Simulation-Free Learning of Population Dynamics with Wasserstein Lagrangian Residuals

Anonymous authors
Paper under double-blind review

Abstract

The dynamics of cells, organisms, and fluids are often modeled as probability distributions evolving over time. Reconstructing and extrapolating this evolution from unpaired snapshots requires assumptions about the underlying process. Wasserstein gradient flows are a common choice, but they cannot describe conservative or periodic dynamics. Lagrangian mechanics in Wasserstein space covers both, but existing methods for learning it are simulation-based: they run a numerical solver at every training step, which makes training expensive. We propose *Double-Stitch*, a simulation-free method that learns these mechanics by penalizing the residual of the equation of motion along a learned population path. We derive this equation from a Clebsch variational principle that does not require gradient velocities, and show that the residual vanishes exactly when the equation holds. We test Double-Stitch on synthetic, single-cell and ocean vortex datasets and find that it matches or outperforms gradient-flow methods and simulation-based WLM on most tasks, while training 4–14 times faster than WLM.

1 Introduction

Learning complex dynamics of populations is a fundamental problem with applications to single-cell biology, animal behavior, and fluid mechanics (Schiebinger et al., 2019; Maury et al., 2010; Petrović et al., 2025). Individual entities are usually not tracked: the population is observed as unpaired sets of samples (snapshots) measured at several time points. Generally, modeling approaches to this problem can be grouped into three families based on the assumptions they make about the data generating process: flow-based, gradient flow, and Lagrangian flow.

Flow-based methods do not make explicit assumptions[1] about the underlying dynamical process (Kapuśniak et al., 2024; Atanackovic et al., 2025). They learn a time-dependent velocity field or a stochastic bridge that transports an observed snapshot to a new time point by an optimal path (e.g., minimizing a kinetic energy). Without additional assumptions, these methods are generally not suited for extrapolation beyond the observed time interval (Guan et al., 2026a).

Gradient flow methods assume that the underlying dynamics follow the steepest descent of a free energy in Wasserstein space, or, in other words, a Wasserstein gradient flow (WGF). The resulting closed system allows learning the time-independent free energy function $\mathcal{F}$ and therefore provides time extrapolation capabilities. However, WGFs are purely dissipative and cannot effectively model dynamics where velocity is not deterministically defined by energy alone (e.g., as in periodic phenomena and conservative systems; see the conservative example in Figure 1).

The Lagrangian methods generalize gradient flows by assuming that the underlying dynamic process is governed by an equation on acceleration instead of velocity (i.e., Newton's second law). This approach was recently proposed for population dynamics (Guan et al., 2026a), building on Lagrangian formulations in Wasserstein space (Neklyudov et al., 2024). However, the existing algorithm (WLM) is simulation-based and computationally expensive because it simulates the trajectory at every training step.

We propose *Double-Stitch*, a simulation-free approach to learning Lagrangian mechanics. Unlike flow-based methods, this method allows extrapolation; unlike WGF-based methods, it models peri-

[1] Some methods can use a reference process, but it is designed for a specific dataset, not used as a general assumption (Petrović et al., 2025).

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

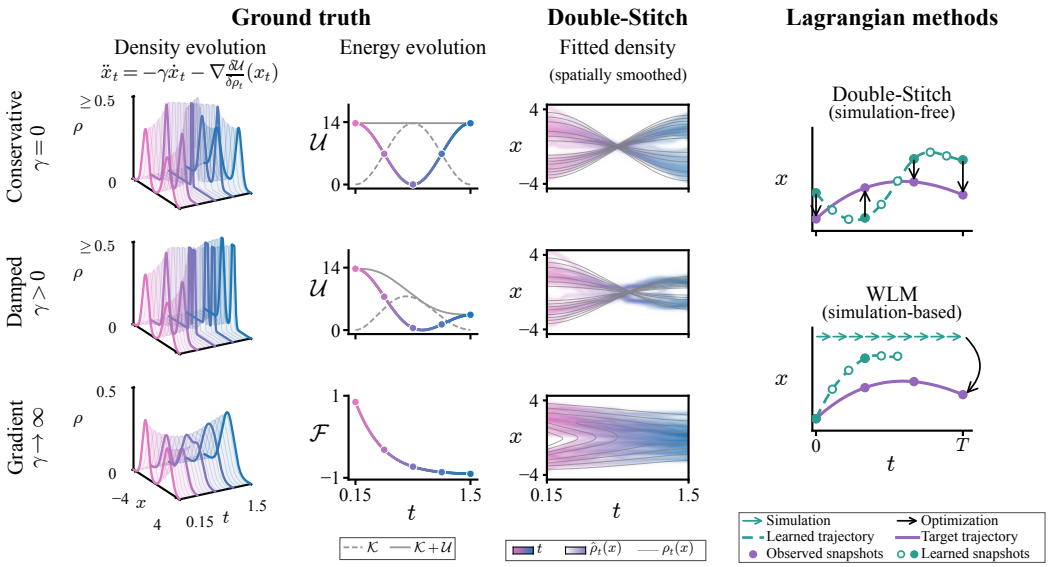

Figure 1: **Simulation-free fitting across three dynamical regimes.** Left: conservative, damped and gradient-flow examples, with analytic densities, energies and Double-Stitch fits. Right: schematic training procedures for Double-Stitch and WLM. Grey curves show kinetic and total energy for the oscillators; the gradient-flow panel shows free energy. Grey contours show analytic densities.

odic and conservative systems (see Figure 1); and it is computationally cheaper than the simulation-based WLM algorithm. Table 1 summarizes the similarities and differences between the existing methods and our approach.

Double-Stitch is based on Stitching (Heinonen et al., 2026), a simulation-free WGF-based algorithm that learns the dynamics by directly optimizing the trajectory to minimize (a) the distance between the learned and observed snapshots, and (b) the residual of the underlying dynamical law. Double in the name refers to the second-order law, compared to the first-order equation for WGF. We re-derive the Wasserstein Lagrangian law without requiring gradient velocities and show its residual vanishes if and only if the dynamics are satisfied.

Contributions.

- We propose a Clebsch-based derivation of the Wasserstein Lagrangian mechanics. This derivation does not restrict the velocities to be gradient, making it applicable to initial conditions with rotational components.
- We propose Double-Stitch, a simulation-free method that fits a latent population path and a neural potential by penalizing the discretized residual of the Lagrangian mechanics.
- We show that Double-Stitch matches or improves upon competing methods for most tasks on synthetic, single-cell, and ocean vortex datasets, while being cheaper to train than WLM.

2 Wasserstein Lagrangian mechanics

2.1 Problem setting

Consider the problem of learning the temporal evolution of a probability distribution $\rho : [0, T] \to \mathcal{P}_2(\mathbb{R}^d)$[2]. We assume that at certain time points $t_0 < \ldots < t_K$ we observe sets of independent samples from that distribution:

$$X_i = \left\{ x_i^{(n)} \right\}_{n=1}^{N_i}, \quad x_i^{(n)} \overset{\text{ind.}}{\sim} \rho_{t_i}, \quad i = 0, \ldots, K.$$

[2]Full notation is given in Section A.

Table 1: Comparison of methods for learning population dynamics. Double-Stitch is a simulation-free method that extrapolates in time and learns both gradient and second-order Lagrangian flows.

Family	Method	Inputs	Extrapolation	Encoded flow		Simulation-free
				Gradient	Lagrangian	
Flow-based	Flow matching, bridges	X_i	✗	✗	✗	✓
	Curly-FM	X_i, reference process	✗*	✗	✗*	✓
Gradient flow	JKOnet family	X_i	✓	✓	✗	✓/✗‡
	Stitching	X_i	✓	✓	✗	✓
Lagrangian flow	WLM	X_i, (v_0)	✓	✓†	✓	✗
	Double-Stitch	X_i, (u_0)	✓	✓†	✓	✓

* Reference process can be used for some extrapolation and flow encoding. † Recovered in the overdamped limit. ‡ Depending on the method.

We will call these sets of samples *snapshots*. The samples in consecutive snapshots need not be paired, i.e. $x_i^{(n)}$ and $x_{i+1}^{(n)}$ may describe distinct individuals.

Our task is to create a model that, given the observed snapshots, will predict the underlying distribution $\hat{\rho}_t$ at a given time point t. The quality of the prediction is evaluated using the p-Wasserstein distance at selected times $\mathcal{T}_{\text{eval}}$:

$$\frac{1}{|\mathcal{T}_{\text{eval}}|} \sum_{t \in \mathcal{T}_{\text{eval}}} W_p(\rho_t, \hat{\rho}_t).$$

We call the task *interpolation* if $\max(\mathcal{T}_{\text{eval}}) < t_K$ and *extrapolation* otherwise.

The dynamics of a continuous-time probability distribution are not identifiable from a finite set of snapshots (Weinreb et al., 2018; Guan et al., 2026a). To make the problem well-posed, additional assumptions on the underlying generating process are made.

2.2 Dynamical assumptions

One common assumption is the conservation of mass, which leads to the *continuity equation* (Ambrosio et al., 2008):

$$\partial_t \rho_t + \nabla \cdot (\rho_t u_t) = 0, \tag{1}$$

where u_t is the velocity. Although the continuity equation does constrain the applicable dynamics, it does not close the system because the velocity evolution is not defined.

A large class of systems exhibits relaxation towards lowest free energy (e.g., diffusion). These processes can be formalized as *Wasserstein gradient flows* (WGF), with their velocity following the steepest descent of the free energy (Jordan et al., 1998; Otto, 2001; Heinonen et al., 2026):

$$u_t = -\nabla \frac{\delta \mathcal{F}}{\delta \rho_t}, \tag{2}$$

where $\mathcal{F}$ is the free energy of the system. This assumption is used in the JKOnet family of methods and Stitching (Bunne et al., 2022; Terpin et al., 2024; Heinonen et al., 2026).

As a first-order condition with a time-independent free energy, WGF cannot describe systems that exhibit periodicity. In this case, a second-order assumption needs to be adopted. Here the velocity is no longer a variable that is defined by the particle positions, but can instead evolve independently.

Ambrosio et al. (2008) show that an absolutely continuous curve satisfying the continuity equation Equation (1) admits multiple compatible velocities. Among these, the canonical velocity v_t provides the solution with the minimum population kinetic energy. We define the canonical velocity through Helmholtz decomposition of the total velocity u_t into a gradient field v_t and a rotational field w_t that moves mass without changing the density:

$$u_t = v_t + w_t, \qquad \nabla \cdot (\rho_t w_t) = 0. \tag{3}$$

Guan et al. (2026a) use this canonical velocity to derive second-order dynamics from the stationary action of the Caldirola–Kanai (CK) Lagrangian (Equation (5)) for linear damping (Caldirola, 1941;

Kanai, 1948):

$$\underbrace{\partial_t v_t + (v_t \cdot \nabla) v_t}_{\text{acceleration}} = \underbrace{-\gamma v_t}_{\text{friction}} \underbrace{-\nabla \frac{\delta \mathcal{U}}{\delta \rho_t}}_{\text{potential force}} . \tag{4}$$

The left-hand side is the acceleration of a particle moving with the flow. Here γ is the friction coefficient and $\mathcal{U}$ is the potential energy. For autonomous systems $\gamma = 0$ corresponds to energy conservation, while $\gamma \to \infty$ recovers WGF (Corollary 1).

The canonical velocity describes the instantaneous change in density, but need not coincide with the observed velocity. For second-order dynamics, however, velocity is part of the state and determines the subsequent motion. We therefore seek a residual for the full velocity u_t, allowing circulation.

2.3 Equations of motion

To derive the equations of motion for the full velocity u_t, we use the Clebsch action principle (Holm et al., 1998; Cotter & Holm, 2009). In addition to continuity, this formulation imposes transport of particle labels: each particle keeps its initial label as it moves. We use the Caldirola–Kanai action

$$\mathcal{A} = \int_0^T \mathrm{e}^{\gamma t} \big(\mathcal{K} - \mathcal{U}\big)\, \mathrm{d}t, \qquad \mathcal{K} = \frac{1}{2} \int \rho_t(x) \|u_t(x)\|^2 \, \mathrm{d}x, \tag{5}$$

where $\mathcal{U}$ is the potential energy and $\gamma \geq 0$ is a constant friction coefficient. The exponential factor produces linear damping, while the kinetic energy retains the full velocity, as in geometric hydrodynamics (Khesin et al., 2021).

Theorem 1 (Lagrangian equations of motion). Under the regularity assumptions in Section B.2, let (ρ_t, u_t) satisfy continuity. For fixed $\mathcal{U}$ and γ, stationarity of Equation (5) under the Clebsch constraints, with density and labels fixed at the temporal endpoints, implies

$$\partial_t u_t + (u_t \cdot \nabla) u_t = -\gamma u_t - \nabla \frac{\delta \mathcal{U}}{\delta \rho_t}. \tag{6}$$

The velocity u_t need not be a gradient.

Proof. See Section B.2. □

Corollary 1 (Recovered dynamics). A smooth solution of Equation (6) with a gradient initial velocity remains canonical and satisfies the WLM dynamics Equation (4). In this canonical class, setting $\gamma = 0$ gives Wasserstein Hamiltonian flow (WHF) (Ambrosio & Gangbo, 2008; Chow et al., 2020). For a time-independent energy, zero damping conserves $\mathcal{K}+\mathcal{U}$, also for general velocities; the formal overdamped limit $\gamma \to \infty$, in slow time $\tau = t/\gamma$, recovers the Wasserstein gradient flow Equation (2).

Proof. See Section B.3. □

Why general velocities. Observed particle velocities need not be canonical. Although replacing them by the canonical velocity preserves the instantaneous change in density, it can alter subsequent motion because velocity is part of the state in second-order dynamics. A residual evaluated along particle trajectories should therefore retain their full velocity. The Clebsch derivation provides this formulation, allowing observed velocities to be used directly as initial conditions without projecting them onto gradient fields. Section B.4 illustrates how this projection can change the subsequent density evolution. The labels and the Lagrange multipliers (see Section B.2) are needed for the derivation and are not learned.

3 Double-Stitch

Lagrangian residual. We turn the equation of motion in Equation (6) into a loss by squaring its discrepancy, averaging over the population, and integrating over time. Along a particle trajectory satisfying $\mathrm{d}x_t/\mathrm{d}t = u_t(x_t)$, this gives

$$\mathcal{R}[\rho, u] = \mathbb{E}_{x_0 \sim \rho_0} \left[\int_0^T \left\| \ddot{x}_t + \gamma \dot{x}_t + \nabla \frac{\delta \mathcal{U}}{\delta \rho_t}(x_t) \right\|^2 \mathrm{d}t \right]. \tag{7}$$

Under the assumptions of Theorem 1, this nonnegative residual vanishes exactly when the equation of motion holds (and vice versa, see Section C.3). It can be evaluated directly on a fitted path without simulating the dynamics.

Objective. Following Stitching, we learn the population dynamics by fitting the learned distribution curve $\hat{\rho}$ directly to the observed snapshots as well as to the assumed dynamics (Heinonen et al., 2026). This means that we aim to minimize a loss with two components: *data loss* $\mathcal{L}_{\text{data}}$, responsible for fitting the observed snapshots, and *residual loss* $\mathcal{L}_{\text{res}}$, responsible for fitting the dynamics:

$$\mathcal{L} \triangleq \mathcal{L}_{\text{data}} + \lambda\, \mathcal{L}_{\text{res}}, \tag{8}$$

where λ is a balancing coefficient. The data loss is computed as a sum of distances between the observed and learned snapshots at the observation times $\mathcal{T}_{\text{obs}}$

$$\mathcal{L}_{\text{data}} \triangleq \sum_{t_i \in \mathcal{T}_{\text{obs}}} \mathcal{D}(\hat{\rho}^\theta_{t_i}, X_i),$$

where $\mathcal{D}$ is a divergence between the learned density and the observed samples, typically the forward KL (negative log-likelihood). The residual loss requires parametrization and discretization of the Lagrangian residual Equation (7).

Distribution parameterization. We parametrize the distribution curve $\hat{\rho} \to \hat{\rho}^\theta$ using kernel density estimation (KDE):

$$\hat{\rho}^\theta_t(x) = \sum_{k=1}^{P} w^\theta_k \phi(x - x^\theta_{t,k}), \qquad \sum_{k=1}^{P} w^\theta_k = 1. \tag{9}$$

Here ϕ is a Gaussian kernel, $x^\theta_{t,k}$ is the learnable center of kernel $k = 1, \dots, P$ at time t, and w^θ_k is its weight in the overall mixture. We can then derive (see Section C.1) a velocity field satisfying continuity for this KDE path

$$u^\theta_t(x) = \sum_{k=1}^{P} r^\theta_{t,k}(x)\, \dot{x}^\theta_{t,k}, \qquad r^\theta_{t,k}(x) = \frac{w^\theta_k \phi(x - x^\theta_{t,k})}{\sum_{l=1}^{P} w^\theta_l \phi(x - x^\theta_{t,l})}, \qquad \sum_{k=1}^{P} r^\theta_{t,k}(x) = 1, \tag{10}$$

where $r^\theta_{t,k}$ are the responsibilities of the kernels at x.

Potentials and forces parametrization. We assume that the potential energy has the form

$$\mathcal{U} = c_V \underbrace{\int_{\mathbb{R}^d} V(x)\ \mathrm{d}\rho(x)}_{\text{external potential}} + c_H \underbrace{\int_{\mathbb{R}^d} \log \rho(x)\ \mathrm{d}\rho(x)}_{\text{internal energy}} + \frac{c_W}{2} \underbrace{\int_{\mathbb{R}^d \times \mathbb{R}^d} W(x - x')\ \mathrm{d}\rho(x)\ \mathrm{d}\rho(x')}_{\text{particle interactions}}. \tag{11}$$

This form is common in applications (Villani, 2003; Heinonen et al., 2026) and usefully has an analytic gradient of the variation with respect to paths $\nabla \frac{\delta \mathcal{U}}{\delta \rho}$ (see Equation (17) in Section C.2). We parameterize the external potential and particle interactions with neural networks $V = V_\theta$ and $W = W_\theta$ and compute the entropy term from the KDE approximation, resulting in a parameterized potential $\mathcal{U}_\theta[\rho]$. In Stitching, this functional is the free energy $\mathcal{F}$ of the gradient flow Equation (2), whose entropy term drives diffusion; here it is the potential energy $\mathcal{U}$ in the second-order law Equation (6), and the same term acts as an isothermal pressure with force $-c_H \nabla \log \rho$.

Empirical objective. To compute the data loss numerically, we simply compute it for the learned KDE density $\hat{\rho}^\theta_t$ and the observed set of particles.

To compute the residual loss, we approximate the population average in Equation (7) by a weighted sum over the learned center paths. We evaluate density-dependent forces using the KDE and approximate the interaction convolution using the centers (Section C.3). The acceleration is the derivative along each center path; this particle approximation differs from the exact material acceleration of the KDE mean velocity, as detailed in Section C.1. Finally, we approximate the time integral by quadrature and the path derivatives by finite differences:

$$\mathcal{L}_{\text{res}}(\theta) = h \sum_{j=0}^{M} \sum_{k} w_k \left\| \Delta_h^2 x_{t_j,k} + \gamma\, \Delta_h x_{t_j,k} + \nabla \frac{\delta \mathcal{U}_\theta}{\delta \rho}\left[\hat{\rho}^\theta_{t_j}\right](x_{t_j,k}) \right\|^2, \tag{12}$$

where Δ_h and Δ_h^2 are the finite differences of the center positions on the time grid $t_0 < \cdots < t_M$ with spacing h, central at the interior nodes and one-sided at the two endpoints, and the terms at $j = 0$ and $j = M$ are halved (trapezoidal rule). Here γ is a learnable friction coefficient.

Initial velocity constraint. When paired initial positions and velocities $\{(x_0^{(n)}, v_0^{(n)})\}_{n=1}^{N_0}$ are available, we add $\lambda_{v_0}\mathcal{L}_{v_0}$ to the objective in Equation (8). Using the KDE responsibilities from Equation (10) and a forward difference on the first grid interval, we set

$$\mathcal{L}_{v_0}(\theta) = \frac{1}{N_0}\sum_{n=1}^{N_0}\left\|\sum_{k=1}^{P} r_{t_0,k}^{\theta}(x_0^{(n)})\,\frac{x_{t_1,k}^{\theta} - x_{t_0,k}^{\theta}}{h} - v_0^{(n)}\right\|^2. \tag{13}$$

This softly constrains the initial velocity field without requiring velocity observations at later times.

Proposition 1 (Training cost). Consider the training cost of one Double-Stitch and WLM training iteration on the same task. Suppose that the number of time steps M, number of particles P, and the data loss evaluation time are the same. Assume that $C(P)$ is the time it takes to make a force evaluation (P particles, including backpropagation), and suppose the hardware can run B force evaluations in parallel. Then, apart from the data loss, one Double-Stitch iteration takes $O(\lceil M/B \rceil\, C(P))$ time while WLM will take $O(M\, C(P))$.

Proof. See Section C.6; it follows directly from the pseudocode of the algorithms in Section C.5. □

4 Experiments

We first demonstrate the difference between the Lagrangian flow Double-Stitch and the gradient flow Stitching on synthetic datasets, then compare the performance of our method against published results on real single-cell and fluid dynamics datasets.

We consider multiple prediction modes: direct, push-forward, and rollout. The direct method is applicable to particle-based methods (i.e., Stitching and Double-Stitch) and essentially scores the particles that are learned directly. Push-forward takes the latest observed snapshot and unrolls the trajectory until the required test time using the learned field. The rollout does the same, but unrolls from the earliest observed snapshot.

We provide additional visualizations, expanded method comparisons, demonstrations of non-identifiability, as well as sensitivity and ablation studies in Section E.

4.1 Synthetic demonstration

First, to illustrate the benefit of second-order modeling, we consider three Gaussian-mixture evolutions: conservative harmonic motion, damped harmonic motion with $\gamma = 1$, and Wasserstein gradient flow. The conservative flow reverses the mixture and conserves total energy. The damped flow is the same process, but the damping reduces the oscillation amplitude. The gradient flow relaxes towards a single Gaussian. Figure 1 shows their densities, energy profiles and Double-Stitch fits (dataset details are given in Section D.1).

We train Stitching (Heinonen et al., 2026) and Double-Stitch with 50 centers, 50 time nodes and 5000 updates on the same three snapshots, holding out two times. For a single seed, the test W_1 of Stitching versus Double-Stitch is 1.324 versus 0.389 on conservative flow, 0.711 versus 0.366 on damped flow, and 0.273 versus 0.292 on gradient flow. Figure 11 in Section E.4 visualizes ground truth and fitted densities produced by both methods.

4.2 Single-cell dynamics

Second, we consider the embryoid-body (EB) single-cell RNA dataset (Moon et al., 2019). It describes the evolution of human embryonic stem cells over 5 snapshots. 3 different training and evaluation protocols have been proposed for this dataset. First, train on all snapshots, holding out 30% of the cells of each for evaluation (Terpin et al., 2024; Persiianov et al., 2025). Second, leave one snapshot out for validation (Tong et al., 2020). Third, leave two snapshots out for validation and average

the performance on them (Shen et al., 2025). We refer to these protocols as leave-particles out (LPO), leave-one-out (L1O), and leave-two-out (L2O) respectively. Since the key methods we compare to, Stitching (Heinonen et al., 2026) and WLM (Guan et al., 2026a) use different protocols, we rerun them alongside our method for consistency. Here we score the particles learned by Double-Stitch directly and use pushforward evaluation for WLM (see dataset details in Section D.1 and training details in Section D.2).

Table 2: **Double-Stitch matches performance of the best gradient and Lagrangian methods on the EB dataset** (W_1 and W_2, mean ± std, 5 seeds). Lower is better. Dashes indicate unreported cells. $V(x)$ and $V(x,t)$ denote static and time-dependent potentials. Stitching and WLM rows are our reruns (Section D.2); Section E.2 also lists their published values.

	Method	LPO, W_1	L1O, W_1	L2O, W_2
Transport, bridges & neural flows	Neural SDE	0.820	—	—
	TrajectoryNet	0.930	0.848	1.980
	SB-FBSDE	0.840	—	—
	NLSB	0.780	0.970	—
	OT-CFM	0.770	0.790 ± 0.068	—
	WLF-OT	0.740	0.742 ± 0.012	—
	WLF-SB	0.730	0.746 ± 0.016	—
	Vanilla-SB	—	—	1.520
	MMSB	—	—	1.420
	DMSB	—	—	1.290
	OT-MFM	—	0.713 ± 0.039	—
WGF energy models	JKOnet	1.340	—	—
	JKOnet*, $V(x)$	1.120	—	1.840
	JKOnet*, $V(x,t)$	0.730	—	3.590
	iJKOnet, $V(x)$	1.050	—	1.120
	iJKOnet, $V(x,t)$	0.580	—	0.920
Simulation-free WGF	Stitching, $V(x)$	0.618 ± 0.007	0.796 ± 0.014	0.821 ± 0.019
	Stitching, $V(x,t)$	0.581 ± 0.010	0.717 ± 0.010	0.786 ± 0.008
Simulation-based WLM	WLM	0.682 ± 0.009	0.734 ± 0.023	0.823 ± 0.041
Simulation-free WLM	Double-Stitch, $V(x)$	0.618 ± 0.001	0.733 ± 0.004	0.794 ± 0.004
	Double-Stitch, $V(x,t)$	0.555 ± 0.005	0.717 ± 0.008	0.799 ± 0.003

We show the results in Table 2 (flow-based and JKOnet models are adapted from Heinonen et al. (2026, Tables 2 and 5) and Guan et al. (2026a, Table 3)). We find that Double-Stitch outperforms or matches WLM using both static and time-variable potential. It also performs better or the same as Stitching for all cases except one (L2O for time dependent potential). Both comparisons use our reruns (see the settings in Section D.2); Section E.2 also lists the published values. Visualization of predictions and extended comparison tables are given in Section E.2.

4.3 Ocean vortex dynamics

Third, we evaluate the methods on two datasets of ocean vortex dynamics in the Gulf of Mexico (Guan et al., 2026a). The first one captures a small vortex and provides an interpolation task: it consists of 9 snapshots of 111 particle positions, with even snapshots used for training and odd ones for testing. The second one captures a large vortex and provides a forecasting task: it consists of 11 snapshots of 400 particle positions and uses the final snapshot for testing (see data details in Section D.1).

Following Guan et al. (2026a), we provide our method with the initial velocity through the initial-velocity constraint introduced in Equation (13). The settings and evaluation protocols are described in Section D.2. At evaluation, both methods receive the observed velocity in pushforward snapshots and the initial velocity for a full rollout.

We present the results in Table 3, where the results for flow-based and reference process methods are adapted from Guan et al. (2026a, Table 2) (see the visualizations and per-snapshot performance in Section E.3). On interpolation, static Double-Stitch matches our WLM rerun and outperforms all other methods except Curly-FM, which relies on reference process. On forecasting, WLM performs best under pushforward, while Double-Stitch still improves on Stitching and reference process methods. Under rollout, static Double-Stitch performs poorly and time-dependent potential brings its error close to WLM, although static Stitching is the best here. Note that at $t = 10$ the time-

Table 3: **Double-Stitch outperforms gradient methods and matches the best Lagrangian method on ocean vortex interpolation** (W_1, mean ± std, 3 seeds). Lower is better. Dashes indicate unreported cells. $V(x)$ and $V(x,t)$ denote static and time-dependent potentials. Stitching and WLM rows are our reruns (Section D.2); Section E.3 also lists WLM's published values.

	Method	Interp. pushfwd.	Forecast pushfwd.	Forecast rollout
Action & continuity	AM	0.376	—	—
	UAM	0.383	—	—
	sAM	0.378	—	—
	DICE	0.244	—	—
Transport & bridges	Vanilla-SB	0.214	—	—
	OT-CFM	0.204	—	—
	OT-MFM	0.071	—	—
Reference process	Curly-FM	0.030	—	—
	SBIRR	0.076	0.567 ± 0.014	1.062 ± 0.111
	Snap-MMD	0.076	0.473 ± 0.013	0.896 ± 0.141
Simulation-free WGF	Stitching, $V(x)$	4.750 ± 0.535	0.205 ± 0.006	0.149 ± 0.007
	Stitching, $V(x,t)$	0.104 ± 0.020	0.170 ± 0.004	0.211 ± 0.092
Simulation-based WLM	WLM	0.064 ± 0.015	0.088 ± 0.023	0.194 ± 0.048
Simulation-free WLM	Double-Stitch, $V(x)$	0.065 ± 0.004	0.134 ± 0.003	1.839 ± 0.539
	Double-Stitch, $V(x,t)$	0.073 ± 0.009	0.119 ± 0.019	0.222 ± 0.008

dependent potential is evaluated beyond the last observed snapshot, which assumes that its learned time dependence extrapolates.

4.4 Computational cost

Finally, we investigate the computational cost of our simulation-free algorithm and compare it to the simulation-based WLM.

First, we measure the practical training time on EB and Oceans (see Table 4). Double-Stitch trains in roughly 4–12 minutes on EB and 7–10 minutes on oceans, while WLM takes 16–54 and 97–145 minutes on identical hardware (see the details in Section D.2).

Second, we match the parameters of the two methods (number of updates, time steps and data) and run them on the synthetic WHF Gaussian mixture flow from Figure 1. We find that WLM takes almost 7.6 times as long (225 vs 30 seconds, Figure 2) to reach the same level of performance (test W_1 of about 0.09).

This illustrates the computational advantage of simulation-free training in this setting.

Table 4: **Double-Stitch requires less training time on EB and Oceans** (minutes, mean ± std, 3 seeds). Static Double-Stitch.

Dataset	Task	WLM	Double-Stitch
EB	LPO	17.75 ± 0.23	3.93 ± 0.02
	L1O	53.92 ± 0.45	11.50 ± 0.78
	L2O	15.80 ± 0.75	3.52 ± 0.03
Oceans	Interp.	96.54 ± 2.41	6.81 ± 0.32
	Forecast	145.05 ± 14.61	10.44 ± 0.10

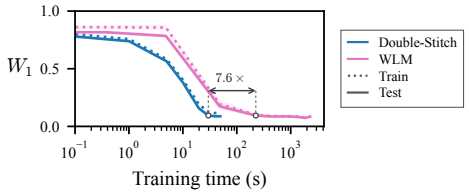

Figure 2: **Training on synthetic WHF** (mean ± std, 3 seeds). Static methods. Arrow shows the training time ratio at test $W_1 \approx 0.09$.

5 Related work

Flow-based methods. Flow matching (Lipman et al., 2023; Tong et al., 2024), dynamic optimal transport (Tong et al., 2020), and Schrödinger bridges (Vargas et al., 2021; Chen et al., 2022; Koshizuka & Sato, 2023; Chen et al., 2023) learn a time-dependent velocity field or stochastic bridge that transports one snapshot into the next according to some optimality criterion (usually the minimal kinetic energy trajectory). Action matching (Neklyudov et al., 2023) fits the potential jointly to all

marginals without a coupling; Neklyudov et al. (2024) generalize the interpolant to any user-specified Lagrangian. As described above, without additional assumptions, these methods are generally not suitable for extrapolation beyond the last observed snapshot (Berlinghieri et al., 2025). However, they can be extended to handle non-gradient flows by adding a reference process (Petrović et al., 2025; Shen et al., 2025).

Gradient flow methods. The JKOnet family of methods imposes WGF via a Jordan–Kinderlehrer–Otto (JKO) scheme, which reduces the problem to learning the potential function $\mathcal{F}$. Various methods offer different ways of learning it (Bunne et al., 2022; Amos et al., 2017; Persiianov et al., 2025), but they all need to unroll the JKO scheme to get the population trajectory. Alternatively, WGF can be softly imposed via a Wasserstein residual, similar to physics-informed collocation losses (Raissi et al., 2019). Stitching (Heinonen et al., 2026) directly optimizes the trajectory and learns the potential to fit the observed snapshots and minimize the residual. GenWGP (Liu & Zhou, 2026) uses the same residual for a known energy but integrates a Neural ODE.

Lagrangian flow methods. Neklyudov et al. (2024) use a given Lagrangian in an action-matching framework. Guan et al. (2026a) use a CK Lagrangian in Wasserstein space with Lagrange multipliers and propose a simulation-based algorithm for learning its energy and friction. We derive the same dynamical law using an alternative approach proposed in geometric hydrodynamics, specifically the Clebsch variational formulation (Cotter & Holm, 2009) of Euler–Arnold and Euler–Poincaré equations (Holm et al., 1998; Khesin et al., 2021).

Residuals and numerics. Penalizing the residual of a governing equation has long been used in physics-informed neural networks (Raissi et al., 2019; Karniadakis et al., 2021). Their optimization is known to suffer from competing loss terms and stiff dynamics (Krishnapriyan et al., 2021; Wang et al., 2021). Related works have enforced some constraints like conservation laws by construction (Richter-Powell et al., 2022). Our method uses both these approaches: dynamics are imposed softly via a residual while continuity is enforced by construction. We find that numerical methods offer useful language to compare methods for learning population dynamics: Stitching and Double-Stitch are collocation methods (directly optimize the trajectory), JKOnet and WLM are shooting methods (simulate the trajectory forward), and multiple shooting isn't represented in the field (Bock & Plitt, 1984).

6 Discussion

Summary. We present Double-Stitch, a simulation-free residual-based method for learning population dynamics. We derive a Lagrangian dynamical law for general (non-gradient) velocities, which leads to a Wasserstein residual allowing learning gradient, conservative, and damped dynamics. We show that our method matches or outperforms Stitching and WLM on single-cell data and ocean interpolation while training 4–14 times faster than WLM, although WLM remains better on ocean forecasting.

Limitations. As a residual-based method and unlike simulation-based methods, Double-Stitch does not guarantee exact satisfaction of the assumed dynamical law. For the same reason, its optimization may be challenging if the initialization is poor or the residuals compete in a complex manner.

Future work. While we focus on the Caldirola–Kanai Lagrangian formulation, future work could investigate alternative Lagrangians (e.g., with Rayleigh dissipation functions) and consider more general mechanics such as active matter and viscous flows. Another line of research could develop hybrid methods that combine the advantages of Double-Stitch and WLM via multiple-shooting.

AI use statement

In this work, we used generative AI tools for creating and editing software code (including implementation of methods, plotting, table extraction, and references formatting), proposing keywords, editing the manuscript to improve readability (including drafting some paragraphs from raw notes), assistance in the writing of proofs and mathematical claims (specifically unifying notation, transcribing and adapting equations, verifying proof correctness), generating synthetic datasets (specifically finding the right parameter settings to make them visual and consistent). We have not used generative AI tools to clean and reformat datasets, discover research topics or identify gaps, assist with translation, suggest a structure for a research paper and support qualitative and thematic data analysis. Tasks such as formulating questions for surveys or interviews, creating artifacts, and transcribing recordings of research material are not applicable to this work. We have reviewed all AI-assisted work (derivations were checked manually and generated code was verified and tested). We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

Ethics statement

This work focuses on machine learning research using publicly available datasets with applications to computational biology and physical sciences. We are not aware of ethical concerns beyond the general ones common to all statistical and data science research.

Reproducibility statement

We describe the proofs in Section B, method derivation in Section C, data details in Section D.1, and training details in Section D.2. The code is provided at https://anonymous.4open.science/r/double-stitching-anon-D9CE.

References

Luigi Ambrosio and Wilfrid Gangbo. Hamiltonian ODEs in the Wasserstein space of probability measures. *Communications on Pure and Applied Mathematics*, 61(1):18–53, 2008. doi: 10.1002/cpa.20188.

Luigi Ambrosio, Nicola Gigli, and Giuseppe Savaré. *Gradient Flows in Metric Spaces and in the Space of Probability Measures*. Birkhäuser Basel, Basel, 2 edition, 2008. doi: 10.1007/978-3-7643-8722-8.

Brandon Amos, Lei Xu, and J. Zico Kolter. Input Convex Neural Networks. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning (ICML)*, volume 70 of *PMLR*, pp. 146–155. PMLR, 2017. URL https://proceedings.mlr.press/v70/amos17b.html.

Lazar Atanackovic, Xi 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 *The Thirteenth International Conference on Learning Representations (ICLR)*, 2025. arXiv:2408.14608.

Renato Berlinghieri, Yunyi Shen, Jialong Jiang, and Tamara Broderick. Oh snapmmd! forecasting stochastic dynamics beyond the schrödinger bridge's end. *arXiv preprint*, 2025. arXiv:2505.16082.

Tobias Blickhan, Jules Berman, Andrew Stuart, and Benjamin Peherstorfer. Dice: Discrete inverse continuity equation for learning population dynamics. *arXiv preprint*, 2025. arXiv:2507.05107.

H.G. Bock and K.J. Plitt. A Multiple Shooting Algorithm for Direct Solution of Optimal Control Problems. *IFAC Proceedings Volumes*, 17(2):1603–1608, July 1984. doi: 10.1016/S1474-6670(17)61205-9.

Charlotte Bunne, Laetitia Papaxanthos, Andreas Krause, and Marco Cuturi. Proximal Optimal Transport Modeling of Population Dynamics. In *Proceedings of the 25th International Conference on*

Artificial Intelligence and Statistics (AISTATS), volume 151 of *PMLR*, pp. 6511–6528, 2022. URL https://proceedings.mlr.press/v151/bunne22a.html.

Piero Caldirola. Forze non conservative nella meccanica quantistica. *Il Nuovo Cimento*, 18(9):393–400, November 1941. doi: 10.1007/bf02960144.

Tianrong Chen, Guan-Horng Liu, and Evangelos A. Theodorou. Likelihood training of schrödinger bridge using forward-backward sdes theory. In *International Conference on Learning Representations (ICLR)*, 2022. arXiv:2110.11291.

Tianrong Chen, Guan-Horng Liu, Molei Tao, and Evangelos A. Theodorou. Deep momentum multi-marginal schrödinger bridge. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. arXiv:2303.01751.

Shui-Nee Chow, Wuchen Li, and Haomin Zhou. Wasserstein Hamiltonian flows. *Journal of Differential Equations*, 268(3):1205–1219, 2020. doi: 10.1016/j.jde.2019.08.046.

C. J. Cotter and D. D. Holm. Continuous and Discrete Clebsch Variational Principles. *Foundations of Computational Mathematics*, 9(2):221–242, April 2009. doi: 10.1007/s10208-007-9022-9.

Vincent Guan, Joseph Janssen, Hossein Rahmani, Andrew Warren, Stephen Zhang, Elina Robeva, and Geoffrey Schiebinger. Identifying drift, diffusion, and causal structure from temporal snapshots, 2024.

Vincent Guan, Lazar Atanackovic, and Kirill Neklyudov. A Call to Lagrangian Action: Learning Population Mechanics from Temporal Snapshots. In *Proceedings of the 43rd International Conference on Machine Learning (ICML)*, 2026a. arXiv:2605.08550. Spotlight.

Vincent Guan, Joseph Janssen, Nicolas Lanzetti, Antonio Terpin, Geoffrey Schiebinger, and Elina Robeva. Gradient-flow SDEs have unique transient population dynamics. In *Proceedings of The 29th International Conference on Artificial Intelligence and Statistics*, volume 300 of *Proceedings of Machine Learning Research*, pp. 2449–2457. PMLR, 2026b. URL https://proceedings.mlr.press/v300/guan26a.html.

Markus Heinonen, Yair Shenfeld, Ricardo Baptista, Daniel Waxman, Dmitry Batenkov, Tim Cooijmans, and Eli Bingham. Wasserstein Residuals: Learning Gradient Flows from Population Dynamics. *arXiv preprint*, 2026. arXiv:2607.04738.

Darryl D Holm, Jerrold E Marsden, and Tudor S Ratiu. The Euler–Poincaré Equations and Semidirect Products with Applications to Continuum Theories. *Advances in Mathematics*, 137(1):1–81, July 1998. doi: 10.1006/aima.1998.1721.

Richard Jordan, David Kinderlehrer, and Felix Otto. The Variational Formulation of the Fokker–Planck Equation. *SIAM Journal on Mathematical Analysis*, 29(1):1–17, January 1998. doi: 10.1137/S0036141096303359.

E. Kanai. On the Quantization of the Dissipative Systems. *Progress of Theoretical Physics*, 3(4):440–442, December 1948. doi: 10.1143/ptp/3.4.440.

Kacper Kapuśniak, Peter Potaptchik, Teodora Reu, Leo Zhang, Alexander Tong, Michael Bronstein, Avishek Joey Bose, and Francesco Di Giovanni. Metric flow matching for smooth interpolations on the data manifold. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. arXiv:2405.14780.

George Em Karniadakis, Ioannis G. Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, May 2021. doi: 10.1038/s42254-021-00314-5.

Boris Khesin, Gerard Misiołek, and Klas Modin. Geometric Hydrodynamics and Infinite-Dimensional Newton's Equations. *Bulletin of the American Mathematical Society*, 58(3):377–442, 2021. doi: 10.1090/bull/1728.

Takeshi Koshizuka and Issei Sato. Neural lagrangian schrödinger bridge: Diffusion modeling for population dynamics. In *International Conference on Learning Representations (ICLR)*, 2023. arXiv:2204.04853.

Aditi S. Krishnapriyan, Amir Gholami, Shandian Zhe, Robert M. Kirby, and Michael W. Mahoney. Characterizing possible failure modes in physics-informed neural networks. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P. S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pp. 26548–26560. Curran Associates, Inc., 2021. arXiv:2109.01050.

Xuechen Li, Ting-Kam Leonard Wong, Ricky T. Q. Chen, and David Duvenaud. Scalable gradients for stochastic differential equations. In Silvia Chiappa and Roberto Calandra (eds.), *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 108 of *Proceedings of Machine Learning Research*, pp. 3870–3882. PMLR, 2020. URL https://proceedings.mlr.press/v108/li20i.html.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow Matching for Generative Modeling. In *The Eleventh International Conference on Learning Representations (ICLR)*, 2023. arXiv:2210.02747.

Chengyu Liu and Xiang Zhou. Generative Path-Finding Method for Wasserstein Gradient Flow. *arXiv preprint*, 2026. arXiv:2604.11519.

Bertrand Maury, Aude Roudneff-Chupin, and Filippo Santambrogio. A Macroscopic Crowd Motion Model of Gradient Flow Type. *Mathematical Models and Methods in Applied Sciences*, 20(10): 1787–1821, October 2010. doi: 10.1142/S0218202510004799.

Robert J. McCann. A convexity principle for interacting gases. *Advances in Mathematics*, 128(1): 153–179, 1997. doi: 10.1006/aima.1997.1634.

Leland McInnes, John Healy, and James Melville. UMAP: Uniform manifold approximation and projection for dimension reduction, 2018.

Kevin R. Moon, David van Dijk, Zheng Wang, Scott Gigante, Daniel B. Burkhardt, William S. Chen, Kristina Yim, Antonia van den Elzen, Matthew J. Hirn, Ronald R. Coifman, Natalia B. Ivanova, Guy Wolf, and Smita Krishnaswamy. Visualizing structure and transitions in high-dimensional biological data. *Nature Biotechnology*, 37(12):1482–1492, 2019. doi: 10.1038/s41587-019-0336-3.

Kirill Neklyudov, Rob Brekelmans, Daniel Severo, and Alireza Makhzani. Action Matching: Learning Stochastic Dynamics from Samples. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning (ICML)*, volume 202 of *PMLR*, pp. 25858–25889. PMLR, 2023. URL https://proceedings.mlr.press/v202/neklyudov23a.html.

Kirill Neklyudov, Rob Brekelmans, Alexander Tong, Lazar Atanackovic, Qiang Liu, and Alireza Makhzani. A Computational Framework for Solving Wasserstein Lagrangian Flows. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning (ICML)*, volume 235 of *PMLR*, pp. 37461–37485. PMLR, 2024. URL https://proceedings.mlr.press/v235/neklyudov24a.html.

Felix Otto. The Geometry of Dissipative Evolution Equations: The Porous Medium Equation. *Communications in Partial Differential Equations*, 26(1-2):101–174, January 2001. doi: 10.1081/PDE-100002243.

Mikhail Persiianov, Jiawei Chen, Petr Mokrov, Alexander Tyurin, Evgeny Burnaev, and Alexander Korotin. Learning of Population Dynamics: Inverse Optimization Meets JKO Scheme. *arXiv preprint*, 2025. arXiv:2506.01502.

Katarina Petrović, Lazar Atanackovic, Viggo Moro, Kacper Kapuśniak, İsmail İlkan Ceylan, Michael Bronstein, Avishek Joey Bose, and Alexander Tong. Curly Flow Matching for Learning Non-gradient Field Dynamics. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. arXiv:2510.26645.

M. Raissi, P. Perdikaris, and G.E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, February 2019. doi: 10.1016/j.jcp.2018.10.045.

Jack Richter-Powell, Yaron Lipman, and Ricky T. Q. Chen. Neural Conservation Laws: A Divergence-Free Perspective. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pp. 38075–38088. Curran Associates, Inc., 2022. doi: 10.52202/068431-2759.

Kenneth Rose, Eitan Gurewitz, and Geoffrey C. Fox. Statistical mechanics and phase transitions in clustering. *Physical Review Letters*, 65(8):945–948, 1990. doi: 10.1103/PhysRevLett.65.945.

Simo Särkkä and Arno Solin. *Applied Stochastic Differential Equations*. Number 10 in Institute of Mathematical Statistics Textbooks. Cambridge University Press, Cambridge, 2019. doi: 10.1017/9781108186735.

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, February 2019. doi: 10.1016/j.cell.2019.01.006.

Yunyi Shen, Renato Berlinghieri, and Tamara Broderick. Multi-marginal Schrödinger Bridges with Iterative Reference Refinement. In Yingzhen Li, Stephan Mandt, Shipra Agrawal, and Emtiyaz Khan (eds.), *Proceedings of The 28th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 258 of *PMLR*, pp. 3817–3825. PMLR, 2025. URL https://proceedings.mlr.press/v258/shen25b.html.

Bernard W. Silverman. *Density Estimation for Statistics and Data Analysis*. Chapman and Hall, London, 1986. doi: 10.1201/9781315140919.

Antonio Terpin, Nicolas Lanzetti, Martín Gadea, and Florian Dörfler. Learning diffusion at lightspeed. *Advances in Neural Information Processing Systems (NeurIPS)*, 37, 2024. arXiv:2406.12616. Oral.

Alexander Tong, Jessie Huang, Guy Wolf, David Van Dijk, and Smita Krishnaswamy. TrajectoryNet: A Dynamic Optimal Transport Network for Modeling Cellular Dynamics. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning (ICML)*, volume 119 of *PMLR*, pp. 9526–9536. PMLR, 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. arXiv:2302.00482.

Francisco Vargas, Pierre Thodoroff, Austen Lamacraft, and Neil Lawrence. Solving Schrödinger bridges via maximum likelihood. *Entropy*, 23(9):1134, 2021. doi: 10.3390/e23091134.

Cédric Villani. *Topics in Optimal Transportation*, volume 58 of *Graduate Studies in Mathematics*. American Mathematical Society, 2003. doi: 10.1090/gsm/058.

Sifan Wang, Yujun Teng, and Paris Perdikaris. Understanding and Mitigating Gradient Flow Pathologies in Physics-Informed Neural Networks. *SIAM Journal on Scientific Computing*, 43(5):A3055–A3081, January 2021. doi: 10.1137/20M1318043.

Caleb Weinreb, Samuel Wolock, Betsabeh K. Tusi, Merav Socolovsky, and Allon M. Klein. Fundamental limits on dynamic inference from single-cell snapshots. *Proceedings of the National Academy of Sciences*, 115(10):E2467–E2476, 2018. doi: 10.1073/pnas.1714723115.

Appendix contents

A Notation **15**

B Derivation of the equations of motion **16**
B.1 Lagrange multipliers 16
B.2 Clebsch action principle 17
B.3 Recovered flows 18
B.4 General and canonical velocities 19

C Derivation of Double-Stitch **19**
C.1 Velocity and acceleration 19
C.2 Energy variation and Wasserstein gradient 20
C.3 Lagrangian residual and empirical approximation 20
C.4 Time discretization 20
C.5 Pseudocode 21
C.6 Computational complexity 22

D Experimental details **23**
D.1 Data 23
D.2 Training and evaluation 24

E Additional results **26**
E.1 Identifiability 26
E.2 Single-cell dynamics 29
E.3 Ocean vortex dynamics 30
E.4 Synthetic demonstration 32
E.5 Ablations and sensitivity studies 32

A Notation

Table 5: Notation

Symbol	Meaning
Indices	
i	Snapshot, $i = 0, \dots, K$
n	Sample within a snapshot, $n = 1, \dots, N_i$
k, l	Learned centers, $k, l = 1, \dots, P$
j	Time step, $j = 0, \dots, M$
Problem setting	
$\mathcal{P}_2(\mathbb{R}^d)$	Probability measures with finite second moment
ρ_t	Population distribution at time t
X_i	Observed snapshot at time t_i
N_i	Number of observations in snapshot i (N if equal)
d	Dimension of the state space
$\mathcal{T}_{\text{obs}}$	Observation times used for fitting
$\mathcal{T}_{\text{eval}}$	Times used for evaluation
$\hat{\rho}_t$	Predicted population distribution
W_p	p-Wasserstein distance
Dynamics and Clebsch variables	
u_t	General population velocity
v_t	Canonical velocity with minimum population kinetic energy
w_t	Density-preserving component of the velocity
x_t	Particle position at time t
$\partial_t f_t(x)$	Time derivative of a field at fixed position
$\mathrm{d}f_t(x_t)/\mathrm{d}t$	Total derivative of a field along a particle trajectory
$\mathcal{F}$	Free energy of a Wasserstein gradient flow
$\mathcal{U}$	Potential energy of the population
$\mathcal{K}$	Kinetic energy of the population
$\nabla(\delta\mathcal{U}/\delta\rho_t)$	Wasserstein gradient of the potential energy
γ	Linear friction coefficient
$\mathcal{A}$	Caldirola–Kanai action
ℓ_t	Material label field
ϕ_t	Continuity multiplier
μ_t	Clebsch multiplier for label transport
Double-Stitch parameterization	
θ	Learned parameters
$\hat{\rho}_t^\theta$	Learned kernel density estimate (KDE)
u_t^θ	Responsibility-weighted mean of the center velocities
$x_{t,k}^\theta$	Position of learned center k at time t
w_k^θ	Time-independent mixture weight of center k
P	Number of particles (learned centers in Double-Stitch, simulated particles in WLM)
ϕ	Smoothing kernel
σ	Kernel bandwidth
$r_{t,k}^\theta(x)$	Responsibility of center k at position x
ρ_t^P	Weighted empirical measure of the centers
V_θ	Learned external potential
W_θ	Learned interaction potential
c_V	External-potential coefficient
c_H	Entropy coefficient
c_W	Interaction coefficient
M	Number of time steps (grid intervals or solver steps)
h	Time-grid spacing
Δ_h	First-difference operator (central; one-sided at the endpoints)
Δ_h^2	Second-difference operator (central; one-sided at the endpoints)
Objectives and computational cost	
$\mathcal{L}$	Training objective; mechanical Lagrangian in the action derivation
$\mathcal{L}_{\text{data}}$	Data loss
$\mathcal{L}_{\text{res}}$	Discretized residual loss
λ	Residual-loss weight
$\mathcal{D}$	Data divergence
$\mathcal{R}$	Population Lagrangian residual
$C(P)$	Time of one force evaluation on P particles, including backpropagation
B	Number of force evaluations the hardware can run at once

B Derivation of the equations of motion

We consider a flow of probability measures $\rho_t \in \mathcal{P}_2(\mathbb{R}^d)$ with density and velocity field u_t smooth in space and time, satisfying the continuity equation (Equation (1)):

$$\partial_t \rho_t + \nabla \cdot (\rho_t u_t) = 0.$$

The classical Lagrangian $\mathcal{L} = \mathcal{K} - \mathcal{U}$ with kinetic energy

$$\mathcal{K} = \frac{1}{2} \int \rho_t(x) \|u_t(x)\|^2 \, \mathrm{d}x$$

and a differentiable potential energy $\mathcal{U}$ with a smooth first variation describe Hamiltonian dynamics. For canonical velocities, these are Wasserstein Hamiltonian flows (WHF) (Chow et al., 2020). Total energy is conserved when $\mathcal{U}$ has no explicit time dependence. The corresponding action is

$$\mathcal{A} = \int_0^T \big(\mathcal{K} - \mathcal{U}\big) \, \mathrm{d}t.$$

To describe dissipative flows as well, we adopt the Caldirola–Kanai Lagrangian $\mathcal{L} = \mathrm{e}^{\gamma t}(\mathcal{K} - \mathcal{U})$, where $\gamma \geq 0$ is a constant friction coefficient (Caldirola, 1941; Kanai, 1948). This leads to a damped action

$$\mathcal{A} = \int_0^T \mathrm{e}^{\gamma t} \big(\mathcal{K} - \mathcal{U}\big) \, \mathrm{d}t.$$

For a time-independent energy, WGF is recovered in the overdamped limit after rescaling time (Section B.3).

Using this action and the continuity constraint, we derive the equations of motion that can be used by a trajectory simulator (for simulation-based methods) or in a residual loss (for simulation-free methods).

B.1 Lagrange multipliers

One common approach to derive the equations of motion is to enforce continuity within the action via a Lagrange multiplier (Guan et al., 2026a). We write the velocity here as v_t for reasons that will become clear soon.

Denote the scalar Lagrange multiplier by ϕ_t and the spatial inner product by $\langle f, g \rangle = \int f(x) \cdot g(x) \, \mathrm{d}x$. Then the constrained action is

$$\mathcal{A} = \int_0^T \mathrm{e}^{\gamma t} \Big[\mathcal{K} - \mathcal{U} + \underbrace{\langle \phi_t, \partial_t \rho_t + \nabla \cdot (\rho_t v_t) \rangle}_{\text{continuity}} \Big] \, \mathrm{d}t.$$

Following the principle of stationary action, we set its first variations to zero and integrate by parts:

$$\frac{\delta \mathcal{A}}{\delta \rho_t} = 0 \implies (\partial_t + v_t \cdot \nabla)\phi_t = \frac{1}{2}\|v_t\|^2 - \frac{\delta \mathcal{U}}{\delta \rho_t} - \gamma \phi_t, \qquad \text{(potential evolution)}$$

$$\frac{\delta \mathcal{A}}{\delta \phi_t} = 0 \implies \partial_t \rho_t + \nabla \cdot (\rho_t v_t) = 0, \qquad \text{(continuity)}$$

$$\frac{\delta \mathcal{A}}{\delta v_t} = 0 \implies v_t = \nabla \phi_t. \qquad \text{(gradient velocity)}$$

Here ∂_t differentiates a field at a fixed spatial position. Along a moving particle with $\mathrm{d}x_t/\mathrm{d}t = v_t(x_t)$, the ordinary time derivative gives the material derivative by the chain rule,

$$\frac{\mathrm{d}}{\mathrm{d}t} f_t(x_t) = \big[\partial_t f_t + (v_t \cdot \nabla) f_t\big](x_t).$$

The first equation, together with the third, gives the damped Hamilton–Jacobi equation and recovers the equations of motion in Guan et al. (2026a, Equations (6)–(8)). In Lagrangian coordinates, with $\frac{\mathrm{d}x_t}{\mathrm{d}t} = v_t(x_t)$, it can be written as

$$\frac{\mathrm{d}^2 x_t}{\mathrm{d}t^2} = -\nabla \frac{\delta \mathcal{U}}{\delta \rho_t}(x_t) - \gamma \frac{\mathrm{d}x_t}{\mathrm{d}t}.$$

The second equation is the continuity condition that we wished to impose.

The third equation restricts the velocity to be a gradient or in other words to be irrotational. This means v_t is the canonical velocity of Ambrosio et al. (2008, Chapter 8) with the smallest kinetic energy among velocity fields producing the same instantaneous change in density (and this is the reason we used v_t instead of u_t to denote it).

However, although the rotational component of the velocity (w_t in the Helmholtz decomposition Equation (3)) does not affect the density instantaneously, it can still affect the motion of the observed particles and therefore density evolution. We discuss this effect in more detail in Section B.4 after introducing the alternative derivation.

B.2 Clebsch action principle

To retain the rotational component of the velocity in the equations of motion, we use the Clebsch action principle (Cotter & Holm, 2009; Holm et al., 1998). Here we use the general velocity u_t, with particle trajectories satisfying $\mathrm{d}x_t/\mathrm{d}t = u_t(x_t)$.

Specifically, define a material label field $\ell_t(x) \in \mathbb{R}^d$ that records the initial position of the particle currently at x. A particle keeps its identity during the flow, so $\ell_t(x_t) = x_0$ and $\ell_0(x) = x$. This gives the following constraint

$$\partial_t \ell_t + (u_t \cdot \nabla)\ell_t = 0.$$

We adjoin this constraint with a vector multiplier $\rho_t \mu_t$ to obtain the action

$$\mathcal{A} = \int_0^T \mathrm{e}^{\gamma t}\Big[\mathcal{K} - \mathcal{U} + \underbrace{\langle \phi_t, \partial_t \rho_t + \nabla\cdot(\rho_t u_t)\rangle}_{\text{continuity}} - \underbrace{\langle \rho_t \mu_t, (\partial_t + u_t\cdot\nabla)\ell_t\rangle}_{\text{label transport}}\Big]\,\mathrm{d}t. \tag{14}$$

Theorem 1 (Lagrangian equations of motion). Let $\rho_t \in \mathcal{P}_2(\mathbb{R}^d)$ be a strictly positive density and u_t a velocity field, both smooth in space and time on $[0, T]$ and satisfying continuity. Suppose the particle flow defined by $\mathrm{d}x_t/\mathrm{d}t = u_t(x_t)$ has a smooth inverse. Fix a constant $\gamma \geq 0$ and a differentiable energy $\mathcal{U}$ with a smooth first variation. Assume finite kinetic energy and action, and sufficient integrability to differentiate under the integrals. Variations are smooth and compactly supported in space, with density and particle labels fixed at the temporal endpoints.

If the Clebsch action in Equation (14) is stationary with respect to ρ_t, u_t and smooth labels and multipliers ℓ_t, ϕ_t, μ_t, then

$$\partial_t u_t + (u_t\cdot\nabla)u_t = -\gamma u_t - \nabla\frac{\delta \mathcal{U}}{\delta \rho_t}.$$

The velocity u_t need not be a gradient.

Proof. Similarly to the previous derivation, we set the variations to zero for stationary action:

$$\begin{aligned}
&\frac{\delta \mathcal{A}}{\delta \rho_t} = 0 \implies (\partial_t + u_t\cdot\nabla)\phi_t = \frac{1}{2}\|u_t\|^2 - \frac{\delta \mathcal{U}}{\delta \rho_t} - \gamma\phi_t, && \text{(potential evolution)}\\
&\frac{\delta \mathcal{A}}{\delta \phi_t} = 0 \implies \partial_t \rho_t + \nabla\cdot(\rho_t u_t) = 0, && \text{(continuity)}\\
&\frac{\delta \mathcal{A}}{\delta u_t} = 0 \implies u_t = \nabla\phi_t + (\nabla\ell_t)^\top \mu_t, && \text{(Clebsch velocity)}\\
&\frac{\delta \mathcal{A}}{\delta \mu_t} = 0 \implies (\partial_t + u_t\cdot\nabla)\ell_t = 0, && \text{(label transport)}\\
&\frac{\delta \mathcal{A}}{\delta \ell_t} = 0 \implies (\partial_t + u_t\cdot\nabla)\mu_t = -\gamma\mu_t, && \text{(multiplier evolution)}
\end{aligned} \tag{15}$$

The first two equations have the same form as before but now with the general velocity u_t.

The third equation adds the label term $(\nabla\ell_t)^\top\mu_t$, which permits nonzero vorticity. Here $\nabla\phi_t$ is not necessarily the canonical velocity v_t because the label term does not have to preserve density.

The fourth equation shows that particle keeps its initial label.

The fifth equation shows that the multiplier is transported with the particles and decays at γ rate because of the exponent in the action.

We use $(\nabla \ell_t)_{ab} = \partial_b \ell_{t,a}$, with the same Jacobian convention for u_t. The equation of motion then follows from differentiating the Clebsch velocity along a particle. For a scalar field f_t, the chain rule gives

$$\frac{\mathrm{d}}{\mathrm{d}t}\big[\nabla f_t(x_t)\big] = \Big[\nabla\big(\partial_t f_t + (u_t \cdot \nabla) f_t\big) - (\nabla u_t)^\top \nabla f_t\Big](x_t).$$

We apply this to ϕ_t and each label component and substitute the stationarity conditions:

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t} u_t(x_t) &= \left[\nabla\left(\frac{1}{2}\|u_t\|^2 - \frac{\delta \mathcal{U}}{\delta \rho_t}\right) - (\nabla u_t)^\top u_t - \gamma u_t\right](x_t) \\ &= -\nabla \frac{\delta \mathcal{U}}{\delta \rho_t}(x_t) - \gamma u_t(x_t), \end{aligned} \tag{16}$$

where we use $(\nabla u_t)^\top u_t = \frac{1}{2}\nabla\left(\|u_t\|^2\right)$. In Lagrangian coordinates, with $\frac{\mathrm{d}x_t}{\mathrm{d}t} = u_t(x_t)$, this becomes

$$\frac{\mathrm{d}^2 x_t}{\mathrm{d}t^2} = -\nabla \frac{\delta \mathcal{U}}{\delta \rho_t}(x_t) - \gamma \frac{\mathrm{d}x_t}{\mathrm{d}t}.$$

This is the same particle equation as before, now allowing a general initial velocity.

□

B.3 Recovered flows

The dynamical law in Equation (16) describes damped flow with a general initial velocity. We now show how it recovers the canonical dynamics of WLM, conservative dynamics, Wasserstein Hamiltonian flow, and Wasserstein gradient flow.

Canonical dynamics (WLM). If the initial velocity is a gradient: $u_0 = v_0 = \nabla \phi_0$. Choosing $\mu_0 = 0$ and using the multiplier evolution in Equation (15), we get $\mu_t = 0$ throughout the smooth flow. The velocity then remains canonical: $u_t = v_t = \nabla \phi_t$. The dynamical law becomes

$$\partial_t \phi_t + \frac{1}{2}\|\nabla \phi_t\|^2 + \frac{\delta \mathcal{U}}{\delta \rho_t} + \gamma \phi_t = 0.$$

Together with continuity, this recovers the Hamiltonian equations of motion of Guan et al. (2026a, Theorem 2.2) and proves Corollary 1.

Conservative dynamics. With $\gamma = 0$ and no explicit time dependence in $\mathcal{U}$ (otherwise its time derivative contributes to the energy balance as well), the flow conserves total energy, including for general velocities. Differentiating the potential energy along the flow and using continuity gives

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathcal{U} = \int \rho_t u_t \cdot \nabla \frac{\delta \mathcal{U}}{\delta \rho_t}\,\mathrm{d}x.$$

Similarly, differentiating the full energy and substituting the dynamical law gives

$$\frac{\mathrm{d}}{\mathrm{d}t}(\mathcal{K} + \mathcal{U}) = \int \rho_t u_t \cdot \left(\partial_t u_t + (u_t \cdot \nabla) u_t + \nabla \frac{\delta \mathcal{U}}{\delta \rho_t}\right) \mathrm{d}x = -\gamma \int \|u_t\|^2 \rho_t \,\mathrm{d}x.$$

Therefore the total energy is conserved for $\gamma = 0$ and dissipated for $\gamma > 0$.

Wasserstein Hamiltonian flow. Setting $\gamma = 0$ in the canonical dynamics gives Wasserstein Hamiltonian flow (Ambrosio & Gangbo, 2008; Chow et al., 2020). Here ρ_t and ϕ_t satisfy continuity and the undamped Hamilton–Jacobi equation. For a time-independent potential energy, the Hamiltonian (total energy) $\mathcal{K} + \mathcal{U}$ is conserved (see above).

Wasserstein gradient flow. For a time-independent energy $\mathcal{U} = \mathcal{F}$, the particle form of Equation (16) has the same form as the one used by Guan et al. (2026a, Proposition 2.4) to recover WGF in the overdamped limit. Explicitly, in slow time $\tau = t/\gamma$, writing $\widetilde{x}_\tau = x_{\gamma\tau}$ and $\widetilde{\rho}_\tau = \rho_{\gamma\tau}$ gives

$$\frac{1}{\gamma^2}\frac{\mathrm{d}^2\widetilde{x}_\tau}{\mathrm{d}\tau^2} + \frac{\mathrm{d}\widetilde{x}_\tau}{\mathrm{d}\tau} = -\nabla\frac{\delta\mathcal{F}}{\delta\widetilde{\rho}_\tau}(\widetilde{x}_\tau).$$

Formally dropping the inertial term as $\gamma \to \infty$ yields $\mathrm{d}\widetilde{x}_\tau/\mathrm{d}\tau = -\nabla\frac{\delta\mathcal{F}}{\delta\widetilde{\rho}_\tau}(\widetilde{x}_\tau)$ and hence the WGF law Equation (2).

B.4 General and canonical velocities

There may be multiple velocity fields that produce the same instantaneous change in density. The canonical velocity v_t has the minimum kinetic energy among such fields (Otto, 2001; Ambrosio et al., 2008).

However, the rotational part of the velocity field can still lead to later changes in the density that the canonical velocity would not be able to capture. For example, consider an elliptical (anisotropic two-dimensional) Gaussian distribution rotating around its center under a harmonic force at the rotation frequency (that would keep the particles together). Its general velocity would simply rotate the distribution. Its canonical velocity would produce the same initial change in density, but canonical velocity would stretch the ellipse along one diagonal and compress it along the other over longer time.

Relation to WLM. Guan et al. (2026a, Theorem 2.2 and Appendix A) derive the dynamical law for canonical velocities ($v_0 = \nabla\phi_0$ in our notation). Their simulation-based algorithm can use the initial velocities directly, without imposing an additional gradient condition (Guan et al., 2026a, Algorithms 1 and 2). Therefore, if the initial velocity is canonical, WLM would preserve it for later time steps. However, if the velocity is general, the overall trajectory would not be canonical anymore. The Clebsch derivation covers smooth general initial velocities, so it allows for the same particle law but with the velocities that can include circulation.

C Derivation of Double-Stitch

C.1 Velocity and acceleration

Consider time-independent weights and a fixed kernel ϕ. We differentiate the KDE path Equation (9) and use $\partial_t\phi(x - x_{t,k}) = -\dot{x}_{t,k}\cdot\nabla\phi(x - x_{t,k})$

$$\partial_t\hat{\rho}_t(x) = -\sum_{k=1}^{P} w_k\,\dot{x}_{t,k}\cdot\nabla\phi(x - x_{t,k}) = -\nabla\cdot\Big(\sum_{k=1}^{P} w_k\,\phi(x - x_{t,k})\,\dot{x}_{t,k}\Big) = -\nabla\cdot\big(\hat{\rho}_t(x)\,u_t(x)\big),$$

so the KDE path satisfies the continuity equation with the velocity field

$$u_t(x) = \sum_{k=1}^{P} r_{t,k}(x)\,\dot{x}_{t,k}, \qquad r_{t,k}(x) = \frac{w_k\,\phi(x - x_{t,k})}{\hat{\rho}_t(x)},$$

which is Equation (10) (the responsibility-weighted mean of the center velocities). Then its material derivative is

$$\partial_t u_t(x) + (u_t\cdot\nabla)u_t(x) = \sum_{k=1}^{P} r_{t,k}(x)\,\ddot{x}_{t,k} - \frac{1}{\hat{\rho}_t(x)}\nabla\cdot\big(\hat{\rho}_t(x)\,\Sigma_t(x)\big),$$

$$\Sigma_t(x) = \sum_{k=1}^{P} r_{t,k}(x)\,\big(\dot{x}_{t,k} - u_t(x)\big)\big(\dot{x}_{t,k} - u_t(x)\big)^\top,$$

where Σ_t is the dispersion of the center velocities about their local mean.

C.2 Energy variation and Wasserstein gradient

Consider a symmetric interaction kernel $W_\theta(z) = W_\theta(-z)$ (e.g., a radial interaction). We compute the first variation of the energy Equation (11):

$$\frac{\delta \mathcal{U}_\theta}{\delta \rho}(x) = c_V V_\theta(x) + c_H\big(\log \rho(x) + 1\big) + c_W (W_\theta * \rho)(x),$$

For an asymmetric kernel, this energy depends only on its even part, so we replace W_θ in the first variation with $\frac{1}{2}(W_\theta(z) + W_\theta(-z))$. For the symmetric case, the analytic Wasserstein gradient is

$$\nabla_x \frac{\delta \mathcal{U}_\theta}{\delta \rho}(x) = c_V \nabla V_\theta(x) + c_H \nabla \log \rho(x) + c_W \nabla (W_\theta * \rho)(x). \tag{17}$$

We then evaluate the entropy term on the KDE path: $\nabla \log \hat{\rho}_t$. Approximating the interaction with the empirical mass measure $\rho_t^P = \sum_k w_k \delta_{x_{t,k}}$ we get the force used in the residual

$$\nabla \frac{\delta \mathcal{U}_\theta}{\delta \rho}\big[\hat{\rho}_t\big](x) \approx c_V \nabla V_\theta(x) + c_H \nabla \log \hat{\rho}_t(x) + c_W \sum_{l=1}^{P} w_l \nabla W_\theta(x - x_{t,l}). \tag{18}$$

C.3 Lagrangian residual and empirical approximation

We define the Lagrangian residual as the integral over time and space of the law deviation from zero (see Equation (6)):

$$\mathcal{R}[\rho, u] = \int_0^T \int \left\| \partial_t u_t + (u_t \cdot \nabla) u_t + \gamma u_t + \nabla \frac{\delta \mathcal{U}}{\delta \rho_t} \right\|^2 \rho_t(x)\, \mathrm{d}x\, \mathrm{d}t. \tag{19}$$

Under the assumptions in Section B.2, a trajectory satisfying the equation of motion has zero residual. Conversely, a zero residual means that the nonnegative integrand vanishes almost everywhere. Then, using smoothness and strict positivity of ρ_t we obtain satisfaction of the equation of motion at every point.

Finally, continuity ensures that particles starting from ρ_0 have density ρ_t at time t. Since $\mathrm{d}u_t(x_t)/\mathrm{d}t = \mathrm{d}^2 x_t/\mathrm{d}t^2$, we can write the same residual along particle trajectories as

$$\mathcal{R}[\rho, u] = \mathbb{E}_{x_0 \sim \rho_0}\left[\int_0^T \left\| \frac{\mathrm{d}^2 x_t}{\mathrm{d}t^2} + \gamma \frac{\mathrm{d}x_t}{\mathrm{d}t} + \nabla \frac{\delta \mathcal{U}}{\delta \rho_t}(x_t) \right\|^2 \mathrm{d}t \right],$$

which gives Equation (7).

We then approximate the integral using the particle representation Equation (7). Specifically, each path has velocity $\dot{x}_{t,k}$ and acceleration $\ddot{x}_{t,k}$ so its mass measure satisfies

$$\int g \, \mathrm{d}\rho_t^P = \sum_{k=1}^{P} w_k \, g(x_{t,k}).$$

We can then evaluate density-dependent forces using the KDE at a finite bandwidth and approximate the interaction using the centers as in Equation (18). The residual is then approximated by

$$\mathcal{R} \approx \int_0^T \sum_{k=1}^{P} w_k \left\| \ddot{x}_{t,k} + \gamma\, \dot{x}_{t,k} + \nabla \frac{\delta \mathcal{U}}{\delta \rho}\big[\hat{\rho}_t\big](x_{t,k}) \right\|^2 \mathrm{d}t.$$

We note that this is a particle approximation to the continuum construction, not an exact evaluation of Equation (19) for the KDE mean velocity.

C.4 Time discretization

We perform finite-difference parameterization on the time grid $t_0 < \cdots < t_M$ with steps $h_j = t_{j+1} - t_j$. The center velocity and acceleration can be replaced by the three-point central differences

at the interior nodes $j = 1, \dots, M-1$,

$$\Delta_h x_{t_j,k} = \frac{x_{t_{j+1},k} - x_{t_{j-1},k}}{h_{j-1} + h_j},$$
$$\Delta_h^2 x_{t_j,k} = \frac{2}{h_{j-1} + h_j} \left(\frac{x_{t_{j+1},k} - x_{t_j,k}}{h_j} - \frac{x_{t_j,k} - x_{t_{j-1},k}}{h_{j-1}} \right),$$

which reduce on a uniform grid $h_j \equiv h$ to

$$\Delta_h x_{t_j,k} = \frac{x_{t_{j+1},k} - x_{t_{j-1},k}}{2h}, \qquad \Delta_h^2 x_{t_j,k} = \frac{x_{t_{j+1},k} - 2x_{t_j,k} + x_{t_{j-1},k}}{h^2}.$$

The time integral is evaluated by the trapezoidal rule at all nodes, setting $h_{-1} = h_M = 0$,

$$\mathcal{R} \approx \sum_{j=0}^{M} \tfrac{1}{2}(h_{j-1} + h_j) \sum_{k=1}^{P} w_k \left\| \Delta_h^2 x_{t_j,k} + \gamma\, \Delta_h x_{t_j,k} + \nabla \frac{\delta \mathcal{U}}{\delta \rho}\big[\hat{\rho}_{t_j}\big](x_{t_j,k}) \right\|^2,$$

which is the loss Equation (12). The two endpoint nodes use a one-sided stencil of the same order.

C.5 Pseudocode

We present the pseudocode for WLM in Algorithm 1 and Double-Stitch in Algorithm 2. The former is adapted from the original paper to our notation (Guan et al., 2026a).

Algorithm 1 WLM

Input

- **Data:** snapshots $\{X_i\}_{t_i \in \mathcal{T}_{\text{obs}}}$, $\mathcal{T}_{\text{obs}} = \{t_0 < \cdots < t_K\}$, represented by samples $\{x_i^{(n)}\}_{n=1}^N$; initial velocities $\{v_0^{(n)}\}_{n=1}^N$ at t_0
- **Model hyperparameters:** divergence $\mathcal{D}$, friction $\gamma \geq 0$, and solver step size Δt
- **Training budget:** number of iterations n_{iters}

Initialization

- **Energy:** network parameters of $\mathcal{U}_\theta : \mathbb{R}^{P \times d} \to \mathbb{R}$
- **Friction:** γ fixed, or initialized when learned

Training

for iter $= 1, \dots, n_{\text{iters}}$ **do**

Rollout

$H \sim \text{Unif}\{1, \dots, K\}$ ▷ random horizon

$\tilde{x}_0^{(n)} \leftarrow x_0^{(n)}, \quad \tilde{v}_0^{(n)} \leftarrow v_0^{(n)}$ ▷ P particles from the observed initial cloud

for $j = 0, \dots, \lceil (t_H - t_0)/\Delta t \rceil - 1$ **do**

$a_j^{(n)} \leftarrow -\nabla_{\tilde{x}^{(n)}} \mathcal{U}_\theta\big(\tilde{x}_j^{(1)}, \dots, \tilde{x}_j^{(P)}\big)$ ▷ for all n

$\tilde{x}_{j+1}^{(n)}, \tilde{v}_{j+1}^{(n)} \leftarrow \text{Leapfrog}\big(\tilde{x}_j^{(n)}, \tilde{v}_j^{(n)}, a_j^{(n)}, \gamma, \Delta t\big)$ ▷ damped leapfrog of Equation (4)

end for

$\hat{X}_i \leftarrow \{\tilde{x}_{j_i}^{(n)}\}_{n=1}^P$ at the steps j_i reaching t_i, $i = 1, \dots, H$

Objective and update

$\mathcal{L}_{\text{WLM}}(\theta, \gamma) \leftarrow \dfrac{1}{H} \displaystyle\sum_{i=1}^{H} \mathcal{D}(\hat{X}_i, X_i)$ ▷ backpropagated through the rollout

$\text{OptStep}_{\mathcal{L}_{\text{WLM}}}\big(\mathcal{U}_\theta,\ \gamma \text{ if learned}\big)$

end for

Output

- **Learned energy and friction:** $\mathcal{U}_\theta, \gamma$
- **Learned law:** Equation (4), simulated by the same integrator from any observed cloud and velocity

Algorithm 2 DOUBLE-STITCH

Input

- **Data:** snapshots $\{X_i\}_{t_i \in \mathcal{T}_{\text{obs}}}$, represented by samples; optionally, paired initial positions and velocities $\{(x_0^{(n)}, v_0^{(n)})\}_{n=1}^{N_0}$
- **Model hyperparameters:** number of learned particles P, time grid $t_0 < \cdots < t_M$ with spacing h, initial KDE bandwidth σ_0, initial friction γ_0, residual-loss weight λ, and initial-velocity weight λ_{v_0} (0 without initial velocities)
- **Training budget:** number of epochs n_{epochs}

Initialization

- **Particle positions:** $\{x^\theta_{t_j,k}\}_{j,k} \leftarrow \text{InitPath}(\{X_i\}_{t_i \in \mathcal{T}_{\text{obs}}}, P, M)$
- **KDE:** bandwidth $\sigma^\theta \leftarrow \sigma_0$ and uniform mixture weights $w^\theta_{1:P} \leftarrow 1/P$
- **Potential:** network parameters V_θ, W_θ and functional coefficients c_V, c_H, c_W
- **Friction:** $\gamma \leftarrow \gamma_0$

Training

for epoch $= 1, \ldots, n_{\text{epochs}}$ **do**

Density and force

$$\hat{\rho}^\theta_{t_j}(x) \leftarrow \sum_{k=1}^{P} w^\theta_k \phi_{\sigma^\theta}(x - x^\theta_{t_j,k}) \qquad \triangleright \text{ Equation (9)}$$

$$g^\theta_{j,k} \leftarrow c_V \nabla V_\theta(x^\theta_{t_j,k}) + c_H \nabla \log \hat{\rho}^\theta_{t_j}(x^\theta_{t_j,k}) + c_W \sum_{l=1}^{P} w^\theta_l \nabla W_\theta(x^\theta_{t_j,k} - x^\theta_{t_j,l}) \qquad \triangleright \text{ Equation (18)}$$

Acceleration residual

$$\Delta_h x^\theta_{t_j,k} \leftarrow \frac{x^\theta_{t_{j+1},k} - x^\theta_{t_{j-1},k}}{2h}, \qquad \Delta_h^2 x^\theta_{t_j,k} \leftarrow \frac{x^\theta_{t_{j+1},k} - 2x^\theta_{t_j,k} + x^\theta_{t_{j-1},k}}{h^2} \qquad \triangleright\, 0 < j < M$$

$\Delta_h x^\theta_{t_j,k}$, $\Delta_h^2 x^\theta_{t_j,k}$ at $j = 0, M$: one-sided second-order stencils

$$\mathcal{L}_{\text{res}}(\theta, \gamma) \leftarrow h \sum_{j=0}^{M} \sum_{k=1}^{P} w^\theta_k \left\| \Delta_h^2 x^\theta_{t_j,k} + \gamma\, \Delta_h x^\theta_{t_j,k} + g^\theta_{j,k} \right\|_2^2 \qquad \triangleright \text{ endpoint terms halved}$$

Initial velocity (optional)

$$\mathcal{L}_{v_0}(\theta) \leftarrow \frac{1}{N_0} \sum_{n=1}^{N_0} \left\| \sum_{k=1}^{P} r^\theta_{t_0,k}(x_0^{(n)}) \frac{x^\theta_{t_1,k} - x^\theta_{t_0,k}}{h} - v_0^{(n)} \right\|_2^2 \qquad \triangleright \text{ Equation (13)}$$

Objective and update

$$\mathcal{L}_{\text{data}}(\theta) \leftarrow \sum_{t_i \in \mathcal{T}_{\text{obs}}} \mathcal{D}(\hat{\rho}^\theta_{t_i}, X_i)$$

$$\mathcal{L}(\theta, \gamma) \leftarrow \mathcal{L}_{\text{data}}(\theta) + \lambda \mathcal{L}_{\text{res}}(\theta, \gamma) + \lambda_{v_0} \mathcal{L}_{v_0}(\theta) \qquad \triangleright \text{ Equation (8)}$$

$$\text{OptStep}_{\mathcal{L}}\big(\{x^\theta_{t_j,k}\}_{j,k},\ \sigma^\theta,\ w^\theta_{1:P},\ V_\theta,\ W_\theta,\ c_V,\ c_H,\ c_W,\ \gamma\big)$$

end for

Output

- **Learned particle positions:** $\{x^\theta_{t_j,k}\}_{j,k}$
- **Learned density:** induced curve $\hat{\rho}^\theta = (\hat{\rho}^\theta_t)_{t \in [0,T]}$; Equation (9)
- **Learned velocity:** induced field $u^\theta = (u^\theta_t)_{t \in [0,T]}$; Equation (10)
- **Learned potentials and friction:** energy $\mathcal{U}_\theta$ induced by external V_θ and interaction W_θ; Equation (11); friction γ

C.6 COMPUTATIONAL COMPLEXITY

Proposition 1 (Training cost). Consider the training cost of one Double-Stitch and WLM training iteration on the same task. Suppose that the number of time steps M, number of particles P, and the data loss evaluation time are the same. Assume that $C(P)$ is the time it takes to make a force evaluation (P particles, including backpropagation), and suppose the hardware can run B force evaluations in parallel. Then, apart from the data loss, one Double-Stitch iteration takes $O(\lceil M/B \rceil\, C(P))$ time while WLM will take $O(M\, C(P))$.

Proof. Since we assume that the parameters are identical, both methods will arrive at a computational grid of the same size. Therefore if we count force evaluations, both methods will require $O(M)$ evaluations.

However WLM has to unroll a simulated trajectory while Double-Stitch stores it as the learned particles. The unroll has to happen sequentially while Double-Stitch is able to compute the residual in parallel among many nodes (see Equation (12)).

WLM evaluates one force for each of M steps and the steps have to be sequential (see Algorithm 1), therefore the complexity is $O(M\,C(P))$.

Double-Stitch on the other hand does not require a node to use another's output (it only reads the positions of its neighbors), meaning we can compute forces for $M+1$ terms in batches of B terms and add up their gradients, since the loss is their sum, making its complexity $\lceil (M+1)/B \rceil\, C(P)$ time.

For $B = 1$ both methods take $O(M\,C(P))$ time, and if both have the same $C(P)$, Double-Stitch is up to $\min(B, M)$ times faster. Note that this compares a single update, not the number of updates needed to fit the data. In reality Double-Stitch also uses KL, which is generally cheaper to compute than Sinkhorn. □

D Experimental details

D.1 Data

Synthetic. The demonstration in Figures 1 and 11 uses three flows: conservative (Hamiltonian), damped (Lagrangian), and gradient (WGF).

The conservative flow is a harmonic process:

$$\ddot{x}_t = -\gamma \dot{x}_t - \omega_0^2 x_t, \quad \omega_0 = 2\pi/2.7, \quad \gamma = 0.$$

The damped flow is the same process but with $\gamma = 1$.

The gradient flow is an Ornstein–Uhlenbeck process:

$$\mathrm{d}x_t = -x_t\,\mathrm{d}t + \sqrt{2}\,\mathrm{d}B_t, \quad \rho_0 = \frac{1}{2}\mathcal{N}(-2.5, 0.30^2) + \frac{1}{2}\mathcal{N}(2.5, 0.65^2), \quad \rho_{t\to\infty} = \mathcal{N}(0,1),$$

where B_t is a standard Brownian motion. The solutions for all three flows are analytic: the particle dynamics are linear in position (plus velocity for the harmonic flow) so each Gaussian component of the initial distribution stays Gaussian (Särkkä & Solin, 2019, Ch. 6).

For training, we sample 100 particles at each of five equally spaced times on $[0.15, 1.5]$. We use even snapshots (0, 2, 4) for training and odd ones (1, 3) for testing (similar to L2O on the EB dataset).

Single-cell data. The embryoid-body (EB) dataset follows human embryonic stem-cell differentiation (Moon et al., 2019). We use the processed data from TrajectoryNet (Tong et al., 2020), also used by WLM and Stitching. It contains 16819 cells in five time bins, retains the first five principal components and standardizes each dimension using its mean and std across all cells. LPO uses a fixed random 30% of cells for validation (11773 for training and 5046 for testing) with the W_1 metric averaged over the four noninitial times. L1O withholds each intermediate snapshot and fits a separate model on each with the W_1 metric averaged over the model performances. L2O trains on snapshots $0, 2, 4$ and evaluates W_2 at snapshots $1, 3$ (we also report the average).

We visualize the EB data in Figure 3 using a common two-dimensional UMAP (McInnes et al., 2018).

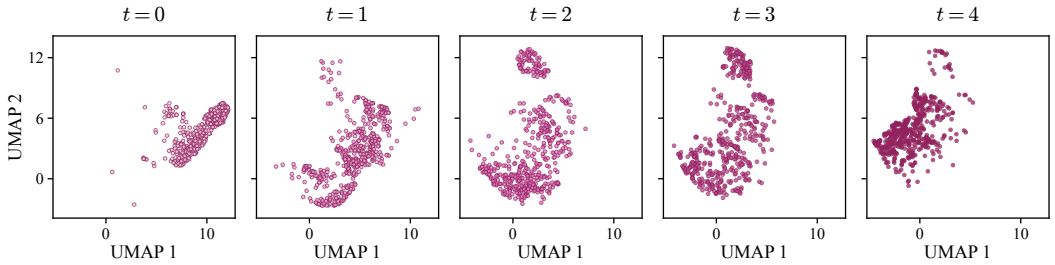

Figure 3: **EB snapshots over time.** Shared UMAP.

Ocean vortex data. HYCOM ocean vortex datasets from the Gulf of Mexico (Guan et al., 2026a) contain 9 snapshots of 111 particles for the small vortex and 11 snapshots of 400 particles for the large vortex. For both, we use WLM's preprocessing. Small vortex offers an interpolation task: even snapshots are used for training and odd ones are held out for testing. The large-vortex dataset offers an extrapolation (or a forecasting task): snapshots 0 to 9 are used for training, and snapshot 10 used for testing. Snapshot times are one time unit apart in all cases, snapshots are considered unpaired.

We visualize the two ocean vortices in Figure 4.

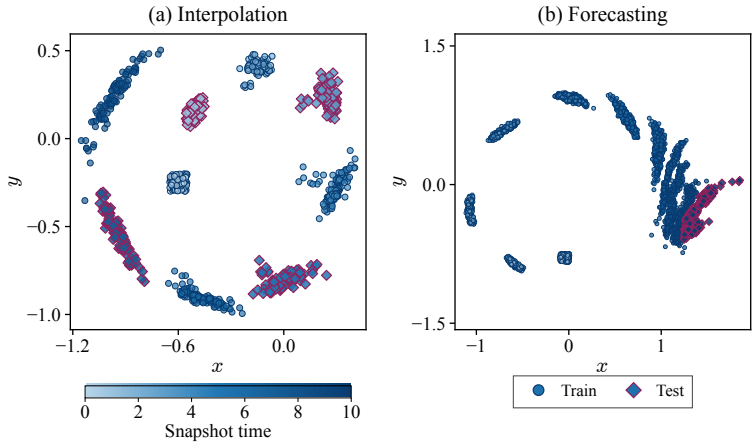

Figure 4: **Ocean vortex snapshots.** Color indicates time. Circular points are used for training, diamonds for testing.

D.2 Training and evaluation

We use settings similar to Stitching and WLM and list the main ones below; the configuration files in our code give the complete settings.

Hardware and uncertainty. We run all timing experiments on NVIDIA RTX 4090 GPUs. We report accuracy as mean ± std over five seeds for EB and three seeds for oceans. Ablations, sensitivity studies and training time comparisons use three seeds. For published results, we retain the reported means and standard deviations.

Double-Stitch. On EB and oceans, we parametrize the potential by an MLP with two hidden layers of 64 SiLU units. We initialize the trajectories by OT-path, the learnable bandwidth at the coarsest scale, and the learnable friction at $\gamma_0 = 1$. On EB, we largely follow the Stitching settings of Heinonen et al. (2026, Appendix G): 50 time nodes, 10000 updates, learned mixture weights and batches of 256, with only the cell coordinates as observations. We use $P = 1000$ centers. On oceans, we use $P = 100$ centers and 49 time nodes for the small vortex, and $P = 400$ centers and

51 time nodes for the large vortex. The latter grid extends to the held-out time. Both ocean tasks use full batches and fixed uniform mixture weights.

We train with Adam for 10000 updates on EB and 40000 on oceans. The learning rate starts at 0.005 and follows cosine decay; the potential network has decoupled weight decay of 1.0. We set the residual-loss weight to $\lambda = 1$. The coefficients c_V and c_H are learned from initial values of 1 and 0.03, respectively, while $c_W = 0$. We treat the mixture weights as constants inside the residual, so this loss does not update them. On oceans, we also use the initial-velocity constraint in Equation (13). Its weight λ_{v_0} is the inverse empirical mean squared initial speed. We evaluate the final checkpoint and use 50 damped-Verlet substeps per snapshot interval when integrating the learned dynamics.

Stitching. We report two settings for our Stitching reruns. On EB, *original settings* follow Appendix G of Heinonen et al. (2026): $P = 100$ centers, 50 time nodes, a potential with two hidden layers of 64 SiLU units, 10000 Adam updates and a learnable bandwidth initialized by Silverman's rule. We scale time to $[0, 1]$, use a $70/30$ cell split, learn the mixture weights and train with batches of 256. We evaluate the learned density directly at the requested time. Our trajectory initialization differs from Appendix G: we use endpoint OT, while the original method couples consecutive snapshots. On oceans, *original settings* use our first-order Stitching configuration with a budget of 40000 updates and early stopping (patience 2000, minimum training-loss improvement 10^{-4}). We evaluate the checkpoint with the lowest training loss.

For *main settings*, we use the Double-Stitch setup with the velocity residual in Equation (2), without the initial-velocity constraint or early stopping. On EB, we use $P = 400$ centers. We report static and time-dependent potentials separately.

WLM. We use the released architectures and solver, with 10000 training updates on EB and 50000 on oceans. On EB, we use the released RNA velocities and a fixed Sinkhorn blur of 1.25. On oceans, we distinguish two settings by how the blur is estimated. The *main settings*, used in Table 3, estimate it from the training snapshots only. The *original settings* estimate it from all snapshots, as in the released code. Both settings use the same position clouds and training/test split, fit only the training snapshots and report the final checkpoint over all three seeds. WLM's attention potential takes time as an input for ocean interpolation, but not for EB or ocean forecasting. We distinguish published results from our reruns in Sections E.2 and E.3.

Initialization sensitivity. We compare three trajectory initializations. Endpoint OT matches the first and last snapshots by optimal transport and linearly interpolates between them. OT-path instead chains optimal-transport matchings between consecutive snapshots. McCann-path approximates each snapshot by a Gaussian with the same moments and follows the Bures–Wasserstein geodesics between them (McCann, 1997).

We also compare two bandwidth initializations. Silverman uses Silverman's rule (Silverman, 1986), $1.06\,\hat{\sigma}_i N^{-1/(d+4)}$ in each dimension i, computed on the first training snapshot. Coarsest uses $\sqrt{\lambda_{\max}}$, the standard deviation along the principal axis of the pooled training snapshots, the coarsest scale at which a Gaussian mixture still resolves structure (Rose et al., 1990). We first vary the bandwidth initialization with OT-path fixed, then vary the trajectory initialization with Silverman bandwidth fixed. For each seed, we keep the network initialization the same across these comparisons.

Prediction. We use the three prediction modes described in Section 4. Direct prediction evaluates the learned KDE $\hat{\rho}_t^\theta$. Pushforward integrates the learned dynamics from the latest observed snapshot before the test time. Rollout integrates from the initial snapshot without restarting at later observations. For second-order dynamics, pushforward also uses the observed velocities at its starting snapshot, while rollout uses only the global initial velocity. On EB, we use direct prediction for Stitching and Double-Stitch, and pushforward for WLM. On oceans, we report each protocol separately.

Training time. Double-Stitch's training time in Table 4 includes initialization, compilation and training callbacks. WLM's covers its native training loop, including monitoring but not model setup. Neither measurement includes the final evaluation. Double-Stitch uses seeds 1–3 for EB LPO/L1O and 0–2 for the other tasks. WLM uses original settings, with seeds 2–4 for EB LPO/L1O and 0–2 for EB L2O and both ocean tasks. L1O sums the training times of the three holdout fits within each seed.

For the particle-count comparison in Table 12, we measure only optimizer execution time, synchronized on the GPU. We use seeds 0–2 for all five tasks.

Synthetic demonstration. For the six fits in Figure 11, we use seed 0, $P = 50$ centers, 50 time nodes and 5000 Adam updates. The learning rate starts at 0.005 and follows cosine decay, without weight decay. Both methods use OT-path initialization, a learnable bandwidth initialized by Silverman's rule, fixed uniform mixture weights, the forward-KL data loss and a static potential with two hidden layers of 64 units. We learn c_H from 0.03 and fix $c_W = 0$. We set λ once at initialization so that the data and residual losses have equal gradient norms with respect to the center paths.

Stitching uses the velocity residual and Double-Stitch uses the Lagrangian residual. For Double-Stitch, we learn friction from $\gamma_0 = 10$ on the Ornstein–Uhlenbeck process and fix $\gamma = 0$ and 1 on the conservative and damped oscillators, respectively. We report the mean held-out W_1 between the fitted and analytic position densities. For visualization, we smooth both methods' fitted densities with $\sigma = 0.1$; the scores use the unsmoothed densities.

Synthetic cost experiment. For Figure 2, we use the harmonic oscillator and initial position distribution from Figure 1, with zero initial velocity. We sample 1000 independent positions at nine observation times on $[0, 1.35]$ and at eight held-out midpoints. Both methods use static potentials with $\gamma = c_H = c_W = 0$ and $c_V = 1$ fixed. We train for 5000 Adam updates over three seeds sharing the same data.

Double-Stitch uses 1000 centers, 41 time nodes and a potential with two hidden layers of 64 SiLU units. We use OT-path initialization, a learnable bandwidth initialized at the coarsest scale, fixed uniform mixture weights, and the forward-KL data loss with the Lagrangian residual weighted by $\lambda = 1$. WLM uses four attention layers with four heads, width 64 and feed-forward width 512. We train it with batches of 256 and a random-horizon debiased Sinkhorn objective, without moving averages. The learning rates are constant at 10^{-3} for Double-Stitch and 10^{-4} for WLM, without weight decay. Double-Stitch's time-grid spacing h and WLM's training integration step are both 0.03375.

Double-Stitch predicts directly from its KDE. WLM rolls out from the initial cloud with zero velocity and 50 substeps between evaluation times. We average training W_1 over the eight positive observation times and test W_1 over the eight held-out midpoints. At common update counts, we plot the mean and std against the mean synchronized execution time of the forward, backward and optimizer operations.

E Additional results

E.1 Identifiability

Snapshots alone generally do not determine the underlying dynamics. Theoretical analysis of this question exists for SDEs: the dynamics are identifiable from the snapshots for (a) linear SDEs with additive noise whose initial distribution does not have generalized rotational symmetries (Guan et al., 2024), and (b) gradient-flow SDEs with constant diffusivity observed away from equilibrium (Guan et al., 2026b). These results do not cover our setting: second-order (Lagrangian) dynamics with finitely many snapshots. We therefore provide examples that demonstrate the challenges of identifiability.

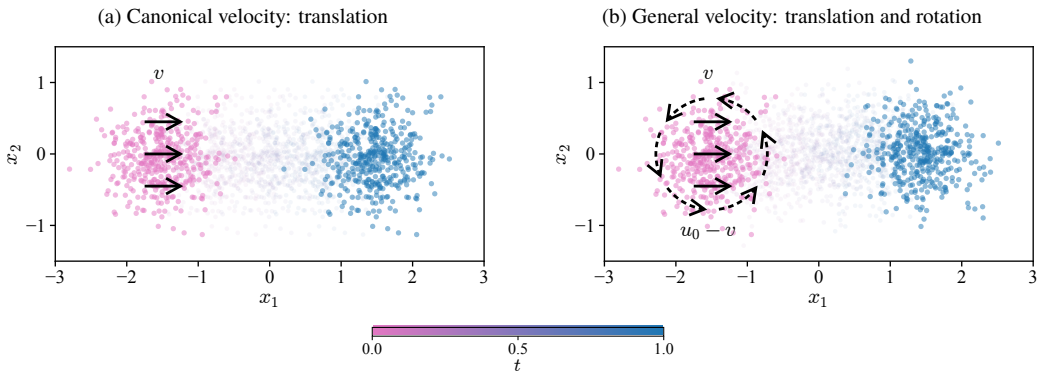

Figure 5: **Canonical and general velocities can produce the same snapshots.** Left: translation with the canonical velocity (solid arrows). Right: translation and rotation (dashed arcs) with a general velocity. Color indicates time.

Canonical and general velocities. Since we operate in Wasserstein spaces, our metric cannot differentiate between populations with the same density distribution but different velocity fields. For this reason, a canonical velocity is often considered. It is irrotational and has minimum kinetic energy among admissible fields (Ambrosio et al., 2008). However, if the initial velocity is not canonical, the dynamical law must be able to take this into account, which is exactly what our Clebsch-based approach does (see Section B.2).

In Figure 5 we show a Gaussian distribution that is transported under constant horizontal velocity from left to right ($\mathcal{N}((-1.5, 0), 0.4^2 I)$, 400 points). On the left its velocity is a pure translation ($u_0 = v_0 = (3, 0)$). On the right it additionally has a rotational component ($\omega = 3\pi/2$), meaning the particles experience an additional force. The density is the same at all times, however the dynamical law has to be written for u.

Different initial velocities. Even with a known potential and friction, a finite set of position snapshots does not necessarily determine the initial velocity. We illustrate this with trajectories of a single particle following Lagrangian dynamics in a circular potential $V(x) = \frac{1}{2}\omega_0^2\|x\|^2$ ($\omega_0 = 4\pi$, $\gamma = 0.2\omega_0$). We set the particle at $x_0 = (1, 0)$ and consider two different initial velocities: $u_0^{(1)} = (0, -\omega_0)$ and $u_0^{(2)} = (0, -\omega_0/2)$.

It is easy to show that both trajectories pass through the same points:

$$x(t_k) = (-1)^k \mathrm{e}^{-\gamma t_k/2} x_0, \qquad t_k = \frac{k\pi}{\omega}, \quad \omega = \sqrt{\omega_0^2 - \gamma^2/4}, \quad k = 0, \dots, 4.$$

Therefore, if the observed snapshots are selected precisely as the intersection points of these trajectories (see Figure 6), the trajectories will not be identifiable.

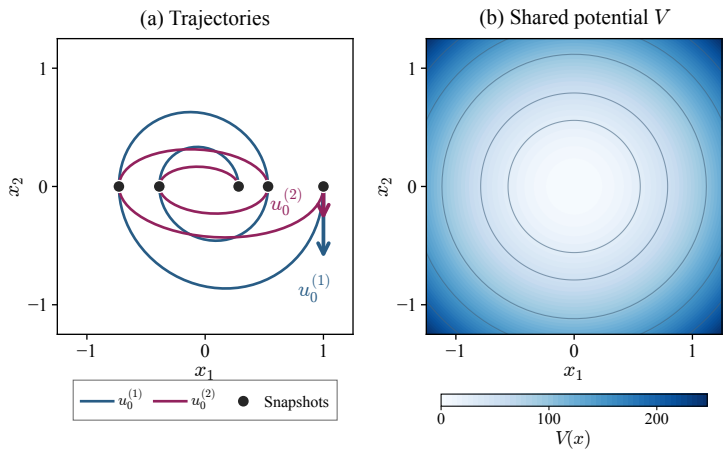

Figure 6: **Snapshots are not identifiable without the initial velocity.** Left: simulated trajectories with their individual initial velocities pass through the same points and are therefore not identifiable from them. Right: shared circular potential.

Different static potentials. The situation is similar in the opposite case: when the initial conditions are the same, multiple static potentials can produce similar trajectories.

We again consider a single particle following Lagrangian dynamics, but now with a single initial condition ($x_0 = (1, 0)$ and $u_0 = (-0.1, -1)$) and under the following potentials:

$$V_{\text{spiral}}(r, \theta) = K\big[1 - \cos\big(\theta - \beta(r - r_0)\big)\big] + U(r), \qquad V_{\text{circular}}(x) = \tfrac{1}{2}\lambda\|x\|^2,$$

where $U'(r) = -a_0\sqrt{1 + \beta^2 r^2}$. We tune the parameters $K = 0.3$, $\beta = 15.47$, $r_0 = 0.96170$, $a_0 = -0.07251$ and $\lambda = 1.01$ using least squares to match the two trajectories.

From Figure 7 we see that we are able to find different potentials that again produce nearly intersecting trajectories. Similar to the previous case, if the observed snapshots are at the intersection points, then the exact potential is non-identifiable.

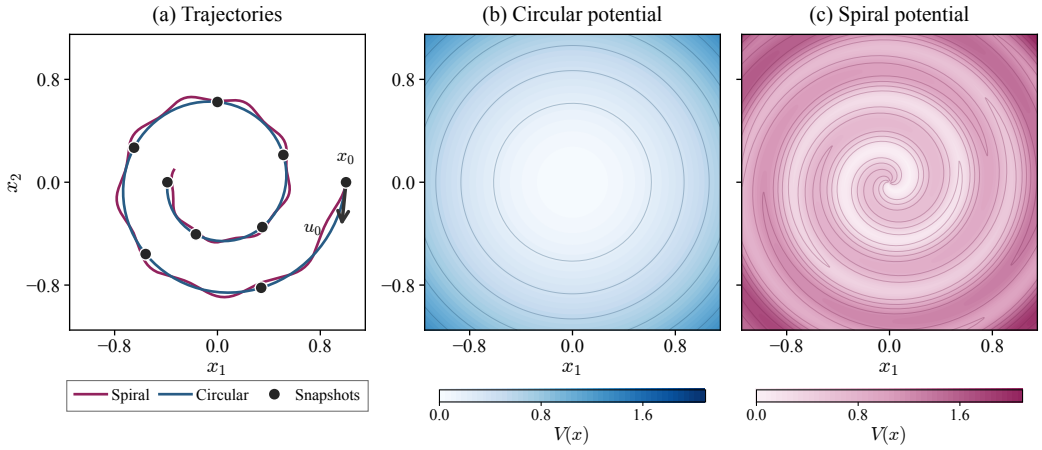

Figure 7: **Potentials are not identifiable from only snapshots.** Left: simulated trajectories with their shared initial velocities pass through the same points and are therefore not identifiable from them. Right: the potentials that produced the corresponding trajectories.

E.2 Single-cell dynamics

Tables 6 to 8 expand the three columns of Table 2 to per-snapshot performance. Each table lists the previously reported results as well as our reruns. We rerun WLM and Stitching with original settings as well and compare them to the settings we used in Section D.2.

Table 6: **Per-snapshot performance on EB LPO** (W_1, mean ± std, 5 seeds, lower is better).

Method	Source / settings	t_1	t_2	t_3	t_4	Mean
Neural SDE	Li et al. 2020	0.690	0.910	0.850	0.810	0.820
TrajectoryNet	Tong et al. 2020	0.730	1.060	0.900	1.010	0.930
SB-FBSDE	Chen et al. 2022	0.560	0.800	1.000	1.000	0.840
NLSB	Koshizuka & Sato 2023	0.680	0.840	0.810	0.790	0.780
OT-CFM	Tong et al. 2024	0.780	0.760	0.770	0.750	0.770
WLF-OT	Neklyudov et al. 2024	0.650	0.780	0.760	0.750	0.740
WLF-SB	Neklyudov et al. 2024	0.630	0.790	0.770	0.740	0.730
JKOnet	Bunne et al. 2022	1.530	1.270	1.130	1.410	1.340
JKOnet*, $V(x)$	Terpin et al. 2024	0.990	1.110	1.060	1.300	1.120
JKOnet*, $V(x,t)$	Terpin et al. 2024	0.690	0.770	0.690	0.780	0.730
iJKOnet, $V(x)$	Persiianov et al. 2025	0.920	1.110	0.950	1.210	1.050
iJKOnet, $V(x,t)$	Persiianov et al. 2025	0.510	0.580	0.570	0.640	0.580
Stitching, $V(x)$	Heinonen et al. 2026	0.460 ± 0.010	0.600 ± 0.010	0.600 ± 0.010	0.650 ± 0.010	0.580 ± 0.010
Stitching, $V(x,t)$	Heinonen et al. 2026	0.440 ± 0.010	0.560 ± 0.010	0.570 ± 0.010	0.600 ± 0.020	0.540 ± 0.010
Stitching, $V(x)$	Main settings	0.509 ± 0.003	0.699 ± 0.009	0.681 ± 0.017	0.803 ± 0.028	0.673 ± 0.005
Stitching, $V(x)$	Original settings	0.507 ± 0.011	0.650 ± 0.017	0.633 ± 0.005	0.682 ± 0.010	0.618 ± 0.007
Stitching, $V(x,t)$	Original settings	0.490 ± 0.012	0.601 ± 0.016	0.595 ± 0.013	0.639 ± 0.008	0.581 ± 0.010
WLM	Main settings	0.660 ± 0.018	0.663 ± 0.016	0.617 ± 0.028	0.789 ± 0.044	0.682 ± 0.009
Double-Stitch, $V(x)$	Main settings	0.546 ± 0.005	0.615 ± 0.011	0.676 ± 0.016	0.634 ± 0.008	0.618 ± 0.001
Double-Stitch, $V(x,t)$	Main settings	0.484 ± 0.005	0.572 ± 0.014	0.582 ± 0.007	0.582 ± 0.004	0.555 ± 0.005

Table 7: **Per-snapshot performance on EB L1O** (W_1, mean ± std, 5 seeds, lower is better).

Method	Source / settings	t_1	t_2	t_3	Mean
TrajectoryNet	Kapuśniak et al. 2024	—	—	—	0.848
NLSB	Kapuśniak et al. 2024	—	—	—	0.970
OT-CFM	Kapuśniak et al. 2024	—	—	—	0.790 ± 0.068
WLF-OT	Kapuśniak et al. 2024	—	—	—	0.742 ± 0.012
WLF-SB	Kapuśniak et al. 2024	—	—	—	0.746 ± 0.016
Reg. CNF	Kapuśniak et al. 2024	—	—	—	0.825 ± 0.429
DSBM	Kapuśniak et al. 2024	—	—	—	1.775 ± 0.429
DSB	Kapuśniak et al. 2024	—	—	—	0.862 ± 0.023
SF2M-Sink	Kapuśniak et al. 2024	—	—	—	1.198 ± 0.342
SF2M-Geo	Kapuśniak et al. 2024	—	—	—	0.879 ± 0.148
SF2M-Exact	Kapuśniak et al. 2024	—	—	—	0.793 ± 0.066
I-CFM	Kapuśniak et al. 2024	—	—	—	0.872 ± 0.087
SB-CFM	Kapuśniak et al. 2024	—	—	—	1.221 ± 0.380
WLF-UOT	Kapuśniak et al. 2024	—	—	—	0.738 ± 0.014
I-MFM	Kapuśniak et al. 2024	—	—	—	0.822 ± 0.042
OT-MFM	Kapuśniak et al. 2024	—	—	—	0.713 ± 0.039
AM	Neklyudov et al. 2023	—	—	—	0.944 ± 0.003
UAM	Neklyudov et al. 2023	—	—	—	0.924 ± 0.004
sAM	Neklyudov et al. 2023	—	—	—	0.939 ± 0.003
DICE	Blickhan et al. 2025	—	—	—	0.826 ± 0.002
WLM	Guan et al. 2026a	—	—	—	0.704 ± 0.021
Stitching, $V(x)$	Main settings	0.844 ± 0.046	0.793 ± 0.020	0.822 ± 0.012	0.819 ± 0.021
Stitching, $V(x)$	Original settings	0.738 ± 0.038	0.807 ± 0.019	0.844 ± 0.014	0.796 ± 0.014
Stitching, $V(x,t)$	Original settings	0.654 ± 0.023	0.726 ± 0.023	0.769 ± 0.007	0.717 ± 0.010
WLM	Main settings	0.713 ± 0.019	0.753 ± 0.060	0.735 ± 0.020	0.734 ± 0.023
Double-Stitch, $V(x)$	Main settings	0.692 ± 0.012	0.703 ± 0.011	0.805 ± 0.004	0.733 ± 0.004
Double-Stitch, $V(x,t)$	Main settings	0.695 ± 0.006	0.704 ± 0.012	0.754 ± 0.007	0.717 ± 0.008

Table 8: **Per-snapshot performance on EB L2O** (W_2, mean ± std, 5 seeds, lower is better).

Method	Source / settings	t_1	t_3	Mean
TrajectoryNet	Tong et al. 2020	2.030 ± 0.040	1.930 ± 0.080	1.980
Vanilla-SB	Vargas et al. 2021	1.490 ± 0.060	1.550 ± 0.030	1.520
MMSB	Shen et al. 2025	1.270 ± 0.030	1.570 ± 0.050	1.420
DMSB	Chen et al. 2023	1.130 ± 0.080	1.450 ± 0.160	1.290
JKOnet*, $V(x)$	Terpin et al. 2024	1.150 ± 0.030	2.530 ± 0.010	1.840
JKOnet*, $V(x,t)$	Terpin et al. 2024	4.410 ± 1.500	2.770 ± 0.200	3.590
iJKOnet, $V(x)$	Persiianov et al. 2025	1.080 ± 0.010	1.150 ± 0.000	1.120
iJKOnet, $V(x,t)$	Persiianov et al. 2025	0.980 ± 0.040	0.850 ± 0.020	0.920
Stitching, $V(x)$	Heinonen et al. 2026	0.850 ± 0.060	0.990 ± 0.030	0.920 ± 0.030
Stitching, $V(x,t)$	Heinonen et al. 2026	0.840 ± 0.050	0.920 ± 0.030	0.880 ± 0.040
Stitching, $V(x)$	Main settings	0.797 ± 0.022	0.893 ± 0.007	0.845 ± 0.014
Stitching, $V(x)$	Original settings	0.752 ± 0.037	0.891 ± 0.007	0.821 ± 0.019
Stitching, $V(x,t)$	Original settings	0.716 ± 0.038	0.856 ± 0.025	0.786 ± 0.008
WLM	Main settings	0.846 ± 0.045	0.800 ± 0.120	0.823 ± 0.041
Double-Stitch, $V(x)$	Main settings	0.755 ± 0.008	0.832 ± 0.004	0.794 ± 0.004
Double-Stitch, $V(x,t)$	Main settings	0.769 ± 0.004	0.829 ± 0.008	0.799 ± 0.003

In Figures 8 and 9 we show Double-Stitch predictions for L1O and L2O scenarios.

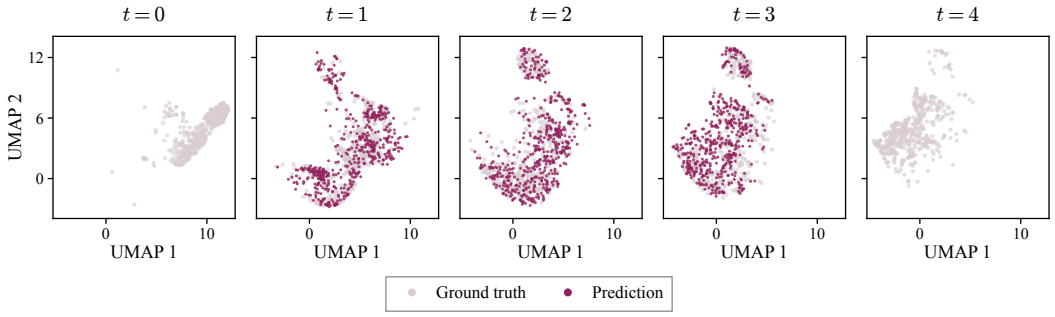

Figure 8: **Double-Stitch predictions on EB L1O** ($V(x)$, seed 0). Shared UMAP. Pale points are ground truth, dark points are predictions.

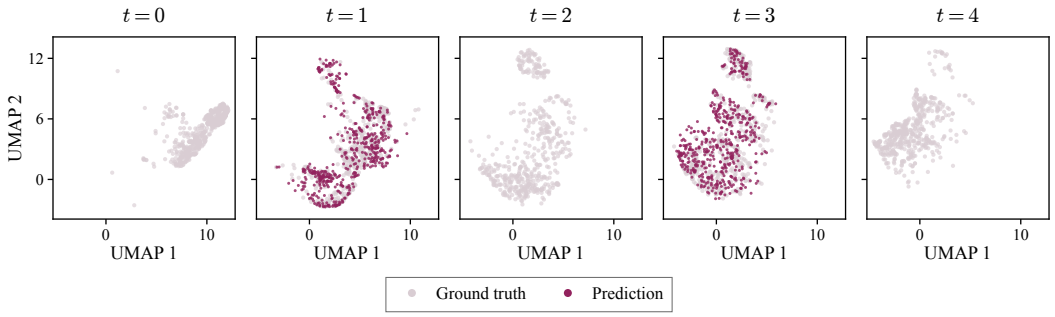

Figure 9: **Double-Stitch predictions on EB L2O** ($V(x)$, seed 0). Shared UMAP. Pale points are ground truth, dark points are predictions.

E.3 Ocean vortex dynamics

In Tables 9 and 10 we show per-snapshot results for interpolation (oceans small vortex) and the performance on three forecasting protocols for extrapolation (oceans large vortex).

Table 9: **Per-snapshot performance on the small ocean vortex** (W_1, mean ± std, 3 seeds, lower is better).

Method	Source / settings	t_1	t_3	t_5	t_7	Mean
AM	Neklyudov et al. 2023	0.358 ± 0.009	0.353 ± 0.020	0.447 ± 0.013	0.346 ± 0.016	0.376
UAM	Neklyudov et al. 2023	0.291 ± 0.008	0.378 ± 0.021	0.514 ± 0.010	0.349 ± 0.009	0.383
sAM	Neklyudov et al. 2023	0.358 ± 0.009	0.354 ± 0.010	0.451 ± 0.016	0.350 ± 0.009	0.378
DICE	Blickhan et al. 2025	0.369 ± 0.019	0.252 ± 0.020	0.240 ± 0.011	0.115 ± 0.009	0.244
Vanilla-SB	Vargas et al. 2021	0.112 ± 0.061	0.178 ± 0.073	0.262 ± 0.043	0.305 ± 0.032	0.214
OT-CFM	Tong et al. 2024	0.148 ± 0.004	0.227 ± 0.008	0.191 ± 0.012	0.250 ± 0.018	0.204
OT-MFM	Kapuśniak et al. 2024	0.107 ± 0.014	0.056 ± 0.014	0.052 ± 0.011	0.070 ± 0.021	0.071
Curly-FM	Petrović et al. 2025	0.019 ± 0.003	0.045 ± 0.005	0.027 ± 0.001	0.030 ± 0.006	0.030
SBIRR	Shen et al. 2025	0.073 ± 0.035	0.087 ± 0.033	0.062 ± 0.010	0.082 ± 0.022	0.076
Snap-MMD	Berlinghieri et al. 2025	0.051 ± 0.003	0.067 ± 0.002	0.131 ± 0.005	0.056 ± 0.001	0.076
WLM, published	Guan et al. 2026a	0.039 ± 0.002	0.028 ± 0.000	0.083 ± 0.002	0.050 ± 0.004	0.050
Stitching, $V(x)$	Main settings	1.538 ± 0.325	1.490 ± 0.331	1.431 ± 0.263	1.601 ± 0.177	1.515 ± 0.273
Stitching, $V(x,t)$	Main settings	0.176 ± 0.043	0.158 ± 0.038	0.144 ± 0.031	0.187 ± 0.028	0.166 ± 0.035
Stitching, $V(x)$	Original settings	8.180 ± 0.643	9.638 ± 1.418	0.475 ± 0.115	0.705 ± 0.046	4.750 ± 0.535
Stitching, $V(x,t)$	Original settings	0.080 ± 0.031	0.059 ± 0.019	0.099 ± 0.068	0.178 ± 0.026	0.104 ± 0.020
WLM	Main settings	0.056 ± 0.027	0.048 ± 0.012	0.086 ± 0.019	0.065 ± 0.039	0.064 ± 0.015
WLM	Original settings	0.037 ± 0.002	0.030 ± 0.005	0.087 ± 0.015	0.083 ± 0.064	0.059 ± 0.015
Double-Stitch, $V(x)$	Main settings	0.068 ± 0.023	0.033 ± 0.004	0.087 ± 0.010	0.070 ± 0.043	0.065 ± 0.004
Double-Stitch, $V(x,t)$	Main settings	0.047 ± 0.025	0.037 ± 0.003	0.132 ± 0.015	0.076 ± 0.007	0.073 ± 0.009

Table 10: **Per-snapshot performance on the large ocean vortex** (W_1, mean ± std, 3 seeds, lower is better). Dashes indicate unreported or unavailable cells.

Method	Source / settings	Rollout $0 \to 10$	Pushfwd. $9 \to 10$	Direct
SBIRR	Shen et al. 2025	1.062 ± 0.111	0.567 ± 0.014	—
Snap-MMD	Berlinghieri et al. 2025	0.896 ± 0.141	0.473 ± 0.013	—
WLM, published	Guan et al. 2026a	0.180 ± 0.038	0.065 ± 0.005	—
Stitching, $V(x)$	Main settings	14.370 ± 6.123	1.388 ± 0.119	0.295 ± 0.032
Stitching, $V(x,t)$	Main settings	0.224 ± 0.017	0.195 ± 0.022	0.212 ± 0.018
Stitching, $V(x)$	Original settings	0.149 ± 0.007	0.205 ± 0.006	0.206 ± 0.001
Stitching, $V(x,t)$	Original settings	0.211 ± 0.092	0.170 ± 0.004	0.175 ± 0.008
WLM	Main settings	0.194 ± 0.048	0.088 ± 0.023	—
WLM	Original settings	0.889 ± 1.126	0.238 ± 0.275	—
Double-Stitch, $V(x)$	Main settings	1.839 ± 0.539	0.134 ± 0.003	0.209 ± 0.013
Double-Stitch, $V(x,t)$	Main settings	0.222 ± 0.008	0.119 ± 0.019	0.194 ± 0.007

In Figure 10 we show Double-Stitch predictions for the small vortex interpolation.

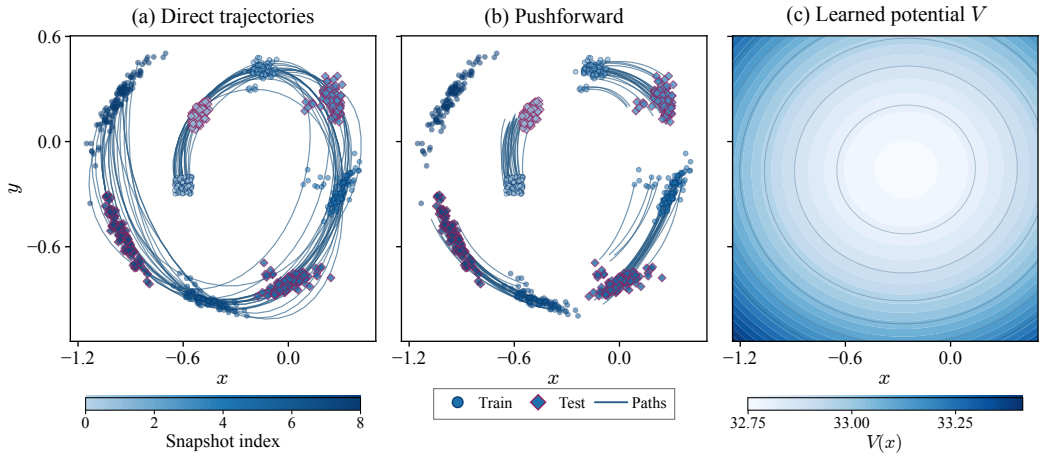

Figure 10: **Double-Stitch predictions on the small ocean vortex** ($V(x)$, seed 0). Left: learned trajectories (used for direct evaluation). Center: pushforward predictions (used for pushforward evaluation). Right: learned potential.

E.4 Synthetic demonstration

Figure 11 shows the ground truth and fitted densities for the three Gaussian mixture flows (see Section 4.1). Each method uses the same training snapshots and fitting budget (see the settings in Section D.2).

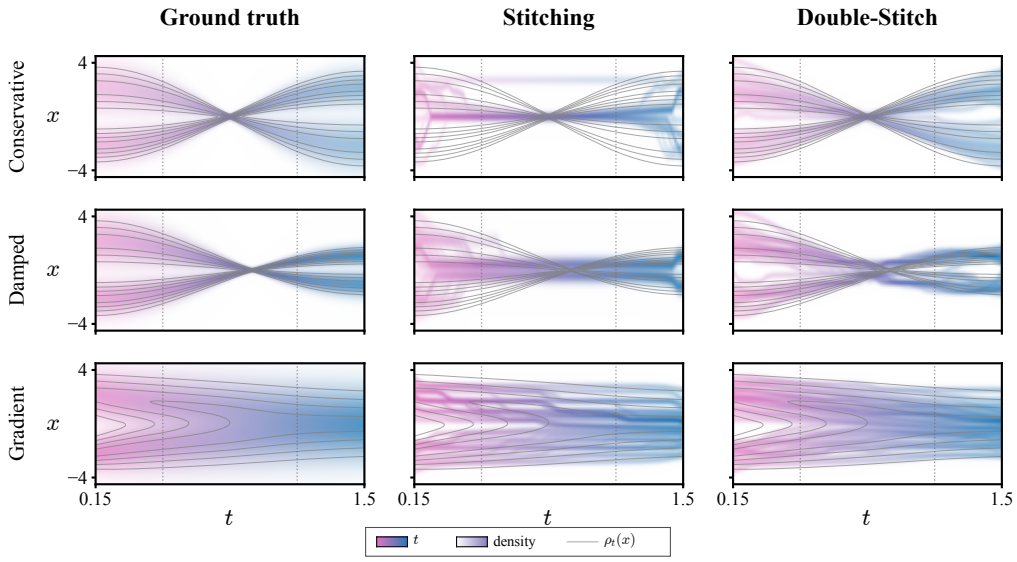

Figure 11: **Double-Stitch learns all three Gaussian-mixture flows.** Left: ground-truth conservative, damped and gradient densities. Center: Stitching, which fails on the conservative flow and underperforms on the damped flow. Right: Double-Stitch, which succeeds on all three. Dotted lines mark the held-out times.

E.5 Ablations and sensitivity studies

Prediction mode and settings. We compare the performance of Stitching, WLM, and Double-Stitch on the ocean datasets across all prediction modes in Table 11. The main and original settings of each method are described in Section D.2. We find that Double-Stitch has similar performance under direct and pushforward evaluations; rollout is worse, but surprisingly improves if a time-dependent potential is used. Stitching scores well under direct evaluation, but not under the other two, likely because its first-order condition is essentially misspecified on this data. For WLM, the original settings are slightly better on interpolation but worse on forecasting.

Table 11: **Double-Stitch and WLM sensitivity to the prediction mode on ocean vortices** (W_1, mean ± std, 3 seeds, lower is better). Dashes indicate unavailable cells.

Method	Settings	Direct	Pushforward	Rollout
Interpolation				
Stitching, $V(x)$	Main settings	0.366 ± 0.015	1.515 ± 0.273	3.858 ± 0.912
Stitching, $V(x,t)$	Main settings	0.125 ± 0.022	0.166 ± 0.035	1.089 ± 0.475
Stitching, $V(x)$	Original settings	0.301 ± 0.016	4.750 ± 0.535	32.764 ± 2.684
Stitching, $V(x,t)$	Original settings	0.104 ± 0.021	0.104 ± 0.020	0.104 ± 0.017
WLM	Main settings	—	0.064 ± 0.015	0.176 ± 0.098
WLM	Original settings	—	0.059 ± 0.015	0.115 ± 0.019
Double-Stitch, $V(x)$	Main settings	0.071 ± 0.019	0.065 ± 0.004	0.270 ± 0.050
Double-Stitch, $V(x,t)$	Main settings	0.063 ± 0.011	0.073 ± 0.009	0.103 ± 0.047
Forecast				
Stitching, $V(x)$	Main settings	0.295 ± 0.032	1.388 ± 0.119	14.370 ± 6.123
Stitching, $V(x,t)$	Main settings	0.212 ± 0.018	0.195 ± 0.022	0.224 ± 0.017
Stitching, $V(x)$	Original settings	0.206 ± 0.001	0.205 ± 0.006	0.149 ± 0.007
Stitching, $V(x,t)$	Original settings	0.175 ± 0.008	0.170 ± 0.004	0.211 ± 0.092
WLM	Main settings	—	0.088 ± 0.023	0.194 ± 0.048
WLM	Original settings	—	0.238 ± 0.275	0.889 ± 1.126
Double-Stitch, $V(x)$	Main settings	0.209 ± 0.013	0.134 ± 0.003	1.839 ± 0.539
Double-Stitch, $V(x,t)$	Main settings	0.194 ± 0.007	0.119 ± 0.019	0.222 ± 0.008

Training time. We investigate how the training time of Double-Stitch depends on the number of particles in Table 12. We find that the training time increases 4–5 times across all tasks when the number of particles increases tenfold.

Table 12: **Double-Stitch training time versus number of particles across datasets** (seconds, mean ± std, 3 seeds).

	EB			Oceans	
Setting	LPO	L1O	L2O	Interp.	Forecast
$P = 100$	28.8 ± 0.2	84.7 ± 3.3	29.0 ± 2.3	104.7 ± 1.0	104.4 ± 3.0
$P = 400$	63.3 ± 0.3	186.8 ± 2.1	62.8 ± 0.6	252.7 ± 6.0	277.1 ± 2.9
$P = 1000$	127.3 ± 2.9	385.5 ± 4.9	125.6 ± 0.4	512.4 ± 16.9	532.5 ± 7.5

Initialization. In Table 13 we compare the performance of Double-Stitch with different trajectory and bandwidth initializations (see definitions in Section D.2). Trajectory initialization has a large effect: endpoint OT has the largest error on all five tasks, and McCann-path is generally close to OT-path except for interpolation on oceans. Bandwidth initialization has a smaller effect: the coarsest initialization is better on ocean interpolation, and the two perform similarly on the other tasks.

Table 13: **Double-Stitch sensitivity to initialization across datasets** (mean ± std, 3 seeds, lower is better).

	EB			Oceans	
Setting	LPO W_1	L1O W_1	L2O W_2	Interp. W_1	Forecast W_1
OT-path, coarsest (ref.)	0.618 ± 0.001	0.733 ± 0.003	0.795 ± 0.006	0.065 ± 0.004	0.134 ± 0.003
OT-path, Silverman	0.614 ± 0.001	0.729 ± 0.004	0.795 ± 0.010	0.077 ± 0.021	0.131 ± 0.013
Endpoint OT, Silverman	0.777 ± 0.004	0.814 ± 0.011	0.814 ± 0.011	0.504 ± 0.274	0.578 ± 0.056
McCann-path, Silverman	0.640 ± 0.004	0.736 ± 0.001	0.791 ± 0.002	0.121 ± 0.009	0.127 ± 0.009

Time-dependent potential. In Table 14 we compare the performance of Double-Stitch with static and time-dependent potentials. Overall, the time-dependent potential outperforms the static potential on EB LPO and ocean forecasting and matches it elsewhere.

Table 14: **Double-Stitch sensitivity to a time-dependent potential across datasets** (mean ± std, 3 seeds, lower is better).

	EB			Oceans	
Setting	LPO W_1	L1O W_1	L2O W_2	Interp. W_1	Forecast W_1
$V(x)$	0.618 ± 0.001	0.733 ± 0.003	0.794 ± 0.005	0.065 ± 0.004	0.134 ± 0.003
$V(x,t)$	0.558 ± 0.004	0.716 ± 0.007	0.800 ± 0.004	0.073 ± 0.009	0.119 ± 0.019

Residual-loss weight. In Table 15 we compare the performance with various residual-loss weights. A smaller weight is slightly better on LPO, L1O and ocean interpolation, while the larger weight shows better results on L2O and forecasting. We hypothesize that this happens because on L2O and forecasting the gaps between observed snapshots are larger and the model relies more on the dynamics.

Table 15: **Double-Stitch sensitivity to the residual-loss weight across datasets** (mean ± std, 3 seeds, lower is better).

	EB			Oceans	
Setting	LPO W_1	L1O W_1	L2O W_2	Interp. W_1	Forecast W_1
$\lambda = 0.3$	0.591 ± 0.005	0.724 ± 0.005	0.805 ± 0.005	0.061 ± 0.002	0.134 ± 0.008
$\lambda = 1$ (ref.)	0.618 ± 0.001	0.733 ± 0.003	0.795 ± 0.006	0.065 ± 0.004	0.134 ± 0.003
$\lambda = 3$	0.633 ± 0.003	0.739 ± 0.006	0.786 ± 0.004	0.071 ± 0.011	0.129 ± 0.004

Initial friction. In Table 16 we compare the performance of Double-Stitch with various initial friction parameters. $\gamma_0 = 0.1$ seems to produce slightly lower errors than $\gamma_0 = 1$ on all five tasks, while $\gamma_0 = 10$ increases the errors on EB and ocean interpolation. For the main experiments we use $\gamma_0 = 1$ (fixed before the sensitivity study, as are the other parameters).

Table 16: **Double-Stitch sensitivity to the initial friction across datasets** (mean ± std, 3 seeds, lower is better).

	EB			Oceans	
Setting	LPO W_1	L1O W_1	L2O W_2	Interp. W_1	Forecast W_1
$\gamma_0 = 0.1$	0.610 ± 0.003	0.728 ± 0.004	0.793 ± 0.003	0.062 ± 0.001	0.113 ± 0.018
$\gamma_0 = 1$ (ref.)	0.618 ± 0.001	0.733 ± 0.003	0.795 ± 0.006	0.065 ± 0.004	0.134 ± 0.003
$\gamma_0 = 10$	0.740 ± 0.003	0.812 ± 0.003	0.827 ± 0.003	0.148 ± 0.005	0.127 ± 0.007

Number of particles. In Table 17 we compare the performance of Double-Stitch with various numbers of particles. On EB, increasing the number of particles reduces the error, with little change on L2O beyond $P = 400$. On oceans the situation is different: the error on the interpolation task increases with P. This is likely because this dataset has only 111 particles per snapshot, so $P = 100$ already matches its size. On forecasting, $P = 1000$ gives the lowest error but needs the highest training time (see Table 12).

Table 17: **Double-Stitch sensitivity to the number of particles across datasets** (mean ± std, 3 seeds, lower is better).

	EB			Oceans	
Setting	LPO W_1	L1O W_1	L2O W_2	Interp. W_1	Forecast W_1
$P = 100$	0.667 ± 0.013	0.766 ± 0.021	0.833 ± 0.003	0.065 ± 0.004	0.122 ± 0.018
$P = 400$	0.629 ± 0.009	0.735 ± 0.002	0.794 ± 0.003	0.079 ± 0.004	0.134 ± 0.003
$P = 1000$	0.618 ± 0.001	0.733 ± 0.003	0.795 ± 0.006	0.099 ± 0.013	0.098 ± 0.008