FLOW MAPS FOR TRAJECTORY INFERENCE TO IMPROVE LEARNED CONDITIONAL LAWS

Anonymous authors
Paper under double-blind review

ABSTRACT

Trajectory inference seeks to reconstruct a stochastic process from unpaired samples of its temporal marginals. Because temporal marginals do not uniquely determine the underlying process, inductive biases are required to select a plausible model. Many recent methods encode these biases by constructing an interpolant process connecting the observed marginals and train a neural ODE or SDE to match its time-dependent marginal distributions. In cellular differentiation, biological knowledge can specify plausible transitions between cell types, which learned dynamics should respect. However, encoding this knowledge in the interpolant alone does not guarantee that the learned model will inherit it, because matching marginals does not ensure matching joint distributions across time. This motivates directly shaping the transition laws of the learned process. A natural approach is to train with differentiable losses evaluated on model-generated trajectories, such as penalties for transitions that violate known lineage relationships. This, however, would conventionally require simulating the dynamics with a numerical solver and differentiating through the simulation, making training expensive. To more efficiently incorporate prior knowledge about transitions, we propose CLIFT, a framework that uses deterministic or stochastic flow maps to impose differentiable constraints on learned conditional laws. Flow maps provide direct, differentiable access to finite-time transitions, making these constraints efficient to optimise. To illustrate the framework, we formalise processes whose transitions follow a prescribed lineage graph as hierarchical processes, and introduce a hierarchical loss that encourages the learned dynamics to respect these rules. Across synthetic and single-cell datasets, CLIFT improves adherence to these transition rules while maintaining competitive marginal distributional fidelity. Comparisons with incorporating such prior knowledge through numerical rollouts demonstrate the substantial computational advantage of using flow maps.

1 INTRODUCTION

In trajectory inference (Hashimoto et al., 2016; Lavenant et al., 2021), we aim to reconstruct a stochastic process $\mathbf{X} = (X_t)_{t\in[0,1]} \sim \mathbb{P}$, where $\mathbb{P} \in \mathcal{P}\big(C([0,1],\mathbb{R}^d)\big)$, from K unpaired sets of samples from the temporal marginals $(p_{t_k})_{k\leq K}$, constructing an approximation $\hat{\mathbf{X}} \sim \hat{\mathbb{P}}$ of the underlying process. This is relevant in applications where one cannot measure systems of interest continuously in time, but can access cross-sectional data, such as when using destructive single-cell RNA-sequencing (scRNA-seq) (Klein et al., 2015; Macosko et al., 2015) to study cellular processes (Lähnemann et al., 2020). Other relevant domains include the study of migratory patterns (Scarvelis & Solomon, 2023) and neural network behaviours (Amad & van der Schaar, 2026).

Trajectory inference is fundamentally ill-posed: a finite collection of temporal marginals does not uniquely determine the underlying path law $\mathbb{P}$ (Lavenant et al., 2021). Additional assumptions, or inductive biases, are therefore needed to select an approximation $\hat{\mathbb{P}}$ from the family of processes consistent with the observed data. Many recent machine learning approaches choose a coupling between the observed marginals and construct interpolants between coupled samples before training a neural ODE or SDE (Chen et al., 2018; Li et al., 2020; Kidger et al., 2021), using matching objectives (Ho et al., 2020; Song et al., 2020; Lipman et al., 2023), to mimic the interpolant's temporal marginals. Specific couplings and interpolants can be chosen to encode assumptions in the temporal

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.

marginals. For example, least-action principles can be encoded by choosing an optimal transport (OT) coupling (Villani et al., 2009) and linear interpolants (Tong et al., 2024a).

A primary use of trajectory inference methods is to estimate a particle's future from its current state: i.e. to estimate the conditional distributions $(p_{s,t}(\cdot \mid x_s))_{t>s}$. In many applications, prior knowledge about plausible transitions can guide the estimation of these conditional distributions. Cellular differentiation is a clear example: lineage trees in developmental biology describe which cell types can give rise to which others (Wagner & Klein, 2020; Boezio et al., 2025), such that a cell that has committed to one lineage should not have descendants in a sibling lineage at any later time. However, it is not trivial to inform a model's conditional laws via bespoke couplings or interpolants. Matching objectives are designed to reproduce the interpolant's temporal marginals, but this does not guarantee that the learned dynamics reproduce its joint distributions across time (§2.3). This motivates imposing the transition constraints directly on the conditional laws of the learned process. For neural ODEs/SDEs parameterised by an instantaneous drift, optimising a loss on model-generated transitions requires differentiating the simulated dynamics, either through a numerical solver or via an adjoint method (Chen et al., 2018). The resulting computational cost can make direct regularisation of the learned conditional laws expensive.

We introduce CLIFT (**C**onditional-**L**aw-**I**nformed **F**low map for **T**rajectory inference), a framework for incorporating transition priors directly into learned dynamics. Our framework learns flow maps that approximate the finite-time solution operators of neural ODEs/SDEs (Boffi et al., 2026; McCallum et al., 2026). Flow maps have received recent attention in generative modelling for the inference-time speedups they enable through few-step sampling (Song et al., 2023; Frans et al., 2025; Geng et al., 2025; Boffi et al., 2026; McCallum et al., 2026), but their potential for trajectory inference remains unexplored.

- We identify limitations in the biases that interpolants encode into neural ODE/SDEs, by showing that the joint law of a neural ODE/SDE need not match that of its training interpolants (§2).
- We define a class of processes—hierarchical processes—that have known constraints on their conditional laws, and describe their relevance to trajectory inference for single-cell data (§3.1).
- We introduce CLIFT, a deterministic or stochastic flow maps with differentiable constraints on its conditional laws, allowing more faithful reconstruction of hierarchical processes (§3.3).
- We instantiate CLIFT on various single-cell datasets with known hierarchical lineage relationships and show that it generates trajectories that are substantially more biologically feasible than existing neural ODE/SDE methods, while remaining competitive in marginal distributional fidelity. We also demonstrate substantial speed-ups over alternate attempts to directly constrain neural ODEs/SDEs without using the flow map interface (§5).

2 Preliminaries

2.1 Neural ODEs and SDEs

A natural way to approximate $\mathbb{P}$ is by modelling its dynamics with an additive-noise neural SDE

$$d\mathbf{X}_t = \left[\mathbf{v}_\theta(t, \mathbf{X}_t) + \frac{g(t)^2}{2}\mathbf{s}_\phi(t, \mathbf{X}_t)\right] dt + g(t)\, d\mathbf{W}_t, \tag{1}$$

where $\mathbf{W}_t$ is a standard Wiener process, $\mathbf{v}_\theta$ is a learnable velocity field, $\mathbf{s}_\phi$ is a learnable score field, and $g(t)$ is a fixed diffusion coefficient. In the deterministic case $g(t) = 0$, this reduces to a neural ODE driven by the velocity field $\mathbf{v}_\theta$. These models can be learnt such that they push samples between observed distributions $(p_{t_k})_{k\leq K}$ along a defined probability path. Recent methods do so using matching objectives, such as score matching (Ho et al., 2020; Song et al., 2020) and flow matching (Liu et al., 2022; Albergo & Vanden-Eijnden, 2023; Lipman et al., 2023).

2.2 Learning Dynamics via Matching Objectives

The dynamics associated with a prescribed stochastic interpolant $\mathbf{I} : [0,1] \times \mathbb{R}^d \times \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^d$ between observed marginals can be learned efficiently with matching objectives (Albergo & Vanden-Eijnden, 2023). For simplicity, we will consider the two-marginal case here, with source p_0 and

target p_1, but this can easily be generalised to K-marginals. $\mathbf{I}$ can be defined as

$$\mathbf{I}_t(\mathbf{x}_0, \mathbf{x}_1, \epsilon) = \alpha_t \mathbf{x}_0 + \beta_t \mathbf{x}_1 + \rho_t \epsilon, \qquad \epsilon \sim \mathcal{N}(0, I) \tag{2}$$

where $\alpha, \beta : [0,1] \to [0,1]$, $\rho : [0,1] \to \mathbb{R}^d$ are continuously differentiable functions satisfying

$$\alpha_0 = \beta_1 = 1, \qquad \alpha_1 = \beta_0 = 0, \qquad \rho_0 = \rho_1 = 0 \tag{3}$$

and the pair $(\mathbf{x}_0, \mathbf{x}_1) \sim \pi$ are drawn from a coupling satisfying the marginal constraints $\int_{\mathbb{R}^d} \pi(\mathbf{x}_0, \mathbf{x}_1) d\mathbf{x}_0 = p_1(\mathbf{x}_1)$, $\int_{\mathbb{R}^d} \pi(\mathbf{x}_0, \mathbf{x}_1) d\mathbf{x}_1 = p_0(\mathbf{x}_0)$. The interpolant induces a probability path $(p_t^{\mathbf{I}})_{t\in[0,1]}$ which can then be approximated via a neural ODE/SDE. The coupling π and interpolant $\mathbf{I}$ determine the temporal marginals that the model will learn, and these objects can therefore be designed to encode various priors about the process.

Under suitable regularity conditions, the interpolant admits both a velocity and score field $\mathbf{v}^\star(t, \mathbf{x}) = \mathbb{E}\left[\dot{\mathbf{I}}_t \mid \mathbf{I}_t = \mathbf{x}\right]$, $\mathbf{s}^\star(t, \mathbf{x}) = \nabla_{\mathbf{x}} \log p_t^{\mathbf{I}}(\mathbf{x})$, which can be learned with matching objectives,

$$\begin{aligned} \mathcal{L}_{\mathrm{FM}}(\theta) &= \mathbb{E}_{t\sim U[0,1],\, (\mathbf{X}_0,\mathbf{X}_1)\sim\pi,\, \epsilon\sim\mathcal{N}(0,I)} \left[\left\| \mathbf{v}_\theta(t, \mathbf{I}_t) - \dot{\mathbf{I}}_t \right\|^2 \right], \\ \mathcal{L}_{\mathrm{SM}}(\phi) &= \mathbb{E}_{t\sim U[0,1],\, (\mathbf{X}_0,\mathbf{X}_1)\sim\pi,\, \epsilon\sim\mathcal{N}(0,I)} \left[\left\| \mathbf{s}_\phi(t, \mathbf{I}_t) + \frac{\epsilon}{\rho_t} \right\|^2 \right], \end{aligned} \tag{4}$$

which are minimised by $\mathbf{v}^\star$ and $\mathbf{s}^\star(t, \mathbf{x})$, respectively. The optimal velocity field $\mathbf{v}^\star$ defines a deterministic flow whose marginals coincide with $(p_t^{\mathbf{I}})_{t\in[0,1]}$, often called the probability-flow ODE

$$\mathbf{X}_0 \sim p_0, \qquad \frac{d\mathbf{X}_t}{dt} = \mathbf{v}^\star(t, \mathbf{X}_t), \qquad \mathbf{X}_t \sim p_t^{\mathbf{I}}, \qquad t \in [0,1]. \tag{5}$$

The score field can also be used to construct a stochastic process with the same marginals (3)

$$\mathbf{X}_0 \sim p_0, \qquad d\mathbf{X}_t = \left[\mathbf{v}^\star(t, \mathbf{X}_t) + \frac{g(t)^2}{2} \mathbf{s}^\star(t, \mathbf{X}_t) \right] dt + g(t)\, d\mathbf{W}_t, \qquad \mathbf{X}_t \sim p_t^{\mathbf{I}}, \qquad t \in [0,1]. \tag{6}$$

2.3 Neural ODEs/SDEs do not Learn The Joint Law of The Interpolants

There is no general guarantee that the neural ODE/SDE reproduces the interpolant's joint distribution across time of the interpolant. Indeed, consider the learnable interpolant allowing to encode prior knowledge directly on it: $\mathbf{m}_t^\theta(\mathbf{x}_0, \mathbf{x}_1) = (1-t)\mathbf{x}_0 + t\mathbf{x}_1 + \mathbf{g}_\theta(t, \mathbf{x}_0, \mathbf{x}_1)$, where $\mathbf{g}_\theta$ is a neural network that vanishes at the endpoints (Kapuśniak et al., 2024; Petrović et al., 2026; Tang et al., 2026). For such interpolants with independent Brownian-bridge noise, the following proposition shows that their joint laws cannot be recovered with a neural ODE/SDE.

Proposition 1 (Failure to recover the interpolant's joint law)**.** *Let $(Z_0, Z_1) \sim \pi \in \Pi(p_0, p_1)$, and B^{br} be a standard Brownian bridge independent of (Z_0, Z_1). Consider the interpolant*

$$I_t^\theta = m_t^\theta(Z_0, Z_1) + \sigma B_t^{\mathrm{br}}, \qquad \sigma > 0. \tag{7}$$

Under the assumptions in Appendix C.1, no Markov ODE/SDE model on $\mathbb{R}^d$ can reproduce its full path law, even if its one-time marginals agree.

Proof in Appendix C.1. Matching the interpolant's temporal marginals does not guarantee recovery of its joint law. Consequently, transition constraints encoded through the interpolant need not be preserved by the learned dynamics. This motivates encoding these constraints directly in the training objective of the learned model.

2.4 Learning the Solution Operator of ODE and SDEs

Rather than representing continuous-time dynamics only through their instantaneous velocity/drift, we can learn the solution operator that directly maps a particle to its future state.

2.4.1 Flow Maps

A flow map is the solution operator to the probability flow ODE, $\Phi_{s,t}(\mathbf{x}) = \mathbf{x} + \int_s^t \mathbf{v}\left(\tau, \Phi_{s,\tau}(\mathbf{x})\right)\, d\tau$. The exact flow map satisfies the identity, semigroup, and tangent properties

$$\Phi_{s,s}(\mathbf{x}) = \mathbf{x}, \qquad \Phi_{u,t}(\Phi_{s,u}(\mathbf{x})) = \Phi_{s,t}(\mathbf{x}),\ s \le u \le t, \qquad \partial_t \Phi_{s,t}(\mathbf{x}) = \mathbf{v}(t, \Phi_{s,t}(\mathbf{x}))\,. \tag{8}$$

A neural flow map can be parametrised as

$$\Phi_\theta(s, t, \mathbf{x}) = \mathbf{x} + (t - s) a_\theta(s, t, \mathbf{x}), \tag{9}$$

where $a_\theta(s, t, \mathbf{x})$ represents the average velocity over the transition. This parameterisation satisfies $\Phi_\theta(s, s, \mathbf{x}) = \mathbf{x}$ by construction, and its diagonal, $a_\theta(t, t, \mathbf{x})$, defines an instantaneous velocity field that can be trained via Eq. 4. Off-diagonal transitions can be trained with various self-distillation losses, such as Lagrangian self-distillation (LSD) (Boffi et al., 2026)

The Itô map $\Psi_{s,t}(\mathbf{x}, \mathbf{W}_{[s,t]}) = \mathbf{x} + \int_s^t f\left(\tau, \mathbf{x}_\tau\right) d\tau + \int_s^t g(\tau)\, d\mathbf{W}_\tau$, is the stochastic analogue, where $\mathbf{W}_{[s,t]}$ a realised Brownian path. The Itô map satisfies an analogue of the tangent condition,

$$\lim_{s \to t} d\Psi_{s,t}(\mathbf{x}_s, \mathbf{W}_{[s,t]}) = f(t, \mathbf{x}_t)\, dt + g(t)\, d\mathbf{W}_t, \tag{10}$$

together with the identity and semigroup properties. McCallum et al. (2026) propose a method to approximate this map with a neural network using an Euler–Maruyama-like parameterisation

$$\Psi_\theta(s, t, \mathbf{x}, \mathbf{W}_{[s,t]}) = \mathbf{x} + (t - s) f_\theta(s, t, \mathbf{x}, \mathbf{W}_{[s,t]}) + g_\theta(s, t, \mathbf{W}_{[s,t]})(\mathbf{W}_t - \mathbf{W}_s), \tag{11}$$

where training follows a similar process to the deterministic setting—training the diagonal portion to match the instantaneous drift of $\mathbf{I}$, and the off-diagonals to satisfy self-consistency. In this case, self-distillation cannot be enforced with the Lagrangian loss, since the time-derivative of a Brownian does not exist, so progressive self-distillation is used instead.

3 Method

We now formalise the class of processes with constraints on admissible transitions, and show how a loss that penalises constraint violations can be beneficial for trajectory inference. We then describe how CLIFT incorporates this loss into its training objective to shape the learned path distribution.

3.1 Hierarchical Processes

Many dynamical systems have restrictions on which transitions between states are admissible, adopting a hierarchical structure where a state may have several possible futures, and commitment to one can restrict subsequent transitions. We can formalise this through a directed acyclic graph of allowable transitions between regions of the state space. A hierarchical process, introduced below, has dynamics that approximately respect these restrictions.

Definition 1 (Hierarchical process)**.** Consider regular closed sets[a] $U_1, \ldots, U_I \subseteq \mathbb{R}^d$, each with nonempty interior, whose union is $\mathbb{R}^d$. Let G be a directed acyclic graph of allowable transitions between these regions, and let $\mathbf{X} = (X_t)_{t \in [0,T]}$ be a continuous stochastic process with path law $\mathbb{P}$ on $C([0, T]; \mathbb{R}^d)$.
We wish to describe processes whose trajectories respect this hierarchy, while allowing for fluctuations near region boundaries. For $\delta > 0$, define the interior core of each region by[b]

$$K_i^\delta = \left\{ x \in U_i : \operatorname{dist}(x, \mathbb{R}^d \setminus U_i) \ge \delta \right\}.$$

Let $\Omega_{G,\delta}$ consist of the continuous paths $\omega \in C([0, T]; \mathbb{R}^d)$ with the following property: whenever ω visits K_i^δ at some time s, every core K_j^δ visited at a later time $t > s$ must correspond either to the same region or to a descendant of i in G, meaning a vertex reachable from i by a directed path. For $\varepsilon \in [0, 1]$, we say that $\mathbb{P}$ is (δ, ε)-hierarchical on $[0, T]$ if

$$\boxed{\mathbb{P}(\Omega_{G,\delta}) \ge 1 - \varepsilon.}$$

Thus, with probability at least $1-\varepsilon$, the trajectory respects the hierarchy between these interior cores throughout the entire interval.

[a] A set U is regular closed if $U = \overline{\mathrm{int}(U)}$.
[b] We assume that δ is sufficiently small that each K_i^δ has nonempty interior.

Some real-world processes can be approximately described by hierarchical processes, such as cellular differentiation. scRNA-seq measures the transcriptome of individual cells, and cells in these datasets are conventionally assigned to discrete types, using marker genes and clustering. These types are related to one another by a lineage tree that records which type can give rise to which (Wagner & Klein, 2020; Boezio et al., 2025), constraining the dynamics of the process.

Differentiation proceeds by a branching sequence of transitions in which a multipotent progenitor becomes progressively restricted. Once a cell commits to one branch it does not revert to the progenitor state or cross into a sibling lineage. Waddington's landscape captures this idea as a ball rolling downhill through successively narrower valleys (Waddington, 1957). Individual fate choices may be stochastic, yet the set of permitted transitions is finite and, for some biological systems, known in advance, so the process is well approximated by a hierarchical process in the sense of Definition 1. To make this precise, let $\mathcal{T} : \mathbb{R}^d \to \{1, \dots, I\}$ denote the annotation map that assigns an expression profile x to its cell type, and $U_i = \overline{\mathcal{T}^{-1}(i)}$ be the region associated with cell type i. The lineage tree then specifies the permitted progressions between regions. Annotation procedures are dataset-dependent, so $\mathcal{T}$ is generally not available in closed form. Instead, we train a classifier $D_\alpha : \mathbb{R}^d \to \{1, \dots, I\}$ to predict cell types, and use its decision regions as estimates of U_i.

3.2 A Hierarchical Loss for Trajectory Inference

For a ground-truth path law $\mathbb{P}$ that is hierarchical according to Definition 1, we seek an approximation $\widehat{\mathbb{P}}$ that also respects this hierarchy. To encourage this, we introduce a hierarchical loss to apply to a neural ODE/SDE, given a classifier, as introduced above, and a lineage graph G.

Definition 2 (Hierarchical loss)**.** Let $\widehat{\mathbb{P}}$ be a candidate path law on $C([0,T];\mathbb{R}^d)$, and let $\widetilde{D}_\alpha : \mathbb{R}^d \to \Delta^{I-1}$ denote the probability output of a fixed classifier, with

$$D_\alpha(x) = \operatorname*{argmax}_{1 \le i \le I} [\widetilde{D}_\alpha(x)]_i.$$

Let $A \in \{0,1\}^{I \times I}$ encode the permitted transitions: $A_{ij} = 1$ if $j = i$ or j is a descendant of i in G, otherwise $A_{ij} = 0$. For two states $x, y \in \mathbb{R}^d$, define their soft transition compatibility by

$$q_{\mathrm{valid}}(x,y) = \widetilde{D}_\alpha(x)^\top A \, \widetilde{D}_\alpha(y).$$

Let μ be a probability distribution over ordered time pairs $0 \le s < t \le T$, sampled independently of the process. The hierarchical loss is

$$\boxed{\mathcal{L}_h(\widehat{\mathbb{P}}) = \mathbb{E}_{\substack{\mathbf{X} \sim \widehat{\mathbb{P}} \\ (s,t) \sim \mu}} [-\log q_{\mathrm{valid}}(X_s, X_t)]} \tag{12}$$

$\mathcal{L}_h$ uses the classifier's probability outputs to provide a differentiable training signal. We now relate this soft objective to the discrete transitions obtained from its predicted classes. The following proposition shows that $\mathcal{L}_h$ bounds the probability of a predicted forbidden transition.

Proposition 2 (Control of predicted transition violations)**.** *Under the setting of Definition 2,*

$$\boxed{\Pr_{\substack{\mathbf{X} \sim \widehat{\mathbb{P}} \\ (s,t) \sim \mu}} \left[A_{D_\alpha(X_s), D_\alpha(X_t)} = 0\right] \le I^2 \mathcal{L}_h(\widehat{\mathbb{P}}).}$$

Proof in Appendix C.2. This proposition justifies using the soft compatibility score to control violations assessed through hard class predictions. It concerns the classifier's inferred regions, however, which may differ from the original regions U_i. The next corollary makes the effect of this approximation explicit. Its proof is in Appendix C.3.

Corollary 1 (Control of violations between the original regions)**.** *Fix $\delta > 0$. Under the same sampling scheme, let ε_{cls} denote the probability that either sampled state belongs to a core K_i^δ but receives a predicted label different from i. Then*[a]

$$\Pr_{\substack{\mathbf{X}\sim\widehat{\mathbb{P}}\\(s,t)\sim\mu}}\left[\exists\, i,j:\ X_s\in K_i^\delta,\ X_t\in K_j^\delta,\ i\not\preceq_G j\right]\le I^2\mathcal{L}_h(\widehat{\mathbb{P}})+\varepsilon_{\text{cls}}.$$

In particular, if $D_\alpha(x)=i$ for every $x\in K_i^\delta$ and every i, then $\varepsilon_{\text{cls}}=0$.

[a] We write $i\not\preceq_G j$ if $j\neq i$ and j is not reachable from i by a directed path in G.

3.3 CLIFT

We now describe CLIFT, our method to train a **c**onditional-**l**aw-**i**nformed **f**low map for **t**rajectory inference, by imposing $\mathcal{L}_h$ on its trajectories (algorithms are in Appendix G). The foundations of the flow map training come from Boffi et al. (2026) in the deterministic case, and McCallum et al. (2026) in the stochastic case, and using these maps as an interface to shape the conditional laws of a neural ODE/SDE is one of our novel contributions.

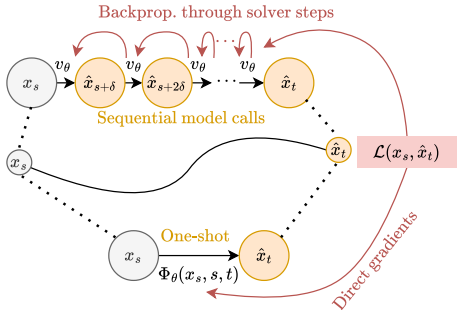


Figure 1: Enforcing constraints on finite-time transitions using a velocity field vs. a flow map. Flow maps can jump along the flow with a single model call, and gradients can pass directly to the model.

CLIFT follows standard flow map training by using the instantaneous drift of $\mathbf{I}$ to train the diagonal portion of the flow map, and self-distillation losses to enforce self-consistency on the off-diagonals. The primary change is the addition of $\mathcal{L}_h$ to assess the validity of the transitions from $\mathbf{I}_s$ to $\Phi_\theta(s,t,\mathbf{I}_s)$, using the already calculated off-diagonal model calls from the self-distillation training step. In doing so, we can shape the conditional law of the model with $\mathcal{L}_h$ without requiring *any* extra model calls beyond usual flow map training. We can set CLIFT to be deterministic or stochastic, depending on which base flow map we use, and we term these CLIFT$_D$ and CLIFT$_S$, respectively.

We can compare this to how $\mathcal{L}_h$ would be calculated using a traditional neural ODE/SDE interface, where only the instantaneous drift is learned, to see the substantial efficiency gains the flow map confers. Computing the transition from $\mathbf{I}_s$ to $\mathbf{X}_t$ would require integration of the neural ODE/SDE, which could require on the order of 10 to 100 model calls. Subsequently, the gradients from $\mathcal{L}_h$ would have to be backpropagated through this solve to update the model (see Figure 1). A complementary stochastic optimal control interpretation of our approach is provided in Appendix D.

4 Related Work

Trajectory inference (Hashimoto et al., 2016; Lavenant et al., 2021) is prominent in single-cell genomics, where destructive scRNA-seq precludes tracking individual cells over time (Macosko et al., 2015; Schiebinger et al., 2019). Here, we focus on methods that allow continuous-time interpolation between observed distributions, rather than inference of cell-level transition probabilities without modelling the intervening probability densities (Lange et al., 2022; Weiler et al., 2024). Such methods are often based on neural ODEs (Chen et al., 2018) or SDEs (Li et al., 2020; Kidger et al., 2021), and most recent approaches are trained via score or flow matching (Shi et al., 2023; Neklyudov et al., 2024; Tong et al., 2024b;a; Kapuśniak et al., 2024; Petrović et al., 2026; Sun et al., 2026).

Successful trajectory inference methods often leverage inductive biases or auxiliary data to generalise beyond the sparse observed times and discover a more faithful approximation of the underlying

process. Typical biases include assuming least-action principles—that the process moves in the most efficient way possible (Yang & Uhler, 2019; Schiebinger et al., 2019; Tong et al., 2020; Scarvelis & Solomon, 2023)—and encouraging inferred paths to traverse dense regions of the data space (Tong et al., 2020; Kapuśniak et al., 2024). Auxiliary data can come from a variety of sources, such as reference RNA velocities (Petrović et al., 2026) or clonal barcodes (Sun et al., 2026). See Table 5 for a comparison of our work with those most related.

Few-step flow-models have been proposed in various forms (Song et al., 2023; Frans et al., 2025; Geng et al., 2025; Boffi et al., 2026; McCallum et al., 2026), to accelerate sampling from flow-based generative models. Two-time flow maps are the most general form, which learn the solution operator of an ODE (Boffi et al., 2026) or SDE (McCallum et al., 2026) between any two time points by enforcing self-consistency in the learning process. While the worth of flow maps is clear in domains such as image generation, there application to trajectory inference is currently unexplored. Here we show that such models can permit efficient imposition of conditional-law constraints during training.

5 Experiments

We now evaluate CLIFT empirically. Firstly, we demonstrate how CLIFT can approximate a synthetic hierarchical stochastic process and utilise prior knowledge on the conditional law (§5.1). We then investigate real-world use cases on scRNA-seq data, comparing to existing neural ODE/SDE methods in terms of biological feasibility, according to known differentiation rules, and distributional fidelity (§5.2). Finally, we examine a further scRNA-seq dataset that contains clonal barcodes, allowing us to derive another measure of biological feasibility according to clone-conditional distributional fidelity (§5.3). Unless otherwise stated, we report the means and standard errors for results over five runs. Extended experimental details are given in the appendix.

5.1 Synthetic Hierarchical Process

We first test whether a lineage prior improves the reconstruction of conditional laws while preserving marginal fidelity. To this end, we construct a two-dimensional stylised SDE model of cellular differentiation. Access to the ground-truth process allows us to evaluate marginal distributions at held-out times, and the distribution of possible futures from a given starting state and time.

The ground-truth process evolves on $\mathbb{R}^2$ over $t \in [0, 2]$, with coordinates representing developmental progress and phenotype. Its marginals follow curved branches: an initial progenitor population separates into two lineages, one of which subsequently divides, yielding three terminal populations (see an illustration in Appendix 2). Diffusion near the branching events makes fate decisions stochastic. We generate $10{,}000$ cells at each of 21 equally spaced times, but use only marginals at $t \in \{0, 1, 2\}$ for fitting; the remaining 18 marginals are reserved for evaluation. A fixed spatial partition assigns five cell types, whose lineage graph permits transitions to the same type or its descendants. A classifier is trained exclusively on cells from the three training times.

A key question in cellular differentiation is how likely a cell in a given state is to reach each terminal fate. Let $F_T \in \{1, 2, 3\}$ denote the terminal fate at $T = 2$. For a cell in state x at time s, the ground-truth and model-predicted fate probabilities are $q^\star_f(s, x) = \mathbb{P}^\star(F_T = f \mid X_s = x)$, $\widehat{q}_f(s, x) = \widehat{\mathbb{P}}(F_T = f \mid X_s = x)$. We estimate these using stochastic continuations from the same conditioning state. We then report **Fate TV**, the total variation distance between the two fate distributions over a common evaluation bank $\{(s_i, x_i)\}_{i=1}^N$. To assess the spatial distribution of possible futures, we also report **Conditional SW$_2$**: the sliced Wasserstein-2 distance between model and ground-truth conditional distributions, averaged over the remaining time horizon. Finally, **Marginal W$_1$** mea-

Diffusion	Method	Fate TV (↓)	Conditional SW$_2$ (↓)	Marginal W$_1$
Ground truth	SF^{2}M	0.1784 ± 0.0013	0.1217 ± 0.0009	0.1428 ± 0.0057
	SSFM	0.1796 ± 0.0014	0.1220 ± 0.0008	0.1422 ± 0.0038
	CLIFT$_S$	$\mathbf{0.1659 \pm 0.0026}$	$\mathbf{0.1158 \pm 0.0009}$	$\mathbf{0.1415 \pm 0.0042}$
$G = 0.2I_2$	SF^{2}M	0.0998 ± 0.0015	0.0989 ± 0.0004	0.1331 ± 0.0020
	SSFM	0.0994 ± 0.0012	0.0982 ± 0.0004	0.1336 ± 0.0025
	CLIFT$_S$	$\mathbf{0.0893 \pm 0.0025}$	$\mathbf{0.0937 \pm 0.0008}$	$\mathbf{0.1304 \pm 0.0016}$

Table 1: Evaluation of methods on two-dimensional synthetic hierarchical process.

sures the average Wasserstein-1 distance between model-generated and ground-truth populations at held-out times. Lower values are better for all three metrics.

We compare three stochastic models: SF^{2}M (Tong et al., 2024b), which learns a neural SDE through score and flow matching, a strong stochastic flow map (SSFM) (McCallum et al., 2026), the base stochastic flow map or our method, and CLIFT$_S$. Comparing SSFM and CLIFT$_S$ isolates the contribution of the lineage prior, with SF^{2}M providing a conventional neural SDE baseline. All methods use OT endpoint couplings. We perform these comparisons under two diffusion settings: supplying the models with the ground-truth diffusion, or a constant isotropic diffusion $G = 0.2I_2$.

Table 1 compares the three methods. SSFM and SF^{2}M obtain similar mean conditional reconstruction errors, while CLIFT$_S$ achieves lower values on both conditional metrics. Relative to SSFM, CLIFT$_S$ reduces mean Fate TV by 7.59% with the ground-truth diffusion and 10.15% with constant diffusion. Conditional SW$_2$ decreases by 5.07% and 4.60%, respectively. These results indicate improved reconstruction of both terminal-fate probabilities and the spatial distributions of possible futures. In Figure 4, we compare a ground-truth conditional endpoint distribution with those predicted by SF^{2}M and CLIFT$_S$.

5.2 Single-Cell Data

Differentiation of embryonic stem cells into neural tube identities. We use the data of Maizels et al. (2024), comprising $44,713$ cells from ten sampling times with annotated cell types from nine classes that represent the stages of differentiation from embryonic stem cells into neural tube identities. The lineage tree between these types is represented in Figure 8. We use the reflexive transitive closure of this graph to define the allowable transitions for $\mathcal{L}_{\mathrm{h}}$: cells may remain in their current class or move to a descendant, but may not move backwards or switch branches after commitment. We represent cells in a 50-dimensional PCA space and learn transport between three evenly spaced observed marginals, reserving seven intermediate populations for distributional evaluation.

Differentiation of hematopoietic stem cells into blood-cell progenitors. The common benchmark CITE-seq (Cite) and Multiome (Multi) datasets, from Lance et al. (2022), comprise $31,240$ and $33,701$ cells, respectively, measured at four times, with the same seven annotated cell types. The associated lineage tree is shown in Figure 9. We follow the experimental design of prior works (Tong et al., 2024b; Kapuśniak et al., 2024), operating in 100-dimensional PCA space and evaluating models in two leave-one-out settings (leaving out either of the middle two time points and training on the other three). Reported results are the average of these two settings.

For all datasets, we train a classifier, for $\mathcal{L}_h$, on the observed marginals. For CLIFT, we also use this classifier to inform the training interpolants, by filtering any interpolants that traverse an invalid sequence of cell types. We compare to only doing interpolant filtering as an ablation.

Across these datasets, we report two metrics. **Viol. (%)** is the percentage of model-generated trajectories, starting at held-out cells from the initial training day, that violate the relevant lineage graph, where we use an unseen classifier, trained on all time points, to annotate cell types along the trajectories. $\mathbf{W}_1$ is the Wasserstein-1 distance between the held-out time points and the model pushforward distributions from the nearest observed marginal. Lower is better for both metrics.

For the deterministic setting, we compare CLIFT$_{\mathrm{D}}$ to flow matching using independent (I-FM) and OT (OT-FM) couplings (Tong et al., 2024a), metric flow matching using an OT coupling (OT-MFM) (Kapuśniak et al., 2024), a branched Schrödinger bridge method (BranchSBM) (Tang et al., 2026), a deterministic flow map using an OT coupling (OT-FMap) (Boffi et al., 2026), OT-FMap with interpolant filtering (OT-FMap (filt.)), and OT-FM with interpolant filtering and $\mathcal{L}_h$ imposed via rollouts (Constr. OT-FM (filt.)). For the stochastic setting, we compare CLIFT$_{\mathrm{S}}$ to SF^{2}M (Tong et al., 2024b), a strong stochastic flow map trained using independent (I-SSFM) and OT couplings (OT-SSFM) (McCallum et al., 2026), OT-SSFM with interpolant filtering (OT-SSFM (filt.)), and an OT neural SDE with interpolant filtering and $\mathcal{L}_h$ imposed via rollouts (Constr. OT-SDE (filt.)). The results are in Table 2.

Across both deterministic and stochastic settings, CLIFT substantially reduces lineage violations relative to the existing baselines, while also being competitive on held-out distributional fidelity. In both settings, CLIFT has the best average rank in terms of Viol. %. In the stochastic setting, CLIFT$_S$ ranks best in terms of W_1, while CLIFT$_D$ ranks third amongst deterministic methods on

Method	Maizels		Cite		Multi	
	Viol. % (↓)	$\mathbf{W}_1$ (↓)	**Viol.** % (↓)	$\mathbf{W}_1$ (↓)	**Viol.** % (↓)	$\mathbf{W}_1$ (↓)
Deterministic Methods						
I-FM	39.38 ± 1.38	10.47 ± 0.04	55.90 ± 0.80	48.90 ± 0.58	60.74 ± 0.51	56.95 ± 0.14
OT-FM	23.13 ± 1.37	10.33 ± 0.03	33.78 ± 0.81	45.44 ± 0.24	35.69 ± 0.51	54.95 ± 0.20
OT-MFM	29.75 ± 3.80	10.13 ± 0.08	31.09 ± 0.80	42.93 ± 0.25	33.91 ± 0.88	51.63 ± 0.16
BranchedSBM	20.31 ± 6.46	12.50 ± 0.31	46.86 ± 0.67	42.87 ± 0.16	49.59 ± 0.34	62.93 ± 0.58
OT-FMap	23.67 ± 1.32	10.30 ± 0.07	32.81 ± 1.12	46.54 ± 0.31	33.92 ± 0.15	54.90 ± 0.37
OT-FMap (filt.)	15.55 ± 0.92	10.09 ± 0.05	16.24 ± 0.09	44.00 ± 0.22	21.53 ± 0.37	53.79 ± 0.34
Constr. OT-FM (filt.)	9.58 ± 1.67	10.17 ± 0.05	16.76 ± 0.18	43.83 ± 0.34	21.23 ± 0.43	53.51 ± 0.22
CLIFT_{D}	9.06 ± 0.89	10.15 ± 0.05	13.16 ± 0.42	43.96 ± 0.12	18.98 ± 0.20	53.69 ± 0.31
Stochastic Methods						
SF^2M-Exact	26.88 ± 0.90	10.59 ± 0.04	31.42 ± 1.40	46.00 ± 0.14	33.21 ± 0.92	54.65 ± 0.56
I-SSFM	46.52 ± 1.76	10.34 ± 0.01	53.21 ± 1.01	48.28 ± 0.18	59.18 ± 0.44	57.71 ± 0.20
OT-SSFM	30.27 ± 0.40	10.30 ± 0.01	33.62 ± 0.48	45.58 ± 0.08	36.46 ± 0.54	55.52 ± 0.03
OT-SSFM (filt.)	22.06 ± 1.00	10.06 ± 0.01	20.38 ± 0.42	45.27 ± 0.13	25.42 ± 0.35	54.40 ± 0.10
Constr. OT-SDE (filt.)	25.33 ± 5.68	10.29 ± 0.01	15.51 ± 0.28	57.22 ± 1.36	40.48 ± 5.91	61.89 ± 2.75
CLIFT_{S}	15.67 ± 1.35	10.04 ± 0.03	15.87 ± 0.30	44.40 ± 0.08	18.85 ± 0.32	54.12 ± 0.08

Table 2: Evaluation of trajectory inference methods across single-cell datasets, in terms of the biological feasibility of individual trajectories (Viol. %) and held-out distributional fidelity (W_1).

this metric. We can see that imposing the lineage-based inductive bias only at the interpolant level, via filtering, is generally beneficial, as the methods with filtering tend to have lower violation rates. Directly constraining the conditional law of the flow model, however, yields substantial gains on top of this. We can also see that attempting to impose $\mathcal{L}_h$ via rollouts is generally worse, particularly in the stochastic setting. This advantage is compounded by the substantial efficiency gains of CLIFT, as Table 3 shows that imposing $\mathcal{L}_h$ with CLIFT is up to $\sim 8\times$ faster.

Overall, these results validate the claim that biological lineage trees can be a useful inductive bias to encode to recover a more faithful cellular differentiation process. Furthermore, they show that directly enforcing conditional-law biases on a neural ODE/SDE can lead to greater adherence on a real-world dataset, and that flow maps provide a natural and practical way to allow this. For a qualitative display of the difference between CLIFT and standard flow maps, we plot some paired trajectories from the OT-FMap and CLIFT_{D} models in Appendix F.2.

Method	Maizels	Cite	Multi
Deterministic			
Constr. OT-FM	148 ± 8	104 ± 3	108 ± 3
CLIFT_D	53 ± 4	37 ± 4	37 ± 6
Stochastic			
Constr. OT-SDE	75 ± 6	65 ± 4	68 ± 5
CLIFT_S	25 ± 5	8 ± 1	8 ± 1

Table 3: Runtime (min.) for constraining a neural ODE/SDE using rollouts vs. CLIFT.

5.3 Single-Cell Data with Clonal Barcodes

To assess biological feasibility from another angle, we experiment on the LARRY dataset (Weinreb et al., 2020), which provides lineage information, in the form of clonal barcodes, alongside cell-type annotations. We use the two-dimensional SPRING representation of this data, from the original work, and set the start and end days as the observed marginals, leaving one held out marginal. We train CLIFT as before, but we can now assess a clone-level notion of biological fidelity. For each clone, we push forward cells from its starting-day population to the held-out day, and measure the W_1 distance to distribution of the same clone at that time, which, averaged across clones, yields a clone-conditional Wasserstein distance (Clone-W_1).

As shown in Table 4, CLIFT_S improves over OT-SSFM and OT-SSFM (filt.) in terms of our previous metrics, and the new Clone-W_1. This reinforces our previous results, while also suggesting that incorporating lineage information via $\mathcal{L}_h$ can improve adherence to an independent, clone-level measure of biological fidelity. Also, since we see that our Viol. % metric correlates with the Clone-W_1, this provides evidence that Viol. % can hold as a useful measure of biological fidelity for datasets without richer barcoding information, as in our previous experiments. For an extended discussion on our takeaways, and limitations of this work, see Appendix A.

Method	Viol. % (↓)	$\mathbf{W}_1$ (↓)	Clone-$\mathbf{W}_1$ (↓)
OT-SSFM	31.02 ± 0.45	0.347 ± 0.006	0.854 ± 0.001
OT-SSFM (filt.)	29.96 ± 0.57	0.336 ± 0.006	0.845 ± 0.001
CLIFT_S	28.41 ± 0.59	0.333 ± 0.004	0.840 ± 0.001

Table 4: Evaluation of trajectory inference methods on two-dimensional LARRY (Weinreb et al., 2020) data.

AI USE STATEMENT

In this work, we used generative AI tools to generate synthetic data sets, help develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, and implement methods.

We have not used generative AI tools to propose or refine hypotheses, design or provide feedback on research methodology or experiments, assist with translation, clean and reformat datasets, support qualitative and thematic data analysis, or interpret results.

Additionally, we used generative AI tools for create or modify scientific figures or images, create or edit software code, and draft parts of a research paper.

We have reviewed all AI-assisted work. All mathematical derivations have been carefully checked and human-verified, as has AI-generated software by the authors. The final version of the manuscript is human written.

We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Michael S. Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. In *International Conference on Learning Representations*, 2023.

Harry Amad and Mihaela van der Schaar. Hyperparameter trajectory inference with conditional lagrangian optimal transport. In *The Fourteenth International Conference on Learning Representations*, 2026.

Giulia L. M. Boezio, Jasper R. L. Depotter, Thomas J. R. Frith, Arthur Radley, and James Briscoe. Hierarchical lineage architecture of human and avian spinal cord revealed by single-cell RNA sequencing. *bioRxiv*, 2025. doi: 10.1101/2025.10.24.684328. Preprint.

Nicholas Boffi, Michael Albergo, and Eric Vanden-Eijnden. How to build a consistency model: Learning flow maps via self-distillation. *Advances in Neural Information Processing Systems*, 38: 33346–33382, 2026.

Michelle Boué and Paul Dupuis. A variational representation for certain functionals of Brownian motion. *The Annals of Probability*, 26(4):1641–1659, 1998. doi: 10.1214/aop/1022855876.

Ricky TQ Chen and Yaron Lipman. Flow matching on general geometries. In *International Conference on Learning Representations*, volume 2024, pp. 47922–47945, 2024.

Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

Rama Cont and David-Antoine Fournié. Functional Itô calculus and stochastic integral representation of martingales. *The Annals of Probability*, 41(1):109–133, 2013. doi: 10.1214/11-AOP721.

Wendell H. Fleming and H. Mete Soner. *Controlled Markov Processes and Viscosity Solutions*, volume 25 of *Stochastic Modelling and Applied Probability*. Springer, New York, second edition, 2006. doi: 10.1007/0-387-31071-1.

Kevin Frans, Danijar Hafner, Sergey Levine, and Pieter Abbeel. One step diffusion via shortcut models, 2025. URL https://arxiv.org/abs/2410.12557.

Zhengyang Geng, Mingyang Deng, Xingjian Bai, Zico Kolter, and Kaiming He. Mean flows for one-step generative modeling. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 75460–75482. Curran Associates, Inc., 2025. doi: 10.52202/085713-2534. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/6d13e085b79d454da5910e4ca82a3d9d-Paper-Conference.pdf.

Tatsunori Hashimoto, David Gifford, and Tommi Jaakkola. Learning population-level diffusions with generative RNNs. In *International Conference on Machine Learning*, pp. 2417–2426. PMLR, 2016.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Peter Holderrieth, Uriel Singer, Tommi Jaakkola, Ricky TQ Chen, Yaron Lipman, and Brian Karrer. Glass flows: Transition sampling for alignment of flow and diffusion models. *arXiv preprint arXiv:2509.25170*, 2025.

Jerry Y Huang, Justin Lin, Sheel Shah, Kartik Nair, and Nicholas M Boffi. How to guide your flow: Few-step alignment via flow map reward guidance. *arXiv preprint arXiv:2604.27147*, 2026.

Kacper Kapuśniak, Peter Potaptchik, Teodora Reu, Leo Zhang, Alexander Tong, Michael Bronstein, Avishek J Bose, and Francesco Di Giovanni. Metric flow matching for smooth interpolations on the data manifold. *Advances in Neural Information Processing Systems*, 37:135011–135042, 2024.

Patrick Kidger, James Foster, Xuechen Li, and Terry Lyons. Efficient and accurate gradients for neural sdes. *Advances in neural information processing systems*, 34:18747–18761, 2021.

Allon M Klein, Linas Mazutis, Ilke Akartuna, Naren Tallapragada, Adrian Veres, Victor Li, Leonid Peshkin, David A Weitz, and Marc W Kirschner. Droplet barcoding for single-cell transcriptomics applied to embryonic stem cells. *Cell*, 161(5):1187–1201, 2015.

David Lähnemann, Johannes Köster, Ewa Szczurek, Davis J McCarthy, Stephanie C Hicks, Mark D Robinson, Catalina A Vallejos, Kieran R Campbell, Niko Beerenwinkel, Ahmed Mahfouz, et al. Eleven grand challenges in single-cell data science. *Genome biology*, 21(1):31, 2020.

Christopher Lance, Malte D. Luecken, Daniel B. Burkhardt, Robrecht Cannoodt, Pia Rautenstrauch, Anna Laddach, Aidyn Ubingazhibov, Zhi-Jie Cao, Kaiwen Deng, Sumeer Khan, Qiao Liu, Nikolay Russkikh, Gleb Ryazantsev, Uwe Ohler, NeurIPS 2021 Multimodal data integration competition participants, Angela Oliveira Pisco, Jonathan Bloom, Smita Krishnaswamy, and Fabian J. Theis. Multimodal single cell data integration challenge: Results and lessons learned. In Douwe Kiela, Marco Ciccone, and Barbara Caputo (eds.), *Proceedings of the NeurIPS 2021 Competitions and Demonstrations Track*, volume 176 of *Proceedings of Machine Learning Research*, pp. 162–176. PMLR, 06–14 Dec 2022. URL https://proceedings.mlr.press/v176/lance22a.html.

Marius Lange, Volker Bergen, Michal Klein, Manu Setty, Bernhard Reuter, Mostafa Bakhti, Heiko Lickert, Meshal Ansari, Janine Schniering, Herbert B. Schiller, et al. Cellrank for directed single-cell fate mapping. *Nature Methods*, 19(2):159–170, 2022.

Hugo Lavenant, Stephen Zhang, Young-Heon Kim, and Geoffrey Schiebinger. Towards a mathematical theory of trajectory inference. *arXiv preprint arXiv:2102.09204*, 2021.

Christian Léonard. A survey of the Schrödinger problem and some of its connections with optimal transport. *Discrete and Continuous Dynamical Systems*, 34(4):1533–1574, 2014. doi: 10.3934/dcds.2014.34.1533.

Xuechen Li, Ting-Kam Leonard Wong, Ricky TQ Chen, and David K Duvenaud. Scalable gradients and variational inference for stochastic differential equations. In *Symposium on Advances in Approximate Bayesian Inference*, pp. 1–28. PMLR, 2020.

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

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

Evan Z Macosko, Anindita Basu, Rahul Satija, James Nemesh, Karthik Shekhar, Melissa Goldman, Itay Tirosh, Allison R Bialas, Nolan Kamitaki, Emily M Martersteck, et al. Highly parallel genome-wide expression profiling of individual cells using nanoliter droplets. *Cell*, 161(5):1202–1214, 2015.

Rory J Maizels, Daniel M Snell, and James Briscoe. Reconstructing developmental trajectories using latent dynamical systems and time-resolved transcriptomics. *Cell systems*, 15(5):411–424, 2024.

Sam McCallum, Zander W Blasingame, Timothy Herschell, Niklas Rindtorff, Alexander Tong, and James Foster. Strong stochastic flow maps. *arXiv preprint arXiv:2606.01086*, 2026.

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*, volume 235 of *Proceedings of Machine Learning Research*, pp. 37461–37485. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/neklyudov24a.html.

Katarina Petrović, Lazar Atanackovic, Viggo Moro, Kacper Kapusniak, Ismail Ilkan Ceylan, Michael Bronstein, Joey Bose, and Alexander Tong. Curly flow matching for learning non-gradient field dynamics. *Advances in Neural Information Processing Systems*, 38:121544–121573, 2026.

Peter Potaptchik, Adhi Saravanan, Abbas Mammadov, Alvaro Prat, Michael S Albergo, and Yee Whye Teh. Meta flow maps enable scalable reward alignment. *arXiv preprint arXiv:2601.14430*, 2026.

Christopher Scarvelis and Justin Solomon. Riemannian metric learning via optimal transport. In *The Eleventh International Conference on Learning Representations*, 2023.

Geoffrey Schiebinger, Jian Shu, Marcin Tabaka, Brian Cleary, Vidya Subramanian, Aryeh Solomon, Joshua Gould, Siyan Liu, Stacie Lin, Peter Berube, et al. Optimal-transport analysis of single-cell gene expression identifies developmental trajectories in reprogramming. *Cell*, 176(4):928–943.e22, 2019.

Yuyang Shi, Valentin De Bortoli, Andrew Campbell, and Arnaud Doucet. Diffusion schrödinger bridge matching. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 62183–62223. Curran Associates, Inc., 2023. doi: 10.52202/075280-2717. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/c428adf74782c2092d254329b6b02482-Paper-Conference.pdf.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 32211–32252. PMLR, 2023.

Yuhao Sun, Zekun Wu, Zixun Huang, and Peijie Zhou. Tracingflow: A simulation-free trajectory inference framework based on second-order dynamics. *arXiv preprint arXiv:2608.21070*, 2026.

Sophia Tang, Yinuo Zhang, Alexander Tong, and Pranam Chatterjee. Branched schrödinger bridge matching. *ArXiv*, pp. arXiv–2506, 2026.

Alexander Tong, Jessie Huang, Guy Wolf, David Van Dijk, and Smita Krishnaswamy. Trajectorynet: A dynamic optimal transport network for modeling cellular dynamics. In *International conference on machine learning*, pp. 9526–9536. PMLR, 2020.

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*, pp. 1–34, 2024a.

Alexander Y. Tong, Nikolay Malkin, Kilian Fatras, Lazar Atanackovic, Yanlei Zhang, Guillaume Huguet, Guy Wolf, and Yoshua Bengio. Simulation-free Schrödinger bridges via score and flow matching. In Sanjoy Dasgupta, Stephan Mandt, and Yingzhen Li (eds.), *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, volume 238 of *Proceedings of Machine Learning Research*, pp. 1279–1287. PMLR, 02–04 May 2024b. URL https://proceedings.mlr.press/v238/tong24a.html.

Cédric Villani et al. *Optimal transport: old and new*, volume 338. Springer, 2009.

Conrad H. Waddington. *The Strategy of the Genes*. Allen & Unwin, London, 1957.

Daniel E. Wagner and Allon M. Klein. Lineage tracing meets single-cell omics: opportunities and challenges. *Nature Reviews Genetics*, 21(7):410–427, 2020. doi: 10.1038/s41576-020-0223-2.

Philipp Weiler, Marius Lange, Michal Klein, Dana Pe'er, and Fabian Theis. Cellrank 2: unified fate mapping in multiview single-cell data. *Nature methods*, 21(7):1196–1205, 2024.

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

KD Yang and C Uhler. Scalable unbalanced optimal transport using generative adversarial networks. In *International Conference on Learning Representations*, 2019.

A Discussion

In this work, we have investigated how to perform trajectory inference on hierarchical processes, when one has some level of knowledge about the allowable transitions according to the hierarchy. This setting is compelling given its relevance to studying cellular differentiation using trajectory inference, a very common application for these methods. We have proposed a method, CLIFT, to efficiently impose biases on the learned conditional laws of a neural ODE/SDE by learning a neural approximation to the solution map of the learned dynamics.

We have shown that hierarchical priors can indeed be constructed in scRNA-seq datasets, where cell types are often available, and that these priors can improve the modelling of the underlying process (§5.2). Furthermore, we have empirically shown that encoding such priors using a bespoke interpolant process is not as effective as directly constraining the learned dynamics, as our interpolant filtering ablations were generally not as capable as the full CLIFT method. This validates our theory that addresses and formalises this same issue (§2). We have also shown that the flow map interface allows much more efficient imposition of hierarchical priors to a neural ODE/SDE, compared to roll outs using the instantaneous drift (Table 3).

This work, and our proposed method, is not without limitations. Of course, the success of CLIFT, and indeed all trajectory inference methods, will depend on the validity of the prior knowledge encoded into the method. If, for example, a lineage tree that does not represent the allowable transitions in the underlying process, is encoded into CLIFT, this will likely worsen the reconstruction. User's must always carefully

B Synthetic hierarchical dataset

We construct a two-dimensional synthetic process to evaluate whether trajectory inference methods recover the distribution of possible futures from a given state. Knowing the ground-truth dynamics allows us to assess conditional distributions as well as population distributions at unobserved times.

The process represents a stylised differentiation system on $t \in [0, 2]$. An initial progenitor population separates into two lineages, one of which subsequently divides, producing three terminal populations. The population centres follow curved paths, while their shapes evolve from approximately isotropic to horizontally elongated and then mildly vertically elongated. These features require a model to capture both the movement and deformation of populations. Stochasticity around the branching events gives cells starting from the same state multiple possible futures.

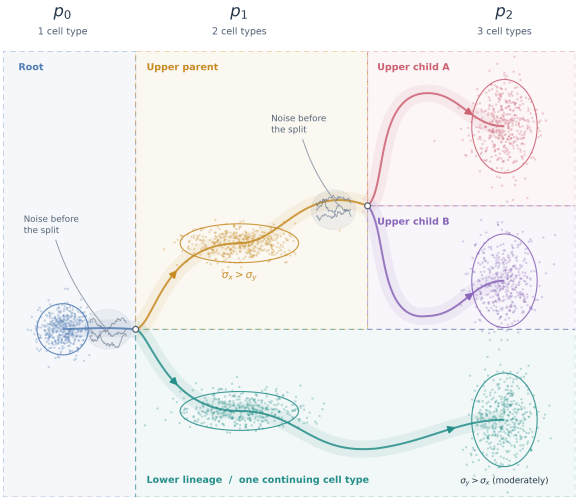


Figure 2: Schematic of the synthetic differentiation process. The initial population separates into a continuing lower lineage and an upper progenitor population, which subsequently divides into two descendants. Coloured regions indicate the cell-type partition. The point clouds, curves, and noise marks illustrate the intended geometry.

B.1 Prescribed marginal distributions

Let $Z_t = (X_t, Y_t) \in \mathbb{R}^2$, where the coordinates represent developmental progress and phenotype. We prescribe its marginal density as a mixture of three Gaussians,

$$p_t(z) = \sum_{k=1}^{3} w_k \mathcal{N}(z; m_k(t), \Sigma_t), \qquad (w_1, w_2, w_3) = (0.5, 0.3, 0.2). \tag{13}$$

The components share a time-dependent diagonal covariance

$$\Sigma_t = S_t^2, \qquad S_t = \mathrm{diag}(s_x(t), s_y(t)).$$

All coordinates and parameters below are expressed in the physical units used for evaluation.

Curved component centres. We use the continuously differentiable transition function

$$H_{a,b}(t) = \begin{cases} 0, & t \le a, \\ 3u^2 - 2u^3, \quad u = \dfrac{t-a}{b-a}, & a < t < b, \\ 1, & t \ge b. \end{cases}$$

The three centres share the progress coordinate

$$m_x(t) = 0.2 + 0.6t + 0.2t^2, \qquad m_k(t) = \begin{pmatrix} m_x(t) \\ \mu_k(t) \end{pmatrix}.$$

Define the common upper-branch trajectory

$$\mu_{\mathrm{p}}(t) = 0.32H_{0.32,0.65}(t) + 0.07H_{0.65,1.00}(t) + 0.17H_{1.00,1.38}(t).$$

The phenotype coordinates are

$$\begin{aligned} \mu_1(t) &= -0.30H_{0.32,0.65}(t) - 0.07H_{0.65,1.00}(t) \\ &\quad - 0.17H_{1.00,1.50}(t) + 0.13H_{1.50,2.00}(t), \\ \mu_2(t) &= \mu_{\mathrm{p}}(t) - 0.43H_{1.38,1.74}(t) + 0.09H_{1.74,2.00}(t), \\ \mu_3(t) &= \mu_{\mathrm{p}}(t) + 0.48H_{1.38,1.76}(t) - 0.12H_{1.76,2.00}(t). \end{aligned}$$

All three centres coincide until $t = 0.32$. The two upper centres remain identical until $t = 1.38$, when they begin to separate. At the three observation times, the centres are

$$\begin{aligned} &m_1(0) = m_2(0) = m_3(0) = (0.2, 0), \\ &m_1(1) = (1, -0.37), \qquad m_2(1) = m_3(1) = (1, 0.39), \\ &m_1(2) = (2.2, -0.41), \qquad m_2(2) = (2.2, 0.22), \quad m_3(2) = (2.2, 0.92). \end{aligned}$$

Time-dependent population shapes. The standard deviations are specified at the following knots:

Time τ_r	$s_x(\tau_r)$	$s_y(\tau_r)$
0.00	0.058	0.058
0.32	0.040	0.048
0.50	0.032	0.040
1.00	0.200	0.044
1.38	0.044	0.048
1.55	0.048	0.052
2.00	0.074	0.090

Writing $s(t) = (s_x(t), s_y(t))^\top$, we interpolate consecutive values using

$$s(t) = \big(1 - H_{\tau_r,\tau_{r+1}}(t)\big)s(\tau_r) + H_{\tau_r,\tau_{r+1}}(t)s(\tau_{r+1}), \qquad t \in [\tau_r, \tau_{r+1}].$$

This gives a continuously differentiable covariance path with positive widths throughout the interval.

B.2 Ground-truth stochastic process

We realise the prescribed density path through a Markov Itô process. Define the component responsibilities

$$r_k(t,z) = \frac{w_k\,\mathcal{N}(z; m_k(t), \Sigma_t)}{p_t(z)}.$$

The score and probability-flow velocity are

$$\nabla_z \log p_t(z) = -\sum_{k=1}^{3} r_k(t,z)\Sigma_t^{-1}\big(z - m_k(t)\big), \tag{14}$$

$$v_t(z) = \sum_{k=1}^{3} r_k(t,z)\left[\dot{m}_k(t) + \dot{S}_t S_t^{-1}\big(z - m_k(t)\big)\right]. \tag{15}$$

The first term in each component velocity moves its centre, while the second accounts for its changing width. Together, they satisfy the continuity equation

$$\partial_t p_t + \nabla \cdot (p_t v_t) = 0.$$

The ground-truth process is

$$\boxed{dZ_t = \left[v_t(Z_t) + \frac{1}{2}G_t G_t^\top \nabla \log p_t(Z_t)\right] dt + G_t\, dW_t,} \tag{16}$$

where W_t is a standard two-dimensional Brownian motion and

$$Z_0 \sim \mathcal{N}\left(\begin{pmatrix}0.2\\0\end{pmatrix}, 0.058^2 I_2\right).$$

The diffusion matrix is

$$G_t = \mathrm{diag}(0.05, g_Y(t)),$$

with

$$g_Y(t)^2 = 0.008^2 + 0.28^2 \exp\left[-\frac{1}{2}\left(\frac{t-0.29}{0.12}\right)^2\right] + 0.24^2 \exp\left[-\frac{1}{2}\left(\frac{t-1.30}{0.13}\right)^2\right].$$

The two diffusion peaks occur shortly before the corresponding component separations, allowing noise to influence branch choices.

Since G_t is independent of the state, the score correction cancels the diffusion term in the Fokker–Planck equation:

$$-\nabla \cdot \left[p_t \left(v_t + \frac{1}{2} G_t G_t^\top \nabla \log p_t \right) \right] + \frac{1}{2} \nabla \cdot \left(G_t G_t^\top \nabla p_t \right) = -\nabla \cdot (p_t v_t).$$

Thus Equation equation 16 has the marginals prescribed by Equation equation 13.

Every particle evolves under the same state- and time-dependent drift. Its state contains only its two coordinates; no persistent component assignment or terminal fate is sampled in advance. Consequently, particles at the same state and time can reach different terminal populations through different Brownian realisations.

B.3 Dataset generation and observation protocol

We simulate 12,000 independent trajectories on $[0, 2]$ using left-point Euler–Maruyama with step size $\Delta t = 0.001$. States are saved at the 21 times

$$\mathcal{T}_{\text{all}} = \{0, 0.1, 0.2, \ldots, 2.0\}.$$

At each saved time, we select 10,000 trajectory indices without replacement. A fresh subset is selected for each snapshot, and trajectory identifiers are withheld from the reconstruction dataset.

Only the three labelled marginals at

$$\mathcal{T}_{\text{train}} = \{0, 1, 2\}$$

are available for fitting and model selection. The remaining 18 marginals are reserved for evaluation. Paired trajectories and the ground-truth drift, score, and transition samples are also excluded from reconstruction training. In the known-diffusion experiments, only the prescribed diffusion coefficient is additionally supplied.

Quantity	Value
Dataset seed	210921
Brownian simulation seed	210922
Time interval	$[0, 2]$
Simulated trajectories	12,000
Euler–Maruyama increments	2,000
Integration step	0.001
Snapshot times	$0, 0.1, \ldots, 2.0$
Cells per snapshot	10,000
Training times	$0, 1, 2$
Evaluation-only snapshot times	Remaining 18 times

Observed snapshots and oracle trajectories are stored separately. Fresh simulations from a specified state and time provide reference conditional distributions for evaluation. The numerical samples approximate the continuous-time process; their accuracy depends on both simulation resolution and Monte Carlo sample size. Figure 3 shows the sampled marginal distributions, illustrating the curved branching structure and changing population shapes.

B.4 Cell types and lineage prior

We assign five cell types, ordered as

$$(\texttt{root}, \texttt{lower}, \texttt{upper}, \texttt{upper_down}, \texttt{upper_up}).$$

The root can give rise to either the lower or upper lineage. The lower lineage retains its identity, while the upper lineage can differentiate into either upper descendant.

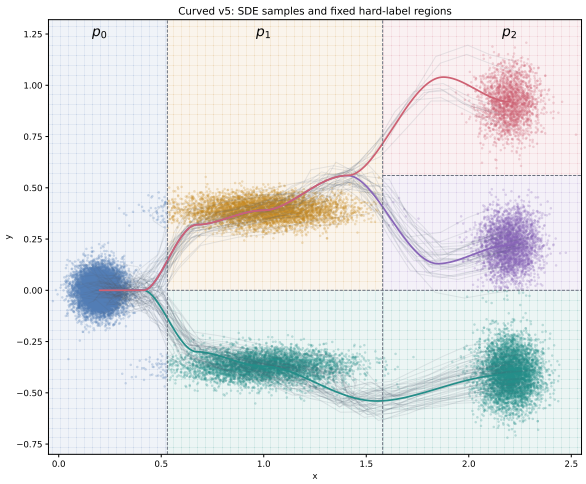


Figure 3: Synthetic branching dataset. Samples at $t \in \{0, 1, 2\}$, the three training times, are coloured by cell type. Solid curves show component centres, and faint lines show simulated trajectories.

The annotation map is the fixed spatial partition

$$\mathcal{T}(x, y) = \begin{cases} 1, & x < 0.53, \\ 2, & x \geq 0.53, \quad y < 0, \\ 3, & 0.53 \leq x < 1.58, \quad y \geq 0, \\ 4, & x \geq 1.58, \quad 0 \leq y < 0.56, \\ 5, & x \geq 1.58, \quad y \geq 0.56. \end{cases}$$

These regions cover the entire state space, including the tails of the Gaussian populations. Their boundaries are fixed as part of the dataset construction. Every observed cell is stored with its integer label and the corresponding exact one-hot vector.

In the class order above, the same-or-descendant matrix is

$$A = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

This matrix supplies the transition prior used by the hierarchical loss $\mathcal{L}_h$.

The annotations describe regions of the continuous state space. They are not additional state variables of the SDE. Gaussian tails and Brownian fluctuations can therefore produce transitions across their boundaries, including occasional transitions incompatible with the declared hierarchy.

B.5 Cell-type classifier

We train a differentiable classifier $\widetilde{D}_\alpha : \mathbb{R}^2 \to \Delta^4$ using only the labelled cells at $t \in \{0, 1, 2\}$. Its inputs are the two cell coordinates. The corresponding hard prediction is

$$D_\alpha(z) = \operatorname*{argmax}_{1 \leq i \leq 5} [\widetilde{D}_\alpha(z)]_i.$$

The available cells are split approximately $80/20$ into training and validation sets within each nonempty $(\text{time}, \text{class})$ stratum. We minimise categorical cross-entropy against the frozen one-hot labels using Adam with learning rate 10^{-2} and full-batch updates. Validation loss is checked every 50 updates. Training stops after 30 checks without an improvement of at least 10^{-6}, subject to a maximum of 20,000 updates.

The classifier was trained using this validation-loss convergence criterion. Training stopped after $8{,}400$ updates and restored the checkpoint selected at update $6{,}900$. Its observed validation accuracy was 100%. After freezing the checkpoint, its mean accuracy across the 18 omitted marginals was 99.85%, with a minimum of 97.92% at an individual time. Agreement with the fixed partition on the diagnostic spatial grid was 99.62%. These omitted-time and grid evaluations were performed after checkpoint selection.

Known hard labels are used directly for observed cells. The frozen classifier supplies differentiable predictions when evaluating the hierarchical loss on generated or intermediate states. The same annotations and classifier are shared across the compared methods.

B.6 Validation and interpretation

The construction was checked by verifying the analytic density, score, and velocity relations and by comparing simulated marginal moments with those of the prescribed Gaussian mixture. These checks support the numerical implementation of the density path and its associated SDE.

We also verified that branching remains stochastic when the conditioning state is fixed. Independent continuations from the same initial progenitor state reached all three terminal populations. Continuations from the same upper-progenitor state at $t = 1$ reached both upper descendants. Thus, the distribution of possible futures reflects randomness in the dynamics as well as variation in initial states.

The hierarchy is approximate. On the stored temporal grid with spacing 0.1, approximately 0.177% of consecutive ground-truth label transitions were incompatible with A, and 2.77% of trajectories contained at least one such transition. These are finite-grid diagnostics of the hard annotations, rather than bounds on continuous-time core-visit violations in Definition 1.

Finally, the three observed marginals and the lineage graph do not uniquely determine the curved ground-truth dynamics or their branch probabilities. The synthetic process provides a known reference against which to measure how accurately the inferred conditional laws recover those properties.

C Aditionnal Propositions

C.1 Recovering the joint distribution law of the stochastic interpolant

In this subsection, we prove Proposition 1. For this, we assume that g_θ is continuous in time, satisfies $g_\theta(0, z_0, z_1) = g_\theta(1, z_0, z_1) = 0$, and that $g_\theta(t_*, Z_0, Z_1)$ is not π-almost surely constant for some $t_* \in (0, 1)$

The key idea is that, with these assumptions, I^θ is non-Markovian. Consequently, no Markov ODE/SDE model on $\mathbb{R}^d$ can reproduce its full path law.

For a Markov process, knowing the states at two times makes observations outside that interval irrelevant to the process between them. For our interpolant, however, the conditional law between those times can still depend on the original endpoints through g_θ. We show that eliminating this dependence would force $g_\theta(t, Z_0, Z_1)$ to be almost surely constant at every time, contradicting the assumption of the proposition.

Proof of Proposition 1. Suppose, for contradiction, that I^θ is Markovian. Recall that the Brownian bridge is independent of (Z_0, Z_1), and that the endpoint conditions give $I^\theta_0 = Z_0$ and $I^\theta_1 = Z_1$ almost surely. For brevity, write

$$g_t(z_0, z_1) = g_\theta(t, z_0, z_1).$$

Fix three interior times $0 < s < t < u < 1$, and set

$$\lambda = \frac{t - s}{u - s}.$$

Conditional on the original endpoints and on $I_s^\theta = x$, $I_u^\theta = y$, the usual Gaussian conditioning formula for a Brownian bridge gives

$$\begin{aligned}&\mathrm{Law}\big(I_t^\theta \,\big|\, I_s^\theta = x,\ I_u^\theta = y,\ Z_0 = z_0,\ Z_1 = z_1\big)\\ &\quad = \mathcal{N}\left((1-\lambda)x + \lambda y + R_{s,t,u}(z_0, z_1),\ \sigma^2 \frac{(t-s)(u-t)}{u-s} I_d\right),\end{aligned}$$

where

$$R_{s,t,u}(z_0, z_1) = g_t(z_0, z_1) - (1-\lambda) g_s(z_0, z_1) - \lambda g_u(z_0, z_1).$$

Indeed, conditioning the centered bridge on its values at s and u gives linear interpolation between those values. The linear endpoint contribution in m_t^θ cancels, leaving precisely the residual $R_{s,t,u}$.

On the other hand, the Markov property at the five ordered times $0 < s < t < u < 1$ implies

$$\mathrm{Law}\big(I_t^\theta \,\big|\, I_s^\theta, I_u^\theta, Z_0, Z_1\big) = \mathrm{Law}\big(I_t^\theta \,\big|\, I_s^\theta, I_u^\theta\big).$$

Thus, once I_s^θ and I_u^θ are fixed, the conditional law displayed above cannot retain any dependence on the original endpoints.

To make this implication precise, observe that, conditional on any endpoint pair, (I_s^θ, I_u^θ) has a non-degenerate Gaussian density that is strictly positive everywhere. Bayes' formula therefore shows that the conditional distribution of (Z_0, Z_1) given $(I_s^\theta, I_u^\theta) = (x, y)$ is equivalent to π: both distributions have the same null sets. Consequently, the Gaussian conditional laws above must agree for π-almost every endpoint pair. Since their covariance matrices are identical, their means must agree as well. It follows that

$$R_{s,t,u}(Z_0, Z_1) \quad \text{is } \pi\text{-almost surely constant.}$$

Now fix the time t_* appearing in the proposition. Choose sequences

$$0 < s_n < t_* < u_n < 1, \qquad s_n \downarrow 0, \qquad u_n \uparrow 1.$$

For each n, the preceding argument shows that $R_{s_n,t_*,u_n}(Z_0, Z_1)$ is almost surely constant. By continuity of g_θ and its vanishing endpoint values,

$$R_{s_n,t_*,u_n}(Z_0, Z_1) \longrightarrow g_{t_*}(Z_0, Z_1) \qquad \text{almost surely.}$$

Taking the intersection of the probability-one sets for this countable sequence, we conclude that the limit is also almost surely constant. This contradicts the assumed nonconstancy of $g_\theta(t_*, Z_0, Z_1)$. Hence I^θ is not Markovian.

Finally, the Markov property is determined by the path law. Any Markov ODE/SDE model with the same path law as I^θ would therefore make I^θ Markovian, which is impossible. Agreement of the one-time marginals does not remove this obstruction. □

C.2 Control of predicted transition violations

In this subsection, we prove Proposition 2. The proof is as follows: the most probable class receives probability of at least $1/I$. If the transition between the two predicted classes is forbidden, their contribution to the total forbidden probability is therefore at least $1/I^2$. The logarithmic fate loss is at least this forbidden probability. Averaging this observation over the sampled transitions gives the desired bound.

Proof of Proposition 2. Fix two states $x, y \in \mathbb{R}^d$, and write

$$p = \widetilde{D}_\alpha(x), \qquad p' = \widetilde{D}_\alpha(y), \qquad i = D_\alpha(x), \qquad j = D_\alpha(y).$$

Since p and p' are probability vectors with I entries, their largest entries satisfy

$$p_i \geq \frac{1}{I}, \qquad p'_j \geq \frac{1}{I}.$$

These inequalities hold for any choice of maximiser when there are ties.

Let $q = q_{\text{valid}}(x, y) = p^\top A p'$. Because the entries of A are either zero or one,

$$1 - q = \sum_{a=1}^{I} \sum_{b=1}^{I} (1 - A_{ab}) p_a p'_b$$

is the total probability assigned to forbidden class pairs. If the predicted transition is forbidden, so that $A_{ij} = 0$, then its contribution gives

$$1 - q \geq p_i p'_j \geq \frac{1}{I^2}.$$

Consequently, for every pair of states,

$$\mathbf{1}_{\{A_{D_\alpha(x), D_\alpha(y)} = 0\}} \leq I^2 (1 - q).$$

We now use that $1 - q \leq -\log q$ for $q \in (0, 1]$. Thus,

$$\mathbf{1}_{\{A_{D_\alpha(x), D_\alpha(y)} = 0\}} \leq I^2 \big[-\log q_{\text{valid}}(x, y) \big].$$

Taking expectations over $\mathbf{X} \sim \widehat{\mathbb{P}}$ and $(s, t) \sim \mu$ yields

$$\begin{aligned} &\Pr_{\substack{\mathbf{X} \sim \widehat{\mathbb{P}} \\ (s,t) \sim \mu}} \left[A_{D_\alpha(X_s), D_\alpha(X_t)} = 0 \right] \\ &\quad \leq I^2 \mathbb{E}_{\substack{\mathbf{X} \sim \widehat{\mathbb{P}} \\ (s,t) \sim \mu}} \left[-\log q_{\text{valid}}(X_s, X_t) \right] \\ &\quad = I^2 \mathcal{L}_{\text{h}}(\widehat{\mathbb{P}}), \end{aligned}$$

which proves the proposition. □

C.3 Control of violations between the original regions

In this subsection, we prove Corollary 1.

Proof of Corollary 1. All probabilities below are taken over $\mathbf{X} \sim \widehat{\mathbb{P}}$ and $(s, t) \sim \mu$.

Let

$$\mathcal{V}_\delta = \left\{ \exists\, i, j :\ X_s \in K_i^\delta,\ X_t \in K_j^\delta,\ i \not\preceq_G j \right\}$$

be the event that the sampled transition violates the hierarchy between the original cores. Define the classification-error event

$$\mathcal{E}_{\text{cls}} = \left\{ \exists\, r \in \{s, t\},\ i \in \{1, \ldots, I\} :\ X_r \in K_i^\delta,\ D_\alpha(X_r) \neq i \right\}.$$

By definition, $\Pr(\mathcal{E}_{\text{cls}}) = \varepsilon_{\text{cls}}$.

Suppose that $\mathcal{V}_\delta$ occurs but $\mathcal{E}_{\text{cls}}$ does not. There are then indices i, j such that $X_s \in K_i^\delta$, $X_t \in K_j^\delta$, and $i \not\preceq_G j$. Since neither endpoint is misclassified,

$$D_\alpha(X_s) = i, \qquad D_\alpha(X_t) = j.$$

The definition of A therefore gives

$$A_{D_\alpha(X_s), D_\alpha(X_t)} = A_{ij} = 0.$$

Consequently,

$$\mathcal{V}_\delta \subseteq \left\{ A_{D_\alpha(X_s), D_\alpha(X_t)} = 0 \right\} \cup \mathcal{E}_{\text{cls}}.$$

Applying the union bound and then Proposition 2, we obtain

$$\begin{aligned} \Pr(\mathcal{V}_\delta) &\leq \Pr\left[A_{D_\alpha(X_s), D_\alpha(X_t)} = 0 \right] + \Pr(\mathcal{E}_{\text{cls}}) \\ &\leq I^2 \mathcal{L}_{\text{h}}(\widehat{\mathbb{P}}) + \varepsilon_{\text{cls}}. \end{aligned}$$

Finally, if $D_\alpha(x) = i$ for every $x \in K_i^\delta$ and every i, the classification-error event is empty. Hence $\varepsilon_{\text{cls}} = 0$, giving the stated special case. □

C.4 NO PREFERENCE AMONG POSSIBLE FUTURES

A lineage graph specifies which transitions are admissible, but does not specify their relative probabilities. For example, a progenitor may give rise to several descendant cell types without the graph indicating how frequently each fate occurs. Prescribing these frequencies through the hierarchical penalty would introduce information beyond the stated lineage prior. The hierarchical loss $\mathcal{L}_h$ in Definition 2 should therefore penalise incompatible transitions while leaving the allocation among admissible futures to be inferred from the observed data and the remaining modelling assumptions. In CLIFT, this allows the lineage prior to complement the matching and self-consistency objectives without specifying a preferred descendant or a target mixture of descendants.

We formalise this property for a fixed source state whose classifier output identifies a single cell type. Using the notation of Definition 2, write the pairwise contribution to $\mathcal{L}_h$ as

$$\ell_h(x, y) = -\log q_{\text{valid}}(x, y) = -\log\Big(\widetilde{D}_\alpha(x)^\top A\, \widetilde{D}_\alpha(y)\Big).$$

Proposition 3 (No preference among admissible futures)**.** *Under the setting of Definition 2, fix $x \in \mathbb{R}^d$ such that $\widetilde{D}_\alpha(x) = e_i$, where e_i is the ith standard basis vector. Let $V_i = \{j \in \{1, \dots, I\} : A_{ij} = 1\}$ denote the admissible future classes, including class i itself. Then*

$$\ell_h(x, y) = -\log\left(\sum_{j \in V_i} [\widetilde{D}_\alpha(y)]_j\right).$$

Consequently, the loss is invariant to redistributing classifier probability among admissible classes while keeping their total probability fixed. In particular, if $\widetilde{D}_\alpha(y)$ is supported on V_i, then

$$\ell_h(x, y) = 0.$$

If $\widetilde{D}_\alpha$ is also differentiable at y, then

$$\nabla_y \ell_h(x, y) = 0.$$

Proof. Since $\widetilde{D}_\alpha(x) = e_i$, compatibility reduces to

$$q_{\text{valid}}(x, y) = e_i^\top A\, \widetilde{D}_\alpha(y) = \sum_{j \in V_i} [\widetilde{D}_\alpha(y)]_j.$$

Thus, $\ell_h(x, y)$ depends only on the total classifier probability assigned to V_i, proving the invariance. When all probability is supported on V_i, this sum equals one and $\ell_h(x, y) = 0$. Since compatibility lies in $[0, 1]$, the loss is nonnegative wherever it is finite. Hence y is a global minimiser of $\ell_h(x, \cdot)$. If the classifier is differentiable at y, so is the loss, because $q_{\text{valid}}(x, y) = 1 > 0$. Its gradient therefore vanishes. □

The proposition concerns the information supplied by the hierarchical term for a fixed, unambiguous source label. With uncertain source predictions, the effective compatibility weights $A^\top \widetilde{D}_\alpha(x)$ may differ between classes admissible under the most likely source label. Moreover, a softmax classifier with finite logits assigns positive probability to every class, so exact support on V_i is an idealisation whenever forbidden classes exist. The result therefore does not guarantee that joint optimisation of the full CLIFT objective preserves relative branch probabilities; it establishes the stated invariance of the pairwise hierarchical penalty.

D A STOCHASTIC-CONTROL PERSPECTIVE ON THE HIERARCHICAL OBJECTIVE

The fate loss expresses prior information about transitions that is not determined by the observed marginals alone. This appendix provides a complementary variational perspective on how such information can shape a stochastic process. Starting from a reference diffusion, we consider a path

law that balances expected fate loss against departure from the reference, while satisfying prescribed endpoint marginals.

In a KL-regularised stochastic-control formulation, departure from the reference path law is measured by relative entropy, which corresponds to a quadratic control-energy cost under Girsanov's theorem. The fate term favours lineage-compatible transitions, while the marginal constraints retain agreement with the observations. This separates the information supplied by the observed distributions from the additional preference imposed on the transition structure.

We first characterise the optimal law for this constrained variational problem. We then examine the consequences of the temporal structure of the fate cost. Because the cost couples states at different times, the corresponding optimal control can depend on the trajectory's history. For a linear compatibility cost and a factorised time-pair distribution, we show that this dependence admits a finite-dimensional augmented-state representation.

This formulation clarifies the modelling role of the fate preference. CLIFT incorporates that preference by jointly optimising the fate, matching, and self-consistency objectives through its flow-map representation. These objectives are distinct from the path-space KL criterion considered here; the optimality and exact marginal constraints of the control formulation therefore concern the stated variational problem.

D.1 Framework and optimal solution

We consider population marginals on the interval $[0, 1]$. On the canonical path space $\Omega = C([0, 1]; \mathbb{R}^d)$, let X be the coordinate process and $(\mathcal{F}_t)$ its natural filtration, with the usual augmentation for stochastic calculus. Let $\mathbb{P}^{\mathrm{ref}}$ denote the law of a reference diffusion

$$dX_t = b(t, X_t)\, dt + \sigma_t\, dW_t, \qquad X_0 \sim p_0.$$

The reference may be obtained through marginal matching. For the endpoint-preserving setting, we assume that $\mathbb{P}_1^{\mathrm{ref}} = p_1$. We take b to be measurable in time, uniformly Lipschitz in space, and of at most linear growth, and assume that σ_t is deterministic and scalar, with

$$0 < \underline{\sigma} \le \sigma_t \le \overline{\sigma} < \infty.$$

A progressively measurable control u_t, adapted to the state filtration and allowed to depend on its history, modifies the dynamics according to

$$dX_t^u = \left[b(t, X_t^u) + \sigma_t^2 u_t\right] dt + \sigma_t\, dW_t^u, \qquad X_0^u \sim p_0.$$

We write $\mathbb{P}^u$ for the controlled path law. Admissible controls satisfy the absolute-continuity, martingale, and finite-energy conditions required for Girsanov's theorem. Under these conditions, the standard entropy–energy identity gives (Léonard, 2014, Section 4)

$$\mathrm{KL}\left(\mathbb{P}^u \| \mathbb{P}^{\mathrm{ref}}\right) = \frac{1}{2}\mathbb{E}_{\mathbb{P}^u}\left[\int_0^1 \sigma_t^2 \|u_t\|^2\, dt\right]. \tag{17}$$

Let $h = \widetilde{D}_\alpha$ be the fixed classifier's probability output and let A encode the permitted transitions, as in Definition 2. Let μ be a probability measure on ordered time pairs, sampled independently of the process. We distinguish the pathwise fate cost

$$\ell_{\mathrm{fate}}(X) = \int_{0 \le s < t \le 1} -\log\big(h(X_s)^\top A h(X_t)\big)\ \mu(ds, dt)$$

from its expectation under a path law:

$$\mathcal{L}_{\mathrm{fate}}(\mathbb{Q}) = \mathbb{E}_{\mathbb{Q}}\left[\ell_{\mathrm{fate}}(X)\right].$$

The associated endpoint-preserving control problem is

$$\inf_{u:\, \mathbb{P}_1^u = p_1} \left\{\mathcal{L}_{\mathrm{fate}}(\mathbb{P}^u) + \mathrm{KL}\left(\mathbb{P}^u \| \mathbb{P}^{\mathrm{ref}}\right)\right\}. \tag{18}$$

The fate term favours compatible transitions, the KL term penalises departure from the reference process, and the terminal constraint retains the prescribed final marginal. The initial marginal remains fixed by construction.

Optimal path law. The following characterisation applies the classical Gibbs variational principle and entropy minimisation with marginal constraints to the fate cost (Léonard, 2014); the variational representation of Brownian path functionals by drift control is also classical (Boué & Dupuis, 1998). We include the derivation to specify the conditional normalisation and our control convention.

Assume that ℓ_{fate} is measurable and finite $\mathbb{P}^{\mathrm{ref}}$-almost surely. For a terminal potential $\Phi : \mathbb{R}^d \to \mathbb{R}$, define

$$A_\Phi(x) = \log \mathbb{E}_{\mathbb{P}^{\mathrm{ref}}} \left[\exp(-\ell_{\mathrm{fate}}(X) - \Phi(X_1)) \,|\, X_0 = x \right],$$

and set

$$\mathcal{D}(\Phi) = \mathbb{E}_{X_0 \sim p_0}[A_\Phi(X_0)] + \mathbb{E}_{Y \sim p_1}[\Phi(Y)].$$

We restrict to potentials for which the conditional exponential expectation is positive and finite p_0-almost surely, $A_\Phi \in L^1(p_0)$, and $\Phi \in L^1(p_1)$.

Proposition 4 (Standard Gibbs characterisation for the fate cost)**.** *Suppose that $\mathcal{D}$ admits a minimiser $\Phi^\star$ over an admissible class of potentials stable under bounded measurable perturbations, and that the tilted law below has finite relative entropy and finite expected fate cost. Then the path law*

$$\frac{d\mathbb{Q}^\star}{d\mathbb{P}^{\mathrm{ref}}}(X) = \exp(-\ell_{\mathrm{fate}}(X) - \Phi^\star(X_1) - A_{\Phi^\star}(X_0)) \tag{19}$$

satisfies

$$\mathbb{Q}_0^\star = p_0, \qquad \mathbb{Q}_1^\star = p_1.$$

It is the unique minimiser of

$$\mathcal{J}(\mathbb{Q}) = \mathbb{E}_{\mathbb{Q}}[\ell_{\mathrm{fate}}(X)] + \mathrm{KL}\left(\mathbb{Q} \| \mathbb{P}^{\mathrm{ref}}\right)$$

over path laws with these endpoints and finite objective.
Define

$$V^\star(t, X_{[0,t]}) = \log \mathbb{E}_{\mathbb{P}^{\mathrm{ref}}} \left[\exp(-\ell_{\mathrm{fate}}(X) - \Phi^\star(X_1)) \,|\, \mathcal{F}_t \right].$$

If $V^\star$ admits a sufficiently regular non-anticipative version for the functional Itô formula, and the resulting control is admissible, then $\mathbb{Q}^\star$ solves Equation 18, with

$$\boxed{u_t^\star = \nabla_x^{\mathrm{vert}} V^\star(t, X_{[0,t]})} \qquad dt \otimes \mathbb{Q}^\star\text{-almost everywhere}. \tag{20}$$

Here, ∇_x^{vert} is the vertical derivative: it differentiates with respect to the current endpoint while holding the preceding path fixed, as in functional Itô calculus (Cont & Fournié, 2013).

Proof. For an admissible potential Φ, define

$$\frac{d\mathbb{Q}^\Phi}{d\mathbb{P}^{\mathrm{ref}}}(X) = \exp(-\ell_{\mathrm{fate}}(X) - \Phi(X_1) - A_\Phi(X_0)).$$

The definition of A_Φ gives

$$\mathbb{E}_{\mathbb{P}^{\mathrm{ref}}} \left[\frac{d\mathbb{Q}^\Phi}{d\mathbb{P}^{\mathrm{ref}}} \,\middle|\, X_0 \right] = 1.$$

Consequently, $\mathbb{Q}^\Phi$ is a probability law and $\mathbb{Q}_0^\Phi = p_0$.

For any bounded measurable function g,

$$\frac{d}{d\varepsilon} \mathcal{D}(\Phi + \varepsilon g)\Big|_{\varepsilon=0} = -\mathbb{E}_{\mathbb{Q}^\Phi}[g(X_1)] + \mathbb{E}_{Y \sim p_1}[g(Y)].$$

At $\Phi^\star$, this derivative vanishes for every such g, implying $\mathbb{Q}_1^{\Phi^\star} = p_1$. Thus $\mathbb{Q}^\star = \mathbb{Q}^{\Phi^\star}$ is feasible.

Let $\mathbb{Q}$ be any other path law with the prescribed endpoints and finite objective. The density of $\mathbb{Q}^\star$ yields

$$\begin{aligned} \mathrm{KL}(\mathbb{Q} \| \mathbb{Q}^\star) = \mathrm{KL}(\mathbb{Q} \| \mathbb{P}^{\mathrm{ref}}) + \mathbb{E}_{\mathbb{Q}}[\ell_{\mathrm{fate}}(X)] \\ + \mathbb{E}_{Y \sim p_1}[\Phi^\star(Y)] + \mathbb{E}_{X_0 \sim p_0}[A_{\Phi^\star}(X_0)]. \end{aligned}$$

Hence

$$\mathcal{J}(\mathbb{Q}) = \mathrm{KL}(\mathbb{Q}\|\mathbb{Q}^\star) - \mathcal{D}(\Phi^\star).$$

Applying the same identity to $\mathbb{Q}^\star$ gives

$$\mathcal{J}(\mathbb{Q}) - \mathcal{J}(\mathbb{Q}^\star) = \mathrm{KL}(\mathbb{Q}\|\mathbb{Q}^\star) \geq 0.$$

Equality holds only when $\mathbb{Q} = \mathbb{Q}^\star$, which establishes optimality and uniqueness.

To identify the control, consider the density process

$$\Lambda_t = \mathbb{E}_{\mathbb{P}^{\mathrm{ref}}}\left[\frac{d\mathbb{Q}^\star}{d\mathbb{P}^{\mathrm{ref}}}\,\middle|\,\mathcal{F}_t\right] = \exp\big(V^\star(t, X_{[0,t]}) - A_{\Phi^\star}(X_0)\big)\,.$$

It is a positive martingale. Applying the functional Itô formula (Cont & Fournié, 2013, Theorem 4.1) and using its martingale property gives

$$d\Lambda_t = \Lambda_t\,\sigma_t\left(\nabla_x^{\mathrm{vert}} V^\star(t, X_{[0,t]})\right)^\top dW_t.$$

Girsanov's theorem therefore implies that

$$W_t^\star = W_t - \int_0^t \sigma_s \nabla_x^{\mathrm{vert}} V^\star(s, X_{[0,s]})\,ds$$

is Brownian motion under $\mathbb{Q}^\star$. Substitution into the reference SDE gives

$$dX_t = \left[b(t, X_t) + \sigma_t^2 \nabla_x^{\mathrm{vert}} V^\star(t, X_{[0,t]})\right] dt + \sigma_t\, dW_t^\star.$$

Comparison with the controlled parameterisation proves Equation 20. □

Interpretation. The initial-state normalisation $A_{\Phi^\star}(X_0)$ preserves p_0, while the optimal terminal potential enforces p_1. These roles are distinct from that of the fate cost, which changes the relative preference among trajectories. For several prescribed observation times, the same construction introduces one potential per noninitial observed marginal, replacing $\Phi(X_1)$ by $\sum_k \Phi_k(X_{t_k})$, subject to the corresponding integrability and dual-attainment assumptions.

The proposition characterises the optimum of the stated regularised criterion. It does not identify the unknown data-generating process or equate this criterion with CLIFT's matching and self-consistency objectives.

D.2 Markovian lifting for a linear fate cost

The reference diffusion is Markov, but its optimal controlled law need not be Markov in X_t alone. The pairwise fate cost couples future states to earlier classifier outputs, so the value functional can depend on the past trajectory.

A linear variant illustrates when this history can be summarised by a finite-dimensional accumulated state. In this subsection only, replace the logarithmic cost by $f(q) = 1 - q$. The identity below is a direct calculation for this particular compatibility score. Once the state is augmented, the control argument uses standard dynamic programming (Fleming & Soner, 2006). We retain the logarithmic fate loss in the definition and training of CLIFT.

Lemma 1 (Augmented-state representation of the linear fate cost)**.** *Let $h = \widetilde{D}_\alpha : \mathbb{R}^d \to \Delta^{I-1}$, and suppose that*

$$\mu(ds, dt) = w_-(s) w_+(t) \mathbf{1}_{\{0 \leq s < t \leq 1\}}\, ds\, dt,$$

where $w_-, w_+ \in L^1([0,1])$ are nonnegative and

$$\int_0^1 \int_0^t w_-(s) w_+(t)\, ds\, dt = 1.$$

Define

$$\ell_{\mathrm{fate}}^{\mathrm{lin}}(X) = \int_{s<t} \left[1 - h(X_s)^\top A h(X_t)\right] \mu(ds, dt)$$

and

$$Z_t = \int_0^t w_-(s)h(X_s)\,ds.$$

Then

$$\boxed{\ell^{\mathrm{lin}}_{\mathrm{fate}}(X) = 1 - \int_0^1 w_+(t)Z_t^\top Ah(X_t)\,dt}. \tag{21}$$

The associated controlled dynamics admit the augmented representation

$$dX_t^u = \left[b(t,X_t^u) + \sigma_t^2 u(t,X_t^u,Z_t^u)\right]dt + \sigma_t\,dW_t^u,$$
$$dZ_t^u = w_-(t)h(X_t^u)\,dt, \qquad Z_0^u = 0.$$

Under the admissibility and duality assumptions of Proposition 4, applied to the linear cost, assume that the augmented control problem is well posed and satisfies the regularity and admissibility conditions for the feedback verification argument below. Then an optimal control can be chosen as

$$u_t^\star = u^\star(t,X_t^\star,Z_t^\star),$$

where $(X^\star, Z^\star)$ denotes the state under this optimal control. Thus the control is Markovian in the augmented state.

Proof. Since μ is a probability measure, the constant part of the linear cost integrates to one. For the compatibility term, Fubini's theorem gives

$$\int_0^1\int_0^t w_-(s)w_+(t)h(X_s)^\top Ah(X_t)\,ds\,dt$$
$$= \int_0^1 w_+(t)\left[\int_0^t w_-(s)h(X_s)\,ds\right]^\top Ah(X_t)\,dt.$$

This proves Equation 21.

The accumulated state is absolutely continuous, with

$$dZ_t = w_-(t)h(X_t)\,dt.$$

Consequently, under a feedback control $u(t,x,z)$, the augmented dynamics depend only on the current time and state (X_t, Z_t). The linear fate cost becomes an additive running functional of that state. The control-energy penalty in Equation 17 is also additive.

More explicitly, write

$$a(t,x,z) = w_+(t)z^\top Ah(x),$$

and let $\Phi^\star_{\mathrm{lin}}$ be an optimal terminal potential for the problem with linear fate cost. Define

$$v^\star(t,x,z) = \log \mathbb{E}^{\mathrm{ref}}_{t,x,z}\left[\exp\left(\int_t^1 a(r,X_r,Z_r)\,dr - \Phi^\star_{\mathrm{lin}}(X_1)\right)\right],$$

where $\mathbb{E}^{\mathrm{ref}}_{t,x,z}$ refers to the uncontrolled augmented process started at $(X_t,Z_t) = (x,z)$. The constant term in the linear fate loss does not affect the optimal control.

The Markov property of the uncontrolled augmented process gives the decomposition

$$V^\star_{\mathrm{lin}}(t,X_{[0,t]}) = -1 + \int_0^t a(r,X_r,Z_r)\,dr + v^\star(t,X_t,Z_t),$$

where $V^\star_{\mathrm{lin}}$ is the value functional of Proposition 4 for the linear cost. The accumulated integral and Z_t are unchanged by a perturbation at the single current time, so the vertical derivative is $\nabla_x v^\star(t,X_t,Z_t)$.

Equivalently, the standard logarithmic transformation and dynamic-programming argument (Fleming & Soner, 2006, Chapters IV and VI) identify $v^\star$ as the value for the running reward a, terminal

reward $-\Phi^\star_{\mathrm{lin}}$, and quadratic control penalty. When $v^\star$ is sufficiently regular, the control-dependent part of its Hamilton–Jacobi–Bellman equation is

$$\sup_u \left\{ \sigma_t^2 u^\top \nabla_x v^\star(t,x,z) - \frac{\sigma_t^2}{2}\|u\|^2 \right\}.$$

Completing the square shows that its maximiser is

$$u^\star(t,x,z) = \nabla_x v^\star(t,x,z).$$

The optimal feedback therefore depends only on the current augmented state. □

Example and scope. If two independent uniform times are sampled and ordered, their joint density is $2\mathbf{1}_{\{0\le s<t\le 1\}}$. The lemma applies with $w_- = 1$ and $w_+ = 2$, giving

$$Z_t = \int_0^t h(X_s)\, ds.$$

This state accumulates the classifier outputs along the past trajectory and remains finite throughout the interval.

For the logarithmic cost used by CLIFT, the contribution of past classifier outputs cannot generally be recovered from the accumulated vector Z_t alone. The lemma therefore does not provide the same reduction for that objective. Likewise, a positive minimum separation $t \ge s + \Delta$ introduces a delayed accumulated state, which generally requires additional history.

The result establishes a Markov representation in (X_t, Z_t) for the stated linear special case. It does not imply that the corresponding process is Markov in the original state X_t alone.

E Results in Stochasticity

This section summarises some important results.

Proposition 5 (Fokker-Planck Equation)**.** *Consider the Itô process: $dX_t = \mu(t, X_t)dt + \sigma(t, X_t)dW_t$. The probability density $p(x,t)$ for X_t satisfies the Fokker-Planck equation:*

$$\frac{\partial p(x,t)}{\partial t} = -\sum_{i=1}^{N} \frac{\partial}{\partial x_i}[\mu_i(x,t)p(x,t)] + \sum_{i=1}^{N}\sum_{j=1}^{N} \frac{\partial^2}{\partial x_i x_j}[D_i j(x,t)p(x,t)]$$

with diffusion tensor $D = \frac{1}{2}\sigma\sigma^t$.

Corollary 2 (Continuity Equation)**.** *Consider the Itô process with coefficient term null, which transform the SDE into an SDE:*

$$\begin{cases} \frac{dX_t}{dt} = u(t, X_t) \\ X_0 \sim p_0 \end{cases}$$

The marginals $(p_t(x))_t$ follow the continuity equation:

$$\frac{\partial p(t,x)}{\partial t} + \sum_{i=1}^{N} \frac{\partial}{\partial x_i}[u_i(x,t)p(x,t)] = 0$$

With operator:

$$\frac{\partial p(t,x)}{\partial t} + \nabla \cdot (p(t,x)u(t,x)) = 0$$

Corollary 3 (ODE-SDE equivalence via the Score)**.** *Consider the ODE:*

$$\begin{cases} \frac{dX_t}{dt} = u(t, X_t) \\ X_0 \sim p_0 \end{cases} \tag{22}$$

Under mild assumptions on $u : (t, x) \to u(t, x)$, this equation defines a path measure. Let be (p_t) its marginals.
Now, consider the SDE, with the nonnegative diffusion schedule $\eta : [0, 1] \to \mathbb{R}_+^*$ can be any function:

$$\begin{cases} dX_t = [u(t, X_t) + \frac{\eta(t)^2}{2} \nabla_x \log p_t(X_t)]dt \ + \ \eta(t)dW_t \\ X_0 \sim p_0 \end{cases} \tag{23}$$

Both path measures have the same marginals (p_t). This can be seen by applying the Fokker-Planck Equation.
Consequently, as each process defined in eq 22 and eq 23 are Markovian process, the only difference is their 2-dimensional transition kernels $p_{s,t}(x, y)$. Choosing between either formulation hence depends on which transition kernels one think better represents the studied problem.

F Experimental Plots

F.1 Synthetic Dataset: Observing Condtional Laws

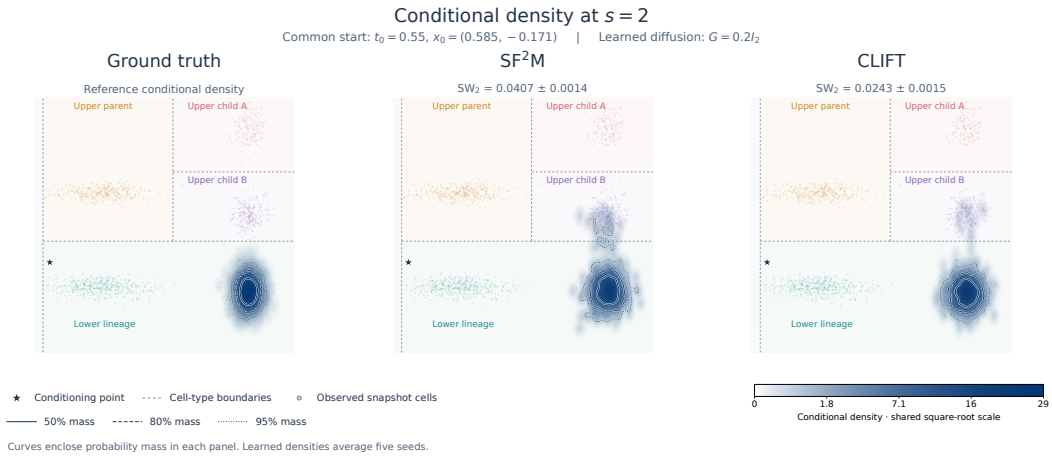
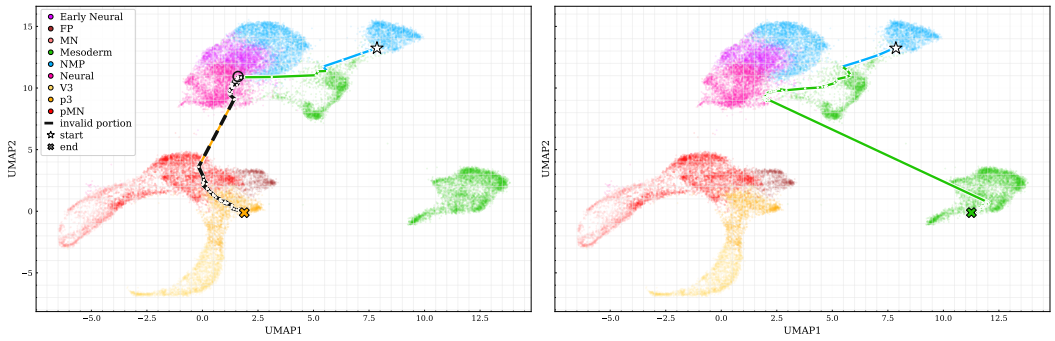


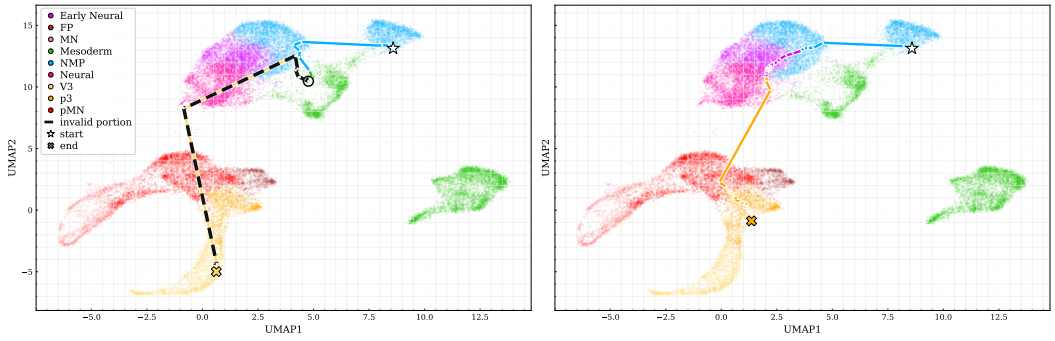
Figure 4: Conditional densities at $s = 2$ from $t_0 = 0.55$, $x_0 = (0.585, -0.171)$ for the ground-truth SDE, SF^{2}M, and CLIFT. Both learned methods use $G = 0.2I_2$. Heatmaps average five training seeds on a shared colour scale; contours enclose 50%, 80%, and 95% probability mass. SW_2 annotations report mean $\pm$ standard deviation across seeds. We can see that CLIFT$_S$ concentrates better around the true fate of the cell, and places less mass in the adjacent, forbidden, fate region than SF^{2}M.

F.2 Trajectory plots

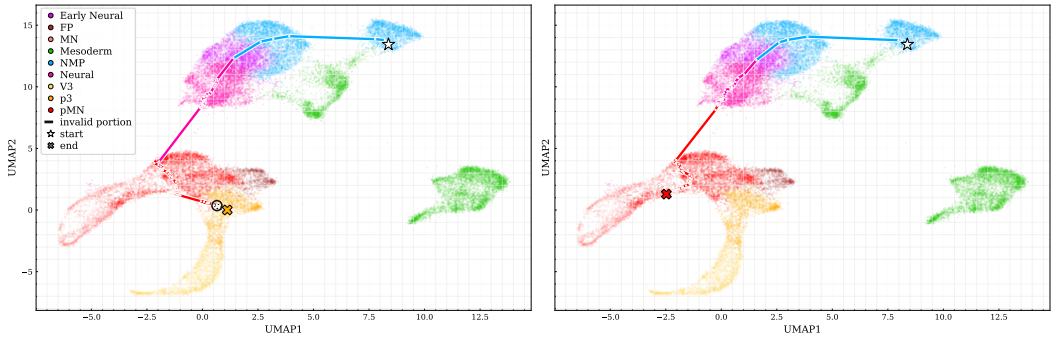
Here, we plot some trajectories on the UMAP of Maizels, Cite, and Multi data. We plot all datapoints, coloured by cell type, and select initial says from the base distribution where the OT flow map leads to a violation. We also plot the same trajectory, but from CLIFT. In general, we can see that the methods produce similar trajectories up to the point where the OT flow map first violates, and CLIFT is able to avoid making the same violation, steering the trajectory the a feasible fate.



(a) OT-FMap produces invalid transition: Mesoderm → Neural



(b) OT-FMap produces invalid transition: Mesoderm → V3



(c) OT-FMap produces invalid transition: pMN → p3

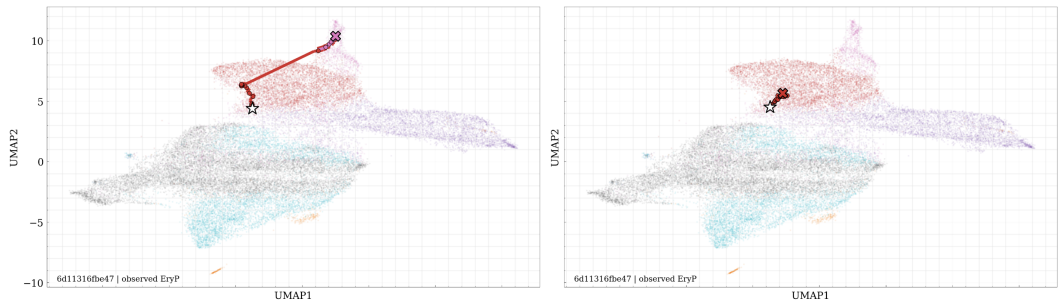
Figure 5: Comparison of selected OT-FMap and CLIFT trajectories on a 2D UMAP of the Maizels data. For plots from OT-FMap, the first invalid transition is circled, and the ensuing infeasible trajectory is dotted in black.

G CLIFT Algorithms

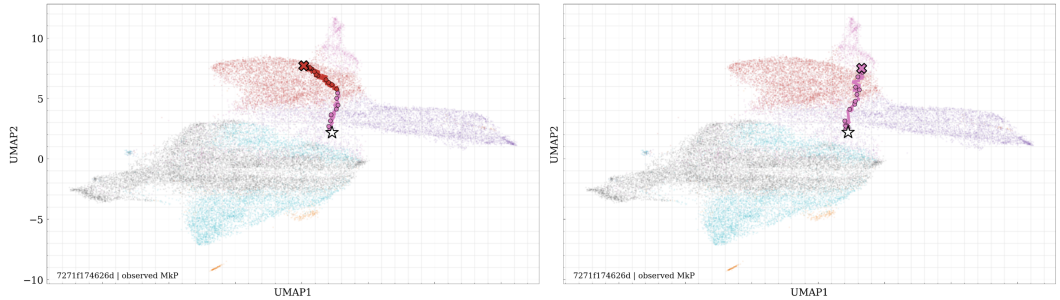
G.1 Fine-tuning for endpoint reconstruction

The CLIFT training objective does not explicitly enforce equality between the model-generated and observed marginals, and therefore does not guarantee exact reconstruction of the observed populations. In our real-world experiments, CLIFT nevertheless achieves competitive W_1 on observed marginals without additional marginal fine-tuning (Table 8).

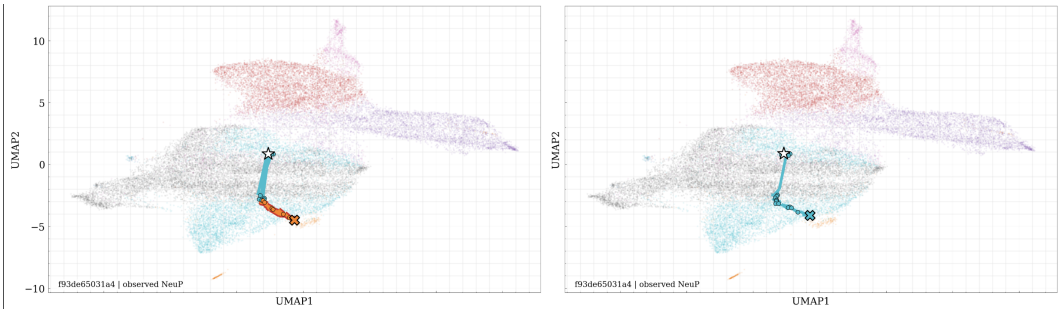
When closer agreement with the observations is desired, we can add a short fine-tuning stage. We describe this procedure for the three observed marginals p_0, p_1, p_2 of the synthetic experiment. We initialise the model from the trained stochastic flow map and retain a frozen copy as a reference. Each update samples a batch from p_0 and generates trajectories through times 1 and 2, continu-



(a) OT produces invalid transition: EryP → MkP



(b) OT produces invalid transition: MkP → EryP



(c) OT produces invalid transition: NeuP → MoP

Figure 6: Comparison of selected OT flow map and constrained flow map trajectories on Cite data.

ing from the generated states at time 1. Both endpoint discrepancies therefore account for errors accumulated along the trajectory.

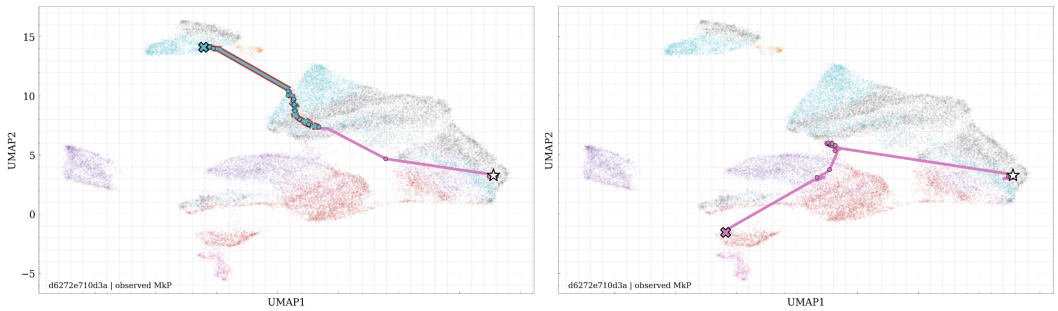
At each observed endpoint, a scalar neural critic f_k, $k \in \{1, 2\}$, learns to distinguish generated states from independent samples of p_k. Writing $\widehat{p}_k^{\theta}$ for the generated marginal, the critic maximises

$$\Delta_k(\theta, f_k) = \mathbb{E}_{X \sim \widehat{p}_k^{\theta}}[f_k(X)] - \mathbb{E}_{Y \sim p_k}[f_k(Y)].$$

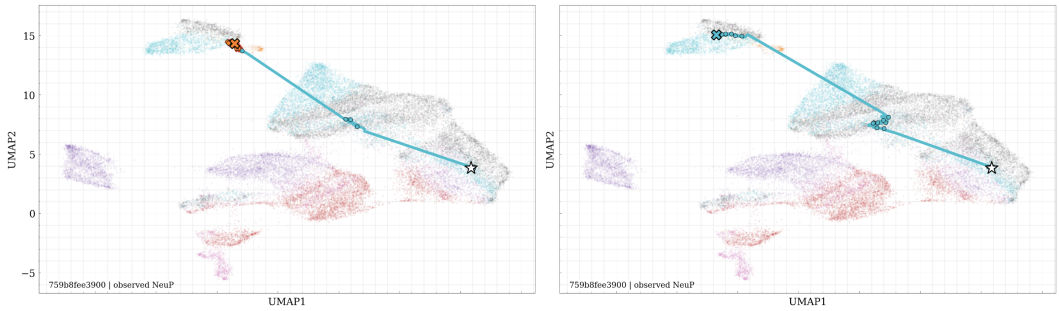
Spectral normalisation and a penalty on gradients exceeding unit norm in standardised coordinates regularise the critics, providing a neural surrogate for a Wasserstein discrepancy. With the critics fixed, we differentiate through the generated trajectories to update the flow map using

$$\mathcal{L}_{\text{ft}}(\theta) = \frac{1}{2} \sum_{k=1}^{2} w_k \Delta_k(\theta, f_k) + \mathcal{R}(\theta; \theta_{\text{ref}}) + \mathcal{L}_{\text{cons}}(\theta).$$

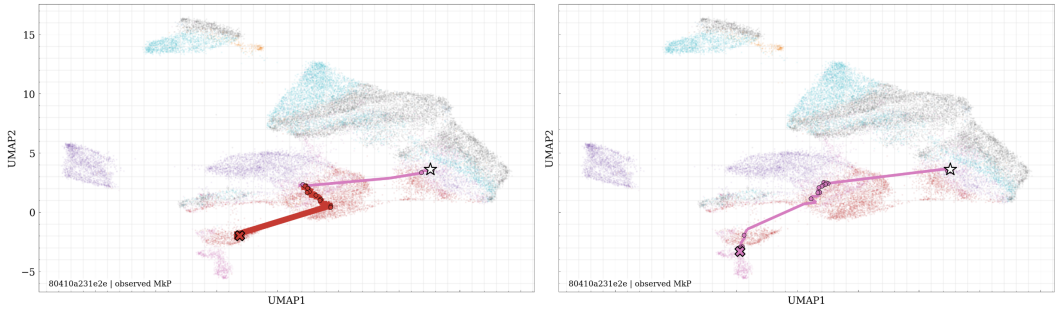
The nonnegative weights w_k adapt to smoothed endpoint gaps relative to small tolerances estimated from sampling variability, increasing pressure on persistent discrepancies. The regulariser $\mathcal{R}$ discourages departure from the trained map by separately penalising squared changes in its drift and noise-dependent finite-step response, weighted by the inverse diffusion covariance and step size. These responses are compared at the same states, times and Brownian inputs along the current trajectories. We also retain the map's self-consistency loss $\mathcal{L}_{\text{cons}}$, which compares a direct transition



(a) OT produces invalid transition: MkP → NeuP



(b) OT produces invalid transition: NeuP → MoP



(c) OT produces invalid transition: MkP → EryP

Figure 7: Comparison of selected OT flow map and constrained flow map trajectories on Multi data.

with two composed transitions. The original matching and hierarchical training losses are omitted during this stage; hierarchical compatibility remains a checkpoint-selection safeguard.

Checkpoint selection uses only validation cells from the observed marginals. We seek an improvement in mean W_1 at times 1 and 2, supported by a majority of three fixed validation draws. We additionally require that the individual endpoint W_1 and W_2 values, including those from a separate rollout starting at observed p_1, and the hierarchical and self-consistency losses remain within preset tolerances of the original model. If no update is accepted, we retain the original checkpoint. Neither omitted-time populations nor ground-truth conditional laws enter fitting or checkpoint selection. This procedure provides an approximate marginal correction, rather than a guarantee of exact marginal equality or unchanged conditional laws.

Since fine-tuning starts from an already trained stochastic flow map, we can use large time steps to reduce the number of function evaluations (NFE) along the differentiable rollout. For example, two map steps per observed interval require four map evaluations along a complete $p_0 \to p_1 \to p_2$ trajectory; reference and consistency terms require additional evaluations. Fine-tuning uses a fixed budget of 75 map updates, with batch size 256 and learning rate 10^{-5}, following 200 critic warm-up updates and using five critic updates per map update. Checkpoints are assessed every five updates using 50 map steps per observed interval. The reported marginal distances use a finer grid of 4,160

Algorithm 1 Constrained deterministic flow map training

Require: Marginals $\{p_{t_k}\}_{k\in[K]}$, coupling π, interpolant I_t, diagonal fraction η, constraint weight λ, batch size B.

Initialise θ
while not converged **do**
  For each $i = 1, \dots, \lfloor \eta B \rfloor$, sample $k^i \sim [K-1]$, $(\mathbf{x}^i_{t_{k^i}}, \mathbf{x}^i_{t_{k^i+1}}) \sim \pi$ and $r^i \sim U[t_{k^i}, t_{k^i+1}]$
  Compute $\mathcal{L}_{\text{diag}} = \frac{1}{\lfloor \eta B \rfloor} \sum_i \left\| \Phi_\theta(r^i, r^i, \mathbf{I}_{r^i}) - \dot{\mathbf{I}}_{r^i} \right\|^2$
  For each $i = 1, \dots, B - \lfloor \eta B \rfloor$, sample $k^i \sim [K-1], (\mathbf{x}^i_{t_{k^i}}, \mathbf{x}^i_{t_{k^i+1}}) \sim \pi, (u^i, v^i) \overset{\text{iid}}{\sim} U[t_{k^i}, t_{k^i+1}]$, and set $s^i = \min\{u^i, v^i\}, t^i = \max\{u^i, v^i\}$
  Compute $\mathcal{L}_{\text{off}} = \frac{1}{B - \lfloor \eta B \rfloor} \sum_i \left\| \partial_t \Phi_\theta(s^i, t^i, \mathbf{I}_{s^i}) - \text{sg}\left[\Phi_{\bar\theta}\left(t^i, t^i, \text{sg}[\Phi_\theta(s^i, t^i, \mathbf{I}_{s^i})]\right)\right] \right\|^2$
  Compute $\mathcal{L}_{\text{fate}} = \frac{1}{B - \lfloor \eta B \rfloor} \sum_i \mathcal{C}(\mathbf{I}_{s^i}, \Phi_\theta(s^i, t^i, \mathbf{I}_{s^i}))$
  Compute $\mathcal{L} = \mathcal{L}_{\text{diag}} + \mathcal{L}_{\text{off}} + \lambda \mathcal{L}_{\text{con}}$
  Update θ with $\nabla \mathcal{L}$.
end while

Algorithm 2 Constrained stochastic flow map training

Require: Marginals $\{p_{t_k}\}_{k\in[K]}$, coupling π, stochastic interpolant I_t, local fraction η, constraint weight λ, batch size B.

Initialise θ and EMA parameters $\bar\theta \leftarrow \theta$
while not converged **do**
  For each $i = 1, \dots, \lfloor \eta B \rfloor$, sample $k^i \sim [K-1]$, $(\mathbf{x}^i_{t_{k^i}}, \mathbf{x}^i_{t_{k^i+1}}) \sim \pi$, a short horizon $h^i > 0$, and $r^i \sim U[t_{k^i}, t_{k^i+1} - h^i]$
  Sample Brownian coefficients $\mathbf{L}^i_{r^i, r^i+h^i}$
  Compute the local Euler–Maruyama target $\mathbf{Y}^i = \mathbf{I}_{r^i} + h^i(\dot{\mathbf{I}}_{r^i} + \frac{1}{2} G^2_{r^i} \frac{-\epsilon}{\rho(r^i)}) + G_{r^i} \mathbf{W}^i_{r^i, r^i+h^i}$
  Compute $\mathcal{L}_{\text{diag}} = \frac{1}{\lfloor \eta B \rfloor} \sum_i \frac{1}{h^i} \left\| \Psi_\theta\left(r^i, r^i + h^i, I_{r^i}, \mathbf{L}^i_{r^i, r^i+h^i}\right) - \mathbf{Y}^i \right\|^2$
  For each $i = 1, \dots, B - \lfloor \eta B \rfloor$, sample $k^i \sim [K-1]$, $(\mathbf{x}^i_{t_{k^i}}, \mathbf{x}^i_{t_{k^i+1}}) \sim \pi$, $(u^i, v^i) \overset{\text{iid}}{\sim} U[t_{k^i}, t_{k^i+1}]$, and set $s^i = \min\{u^i, v^i\}$, $t^i = \max\{u^i, v^i\}$ and $m^i = (s^i + t^i)/2$
  Sample Brownian coefficients $\mathbf{L}^i_{s^i, m^i}$ and $\mathbf{L}^i_{m^i, t^i}$, and compute $\mathbf{L}^i_{s^i, t^i} = \mathbf{L}^i_{s^i, m^i} \star \mathbf{L}^i_{m^i, t^i}$
  Compute the EMA two-step target $\bar{\mathbf{Z}}^i_{t^i} = \Psi_{\bar\theta}\left(m^i, t^i, \Psi_{\bar\theta}\left(s^i, m^i, \mathbf{I}_{s^i}, \mathbf{L}^i_{s^i, m^i}\right), \mathbf{L}^i_{m^i, t^i}\right)$
  Compute $\mathcal{L}_{\text{off}} = \frac{1}{B - \lfloor \eta B \rfloor} \sum_i \frac{1}{t^i - s^i} \left\| \Psi_\theta\left(s^i, t^i, \mathbf{I}_{s^i}, \mathbf{L}^i_{s^i, t^i}\right) - \text{sg}[\bar{\mathbf{Z}}^i_{t^i}] \right\|^2$
  Compute $\mathcal{L}_{\text{con}} = \frac{1}{B - \lfloor \eta B \rfloor} \sum_i \mathcal{C}\left(\mathbf{I}_{s^i}, \Psi_\theta\left(s^i, t^i, \mathbf{I}_{s^i}, \mathbf{L}^i_{s^i, t^i}\right)\right)$
  Compute $\mathcal{L} = \mathcal{L}_{\text{diag}} + \mathcal{L}_{\text{off}} + \lambda \mathcal{L}_{\text{con}}$
  Update θ with $\nabla \mathcal{L}$, and update EMA parameters $\bar\theta$
end while

steps per interval, with a 2,080-step refinement check. We applied this stage only to the synthetic experiment with $G = 0.2 I_2$, as this was the only setting in which CLIFT's W_1 on the observed endpoint marginals was statistically higher than that of the baselines before fine-tuning. In a single-seed experiment, using just two map steps per observed interval during fine-tuning reduced mean observed-marginal W_1 to slightly below that of SF^{2}M for the same seed. Held-out-marginal W_1 also decreased, from 0.1332 to 0.1306, compared with 0.1345 for SF^{2}M. Fate TV and conditional SW_2 remained essentially unchanged. These are descriptive results from one seed.

The five-seed results in Table 1 use twelve fine-tuning steps per observed interval, selected using observed validation data. This reduces mean observed-marginal W_1 by 13.43% to 0.0346, close to the SF^{2}M and SSFM means of 0.0342, while achieving a lower mean held-out-marginal W_1. Fate TV decreases by 0.10% and conditional SW_2 increases by 0.06% relative to the original CLIFT models. All five seeds are included, including one that retains its original checkpoint. These results

show that marginal reconstruction can be improved with little change in the metrics evaluating the conditional laws.

H Extended Related Works

Method	Coupling	Interpolant	Direct	Stochastic	Desired reconstruction
CFM (Lipman et al., 2023)	Independent	$(1-t)x_0 + tx_1$	✗	✗	$p_t(\cdot)$
OT-CFM (Tong et al., 2024a)	OT	$(1-t)x_0 + tx_1$	✗	✗	$p_t(\cdot)$
RCFM (Chen & Lipman, 2024)	Independent	$\exp_{x_1}(\kappa(t)\log_{x_1}(x_0))$	✗	✗	$p_t(\cdot)$
OT-MFM (Kapuśniak et al., 2024)	OT	$(1-t)x_0 + tx_1 + \rho(t)\varphi_\theta(x_0, x_1, t)$	✗	✗	$p_t(\cdot)$
TracingFlow (Sun et al., 2026)	Bio-informed OT	$(1-t)x_0 + tx_1$	✗	✗	$p_t(\cdot)$
Curly-FM (Petrović et al., 2026)	OT	$(1-t)x_0 + tx_1 + \rho(t)\varphi_\theta(x_0, x_1, t) + \sigma\rho(t)\,\epsilon$	✗	✓	$p_t(\cdot)$
SF^{2}M (Tong et al., 2024b)	EOT	$(1-t)x_0 + tx_1 + \sigma\rho(t)\,\epsilon$	✗	✓	$p_t(\cdot)$
Branched SBM (Tang et al., 2026)	EOT	$(1-t)x_0 + tx_{1,k} + \rho(t)\varphi_\theta(x_0, x_{1,k}, t)$	✗	✓	$\{p_{t,k}(\cdot)\}_{k=1}^K$
CLIFT	OT	$(1-t)x_0 + tx_1 + \sigma\rho(t)\,\epsilon$	✓	✓	$p_t(\cdot), p_{t\|s}(\cdot\|X_s = x)$

Table 5: Comparison of flow-based trajectory inference methods in terms of the biases they impose, where they do in the learning process, whether they learn an ODE or SDE, and their main priority for reconstruction. 'Direct' refers to if the bias is imposed directly on the learned drift.

Learning finite-time transitions. Consistency models, shortcut models, and flow maps reduce the cost of sampling from generative models by replacing numerical integration with learned finite-time predictions (Song et al., 2023; Frans et al., 2025; Geng et al., 2025; Boffi et al., 2026). Two-time flow maps approximate the solution operator of an ODE between arbitrary times. For a fixed intermediate state, this operator produces a deterministic continuation. Stochastic flow maps introduce additional randomness to represent a distribution of possible futures. Within this setting, weak stochastic maps represent conditional distributions, while strong stochastic flow maps approximate the SDE solution associated with a specified Brownian realisation (Potaptchik et al., 2026; McCallum et al., 2026). The strong formulation additionally supports composition of transitions along the same driving Brownian path.

Applications to reward guidance. Reward alignment seeks to favour generated samples with desired properties while retaining fidelity to a reference model. In stochastic-control formulations, guidance often requires conditional expectations of terminal rewards, such as $V_t(x) = \log \mathbb{E}[\exp(r(X_1)) \mid X_t = x]$. Efficiently evaluating these quantities motivates methods for sampling possible futures from intermediate states. GLASS Flows construct conditional inner flows from a pretrained model, enabling stochastic transitions between general times through ODE integration (Holderrieth et al., 2025). Meta Flow Maps amortise conditional posterior transports with flow maps, providing differentiable endpoint samples for value-function estimation, inference-time steering, and fine-tuning (Potaptchik et al., 2026). A complementary approach is developed by Huang et al. (2026), who formulate guidance through deterministic optimal control and use flow-map endpoint sensitivities to construct efficient guidance. These works demonstrate how access to finite-time predictions can make reward-based adaptation more practical.

The transition law represented by the sampler. Applying these constructions to trajectory inference requires identifying which conditional law they represent. GLASS Flows define transitions through a correlated Gaussian observation model; particular choices recover the transition kernels of their reference diffusion. Meta Flow Maps target conditional endpoint distributions of the training interpolant, which coincide with those of a specified reference SDE under the diffusion construction considered in that work. Their auxiliary flow transports noise to the requested posterior, and its intermediate states do not generally represent the physical evolution of the reference process. These constructions therefore do not automatically supply the transition kernels of an arbitrary neural SDE, particularly when an additional training objective changes its drift. As discussed in Section 2, agreement of temporal marginals alone does not establish agreement of transition laws.

Strong stochastic flow maps for CLIFT. Our objective requires optimising a transition cost with respect to the parameters of the dynamics being learned. For direct reparameterisation gradients, we need sampled transitions whose dependence on these parameters corresponds to the current model.

An interpolant-posterior sampler does not generally provide this connection: differentiating through it optimises its induced transition behaviour, which need not coincide with that of the neural SDE. We adopt strong stochastic flow maps (McCallum et al., 2026) because they provide a parameterisation of the SDE solution operator that can be trained jointly with the local dynamics. Local matching and self-consistency objectives connect finite-time predictions to the learned drift and prescribed diffusion, using consistent Brownian realisations across composed transitions. The fate penalty can consequently reuse the off-diagonal predictions already computed during training and propagate gradients directly to the shared model parameters, without differentiating through a numerical SDE rollout. Our choice is motivated by this direct coupling between finite-time transitions and trainable dynamics. A weak transition map could also support the pairwise objective if trained to maintain the corresponding model-dependent kernels; the posterior-sampling constructions above do not provide that correspondence automatically.

I Experiment Details

I.1 Maizels dataset

The dataset contains cells collected on days: D3, D3.2, D3.4, D3.6, D3.8, D4, D5, D6, D7, and D8.

Data and training marginals. We trained on the three observed marginals

$$\mathcal{T}_{\text{train}} = \{\text{D3}, \text{D3.8}, \text{D8}\},$$

using the physical-time parameterisation

$$\tau(d) = \frac{d-3}{8-3},$$

so that the three training times were 0, 0.16, and 1. Training pairs were divided equally between the adjacent intervals $\text{D3} \to \text{D3.8}$ and $\text{D3.8} \to \text{D8}$. At each retained time point, a fixed 10% of cells was excluded from flow-model training and used only for checkpoint validation and trajectory diagnostics.

Lineage-filtered training pairs. The lineage tree used is:

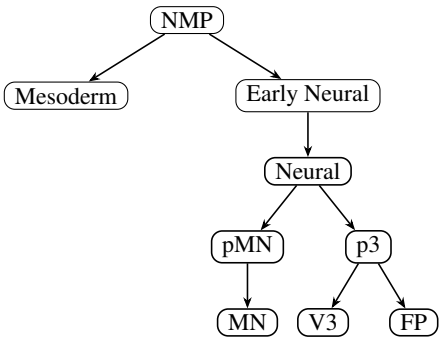


Figure 8: Maizels et al. (2024) lineage tree.

We used the reflexive transitive closure of this graph, permitting a cell to remain in its current type or transition to any descendant.

For interpolant filtering, a pair was retained only if its coupled endpoints satisfied the lineage relation and if the classifier predictions at 50 interior points of the straight-line interpolant formed a lineage-valid sequence.

Constrained flow-map objective. We base our implementation off the code from (Boffi et al., 2026), found here `https://github.com/nmboffi/flow-maps`. The flow map was parameterised as

$$\Phi_\theta(s,t,x) = x + (t-s)a_\theta(s,t,x),$$

Hyperparameter	Setting
Flow-model architecture	$52 \to 1024 \to 1024 \to 1024 \to 50$ MLP
Activation	GELU
Retained-day holdout fraction	10%
Minibatch size	128
Optimiser	RAdam, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$
Learning rate	10^{-3}, selected from $\{3 \times 10^{-4}, 10^{-3}, 3 \times 10^{-3}\}$
Gradient clipping	Global norm 10
Maximum optimiser steps	50,000
Checkpoint validation	Every 100 optimiser steps
Early-stopping patience	10 validation checks
Pair-filter checkpoints	50 interior interpolant points
Lineage weight λ_{lin}	1000, selected from $\{100, 350, 1000, 1500\}$

Table 6: Training hyperparameters for the Maizels constrained flow map.

where a_θ is an MLP receiving the concatenated input (s, t, x). Each minibatch contained 75% diagonal examples, $s = t$, and 25% off-diagonal examples, $s < t$. Diagonal examples were trained using the standard flow-matching regression loss, while off-diagonal examples used Lagrangian self-distillation.

The complete minibatch objective can therefore be written as

$$\mathcal{L} = 0.75\,\mathcal{L}_{\mathrm{FM}} + 0.25\,\mathcal{L}_{\mathrm{LSD}} + \lambda_{\mathrm{lin}}\mathcal{L}_{\mathrm{lin}}.$$

The implementation rescales the constraint contribution so that λ_{lin} is not reduced by the 25% off-diagonal batch fraction. The classifier parameters were frozen, but gradients were propagated through its input and hence into the flow map when optimising $\mathcal{L}_h$.

Checkpoint and hyperparameter validation. We distinguish the validation used for early stopping from that used for hyperparameter selection. During each training run, the complete training objective was evaluated every 100 steps on the 10% held-out subsets of D3, D3.8, and D8. Training stopped after ten checks without improvement, and the checkpoint with the lowest validation loss was retained.

The intermediate days D3.4 and D6 were never used to construct training pairs. They were reserved for outer-loop hyperparameter selection. Each hyperparameter setting listed in the above table was trained, and settings were ranked by the W_1 on these validation times.

Empirical W_1 distance. For an omitted day d, cells from its nearest observed marginal to the left were transported directly to d. Thus, D3 cells were used for D3.2, D3.4, and D3.6, while D3.8 cells were used for D4, D5, D6, and D7. If

$$\widehat{\mu}_d = \frac{1}{n}\sum_{i=1}^{n} \delta_{\Phi_\theta(\tau_\ell, \tau_d, x_i)} \quad \text{and} \quad \widehat{\nu}_d = \frac{1}{m}\sum_{j=1}^{m} \delta_{y_j}$$

denote the predicted and observed empirical distributions, respectively, we computed

$$W_1(\widehat{\mu}_d, \widehat{\nu}_d) = \min_{\pi \in \Pi(\mathbf{1}_n/n, \mathbf{1}_m/m)} \sum_{i,j} \pi_{ij} \|\widehat{x}_i - y_j\|_2.$$

We use 50 integrated flow map sampling steps to get the model-generated distributions. This is the exact uniform-mass Earth Mover's Distance with Euclidean ground cost, solved in the original 50-dimensional PCA space using POT. All available source and target cells were included.

Classifier-based lineage violation rate. Lineage violations were evaluated using the independent classifier trained on all ten sampling days. For each run, all 320 cells in the 10% D3 holdout were propagated from D3 to D8 using composed flow map sampling with 50 integration steps. Each trajectory was classified at 50 points spanning the trajectory. Starting from the cell's annotated D3 type, a trajectory was declared invalid if any successive classifier prediction was not a permitted descendant of the preceding type. Self-transitions and jumps to any downstream descendant were valid, whereas backward transitions and switches between committed branches were violations.

Cell-type classifiers. Both classifiers used the architecture

$$\begin{aligned}\mathrm{Standardise}(50) &\to \mathrm{Linear}(50, 128) \to \mathrm{BN} \to \mathrm{ReLU} \\ &\to \mathrm{Dropout}(0.2) \to \mathrm{Linear}(128, 64) \to \mathrm{BN} \to \mathrm{ReLU} \\ &\to \mathrm{Dropout}(0.2) \to \mathrm{Linear}(64, 9).\end{aligned}$$

The classifier used for pair filtering and the differentiable lineage loss was trained only on D3, D3.8, and D8. These cells were divided using a stratified 85/15 train/validation split, and feature-wise standardisation was fitted only on the classifier-training split. The best checkpoint was selected by validation accuracy and attained 95.16% validation accuracy. It was then frozen throughout flow-map training.

A separate classifier with the same architecture was trained on all sampling days. Its best validation-loss checkpoint attained 95.74% held-out test accuracy. This all-days classifier was used only to calculate the final violation rate and was never used to construct flow-model training pairs or compute the flow-model loss.

I.2 CITE-SEQ AND MULTIOME EXPERIMENTS

Unless stated otherwise, the CITE-seq and Multiome experiments follow the Maizels setup described above.

Data and temporal splits. Both datasets contain observations at D2, D3, D4, and D7, represented using their provided 100-dimensional PCA with the preprocessing of Tong et al. (2024b). We conduct two leave-one-timepoint-out experiments for each dataset. When D3 is held out, the model is trained on D2, D4, and D7; when D4 is held out, it is trained on D2, D3, and D7. Training pairs are balanced between the two adjacent retained intervals. Each retained marginal is divided into 90% training cells and 10% validation cells, while the complete omitted marginal is excluded from flow-map training.

We follow the set-up of metric flow matching (Kapuśniak et al., 2024) and use equally spaced global time coordinates

$$(\tau_{\mathrm{D2}}, \tau_{\mathrm{D3}}, \tau_{\mathrm{D4}}, \tau_{\mathrm{D7}}) = \left(0, \frac{1}{3}, \frac{2}{3}, 1\right).$$

Unlike the Maizels experiment, no additional days are available for hyperparameter validation. We therefore transfer the Maizels-selected hyperparameters without further dataset-specific tuning.

Lineage constraint and classifiers. CITE-seq and Multiome use the seven cell types

$$\{\mathrm{BP}, \mathrm{EryP}, \mathrm{HSC}, \mathrm{MasP}, \mathrm{MkP}, \mathrm{MoP}, \mathrm{NeuP}\}.$$

The lineage tree used is shown below.

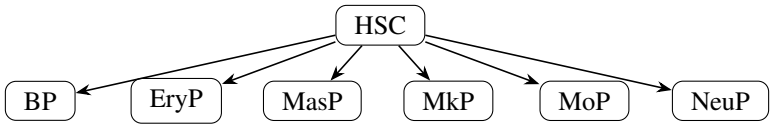


Figure 9: Cite/Multi lineage tree.

HSC denotes hematopoietic stem cell, BP basophil progenitor, EryP erythroid progenitor, MasP mast-cell progenitor, MkP megakaryocyte progenitor, MoP monocyte progenitor, and NeuP neutrophil progenitor. The stem cell can differentiate to any of the progenitors, and, once committed, a cell must remain in its fate.

Separate classifiers are trained for CITE-seq and Multiome because their PCA coordinate systems differ. For a run holding out day d, pair filtering and the differentiable lineage constraint use a classifier trained on all days except d. A separate classifier trained on all four days is reserved exclusively for final evaluation. All classifiers have architecture

Table 7: Settings shared by the stochastic flow-map experiments.

Setting	Value
Brownian coefficients	3 per state dimension
Bridge-noise scale γ	0.2
Diffusion scale σ	0.2
Drift network	3 hidden layers, width 512, SiLU
Diffusion network	2 hidden layers, width 512, SiLU
Uncertainty network	1 hidden layer, width 64, SiLU
Local/semigroup fraction	$0.75/0.25$
Local horizon fraction	Uniform on $[0.01, 0.02]$
Non-local horizon	Log-uniform; maximum increases from 0.02 to 0.98 during the first half of training
Endpoint exclusion	0.005 of each retained interval
EMA decay	0.999
Optimizer	AdamW, $\beta_1 = 0.9$, $\beta_2 = 0.99$, weight decay 10^{-4}
Learning-rate schedule	Linear warm-up to 10^{-3} followed by cosine decay
Lineage weight	10
Validation	Every 100 steps on a fixed batch of 1,024 held-out pairs
Early stopping	Patience 10 validation checks; minimum improvement 0

Evaluation. Distributional accuracy is evaluated separately at the omitted D3 or D4 marginal using the same empirical W_1 estimators as for Maizels. Lineage violations are evaluated over complete D2-to-D7 trajectories originating from the held-out 10% of D2 cells.

I.3 Stochastic flow-map experiments

The stochastic experiments use the same data representations, observation times, train–test splits, lineage graphs, and cell-type classifiers as their deterministic counterparts.

Stochastic interpolant. For a paired sample (x_0, x_1) and normalised position $\tau \in (0, 1)$ within an interval of duration Δ, we use

$$X_\tau = (1-\tau)x_0 + \tau x_1 + \gamma \sin(\pi\tau)\, \boldsymbol{\rho} \odot \epsilon, \qquad \epsilon \sim \mathcal{N}(0, I), \tag{24}$$

$$g_\tau = \sigma \boldsymbol{\rho} \sqrt{\frac{\sin(\pi\tau)}{\Delta}}, \tag{25}$$

where $\boldsymbol{\rho}$ is the per-feature standard deviation estimated from the retained training populations. We set the bridge-noise scale $\gamma = 0.2$ and diffusion scale $\sigma = 0.2$.

The strong stochastic flow map is based on the implementation and code of (McCallum et al., 2026), found here: `https://github.com/sammccallum/ssfm`.

For the full stochastic-flow-map models, 75% of each batch is used for the local stochastic-interpolant loss and 25% for semigroup consistency against an exponential-moving-average (EMA) target.

Because the stochastic interpolant is not represented by its straight-line mean, the standard interpolant filtering variants are changed, to only apply filtering on the annotated endpoints.

Hyperparameters are selected using the mean stochastic W_1 distance on the Maizels days D3.4 and D6. The learning-rate grid is $\{3 \times 10^{-4}, 10^{-3}, 3 \times 10^{-3}\}$; constrained variants additionally use lineage weights $\{1, 10, 100\}$. The diffusion and bridge-noise scales remain fixed at 0.2.

Stochastic evaluation metrics. To account for stochasticity, the same metrics as in the deterministic case are averaged over 3 noise draws.

I.4 LARRY experiments

Unless stated otherwise, the LARRY experiments follow the Maizels setup.

Data and temporal split. The LARRY data contain cells observed at D2, D4, and D6. We use the authors' published two-dimensional SPRING coordinates, centred and divided by a single global root-mean-square scale so that the geometry and aspect ratio of the published embedding are preserved. The model is trained only between D2 and D6, with

$$\tau_{\mathrm{D2}} = 0, \qquad \tau_{\mathrm{D4}} = \frac{1}{2}, \qquad \tau_{\mathrm{D6}} = 1,$$

and evaluated at the unseen intermediate D4 marginal. We retain 90% of the D2 and D6 cells for training and reserve 10% for validation.

Again, we do not tune hyperparameters using D4; the common hyperparameters are inherited from the Maizels setup.

Lineage constraint and classifiers. The annotated cell types are

$$\mathcal{C} = \{\mathrm{Baso}, \mathrm{Ccr7_DC}, \mathrm{Eos}, \mathrm{Erythroid}, \mathrm{Lymphoid}, \mathrm{Mast}, \\ \mathrm{Meg}, \mathrm{Monocyte}, \mathrm{Neutrophil}, \mathrm{Undifferentiated}, \mathrm{pDC}\}.$$

The permitted transition relation is

$$\mathcal{R} = \{(c, c) : c \in \mathcal{C}\} \cup \{(\mathrm{Undifferentiated}, c) : c \in \mathcal{C} \setminus \{\mathrm{Undifferentiated}\}\}.$$

Thus, undifferentiated cells may remain undifferentiated or transition into any annotated mature population, whereas mature cells may only remain within their current population.

Two classifiers are trained. The classifier used for pair filtering and the differentiable constraint is trained using only D2 and D6, whereas the evaluation-only classifier is trained using D2, D4, and D6.

Evaluation. Clone-conditioned evaluation is additionally performed between clones represented at both D2 and D4. For each eligible clone, 32 independent descendants are sampled from every D2 source cell, the resulting pooled distribution is compared with all observed target cousins using exact W_1, and the procedure is repeated for three noise draws. Clones must contain at least one D2 source cell and ten target cells.

I.5 Baselines

Metric Flow Matching (MFM). MFM is based on the implementation and code of (Kapuśniak et al., 2024), found here `https://github.com/kkapusniak/metric-flow-matching`. It first learns a data-dependent Riemannian metric and a neural approximation to its geodesics. A velocity field is then trained by flow matching along these learned paths and integrated as an ODE at evaluation time. For multiple observed marginals, separate metric and geodesic models are fitted to adjacent intervals; we consider its minibatch-OT endpoint coupling variant.

BranchSBM. BranchSBM is based on the implementation and code of (Tang et al., 2026), found here `https://github.com/sophtang/BranchSBM`. It represents the terminal distribution as a collection of branches and jointly learns branch-specific trajectories, growth, and terminal masses. We define one branch for every cell type observed at the final day. Intermediate training marginals inform its learned RBF metric but are not introduced as additional fitting targets, as this is fundemental a two-marginal method.

SF^{2}M. SF^{2}M is based on the implementation and code of (Tong et al., 2024b), found here: `https://github.com/atong01/conditional-flow-matching`. It uses Schrödinger-bridge conditional flow matching to learn both a probability-flow velocity field and a score field. Endpoint pairs are obtained using minibatch optimal transport, and samples are generated by Euler–Maruyama integration of the resulting stochastic dynamics. We use diffusion scale $\sigma = 0.2$ to match the other stochastic set-ups.

Flow matching. The flow-matching baseline (Lipman et al., 2023) trains a time-dependent velocity field using conditional flow-matching targets constructed from paired observations and linear interpolants. At evaluation time, the learned velocity field is integrated numerically from the source marginal to the requested time. We use both independently sampled endpoint pairs and OT endpoint couplings.

Strong stochastic flow map (SSFM). SSFM is based on the implementation and code of (McCallum et al., 2026), found here: `https://github.com/sammccallum/ssfm`.

Flow map. The deterministic flow-map is based of the implementation and the code from (Boffi et al., 2026), found here `https://github.com/nmboffi/flow-maps`.

To keep evaluation consistent, we use 50 integration steps for all methods to assess all metrics.

J Extended Flow Map Preliminaries

Here, we write the self-distillation losses most often used for flow map training. Boffi et al. (2026) propose Lagrangian self-distillation

$$\mathcal{L}_{\mathrm{LSD}}(\theta) = \mathbb{E}_{s<t}\left[\left\|\partial_t \Phi_\theta(s,t,\mathbf{I}_s) - \mathrm{sg}\big[\Phi_\theta\big(t,t,\mathrm{sg}[\Phi_\theta(s,t,\mathbf{I}_s)]\big)\big]\right\|^2\right], \tag{26}$$

and we follow their implementation for the deterministic flow map settings.

For strong stochastic flow maps, (McCallum et al., 2026) propose progressive self-distillation

$$\mathcal{L}_{PSD}(\theta) = \mathbb{E}_{s<u<t,\mathbf{I}_s,\mathbf{W}_{[s,t]}}\left[\left\|\Psi_\theta(s,t,\mathbf{I}_s,\mathbf{W}_{[s,t]}) - \mathrm{sg}\left[\Psi_\theta(u,t,\mathrm{sg}\left[\Psi_\theta(s,u,\mathbf{I}_s,\mathbf{W}_{[s,u]})\right],\mathbf{W}_{[u,.t]})\right]\right\|^2\right]. \tag{27}$$

K Observed Marginal Reconstruction

Here, we report the empirical values from our real dataset experiments of W_1 distance from the observed (training) marginals. This is to show that CLIFT performs competitively in terms of this metric, despite not enforcing reconstruction of the training distributions with a further loss, to a method that does have theoretically guaranteed reconstruction of the training marginals.

Since, in the Maizels, Cite, and Multi datasets, we use multiple training marginals, we compare two settings for model distributional fidelity. To compare the reconstruction of a given training marginal, we can push forward samples from the nearest earlier training marginal, or we can push samples always from the initial p_0 distribution.

Dataset	Method	Nearest marginal pushforward	Base pushforward
Maizels	OT-SSFM	[D3 $\to$ D3.8] 5.72 ± 0.01 [D3.8 $\to$ D8] 8.20 ± 0.01	[D3 $\to$ D8] 8.41 ± 0.08
	CLIFT$_S$	[D3 $\to$ D3.8] 6.03 ± 0.01 [D3.8 $\to$ D8] 7.81 ± 0.01	[D3 $\to$ D8] 9.18 ± 0.09
Cite	OT-SSFM	[D2 $\to$ D4] 41.44 ± 0.18 [D4 $\to$ D7] 42.11 ± 0.06	[D2 $\to$ D7] 50.65 ± 0.22
	CLIFT$_S$	[D2 $\to$ D4] 40.03 ± 0.14 [D4 $\to$ D7] 43.46 ± 0.02	[D2 $\to$ D7] 47.51 ± 0.18
Multi	OT-SSFM	[D2 $\to$ D4] 43.84 ± 0.26 [D4 $\to$ D7] 46.66 ± 0.15	[D2 $\to$ D7] 51.69 ± 0.67
	CLIFT$_S$	[D2 $\to$ D4] 43.10 ± 0.05 [D4 $\to$ D7] 47.56 ± 0.03	[D2 $\to$ D7] 49.96 ± 0.23
Synthetic ($G = 0.2I_2$)	OT-SSFM	$[p_0 \to p_1]$ 0.0391 ± 0.0091 $[p_1 \to p_2]$ 0.0347 ± 0.0009	$[p_0 \to p_2]$ 0.0294 ± 0.0035
	CLIFT$_S$	$[p_0 \to p_1]$ 0.0382 ± 0.0031 $[p_1 \to p_2]$ 0.0348 ± 0.0044	$[p_0 \to p_2]$ 0.0310 ± 0.0045
Synthetic (ground-truth G)	OT-SSFM	$[p_0 \to p_1]$ 0.0471 ± 0.0078 $[p_1 \to p_2]$ –	$[p_0 \to p_2]$ 0.0366 ± 0.0067
	CLIFT$_S$	$[p_0 \to p_1]$ 0.0499 ± 0.0090 $[p_1 \to p_2]$ –	$[p_0 \to p_2]$ 0.0396 ± 0.0079

Table 8: Training marginal reconstruction of CLIFT$_S$ vs OT-SSFM. Synthetic results report the mean and standard deviation across five training seeds. Dashes indicate unavailable measurements.