0.693_{\pm 0.062}$	$0.320_{\pm 0.084}$	$0.221_{\pm 0.109}$	$0.492_{\pm 0.163}$	$0.057_{\pm 0.023}$	$0.027_{\pm 0.013}$	$0.479_{\pm 0.012}$
scNODE	W_1	$0.759_{\pm 0.010}$	$0.887_{\pm 0.075}$	$7.822_{\pm 0.359}$	$0.783_{\pm 0.246}$	$143.751_{\pm 1.692}$	$43.047_{\pm 1.030}$	$94.268_{\pm 1.536}$
	SW1	$0.481_{\pm 0.008}$	$0.541_{\pm 0.070}$	$4.381_{\pm 0.233}$	$0.515_{\pm 0.154}$	$34.896_{\pm 1.137}$	$6.168_{\pm 0.831}$	$10.544_{\pm 1.150}$
	W_2	$0.827_{\pm 0.010}$	$0.952_{\pm 0.056}$	$9.633_{\pm 0.649}$	$0.789_{\pm 0.239}$	$166.307_{\pm 1.709}$	$44.610_{\pm 2.552}$	$94.623_{\pm 1.388}$
	MMD	$0.403_{\pm 0.015}$	$0.324_{\pm 0.091}$	$0.060_{\pm 0.004}$	$1.302_{\pm 0.104}$	$0.211_{\pm 0.018}$	$0.659_{\pm 0.139}$	$1.077_{\pm 0.250}$
PRESCIENT	W_1	$0.849_{\pm 0.034}$	$0.532_{\pm 0.040}$	$3.498_{\pm 0.136}$	$0.153_{\pm 0.009}$	$143.797_{\pm 0.232}$	$12.169_{\pm 0.325}$	$23.328_{\pm 0.086}$
	SW1	$0.525_{\pm 0.020}$	$0.280_{\pm 0.025}$	$1.951_{\pm 0.087}$	$0.097_{\pm 0.006}$	$36.244_{\pm 0.062}$	$0.965_{\pm 0.068}$	$1.883_{\pm 0.013}$
	W_2	$0.894_{\pm 0.032}$	$0.660_{\pm 0.052}$	$3.845_{\pm 0.148}$	$0.186_{\pm 0.007}$	$167.681_{\pm 0.273}$	$12.873_{\pm 0.393}$	$23.397_{\pm 0.086}$
	MMD	$0.483_{\pm 0.007}$	$0.093_{\pm 0.007}$	$0.077_{\pm 0.005}$	$0.400_{\pm 0.055}$	$0.281_{\pm 0.001}$	$0.054_{\pm 0.008}$	$0.177_{\pm 0.002}$
JKOnet*	W_1	$1.288_{\pm 0.067}$	$5.246_{\pm 0.447}$	$3.282_{\pm 0.187}$	$2.332_{\pm 0.876}$	$134.105_{\pm 2.073}$	$52.395_{\pm 1.119}$	$63.051_{\pm 1.367}$
	SW1	$0.797_{\pm 0.044}$	$3.290_{\pm 0.413}$	$1.770_{\pm 0.086}$	$1.561_{\pm 0.587}$	$30.058_{\pm 0.976}$	$8.221_{\pm 0.713}$	$7.076_{\pm 0.325}$
	W_2	$1.328_{\pm 0.060}$	$8.726_{\pm 0.704}$	$3.857_{\pm 0.286}$	$2.743_{\pm 0.891}$	$160.195_{\pm 2.727}$	$76.620_{\pm 1.496}$	$75.024_{\pm 2.428}$
	MMD	$0.594_{\pm 0.046}$	$0.180_{\pm 0.048}$	$0.053_{\pm 0.007}$	$0.868_{\pm 0.089}$	$0.062_{\pm 0.006}$	$0.303_{\pm 0.095}$	$0.206_{\pm 0.028}$
WLM	W_1	$0.602_{\pm 0.032}$	$0.318_{\pm 0.028}$	$1.595_{\pm 0.047}$	$0.941_{\pm 0.264}$	$55.019_{\pm 0.370}$	$13.345_{\pm 0.517}$	$25.805_{\pm 0.638}$
	SW1	$0.362_{\pm 0.020}$	$0.160_{\pm 0.019}$	$0.656_{\pm 0.050}$	$0.520_{\pm 0.246}$	$6.699_{\pm 0.206}$	$0.988_{\pm 0.140}$	$1.929_{\pm 0.093}$
	W_2	$0.688_{\pm 0.037}$	$0.413_{\pm 0.034}$	$2.152_{\pm 0.165}$	$1.845_{\pm 0.848}$	$71.177_{\pm 0.327}$	$13.969_{\pm 0.611}$	$26.921_{\pm 1.159}$
	MMD	$0.255_{\pm 0.019}$	$0.028_{\pm 0.010}$	$0.011_{\pm 0.002}$	$0.366_{\pm 0.127}$	$0.005_{\pm 0.000}$	$0.055_{\pm 0.015}$	$0.169_{\pm 0.023}$
DisKO	W_1	$0.249_{\pm 0.013}$	$0.146_{\pm 0.005}$	$1.454_{\pm 0.110}$	$0.097_{\pm 0.024}$	$54.241_{\pm 0.830}$	$9.114_{\pm 0.143}$	$18.973_{\pm 0.330}$
	SW1	$0.135_{\pm 0.010}$	$0.056_{\pm 0.004}$	$0.656_{\pm 0.087}$	$0.062_{\pm 0.019}$	$7.039_{\pm 0.500}$	$0.559_{\pm 0.007}$	$0.940_{\pm 0.039}$
	W_2	$0.294_{\pm 0.014}$	$0.195_{\pm 0.008}$	$1.827_{\pm 0.104}$	$0.115_{\pm 0.023}$	$71.260_{\pm 1.239}$	$9.372_{\pm 0.109}$	$19.356_{\pm 0.324}$
	MMD	$0.030_{\pm 0.005}$	$0.003_{\pm 0.001}$	$0.011_{\pm 0.004}$	$0.085_{\pm 0.046}$	$0.007_{\pm 0.001}$	$0.013_{\pm 0.000}$	$0.036_{\pm 0.004}$

Prediction distribution visualizations. Figures B.2 through B.8 compare the reference and predicted distributions for OU, Duffing, Boids, Ocean, CAMELS, Pancreas, and WOT iPSC, respectively. The grid figures show the reference, DisKO, DeepRUOT, MIOFlow, and WLM in successive rows, with columns covering selected training and extrapolation times and the source snapshot repeated in each row. Reference contours provide a common spatial comparison at each target time. Ocean uses one panel per method, with time encoded by color. OU, Duffing, Boids, and Ocean are shown in their two-dimensional observation coordinates; CAMELS is shown through halo mass and speed, and the two single-cell datasets through their first two principal component coordinates.

For OU (Figure B.2), DisKO follows the location and spread of the compact reference cloud over the displayed future times. DeepRUOT predictions move outside the plotting range, while MIOFlow and WLM produce displaced or deformed distributions. For Duffing (Figure B.3), DisKO retains the two prominent lobes of the future reference distribution. DeepRUOT concentrates near the center, MIOFlow produces a narrow band, and WLM increasingly concentrates along thin curved structures.

For Boids (Figure B.4), DisKO broadly reproduces the changes between ring, multiple-lobe, and compact configurations, with visible blurring and displacement at later times. DisKO also recovers substantial spatial structure. DeepRUOT and MIOFlow miss much of the spatial extent or arrangement of the future reference populations. For Ocean (Figure B.5), DisKO follows the sequence of reference locations and places the future population near the reference endpoint, with a residual offset and differences in spread. For CAMELS (Figure B.6), both DisKO and WLM reproduce much of the increasing spread in the mass and speed coordinates, while differences in the tails and local density remain visible.

For Pancreas (Figure B.7), DisKO captures the dominant compact population at the two future observations, with remaining differences in orientation and diffuse tails. For WOT iPSC (Figure B.8), DisKO follows much of the changing population shape, while the predictions at the latest times remain displaced toward lower values of the second principal component relative to the reference.

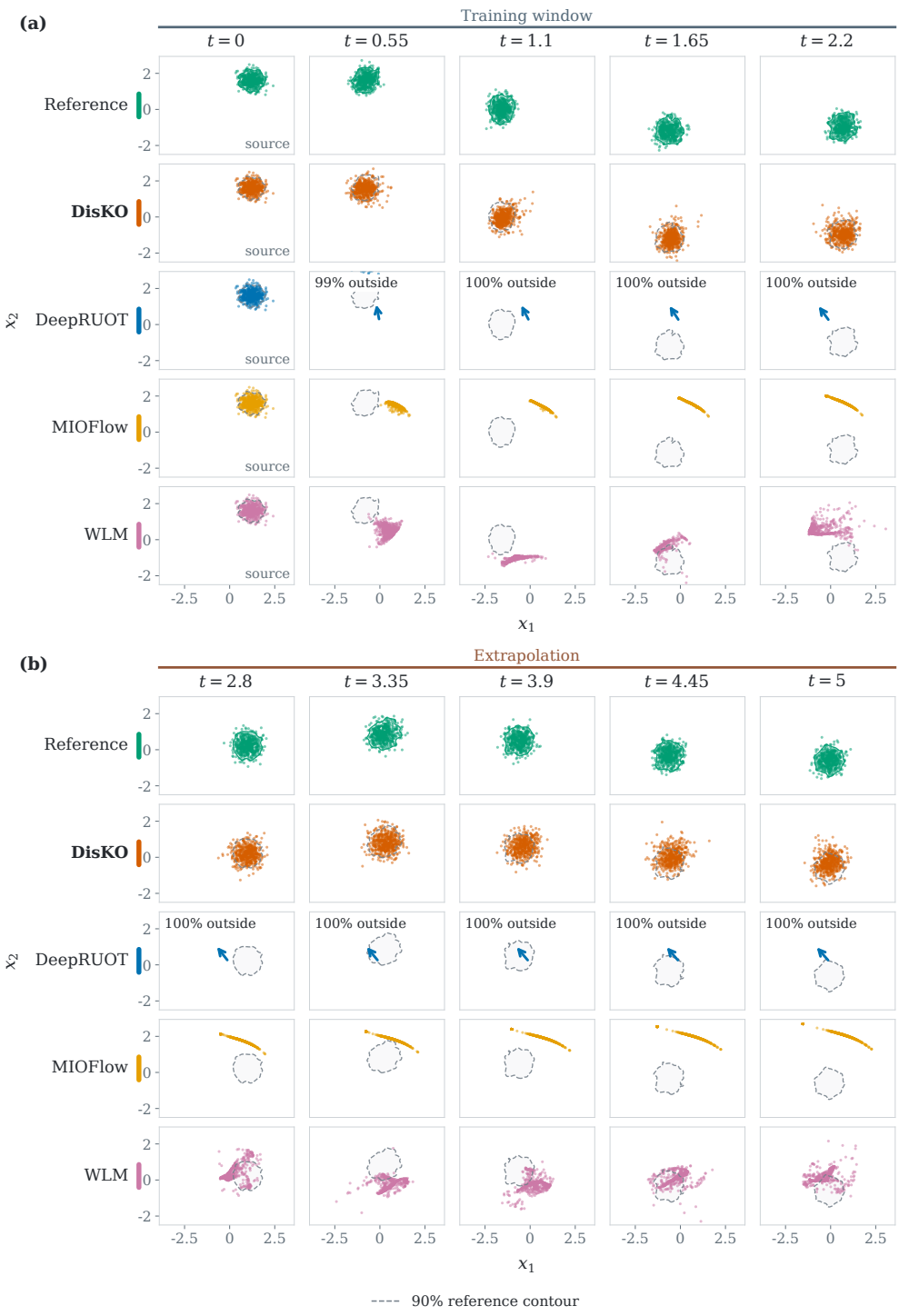

Figure B.2: OU prediction distributions in the observation coordinates x_1 and x_2. Rows show the reference distribution, DisKO, DeepRUOT, MIOFlow, and WLM; columns show the times printed above each panel. Panel (a) covers the training window and panel (b) covers extrapolation. Dashed contours indicate the 90% reference contour.

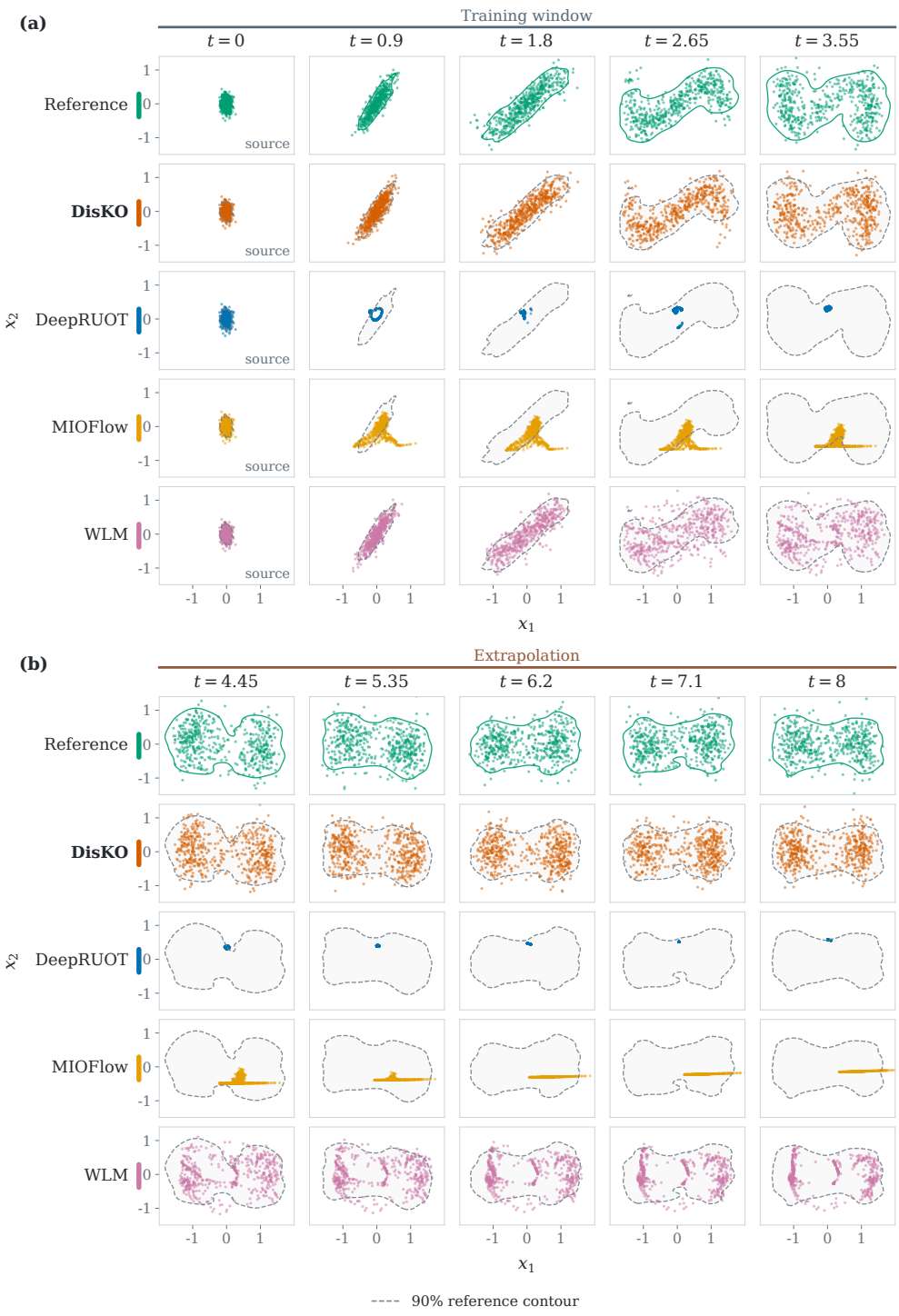

Figure B.3: Duffing prediction distributions in the observation coordinates x_1 and x_2. Rows show the reference distribution, DisKO, DeepRUOT, MIOFlow, and WLM. Panel (a) covers the training window and panel (b) covers extrapolation, at the times shown above the columns. Dashed contours indicate the 90% reference contour.

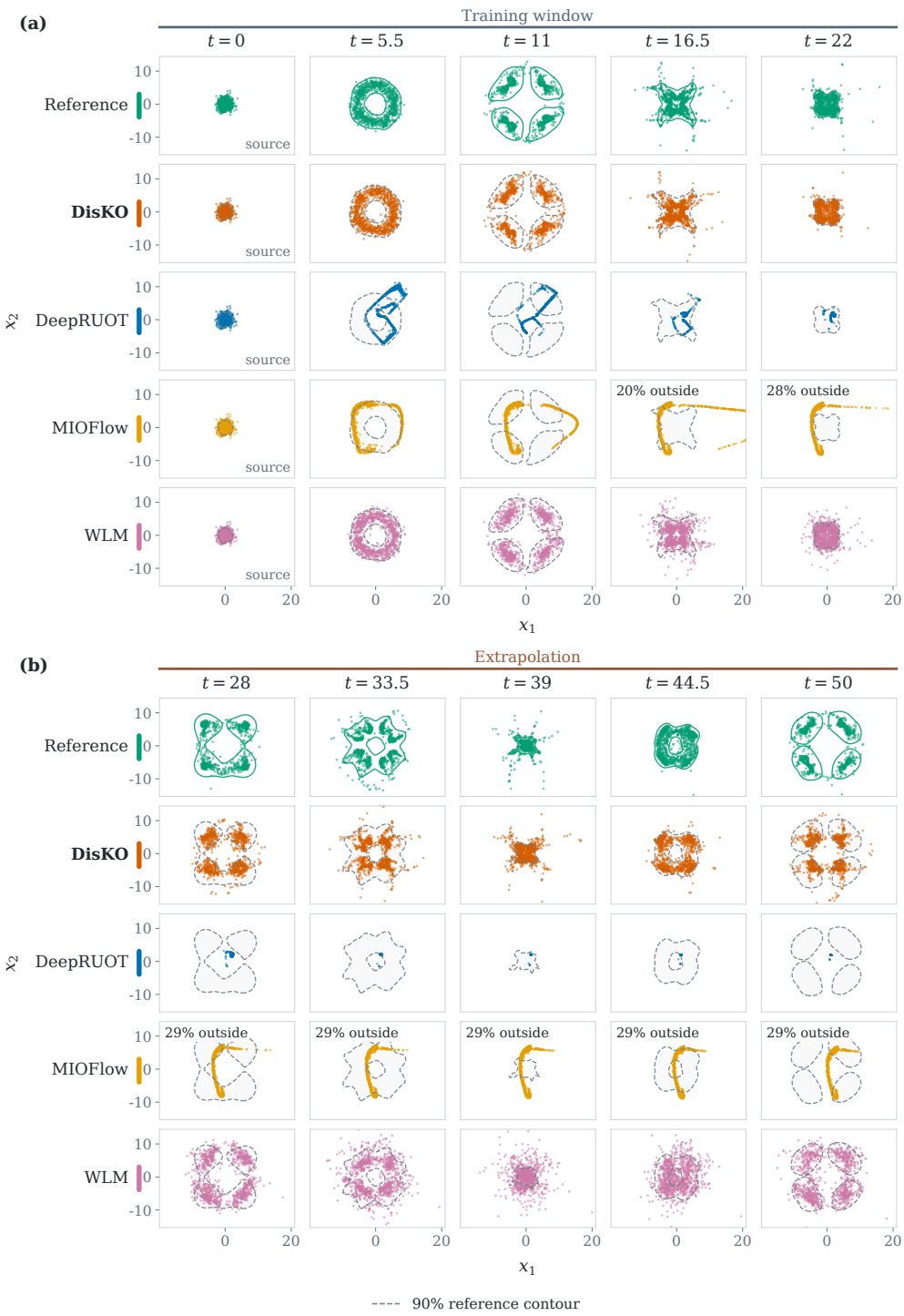

Figure B.4: Boids prediction distributions in the observation coordinates x_1 and x_2. Rows show the reference distribution, DisKO, DeepRUOT, MIOFlow, and WLM. Panel (a) covers the training window and panel (b) covers extrapolation, at the times shown above the columns. Dashed contours indicate the 90% reference contour.

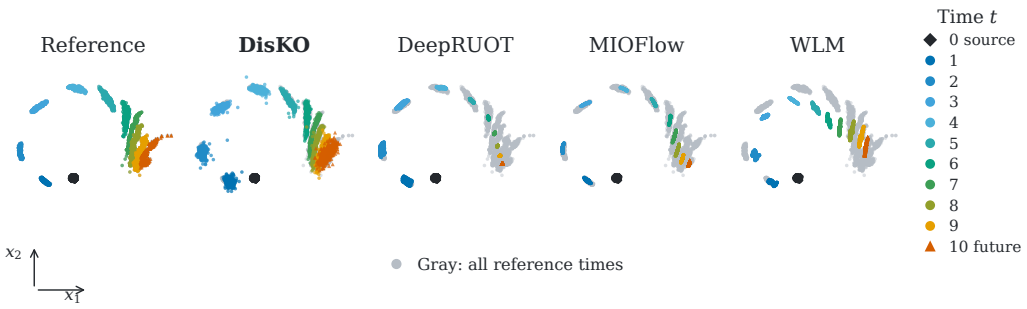

Figure B.5: Ocean prediction distributions in the observation coordinates x_1 and x_2. Panels show the reference distribution, DisKO, DeepRUOT, MIOFlow, and WLM from left to right. Colors indicate time, with the source at $t = 0$ and the future observation at $t = 10$. Gray points show reference samples from all times.

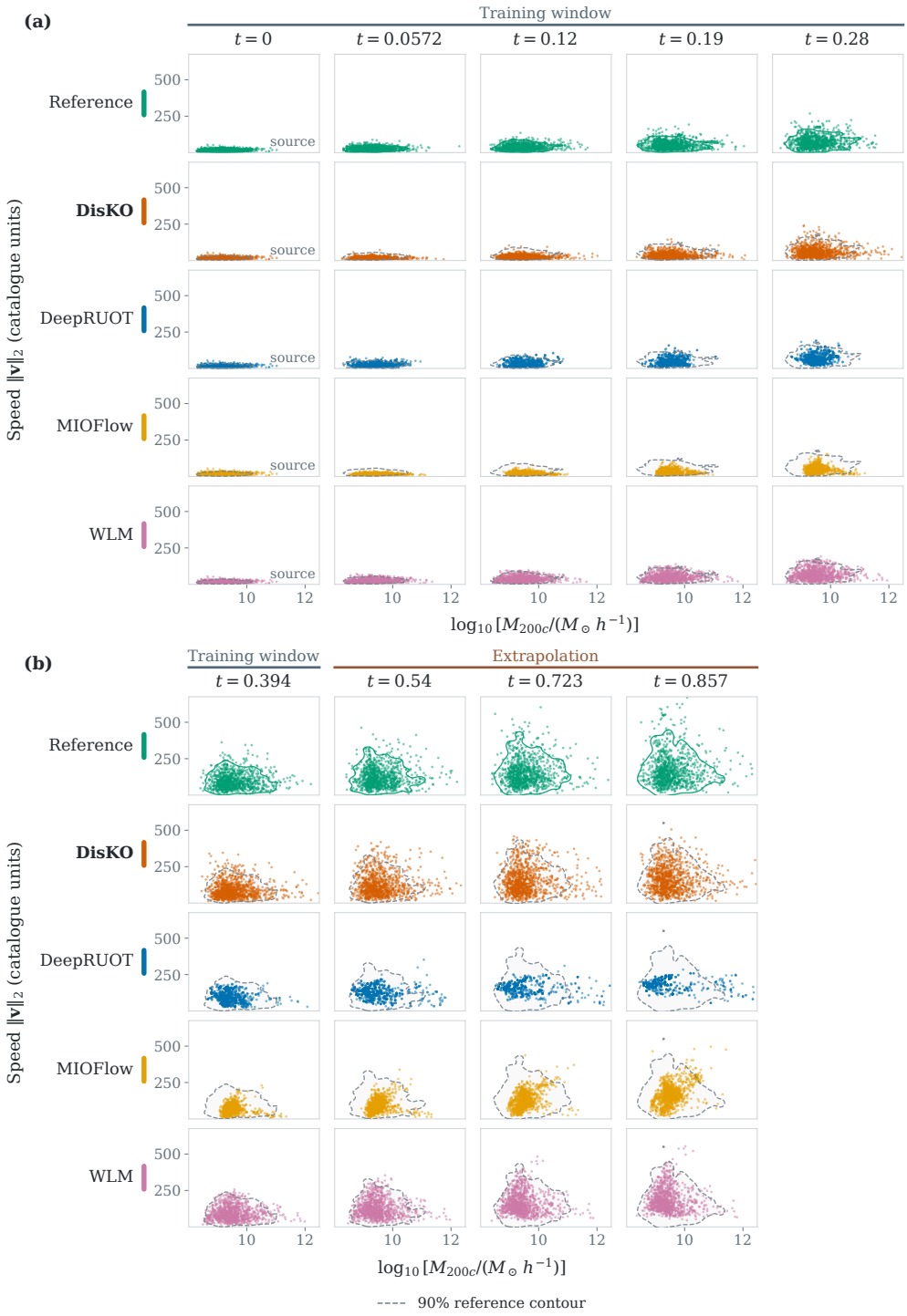

Figure B.6: CAMELS prediction distributions over halo mass and speed, with coordinates $\log_{10}[M_{200c}/(M_\odot h^{-1})]$ and $\|v\|_2$ in catalogue units. Rows show the reference distribution and the four methods in the same order as Figure B.2. The bars above the columns identify training and extrapolation times. Dashed contours indicate the 90% reference contour.

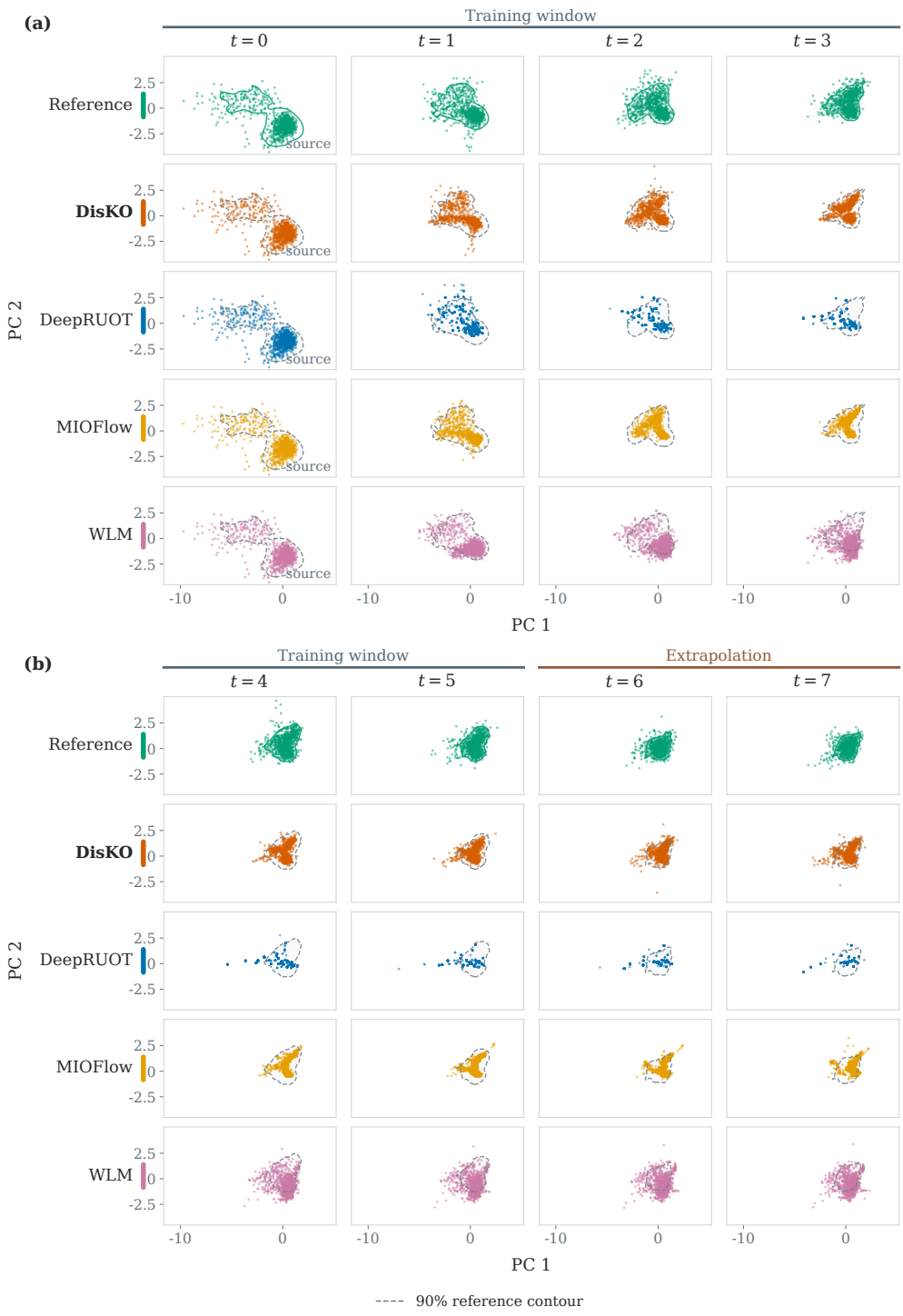

Figure B.7: Pancreas prediction distributions in the first two principal component coordinates. Rows show the reference distribution, DisKO, DeepRUOT, MIOFlow, and WLM. Columns show the indicated times; the bars identify the training window and extrapolation at $t = 6$ and $t = 7$. Dashed contours indicate the 90% reference contour.

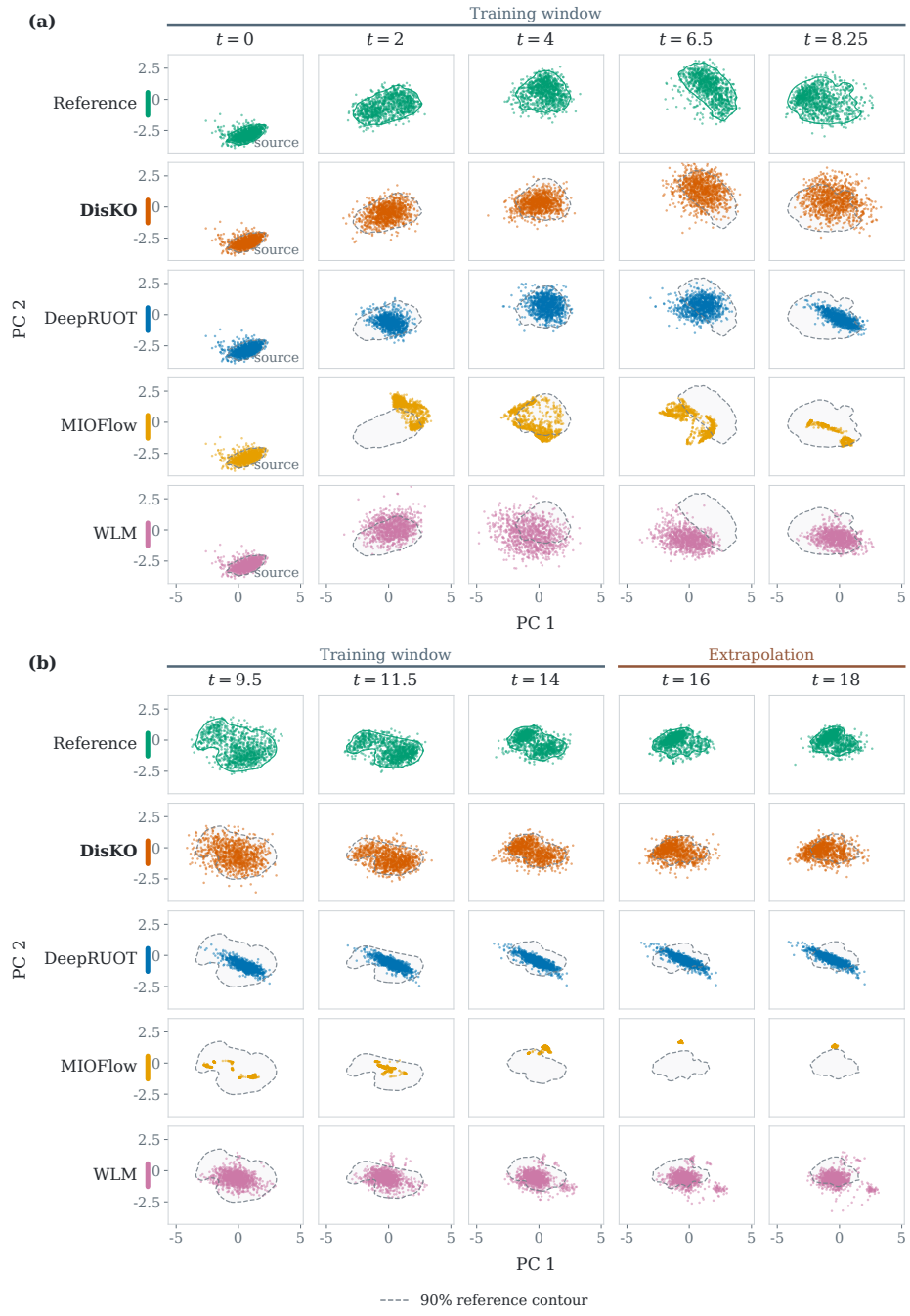

Figure B.8: WOT iPSC prediction distributions in the first two principal component coordinates. Rows show the reference distribution, DisKO, DeepRUOT, MIOFlow, and WLM. Columns show the indicated times, with training and extrapolation intervals identified by the bars above them. Dashed contours indicate the 90% reference contour.

C KOOPMAN SPECTRAL APPROXIMATION

We examine the approximation of Koopman eigenvalues and eigenfunctions on Circle and Torus using discrete and continuous parameterizations of DisKO. Following the terminology of Brunton et al. (2022); Klus et al. (2020), an eigenfunction is a scalar observable satisfying an eigenvalue relation with the operator or its generator. Here its argument is a probability measure, so it is a distribution observable as defined in Section 3.2. We reserve the term Koopman mode for a coefficient vector in the eigenfunction expansion of a vector-valued observable. The figures below display eigenvalue estimates and evaluations of learned eigenfunctions on distributions.

C.1 ANALYTICAL EIGENVALUES AND EIGENFUNCTIONS

Write $\vartheta_t \in \mathbb{T}^q$ for the angular state, where $q = 1$ for Circle and $q = 2$ for Torus. The dynamics and the generator acting on smooth periodic state observables are

$$d\vartheta_{j,t} = \omega_j\, dt + \sqrt{2\kappa_j}\, dW_{j,t} \pmod{2\pi}, \qquad \mathcal{G}f = \sum_{j=1}^{q} \left(\omega_j \partial_j f + \kappa_j \partial_{jj} f\right), \tag{20}$$

with independent Brownian motions. Circle uses $(\omega_1, \kappa_1) = (1, 0.01)$; Torus uses $\omega = (1, 1.7)$ and $\kappa = (0.01, 0.014)$. For $k \in \mathbb{Z}^q$, the Fourier function $\phi_k(\vartheta) = e^{\mathrm{i}k\cdot\vartheta}$ satisfies

$$\mathcal{G}\phi_k = \lambda_k \phi_k, \qquad \lambda_k = -\sum_{j=1}^{q} \kappa_j k_j^2 + \mathrm{i}\sum_{j=1}^{q} \omega_j k_j. \tag{21}$$

This follows directly from $\partial_j \phi_k = \mathrm{i}k_j \phi_k$ and $\partial_{jj}\phi_k = -k_j^2 \phi_k$. The real and imaginary parts of λ_k determine the decay rate and angular frequency, respectively.

Let $\mu_t = T_t \mu_0$ be the angular state distribution and define $h_k(\mu) = \int \phi_k\, d\mu$. Taking expectations in Itô's formula yields

$$\frac{d}{dt} h_k(\mu_t) = \int \mathcal{G}\phi_k\, d\mu_t = \lambda_k h_k(\mu_t), \qquad (\mathcal{K}_t h_k)(\mu) = e^{t\lambda_k} h_k(\mu). \tag{22}$$

Thus h_k is an eigenfunction of the distributional Koopman generator $\mathcal{L}$ with eigenvalue λ_k, and of $\mathcal{K}_{\Delta t}$ with eigenvalue $\rho_k = e^{\Delta t \lambda_k}$. The observation maps are $P(\vartheta) = (\cos\vartheta, \sin\vartheta)$ on Circle and the concatenated sine and cosine pairs on Torus. They are injective on the respective periodic state spaces, so the observed distribution $\nu = P_{\#}\mu$ determines $h_k(\mu)$.

All spectral comparisons use $\Delta t = 0.1$. Circle has $\lambda_k = -0.01k^2 + \mathrm{i}k$, with target indices $k = 1, \ldots, 6$. Torus has $\lambda_{(m,n)} = -(0.01m^2 + 0.014n^2) + \mathrm{i}(m + 1.7n)$, with target indices $(1,0)$, $(0,1)$, $(1,1)$, $(1,-1)$, $(2,0)$, and $(0,2)$. These prescribe a finite set of reference eigenpairs. The constant eigenfunction has generator eigenvalue zero and operator eigenvalue one; the index $-k$ gives the complex conjugate of the eigenpair for k.

For an initial mixture of wrapped Gaussians with weights π_ℓ, centers a_ℓ, and angular covariance matrices Σ_ℓ, the reference values at every time are available analytically,

$$h_k(\mu_t) = e^{t\lambda_k} \sum_{\ell} \pi_\ell \exp\!\left(\mathrm{i}k \cdot a_\ell - \tfrac{1}{2} k^\top \Sigma_\ell k\right). \tag{23}$$

Reference values use the initial mixture parameters and Equation 23; learned values are computed from finite empirical snapshots.

C.2 DISCRETE AND CONTINUOUS DISKO

The two models share the encoder and conditional flow decoder architecture, training distributions, particle subsets, and pretrained weights within each dataset. The encoder has a Deep Sets construction of depth three and width 256, with latent dimension 64; the decoder velocity network has depth five and width 256. Both systems use 128 training population sequences with 1,024 particles per snapshot. Circle has 16 validation and 32 test sequences, and Torus has 24 validation and 48 test sequences. Training uses 81 snapshots over $0 \le t \le 8$; the subsequent 40 snapshots cover

$8.1 \leq t \leq 12$. We report the final models after 10,000 joint optimization steps, with seed zero for both parameterizations.

For the discrete model, $z_{n+1} = Kz_n + b$ at interval Δt. For the continuous model, $\dot{z} = Az + c$ and propagation uses an exact matrix exponential. With $\bar{z} = [z^\top, 1]^\top$, these become

$$\bar{z}_{n+1} = M\bar{z}_n, \quad M = \begin{pmatrix} K & b \\ 0 & 1 \end{pmatrix}, \qquad \bar{z}(t + \Delta t) = e^{\Delta t B}\bar{z}(t), \quad B = \begin{pmatrix} A & c \\ 0 & 0 \end{pmatrix}. \tag{24}$$

If $M^\top w = \widehat{\rho} w$, the scalar function $u(\nu) = w^\top [E(\nu)^\top, 1]^\top$ evolves by $\widehat{\rho}$ under the discrete latent model. If $B^\top w = \widehat{\lambda} w$, the analogous continuous latent observable evolves by $e^{t\widehat{\lambda}}$. Under exact encoder closure these are Koopman eigenfunctions on the represented distribution family; in the learned models they are candidate approximations, evaluated below against h_k. The transpose is the ordinary transpose, consistent with the complex bilinear expression $w^\top \bar{z}$. Only the known constant eigenfunction introduced by augmentation is removed from the computed eigenpairs.

The discrete operator is refitted by ridge regression after re-encoding all 10,240 adjacent training snapshot pairs. Let X contain augmented source encodings as columns, Y the corresponding target encodings, and $\Theta = [K\ b]$. The update is

$$\widehat{\Theta} = \operatorname*{argmin}_{\Theta} \|Y - \Theta X\|_F^2 + \eta \|\Theta\|_F^2 = YX^\top (XX^\top + \eta I)^{-1}, \qquad \eta = 10^{-3}. \tag{25}$$

The residual term is a sum over snapshot pairs, and the penalty includes the affine offset. The implementation evaluates this solution by a singular value decomposition in double precision. The refitted dynamics are held fixed during each subsequent gradient update of the encoder and decoder. The continuous model optimizes A, c by differentiating through the matrix exponential. Each ridge refit minimizes the regularized one-step latent residual for the current encoder. This may help preserve harmonic structure as the learned observable space changes, although the different initialization procedures and dynamics update budgets prevent isolating its contribution from this comparison. Table C.1 summarizes the optimization differences.

Table C.1: Training configurations for the spectral experiments. Pretraining weights are shared within each dataset. The initialization procedures and dynamics update budgets differ between the two parameterizations.

Setting	DisKO (discrete)	DisKO (continuous)
Encoder/decoder pretraining	40,000 shared steps	40,000 shared steps
Dynamics initialization	Full ridge fit	6,000 gradient steps
Joint optimization	10,000 steps	10,000 steps
Dynamics update	Full ridge refit at each step	AdamW, learning rate 3×10^{-4}
Dynamics regularization	Ridge 10^{-3} on K, b	Zero weight decay on A, c
Encoder/decoder optimizer	AdamW, learning rate 3×10^{-4}, weight decay 10^{-5}	Same
Joint training horizons	$1, 2, 4, 8, 16, 32$ steps	Same
Training batch	8 snapshot pairs, 1,024 particles per snapshot	Same

Continuous dynamics initialization uses horizons $1, 2, 4$ for its first 3,000 updates and $1, 2, 4, 8, 16, 32$ for its remaining 3,000 updates. The discrete run performs 10,002 closed-form fits including initialization; the continuous run performs 16,000 gradient updates to the dynamics. The joint loss in Equation 11 uses $(\lambda_{\text{pred}}, \lambda_{\text{rec}}, \lambda_{\text{lat}}, \lambda_{\text{dist}}) = (1, 0.5, 0.05, 0.2)$, with $\beta = 0.25$ and $\gamma = 0.5$ in Equation 10. The distribution loss uses 128 generated samples, eight sampling steps, and 32 sliced projections; gradients are clipped at norm five. These results compare the two complete training configurations and provide evidence from a single seed.

C.3 Eigenvalue Approximation

We compare the eigenvalues of K with those of $e^{\Delta t A}$ in the common operator domain,

$$\widehat{\rho}_j^{\text{disc}} \in \text{spec}(K), \qquad \widehat{\rho}_j^{\text{cont}} = e^{\Delta t \widehat{\lambda}_j}, \quad \widehat{\lambda}_j \in \text{spec}(A). \tag{26}$$

This comparison avoids the frequency ambiguity associated with taking a logarithm of a discrete eigenvalue. For each system and model, a one-to-one assignment minimizes the total complex-plane distance between the six reference eigenvalues and the learned candidates. Assignment uses eigenvalue distances alone. The absolute error for a matched pair is $e_k = |\widehat{\rho}_k - \rho_k|$; its mean gives equal weight to each of the six target indices.

Table C.2 reports all reference and estimated values. Figures C.1 and C.2 show the full learned spectra together with the six reference pairs and detailed views of the selected representatives. On Circle, both models approximate the first three eigenvalues with absolute errors below 0.006. At $k = 4$, the errors are 0.0179 for the discrete model and 0.2145 for the continuous model; both models have larger errors at $k = 5, 6$. The mean errors over all six indices are 0.050311 and 0.175543, respectively. On Torus, the continuous model has smaller errors at every selected index, with mean error 0.002256 compared with 0.008816 for the discrete model.

Table C.2: Eigenvalues of $\mathcal{K}_{0.1}$ and their approximations. Each column corresponds to one prescribed Fourier index; conjugate partners are omitted. For the continuous model, $\widehat{\rho}_k = \exp(0.1\widehat{\lambda}_k)$. Complex values are rounded to four decimals, with real and imaginary parts on separate lines. Absolute errors are computed before rounding. All estimates use the final model from seed zero.

Circle						
	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$	$k=6$
Ground truth	0.9940 +0.0997i	0.9762 +0.1979i	0.9468 +0.2929i	0.9064 +0.3832i	0.8559 +0.4676i	0.7962 +0.5447i
DisKO (discrete)	0.9927 +0.0996i	0.9731 +0.1969i	0.9419 +0.2895i	0.8974 +0.3678i	0.8301 +0.3715i	0.7937 +0.3706i
DisKO (continuous)	0.9927 +0.0997i	0.9738 +0.1981i	0.9410 +0.2919i	0.9125 +0.1688i	0.8901 +0.0941i	0.8627 +0.0956i
Abs. error (discrete)	0.001315	0.003191	0.005928	0.017900	0.099453	0.174081
Abs. error (continuous)	0.001314	0.002410	0.005868	0.214545	0.375084	0.454035
Torus						
	$k=(1,0)$	$k=(0,1)$	$k=(1,1)$	$k=(1,-1)$	$k=(2,0)$	$k=(0,2)$
Ground truth	0.9940 +0.0997i	0.9842 +0.1689i	0.9615 +0.2661i	0.9952 −0.0698i	0.9762 +0.1979i	0.9375 +0.3316i
DisKO (discrete)	0.9916 +0.0995i	0.9814 +0.1682i	0.9550 +0.2634i	0.9886 −0.0689i	0.9636 +0.1933i	0.9196 +0.3212i
DisKO (continuous)	0.9923 +0.1001i	0.9830 +0.1683i	0.9581 +0.2653i	0.9925 −0.0694i	0.9750 +0.1974i	0.9345 +0.3311i
Abs. error (discrete)	0.002404	0.002892	0.006962	0.006614	0.013348	0.020678
Abs. error (continuous)	0.001781	0.001377	0.003430	0.002690	0.001239	0.003020

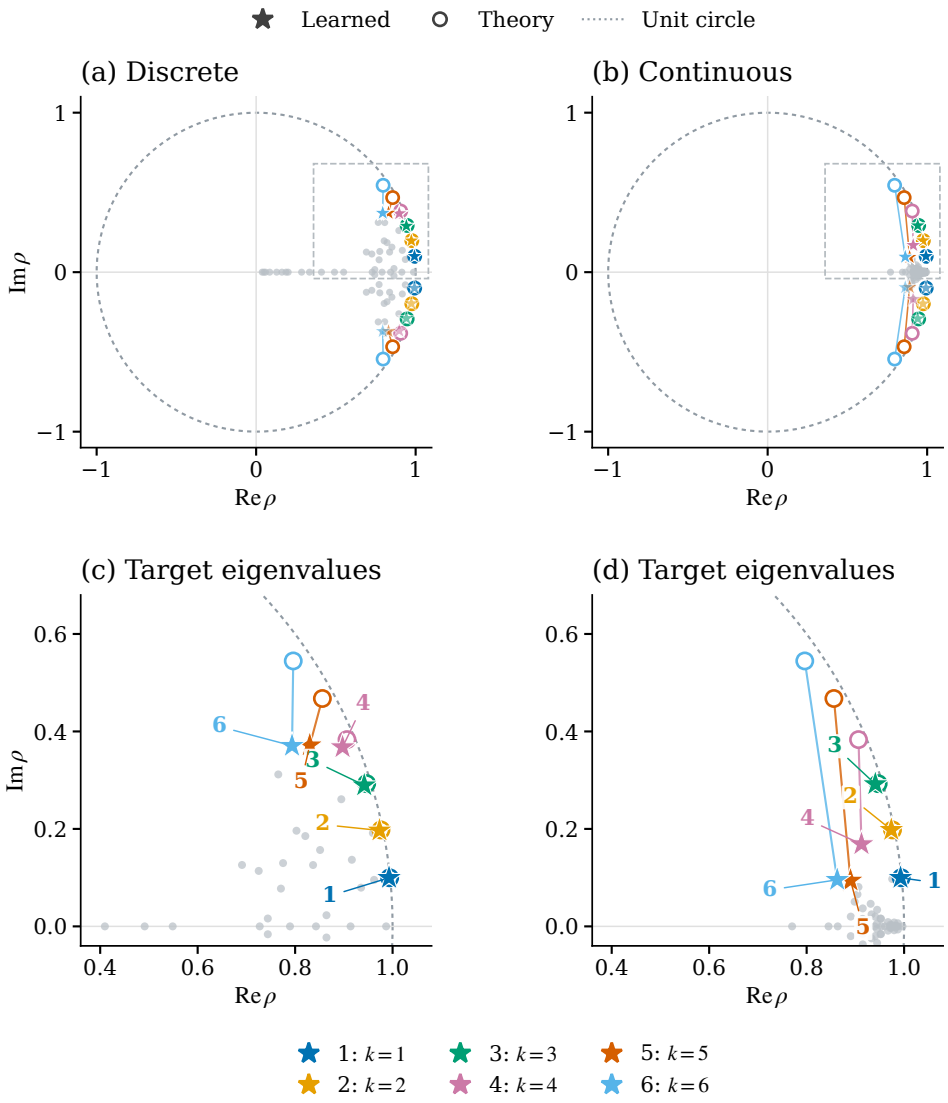

Figure C.1: Circle eigenvalue approximation at $\Delta t = 0.1$. Discrete and Continuous denote the two DisKO parameterizations. Circles show the analytical eigenvalues, stars the matched learned values, and gray points the remaining learned eigenvalues. Upper panels include conjugate partners; lower panels enlarge the selected representatives. Colors index $k = 1, \ldots, 6$. Both axes have equal scale. A matched star denotes the assigned estimate for a reference value.

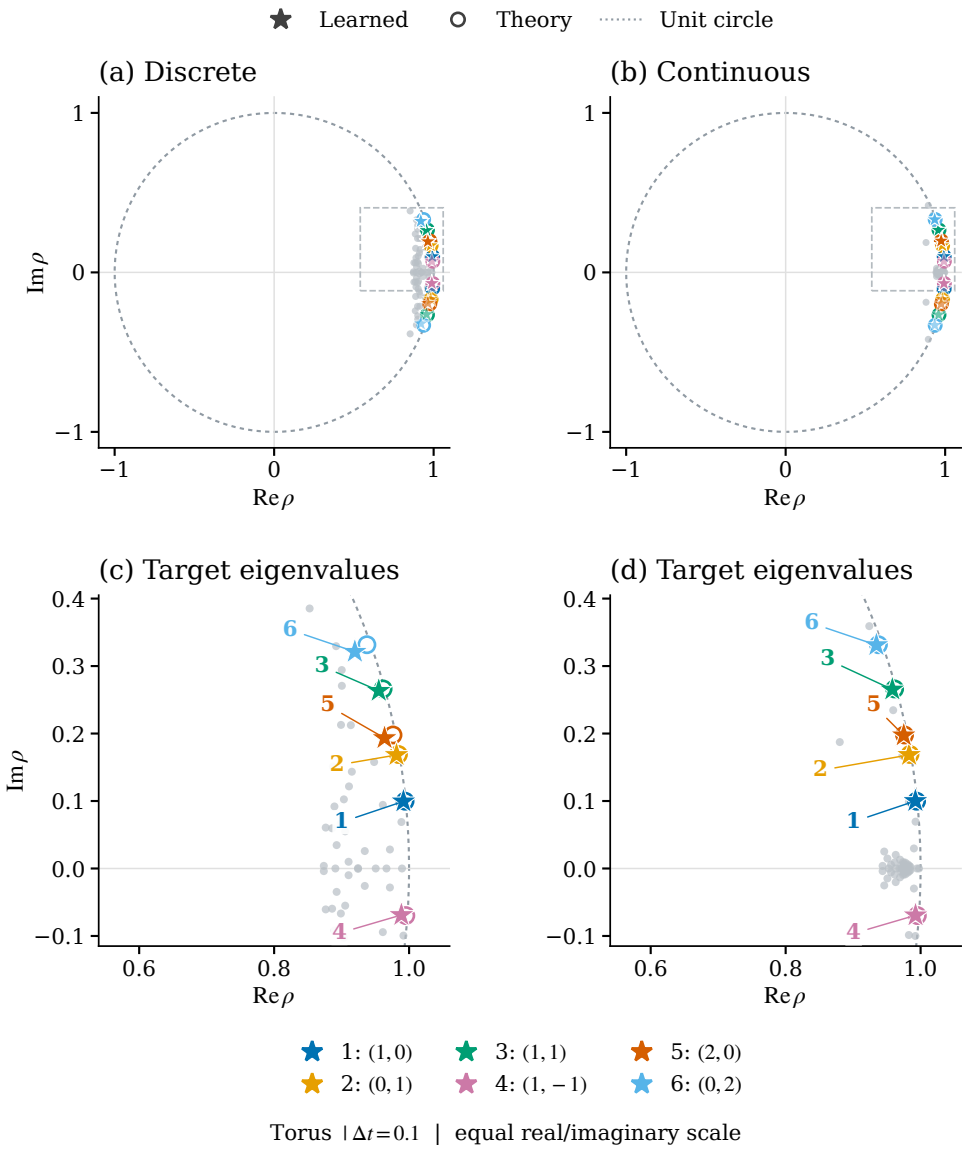

Figure C.2: Torus eigenvalue approximation at $\Delta t = 0.1$, with the same conventions as Figure C.1. The six Fourier indices identify the reference eigenpairs selected for evaluation. The gray points show the additional eigenvalues of the learned finite-dimensional operators.

C.4 Eigenfunction Evaluation on Distributions

The learned function associated with a matched eigenvalue is evaluated as $u_k(\widehat{\nu}) = w_k^\top [E(\widehat{\nu})^\top, 1]^\top$. Its eigenvector has an arbitrary nonzero complex scale. For visual comparison, we fit one complex scalar per eigenfunction using 640 training snapshots, comprising all 128 training sequences at times $0, 2, 4, 6, 8$,

$$\alpha_k = \frac{\sum_{s\in\mathcal{C}} \overline{u_k(\widehat{\nu}_s)} h_k(\mu_s)}{\sum_{s\in\mathcal{C}} |u_k(\widehat{\nu}_s)|^2}, \qquad \widehat{h}_k = \alpha_k u_k. \tag{27}$$

Here $\mathcal{C}$ is the calibration set; the fit has no intercept and is fixed for all subsequent plots. This post-training alignment resolves eigenvector amplitude and phase for evaluation. Analytical values are used only for this alignment and the spectral diagnostics.

Figure C.3 compares the real parts of $\widehat{h}_k(\widehat{\nu}_t)$ and $h_k(\mu_t)$ on test sequences at $8 < t \leq 12$. Each Circle panel contains $32 \times 40 = 1,280$ snapshots and each Torus panel contains $48 \times 40 = 1,920$ snapshots per method. The displayed correlation is computed from the full complex values,

$$C_k = \frac{\left|\sum_s \overline{(u_{k,s} - \bar{u}_k)}(h_{k,s} - \bar{h}_k)\right|}{\left(\sum_s |u_{k,s} - \bar{u}_k|^2\right)^{1/2} \left(\sum_s |h_{k,s} - \bar{h}_k|^2\right)^{1/2}}. \tag{28}$$

The plotted points use independently encoded observed snapshots at each time and assess eigenfunction approximation on these distributions. For Circle, future correlations for the discrete model are 0.999, 0.994, and 0.976 at $k = 1, 2, 3$, compared with 0.998, 0.834, and 0.569 for the continuous model. For Torus, both models have correlations above 0.995 for the two fundamental indices, while higher and mixed indices have lower correlations. Across all six Torus indices, the mean future correlations are 0.881 and 0.849, respectively. Together with the eigenvalue errors, these results show that the quality of the scalar eigenfunctions and the accuracy of their eigenvalues can differ between parameterizations.

The remaining figures evaluate the same learned functions on a controlled family of distributions. For a wrapped Gaussian $\mu_{\vartheta,\sigma}$ with center ϑ and angular covariance $\sigma^2 I$, its Fourier moment is

$$h_k(\mu_{\vartheta,\sigma}) = \exp\!\left(-\tfrac{1}{2}\sigma^2 \|k\|_2^2\right) e^{\mathrm{i}k\cdot\vartheta}, \qquad \sigma = 0.15. \tag{29}$$

Each center in the plot therefore indexes a probability distribution supplied to the encoder. Probes use 1,024 common Gaussian offsets obtained from Sobol normal nodes and their negatives, with 256 Circle centers and a 64×64 uniform angular grid on Torus. The training calibration in Equation 27 is reused without fitting to the probes. The normalized root mean square error on the probe grid $\mathcal{P}$ is

$$\mathrm{NRMSE}_k = \left(\frac{\sum_{\vartheta\in\mathcal{P}} |\widehat{h}_k(\widehat{\nu}_{\vartheta,\sigma}) - h_k(\mu_{\vartheta,\sigma})|^2}{\sum_{\vartheta\in\mathcal{P}} |h_k(\mu_{\vartheta,\sigma})|^2}\right)^{1/2}. \tag{30}$$

Figure C.4 displays both components for Circle indices $k = 1, 2, 3$. At $k = 3$, the probe NRMSE is 0.123 for the discrete model and 0.697 for the continuous model, despite their similar eigenvalue errors. Figure C.5 displays all six Torus indices, including the pointwise complex absolute errors. The real and imaginary components use a common color range across both systems and methods. Torus geometry is used only to display the two angular center coordinates; the encoder receives four-dimensional point clouds. Errors are evaluated uniformly over the angular grid. These probes characterize the restrictions of learned distribution observables to the chosen Gaussian family. Diffusion increases the Gaussian width over time, so a static comparison on the fixed-width family supports conclusions about function approximation on that family. The nonlinear set encoder permits more general distribution observables than linear Fourier moments, and accurate approximation across all probability measures is outside the scope of these finite evaluations.

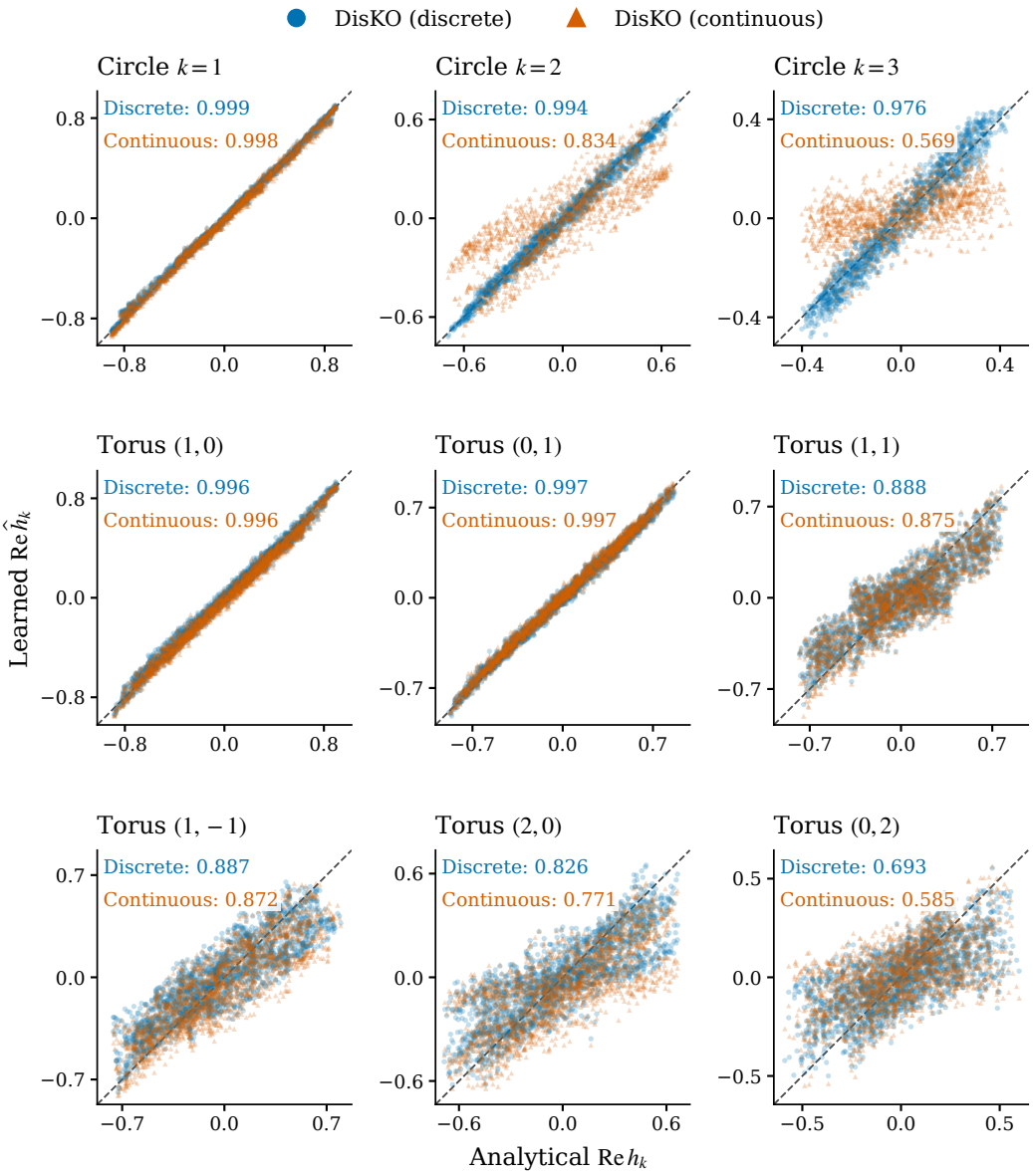

Figure C.3: Eigenfunction values on observed future snapshots from test sequences. The top row shows Circle indices $k = 1, 2, 3$; the remaining rows show all six Torus indices. Each panel overlays DisKO (discrete) and DisKO (continuous), with the equality line dashed. Coordinates display real parts after the training calibration; annotations give centered complex correlations. Each point is obtained by encoding the observed snapshot at that time, so the figure assesses the learned functions on future distributions.

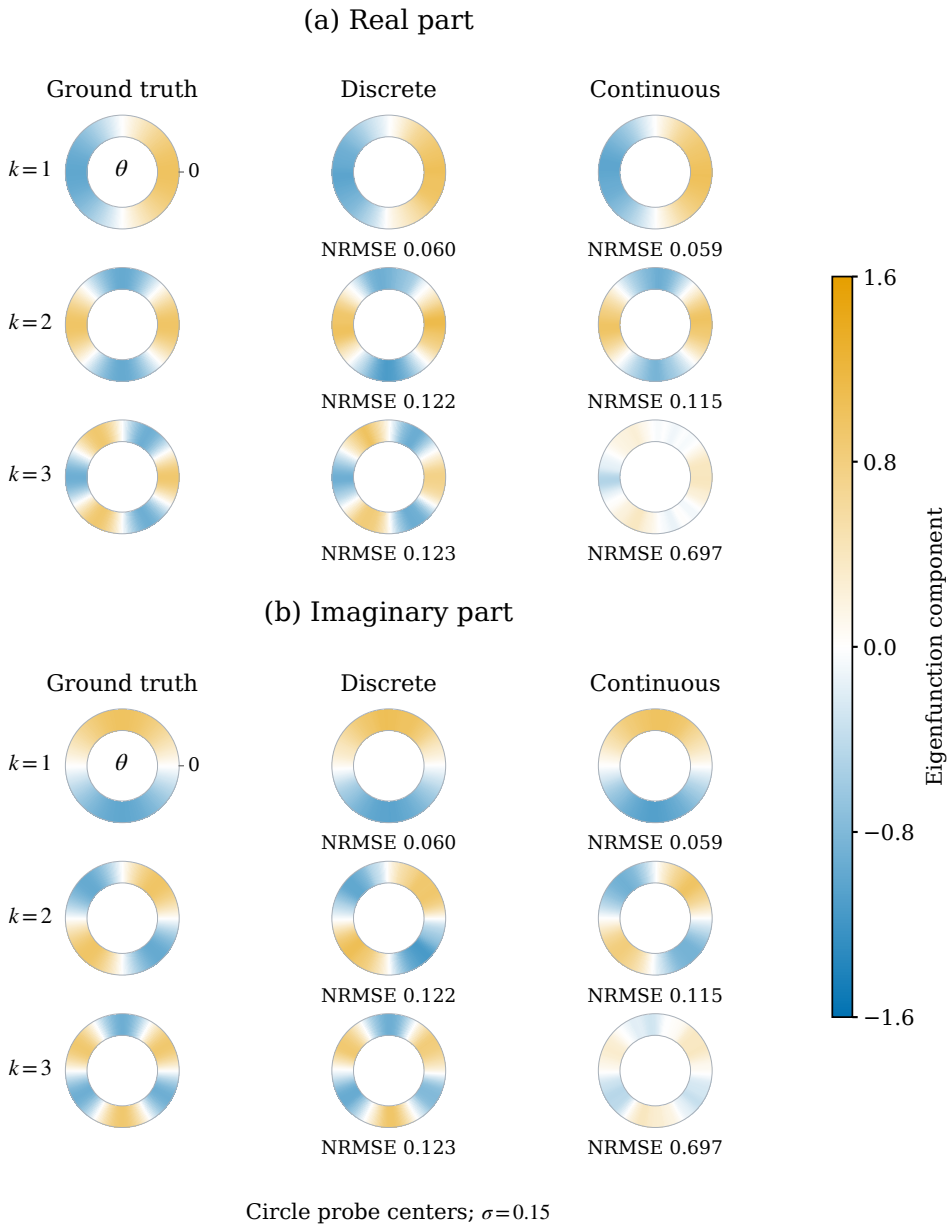

Figure C.4: Circle eigenfunctions evaluated on translated wrapped Gaussian probes. Panels (a) and (b) display real and imaginary components using the same color scale. Each angle is the center of a distribution of width $\sigma = 0.15$; ring radius and thickness are fixed. Columns compare the analytical reference with the discrete and continuous DisKO functions. NRMSE uses the full complex error and is consequently the same in both component panels.

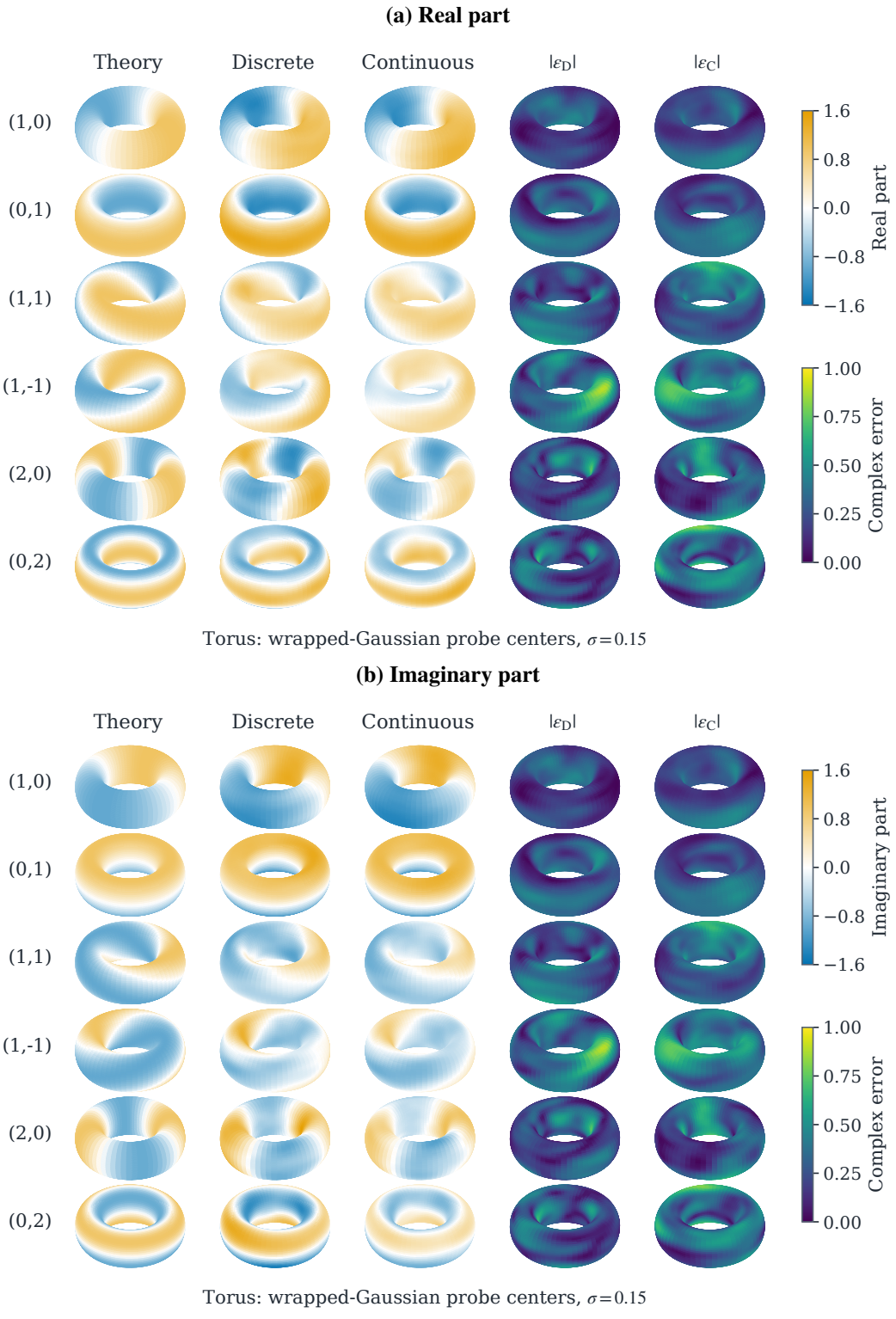

Figure C.5: Torus eigenfunctions on wrapped Gaussian probes. Rows are the six Fourier indices; the first three columns show the reference and the two DisKO approximations. The final two columns show $|\widehat{h}_k - h_k|$ for the discrete and continuous models. These complex error fields are identical in panels (a) and (b). All color ranges are shared, and surface coordinates index probe centers.

D LATENT DYNAMICAL CONSISTENCY

We assess approximate closure in Equation 6 by comparing latent forecasts with encodings of observed snapshots. Exact closure with the corresponding affine generator implies agreement at the population level.

We display the first test population sequence of the OU benchmark (Appendix A), starting from a Gaussian distribution. Observations are spaced by 0.05, training covers $0 \le t \le 2.5$, and extrapolation covers $2.5 < t \le 5$. At each time, 512 particles are sampled without replacement from the available 1,024 particles and standardized using statistics fitted to training populations within the training interval. The same sampled snapshots are supplied to five models with seeds $s = 0, \dots, 4$. Each Deep Sets encoder produces a 32-dimensional representation. After training, the encoder and continuous affine dynamics are fixed, and we compute

$$z_s^{\mathrm{enc}}(t) = E_s(\widehat{\nu}_t), \qquad \widetilde{z}_s(t) = F_t^{(s)}(E_s(\widehat{\nu}_0))\,. \tag{31}$$

Here $F_t^{(s)}$ is evaluated through the matrix exponential in Equation 7. Every forecast originates at $t = 0$ and continues through the training boundary; subsequent snapshot encodings serve solely as references.

For each model, we jointly fit PCA to the centered reference and forecast trajectories, retaining two components without whitening. Reference trajectories are aligned to seed zero by an orthogonal transformation and translation, with the same transformation applied to forecasts and no rescaling. Both visualization steps use the full interval after prediction. The retained components explain 77.66% to 83.51% of joint variance, depending on the model. Figure D.1 shows means and sample standard deviations in these aligned coordinates.

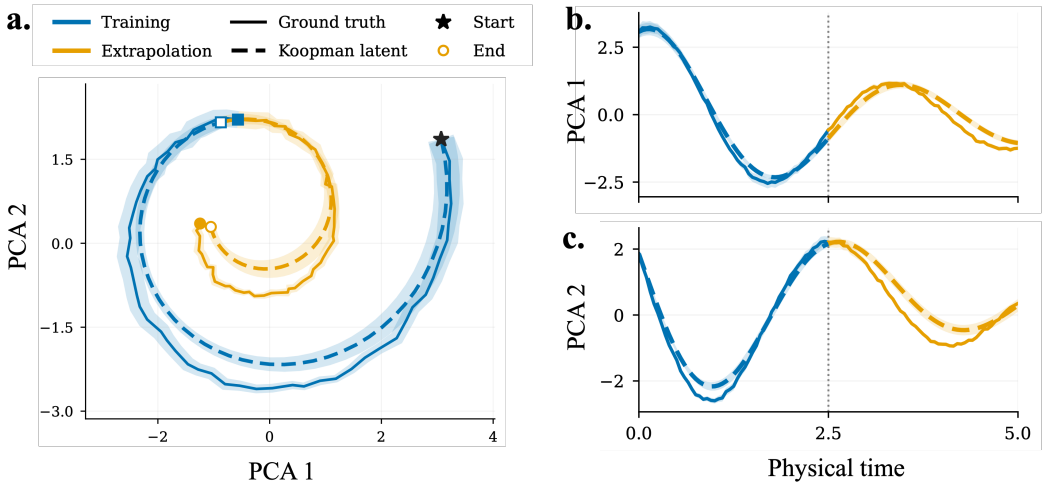

Figure D.1: OU latent evolution. Solid curves (Ground truth) show reference encodings; dashed curves (Koopman latent) show forecasts. Blue and gold indicate training and future time ranges; all snapshots belong to the test population. Curves are means over five models after PCA and rigid alignment. Shading is one sample standard deviation, normal to each mean curve in (a) and coordinatewise in (b,c). Stars, squares, and circles mark $t = 0$, 2.5, and 5; filled and open markers denote reference and forecast.

The forecast follows the reference's rotation and inward progression, reproducing the main oscillation within the training interval and continuing the contracting trajectory beyond $t = 2.5$. Amplitude differences are visible near the troughs of the second coordinate. The clearest future discrepancy occurs near $t = 4$, where the predicted minimum is shallower and slightly later. These observations support approximate dynamical consistency in the dominant projected coordinates for this test sequence, with residual amplitude and timing errors. Projection and averaging restrict this evidence to the displayed dynamics.

E Ablation Study

We examine how the form of latent dynamics affects distribution extrapolation on the OU benchmark. The continuous DisKO model uses the affine generator $\dot{z} = Az + c$, evaluated through the matrix exponential in Equation 7. For comparison, we replace this generator with an autonomous Neural ODE,

$$\dot{z} = W_2 \tanh(W_1 z + b_1) + b_2, \tag{32}$$

with latent dimension 64 and hidden width 32. The two dynamics modules have similar parameter counts, 4,160 for the affine generator and 4,192 for the Neural ODE. Both models use the same Deep Sets encoder and conditional flow matching decoder architectures, with width 256 and depths three and five, respectively.

We use 128 training initial distributions and all 64 test initial distributions from the OU dataset in Appendix A.1; 16 further initial distributions are reserved for validation. Training covers $0 \leq t \leq 2.5$, with snapshots spaced by 0.05 and 1,024 particles per snapshot. Standardization is fitted to the training distributions within this interval. For each of five training seeds, the models share encoder and decoder weights obtained from 10,000 pretraining updates. Each then receives 1,000 updates of the dynamics alone, followed by 10,000 joint updates of all components. Both dynamics modules are optimized by gradient descent with the same losses, minibatch sampling, and learning rate 3×10^{-4}. We evaluate the final checkpoints.

Each test rollout encodes only the initial snapshot and propagates recursively to $t = 5$, without incorporating subsequent observations. The Neural ODE uses fourth-order Runge–Kutta integration with maximum step 0.05. Halving this step changes latent trajectories by at most 3.1×10^{-6} in relative Euclidean norm across the five models. Both decoders generate 512 samples with 32 sampling steps, and errors are computed against 512 reference samples in the original observation coordinates. We use 256 samples per distribution for W_1 and W_2, 128 projection directions for SW_1, and the biased RBF estimate of MMD^2 defined in Appendix A.3. Reference samples and evaluation random seeds are matched between methods.

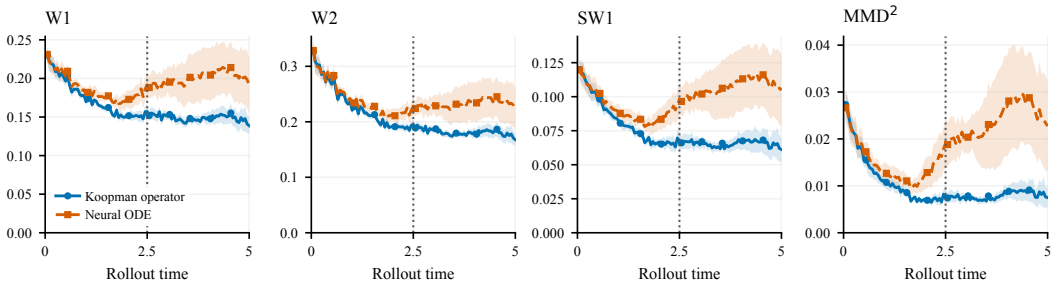

Figure E.1: OU distribution prediction errors for the continuous Koopman model and the Neural ODE. At each time, errors are averaged over 64 test initial distributions within each run, then over five training seeds. Shading denotes one sample standard deviation across seeds. The dotted vertical line marks the end of training at $t = 2.5$. All predictions originate at $t = 0$.

Figure E.1 shows lower future prediction errors and smaller variation across training seeds for the Koopman model in all four metrics. Averaged over $2.5 < t \leq 5$, its W_1 error is 0.149 ± 0.007, compared with 0.201 ± 0.026 for the Neural ODE; the corresponding SW_1 errors are 0.065 ± 0.005 and 0.107 ± 0.018. The Neural ODE exhibits a more pronounced increase in SW_1 and MMD^2 near the training boundary and into the future interval. These results support the affine Koopman parameterization as an effective inductive bias for accurate and reproducible extrapolation under this training protocol.

F Data Efficiency

We examine the data requirements of continuous DisKO on the OU benchmark by varying two aspects of the training data. Let $R = |\mathcal{I}_{\text{train}}|$ denote the number of training initial distributions, each providing a population sequence of snapshots, and let N denote the number of available particle samples per training snapshot. Thus, R controls the number of distribution evolutions represented in training, while N controls the sampling budget within each snapshot.

We vary $R \in \{16, 32, 64, 128\}$ using 1,024 particles per snapshot input during training and evaluation. Separately, we vary $N \in \{128, 256, 512, 1024\}$ with $R = 128$, keeping each training and evaluation input at 128 particles. In the latter experiment, N specifies the available training sample pool from which inputs are drawn. The two experiments are interpreted within their respective input budgets. Each setting uses three training seeds, and the reported scores average prediction errors over the future time window.

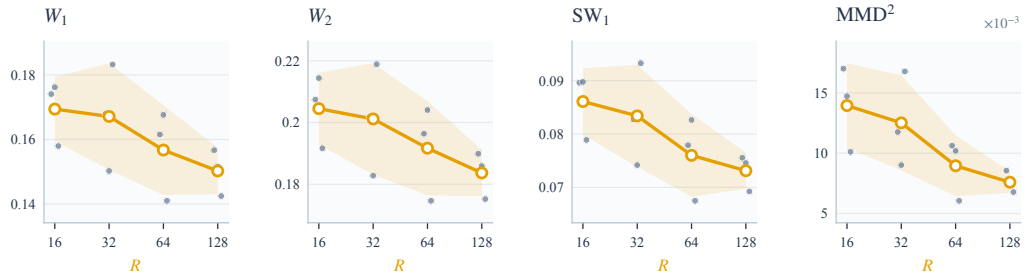

Figure F.1: Effect of the number of training initial distributions R on future OU prediction errors. Colored circles and lines show means over three training seeds; shading denotes one sample standard deviation. Gray points show individual seeds, slightly displaced horizontally for visibility. Inputs contain 1,024 particles. The horizontal axis is logarithmic with base two.

Figure F.1 shows decreasing mean errors in all four metrics as R increases. From $R = 16$ to $R = 128$, W_1 decreases from 0.169 ± 0.010 to 0.150 ± 0.007, and SW_1 decreases from 0.086 ± 0.006 to 0.073 ± 0.003. This trend supports the benefit of exposing the model to more initial distributions.

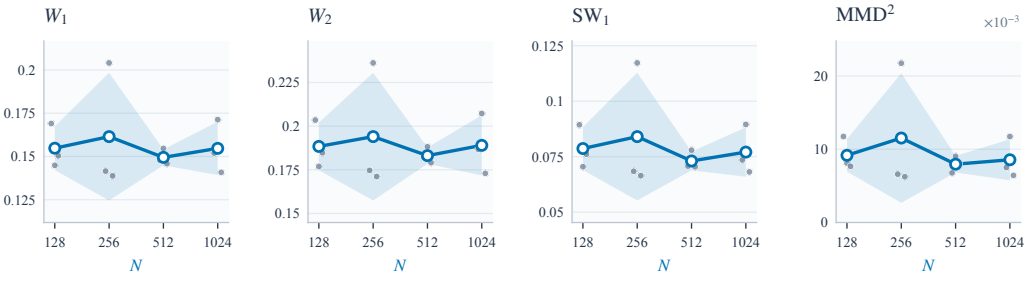

Figure F.2: Effect of the available samples per training snapshot N, with $R = 128$ training initial distributions and 128 particles per input. Means, sample standard deviations, and individual seeds follow Figure F.1. Errors summarize the future time window. The horizontal axis is logarithmic with base two; the MMD^2 axis in both figures uses units of 10^{-3}.

Figure F.2 shows modest, nonmonotonic changes with N under the fixed input budget. Mean W_1 ranges from 0.149 to 0.161, with the lowest mean error in all four metrics at $N = 512$. Variation across seeds is largest at $N = 256$. Over the tested range, increasing the available sample pool yields limited additional accuracy with 128 particles per input.

G DISTRIBUTIONAL KOOPMAN REPRESENTATION

We describe the distributional Koopman representation underlying DisKO, the role of learned distribution observables, and the information required for predictive closure. The results separate the expressiveness of an observable family from the dynamical consistency of its learned finite-dimensional representation.

G.1 DISTRIBUTION OBSERVABLES AND KOOPMAN OPERATORS

Let $\mathcal{X}$ be a state space and $\mathfrak{M} \subseteq \mathcal{P}(\mathcal{X})$ an invariant family of probability distributions. Distribution evolution is described by maps $T_t : \mathfrak{M} \to \mathfrak{M}$ satisfying

$$T_0 = \mathrm{Id}, \qquad T_{t+s} = T_t \circ T_s, \qquad t, s \geq 0. \tag{G.1}$$

Autonomous ODEs induce such evolution by transporting an initial distribution through their solution maps. Time-homogeneous Markov SDEs induce it through their transition probabilities. In both cases, the initial law determines the subsequent law, even when individual state trajectories are stochastic.

A distribution observable is a scalar functional $h : \mathfrak{M} \to \mathbb{C}$. Complex values accommodate oscillatory eigenfunctions; encoder coordinates are real valued. The distributional Koopman operator acts by composition (Oprea et al., 2025),

$$(\mathcal{K}_t h)(\mu) = h(T_t \mu). \tag{G.2}$$

It is linear in h and satisfies $\mathcal{K}_{t+s} = \mathcal{K}_t \mathcal{K}_s$. Thus, nonlinear distribution evolution admits a linear description on a space of distribution observables. On an invariant Banach space $\mathcal{H}$ where $(\mathcal{K}_t)_{t \geq 0}$ is strongly continuous, its generator is

$$\mathcal{L} h = \lim_{t \downarrow 0} \frac{\mathcal{K}_t h - h}{t}, \qquad h \in \mathrm{Dom}(\mathcal{L}), \tag{G.3}$$

with the limit taken in the norm of $\mathcal{H}$. For example, when $\mathfrak{M}$ is compact and $(t, \mu) \mapsto T_t \mu$ is continuous, $\mathcal{H} = C(\mathfrak{M}; \mathbb{C})$ with the uniform norm gives a strongly continuous contraction semigroup.

The connection to state observables is explicit. Let $(S_t \phi)(x) = \mathbb{E}[\phi(X_t) \mid X_0 = x]$ be a Markov semigroup on state functions. For $h_\phi(\mu) = \int \phi \, d\mu$, its induced law evolution obeys

$$(\mathcal{K}_t h_\phi)(\mu) = \int S_t \phi \, d\mu = h_{S_t \phi}(\mu). \tag{G.4}$$

This identity relates state observables to linear functionals of a distribution (Oprea et al., 2025). Distribution observables also include nonlinear statistics, such as $\mathrm{Var}_\mu[\phi] = h_{\phi^2}(\mu) - h_\phi(\mu)^2$ whenever these moments exist.

G.2 LEARNED OBSERVABLE SPACES AND APPROXIMATE CLOSURE

Nonlinear moment representations. Consider functionals formed from finitely many moments,

$$h(\mu) = g\big(a_1(\mu), \ldots, a_q(\mu)\big), \qquad a_\ell(\mu) = \int \phi_\ell \, d\mu. \tag{G.5}$$

The readout g permits nonlinear interactions between statistics of the entire distribution. Their expressive power has a precise interpretation in a compact setting. Assume that $\mathcal{X}$ is a compact metric space, $\mathfrak{M} = \mathcal{P}(\mathcal{X})$ carries the weak topology, and S_t is a strongly continuous Markov semigroup on $C(\mathcal{X})$ with generator $\mathcal{A}$. Its dual law evolution induces a strongly continuous $\mathcal{K}_t$ on $C(\mathfrak{M})$. Let $\mathfrak{P}$ contain all functionals in Equation G.5 with arbitrary finite q, polynomial g, and real-valued $\phi_\ell \in \mathrm{Dom}(\mathcal{A})$, together with their complex linear combinations.

Proposition G.1 (Richness of polynomial moment observables) *Under these assumptions, $\mathfrak{P}$ is dense in $C(\mathfrak{M}; \mathbb{C})$ and is a core for $\mathcal{L}$. In particular, every $h \in \mathrm{Dom}(\mathcal{L})$ admits $p_n \in \mathfrak{P}$ with $\|p_n - h\|_\infty \to 0$ and $\|\mathcal{L} p_n - \mathcal{L} h\|_\infty \to 0$.*

Proof sketch. (Adapted from the core criterion in Engel & Nagel, 2000, Proposition II.1.7.) The moment algebra contains constants and separates probability measures because $\mathrm{Dom}(\mathcal{A})$ is dense in $C(\mathcal{X})$; the Stone–Weierstrass theorem gives density. For a polynomial g, the uniform generator limit and the chain rule give

$$(\mathcal{L}h)(\mu) = \sum_{\ell=1}^{q} \partial_\ell g\big(a_1(\mu), \dots, a_q(\mu)\big) \int \mathcal{A}\phi_\ell \, d\mu. \tag{G.6}$$

Thus $\mathfrak{P} \subseteq \mathrm{Dom}(\mathcal{L})$. Equation G.4 and $S_t \,\mathrm{Dom}(\mathcal{A}) \subseteq \mathrm{Dom}(\mathcal{A})$ imply $\mathcal{K}_t\mathfrak{P} \subseteq \mathfrak{P}$. The cited criterion then establishes the core property. □

This result makes nonlinear moment functionals a rich starting point for distribution representations. Its scope is the complete moment family under the compact Markov assumptions above. A fixed finite dictionary, or a trained neural encoder, requires a separate assessment of approximation and closure.

Learned distribution observables. An encoder E defines each observable by $h_j(\mu) = E_j(\mu)$ and collects them in $\mathbf{h} = (h_1, \dots, h_m)^\top$. For an empirical snapshot $\widehat{\nu} = N^{-1}\sum_{i=1}^{N} \delta_{y_i}$, a moment encoder evaluates Equation G.5 by replacing each integral with $N^{-1}\sum_i \phi_\ell(y_i)$. Neural mean aggregation admits approximation results for continuous functionals on compact families of probability measures (Pevný & Kovařík, 2019). DisKO parameterizes its coordinates with permutation-invariant set networks (Zaheer et al., 2017; Lee et al., 2019) and learns them jointly with latent propagation and distribution reconstruction. This gives a data-driven finite-dimensional observable space

$$V_E = \mathrm{span}\{1, h_1, \dots, h_m\}. \tag{G.7}$$

The approximation results motivate expressive parameterizations. The dynamical interpretation of the learned space depends additionally on its behavior under $\mathcal{K}_t$.

Affine propagation and closure. The continuous DisKO model uses $\dot{z} = Az + c$ with flow F_t. Including the constant observable yields

$$\overline{\mathbf{h}}(\mu) = \begin{pmatrix} \mathbf{h}(\mu) \\ 1 \end{pmatrix}, \qquad B = \begin{pmatrix} A & c \\ 0 & 0 \end{pmatrix}, \qquad \begin{pmatrix} F_t(z) \\ 1 \end{pmatrix} = e^{tB} \begin{pmatrix} z \\ 1 \end{pmatrix}. \tag{G.8}$$

If $h_j \in \mathrm{Dom}(\mathcal{L})$ and $\mathcal{L}\mathbf{h} = A\mathbf{h} + c$ on $\mathfrak{M}$, uniqueness of the resulting linear ODE gives $\overline{\mathbf{h}}(T_t\mu) = e^{tB}\overline{\mathbf{h}}(\mu)$. The space V_E is then Koopman invariant, realizing exact closure in the chosen coordinates (Brunton et al., 2016a).

For learned coordinates, define the finite-time closure residual

$$R_t(\mu) = \overline{\mathbf{h}}(T_t\mu) - e^{tB}\overline{\mathbf{h}}(\mu). \tag{G.9}$$

This definition only requires observable values and can be evaluated from distribution snapshots. Small residuals across relevant distributions and prediction times express approximate closure under the learned propagation. Latent consistency training estimates this discrepancy at observed time pairs using empirical encodings. Its behavior on future snapshots provides a separate test of dynamical consistency.

Eigenfunctions within the learned space. Let $w^\top B = \lambda w^\top$ be a left eigenvector relation over $\mathbb{C}$, and define $h_w(\mu) = w^\top \overline{\mathbf{h}}(\mu)$, assuming $h_w \not\equiv 0$. Multiplying Equation G.9 by $w^\top$ gives

$$(\mathcal{K}_t h_w)(\mu) - e^{t\lambda} h_w(\mu) = w^\top R_t(\mu). \tag{G.10}$$

Here $\top$ denotes ordinary transpose, so w specifies the coefficients of a complex linear combination of real observables. Exact closure therefore produces distributional Koopman eigenfunctions. Under approximate closure, these combinations are candidate eigenfunctions whose evolution can be checked through Equation G.10. For $\lambda = -\alpha + i\omega$, $\alpha > 0$ gives exponential decay and ω gives angular oscillation frequency.

The discrete model has the same interpretation with $\overline{K} = \left(\begin{smallmatrix} K & b \\ 0 & 1 \end{smallmatrix}\right)$, $\overline{K}^n$ replacing e^{tB} at $t = n\Delta t$, and an eigenvalue ρ replacing $e^{\Delta t\lambda}$. Eigenvectors select combinations within V_E; individual encoder coordinates generally combine several dynamical components. Empirical agreement with analytic eigenfunctions consequently tests the dynamical content of the learned observable space. The residual identity alone supplies no bound on the distance to an exact eigenvalue.

Connection to distribution prediction. The decoder D is trained to reconstruct the modeled distribution from $E(\mu)$ and to generate its future law from $F_t(E(\mu))$. Exact reconstruction on a family requires E to be injective there. A continuous injective encoder on a compact family has a continuous inverse on its image. This explains the complementary roles of reconstruction and closure in retaining distributional information and propagating it consistently; practical decoder accuracy remains an empirical property.

G.3 Observation Sufficiency and Identifiability

Suppose that a measurable observation map $P : \mathcal{X} \to \mathcal{Y}$ provides the law $\nu = P_{\#}\mu$, where $P_{\#}$ denotes pushforward. The observed family is $\mathfrak{N} = P_{\#}\mathfrak{M}$. Full observation corresponds to $P = \mathrm{Id}$; partial observation can merge distinct full distributions into the same observed law.

Proposition G.2 (Sufficiency for autonomous observed evolution) *There is a semigroup $\overline{T}_t : \mathfrak{N} \to \mathfrak{N}$ satisfying $\overline{T}_t(P_{\#}\mu) = P_{\#}T_t\mu$ for all $\mu \in \mathfrak{M}$ and $t \geq 0$ if and only if*

$$P_{\#}\mu = P_{\#}\mu' \quad \Longrightarrow \quad P_{\#}T_t\mu = P_{\#}T_t\mu' \quad \text{for every } t \geq 0. \tag{G.11}$$

Proof. An observed evolution must assign identical futures to identical inputs, which proves necessity. Under Equation G.11, define $\overline{T}_t(\nu) = P_{\#}T_t\mu$ using any μ with $P_{\#}\mu = \nu$. The condition makes this definition independent of the chosen preimage. The identity and composition laws follow from those of T_t. □

This condition concerns autonomous evolution on observed laws. Generator statements additionally require the strong continuity conditions in Section G.1. If Equation G.11 fails at time t, any predictor Q_t using only the current observed law has an unavoidable ambiguity. For a metric d on the predicted distributions and the common input $\nu = P_{\#}\mu = P_{\#}\mu'$, the triangle inequality gives

$$\max\{d\big(Q_t(\nu), P_{\#}T_t\mu\big), d\big(Q_t(\nu), P_{\#}T_t\mu'\big)\} \geq \frac{1}{2}d\big(P_{\#}T_t\mu, P_{\#}T_t\mu'\big). \tag{G.12}$$

This obstruction persists with exact population distributions and therefore precedes finite sampling or optimization error.

For example, consider $\dot{x} = v$, $\dot{v} = 0$ on $\mathbb{R}^2$, with $P(x, v) = x$. The full initial laws $\delta_{(0,1)}$ and $\delta_{(0,-1)}$ both give the observed law δ_0. At time $t > 0$, their observed laws are δ_t and δ_{-t}, whose W_1 distance is $2t$. At least one prediction from the common current snapshot thus has W_1 error at least t. Additional measurements or temporal context can provide information about the missing velocity.

In DisKO, $h_j(\mu) = E_j(P_{\#}\mu)$ factors through the observation map. Consequently, every learned observable assigns the same value to full laws with identical observed distributions. Predicting the entire future observed law requires the sufficiency condition above. Predicting a selected set of observables requires only that those future values be determined by the current representation. These requirements distinguish closure of selected statistics from information sufficient for distribution reconstruction.

Identification from finitely sampled times. Even with sufficient observations, the sampling times and initial laws constrain which dynamics can be identified. For instance, the circle rotation $\theta_t = \theta_0 + \omega t \pmod{2\pi}$ induces $T_t^{(\omega)}\mu = (\theta \mapsto \theta + \omega t)_{\#}\mu$. For observations on a uniform grid with spacing Δt,

$$T_{n\Delta t}^{(\omega)} = T_{n\Delta t}^{(\omega + 2\pi k/\Delta t)}, \qquad n \in \mathbb{N}_0, \quad k \in \mathbb{Z}. \tag{G.13}$$

Distinct continuous-time frequencies produce identical sampled evolution for every initial distribution. Recovering a generator therefore requires additional information, such as a frequency restriction or informative sampling times. Initial distributions must also excite the observables whose evolution is to be learned. Together, observation sufficiency and informative sampling delimit the identification problem. Convergence and stability of jointly learning the encoder, operator, and decoder require further analysis within an identifiable model class.