GRAPHFM: GRAPH-CONSTRAINED FLOW MATCHING FOR DYNAMICS RECONSTRUCTION FROM TEMPORAL SNAPSHOTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Reconstructing continuous population dynamics from sparse, unpaired temporal snapshots is a fundamental yet underdetermined problem in many scientific applications. Many vector fields can reproduce the same finite collection of temporal marginals. Existing flow-matching methods reduce this ambiguity through assumptions on endpoint couplings, interpolation geometry, or temporal smoothness, but typically treat the variables within each state as unstructured. This overlooks known interaction structure in systems such as gene-regulatory and road networks. We introduce Graph-Constrained Flow Matching (GraphFM), which incorporates a variable-interaction graph into both conditional interpolation and vector-field learning. GraphFM parameterizes an endpoint-constrained graph neural interpolant whose message-passing structure restricts each variable to depend only on its graph-defined receptive field. To exploit multiple observed marginals, we train the interpolant with a hierarchical objective combining distributional matching at observed time points, multi-anchor trajectory regularization induced by a multi-marginal coupling, and local temporal smoothness. The derivatives of the learned trajectories then supervise a graph-aware velocity field defined over the same interaction graph. We evaluate GraphFM on synthetic dynamics, single-cell trajectory inference, spatial-transcriptomic interpolation, and traffic-state imputation. GraphFM improves or matches strong multi-marginal flow-matching baselines across these settings, while graph-corruption experiments demonstrate the value of informative structural priors. Our results show that variable-level interaction structure provides a effective inductive bias for reconstructing dynamics from temporal snapshots.

1 INTRODUCTION

Many scientific systems are observed not as complete trajectories, but as population snapshots collected at a limited number of time points. Examples include destructive single-cell measurements (Ding et al., 2022), serial tissue profiling (Saelens et al., 2019), repeated cross-sectional studies (Abbott & Funk, 2022), and network-wide environmental or traffic observations (Franzke et al., 2015). Samples observed at different times are generally unpaired: one observes a sequence of temporal marginals, but not the trajectory followed by any individual sample. Recovering continuous dynamics from such data is therefore intrinsically underdetermined. Distinct couplings, interpolation trajectories, and vector fields can agree with the same finite collection of marginals. Dynamics reconstruction consequently requires inductive biases that select plausible trajectories rather than merely reproduce the observed snapshots.

Flow Matching (FM) provides a simulation-free framework for learning continuous vector fields that transport samples between unpaired distributions (Lipman et al., 2023; Tong et al., 2024b). The inferred dynamics depend on modeling choices, including how samples are coupled across marginals and how conditional interpolation trajectories are constructed between them. Existing approaches constrain these choices using optimal-transport couplings (Tong et al., 2024a), Euclidean or geometry-aware interpolation (Kapuśniak et al., 2024), spline-based multi-marginal trajectories (Rohbeck et al., 2025), and learned neural interpolants aligned with observed intermediate marginals (Kviman et al., 2026). These inductive biases regulate where trajectories travel and how

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

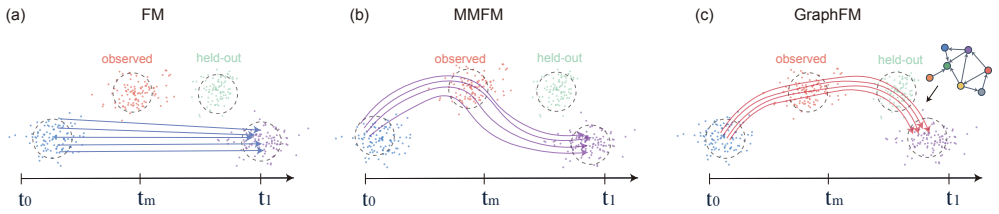


Figure 1: From Flow Matching to Graph-constrained Flow Matching. (a) FM constructs conditional paths between source and target marginals. (b) MMFM further uses intermediate marginals to constrain the paths. (c) GraphFM additionally incorporates variable-level graph constraints to restrict the dependency structure of the learned dynamics.

smoothly they evolve, but generally treat the variables within each state as unstructured. Existing graph-aware flow models use graph structure to construct the source distribution or parameterize the velocity field, while retaining a fixed, typically linear, conditional interpolation rule; they do not constrain the conditional interpolation trajectories from which flow-matching supervision is constructed (Zhang et al., 2026). Such relational structure is important in structured dynamical systemsm where evolutions is governed by interactions among variables, including road-network connectivity in traffic systems and gene-regulatory interactions in cellular systems (Wang et al., 2026). Treating these variables as an unstructured vector permits dependencies that may be inconsistent with known system organization. Incorporating this structure therefore offers a principled way to constrain the space of admissible dynamics.

To overcome this limitation, we formalize variable-interaction structure as graph-local constraints on conditional interpolation trajectories and introduce GraphFM, a multi-marginal flow-matching framework that incorporates a given variable graph into both interpolation-trajectory construction and vector-field learning (Fig. 1). Specifically, GraphFM parameterizes a graph neural interpolant such that the evolution of each variable depends only on information propagated through its graph-defined receptive field. This provides a hard architectural constraint on cross-variable dependencies rather than a soft regularization penalty. To exploit observations from multiple time points, we optimize the interpolant with a hierarchical objective that combines distributional matching at observed temporal anchors, multi-anchor trajectory regularization induced by a multi-marginal coupling, and local temporal smoothness. The temporal derivatives of the resulting graph-constrained trajectories are then used to supervise a graph-aware velocity field defined over the same variable graph. GraphFM therefore maintains structural consistency between the trajectories used to construct the training signal and the continuous dynamics learned for inference.

Our main contributions are the following:

(1) We introduce GraphFM, a graph-constrained flow-matching framework that incorporates the same variable graph into both multi-marginal conditional-path construction and continuous vector-field learning, so that structural constraints are preserved throughout dynamics process.

(2) We develop a hierarchical interpolation objective that combines marginal matching, multi-anchor trajectory regularization, and local temporal smoothness, enabling sparse temporal snapshots to constrain the learned trajectories at complementary scales.

(3) We demonstrate that GraphFM can improve dynamics reconstruction across benchmarks from distinct domains, including synthetic dynamical systems, biological data, and traffic networks. We also study its dependence on graph quality through controlled graph-corruption experiments.

2 METHOD

Background. Let p_0 and p_1 denote the source and target distributions on $\mathbb{R}^d$. A time-varying vector field $v : [0, 1] \times \mathbb{R}^d \to \mathbb{R}^d$ defines an ordinary differential equation (ODE),

$$\frac{dx_t}{dt} = v_t(x_t), \tag{1}$$

where $v_t(x)$ denotes $v(t, x)$. The corresponding flow map $\phi_t : \mathbb{R}^d \to \mathbb{R}^d$ is defined by $x_t = \phi_t(x_0)$ for $x_0 \sim p_0$. The flow transports p_0 along a probability path $(p_t)_{t\in[0,1]}$, i.e., $(\phi_t)_{\#}p_0 = p_t$.

Given a probability path $(p_t)_{t\in[0,1]}$ connecting p_0 and p_1, the goal of FM (Lipman et al., 2023) is to learn a vector field whose induced flow realizes this transport. FM parameterizes the vector field as $u_\theta(x, t)$ and trains it to match a target vector field $v_t(x)$ that generates the prescribed probability path:

$$\mathcal{L}_{\mathrm{FM}}(\theta) = \mathbb{E}_{t\sim\mathcal{U}[0,1],\, x\sim p_t}\left[\|u_\theta(x,t) - v_t(x)\|_2^2\right].$$

Directly optimizing this objective is generally intractable, however, because the marginal vector field $v_t(x)$ associated with p_t is typically intractable.

Conditional Flow Matching (CFM) (Lipman et al., 2023; Tong et al., 2024b) avoids direct access to the marginal vector field by constructing tractable conditional probability paths $p_t(x \mid z)$ with corresponding conditional vector fields $u_t(x \mid z)$. For transport between two endpoint distributions, we let $z = (x_0, x_1) \sim \pi$, where $\pi \in \Pi(p_0, p_1)$ is a coupling between the source and target distributions. The conditional paths induce the marginal path through $p_t(x) = \int p_t(x \mid z)q(z)\,dz$, with $q(z) = \pi(x_0, x_1)$. CFM then optimizes

$$\mathcal{L}_{\mathrm{CFM}}(\theta) = \mathbb{E}_{t\sim\mathcal{U}[0,1],\, z\sim q(z),\, x\sim p_t(x|z)}\left[\|u_\theta(x,t) - u_t(x \mid z)\|_2^2\right]. \tag{2}$$

Importantly, the CFM objective has the same gradient with respect to θ as the corresponding FM objective, while requiring only samples from the conditional paths and evaluations of their tractable conditional vector fields (Lipman et al., 2023).

The choice of $p_t(x \mid z)$ is therefore an important modeling decision, as it determines the intermediate states and target velocities used to supervise the learned vector field $u_\theta(x, t)$. A common choice is a Gaussian conditional probability path (Lipman et al., 2023),

$$p_t(x \mid z) = \mathcal{N}\big(x; \mu_t(z), \sigma_t^2(z)I\big),$$

for which the corresponding conditional vector field is available in closed form. Different choices of the mean path $\mu_t(z)$, variance schedule $\sigma_t(z)$, and endpoint coupling induce different conditional trajectories and therefore different supervision signals for the learned dynamics. Beyond standard linear and diffusion-based paths (Lipman et al., 2023), existing approaches have introduced manifold-aware geodesic interpolants (Kapuśniak et al., 2024), multi-marginal spline interpolants (Rohbeck et al., 2025), and learned neural interpolants that match observed intermediate marginals (Kviman et al., 2026). See § 3 for a discussion.

2.1 GRAPH CONSTRAINED NEURAL INTERPOLANTS

In the multi-marginal setting, observations are available at K discrete time points $0 = t_1 < t_2 < \cdots < t_K = 1$, where samples observed at time t_m follow the corresponding marginal distribution q_{t_m}, i.e., $x_{t_m} \sim q_{t_m}, m = 1, \ldots, K$. Given a coupling π between q_0 and q_1 and a variable graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, we parameterize the interpolant as

$$G_\phi(x_0, x_1, t; \mathcal{G}) = (1-t)x_0 + tx_1 + t(1-t)R_\phi(x_0, x_1, \mathcal{G}, t), \tag{3}$$

where R_ϕ is a graph neural network (GNN) whose message-passing structure is specified by $\mathcal{G}$.

To incorporate the observed intermediate marginals, we require

$$\left(G_\phi(\cdot, \cdot, t_m; \mathcal{G})\right)_{\#}\pi = q_{t_m}, \tag{4}$$

where π is a coupling over endpoint pairs (x_0, x_1) and $(\cdot)_{\#}$ denotes the pushforward operator. By construction, $G_\phi(x_0, x_1, 0; \mathcal{G}) = x_0$ and $G_\phi(x_0, x_1, 1; \mathcal{G}) = x_1$, so the endpoint constraints in equation 4 are satisfied automatically.

We introduce a graph-local dependency constraint through the variable graph $\mathcal{G}$, such that the prediction at each node depends only on information within its graph-defined receptive field.

Definition 1. *Let $\mathcal{N}_i^{(L)}$ denote the L-hop receptive field of node i under $\mathcal{G}$, including node i itself. A mapping $F_{\mathcal{G}}$ satisfies an L-local graph dependency constraint if the output at node i depends only*

on input features associated with nodes in $\mathcal{N}_i^{(L)}$. For a differentiable mapping, this is equivalently expressed as

$$j \notin \mathcal{N}_i^{(L)} \quad \Longrightarrow \quad \frac{\partial F_{\mathcal{G},i}(X)}{\partial X_j} = 0. \tag{5}$$

Message-passing GNNs naturally satisfy this constraint. Since each layer aggregates information only from neighboring nodes in $\mathcal{G}$, an L-layer GNN can propagate information at most across L hops. Consequently, the representation at node i depends only on nodes in $\mathcal{N}_i^{(L)}$. We next show that the endpoint-constrained interpolant G_ϕ in equation 3 preserves this graph-local dependency structure.

Proposition 1 (Preservation of graph-local dependency)**.** *Let $R_\phi(x_0, x_1, \mathcal{G}, t)$ be an L-local message-passing GNN, and consider the endpoint-constrained interpolant G_ϕ defined in equation 3. Assume that each L-hop receptive field $\mathcal{N}_i^{(L)}$ contains node i itself. Then G_ϕ preserves the same L-local graph dependency. Specifically, for any nodes i and j such that $j \notin \mathcal{N}_i^{(L)}$,*

$$\frac{\partial G_{\phi,i}(x_0, x_1, t; \mathcal{G})}{\partial x_{0,j}} = 0, \qquad \frac{\partial G_{\phi,i}(x_0, x_1, t; \mathcal{G})}{\partial x_{1,j}} = 0. \tag{6}$$

Thus, the interpolant introduces no dependency outside the L-hop receptive field specified by $\mathcal{G}$.

Proposition 1 establishes that the interpolant respects the prescribed graph-local dependency structure. We next introduce the objective used to learn $\mathcal{G}$ from the observed temporal marginals.

2.2 Hierarchical Interpolation Objective

The graph-constrained parameterization restricts the dependency structure of the conditional paths. We further learn their temporal geometry through three levels of supervision: marginal consistency at observed times, trajectory regularization through intermediate anchors, and local temporal smoothness.

Level I: Distribution-level marginal consistency. We first align the interpolated population with each observed intermediate marginal:

$$\mathcal{L}_{\text{dist}} = \frac{1}{K-2} \sum_{m=2}^{K-1} \text{SW}^2 \left((G_\phi(\cdot, \cdot, t_m; \mathcal{G}))_{\#} \pi, q_{t_m} \right), \tag{7}$$

where SW^2 is instantiated as the squared sliced-Wasserstein distance (see § A.3.1). This term constrains the population-level state of the interpolant at the observed temporal anchors; the endpoint marginals are satisfied by construction through equation 3.

Level II: Multi-anchor trajectory regularization. Distribution-level marginal matching does not uniquely determine how individual conditional trajectories evolve between the observed times. ALI-CFM addresses this ambiguity by sampling one intermediate marginal and constructing a three-point reference path through the source, intermediate, and target samples (Kviman et al., 2026). We extend this construction from one intermediate anchor to a configurable set of intermediate anchors.

Specifically, let

$$\mathcal{M} = \{m_1, \ldots, m_M\} \subseteq \{2, \ldots, K-1\}, \qquad m_1 < \cdots < m_M,$$

denote the selected intermediate-time indices, where M is configurable. For convenience, we additionally define $m_0 = 1$ and $m_{M+1} = K$, such that $t_{m_0} = 0$ and $t_{m_{M+1}} = 1$.

Instead of constructing an independent bridge for each intermediate marginal, we use a multi-marginal optimal transport coupling

$$\Gamma^{(\mathcal{M})} \in \Pi \left(q_{t_{m_0}}, q_{t_{m_1}}, \ldots, q_{t_{m_{M+1}}} \right), \tag{8}$$

A sample from $\Gamma^{(\mathcal{M})}$ therefore defines the discrete coupled path

$$\left(x_{t_{m_0}}, x_{t_{m_1}}, \ldots, x_{t_{m_{M+1}}} \right) \sim \Gamma^{(\mathcal{M})}.$$

When $M = 1$, this construction reduces to the single-intermediate-anchor bridge used in ALI-CFM.

For each pair of consecutive anchors, we define the piecewise-linear reference trajectory

$$\ell_{\mathrm{pw}}(t) = \frac{t_{m_{r+1}} - t}{t_{m_{r+1}} - t_{m_r}} x_{t_{m_r}} + \frac{t - t_{m_r}}{t_{m_{r+1}} - t_{m_r}} x_{t_{m_{r+1}}}, \qquad t \in [t_{m_r}, t_{m_{r+1}}], \tag{9}$$

for $r = 0, \ldots, M$. The complete-path trajectory regularizer is

$$\mathcal{L}_{\mathrm{ref}} = \mathbb{E}_{r,\mathbf{x},t}\left[\|G_\phi(x_0, x_1, t; \mathcal{G}) - \ell_{\mathrm{pw}}(t)\|_2^2\right]. \tag{10}$$

where $x_0 = x_{t_{m_0}}$, $x_1 = x_{t_{m_{M+1}}}$, $r \sim \mathcal{U}\{0, \ldots, M\}$, $\mathbf{x} = \left(x_{t_{m_0}}, x_{t_{m_1}}, \ldots, x_{t_{m_{M+1}}}\right) \sim \Gamma^{(\mathcal{M})}$ denote a coupled path containing one matched sample from each selected marginal, and $t \sim \mathcal{U}[t_{m_r}, t_{m_{r+1}}]$. Averaging over the $M + 1$ intervals prevents the regularization magnitude from increasing with the number of selected anchors.

Level III: Local temporal smoothness. Finally, we regularize short-scale variations in the conditional velocity using the centered second-order difference $\Delta_h^2 G_\phi(t) = \frac{G_\phi(t+h) - 2G_\phi(t) + G_\phi(t-h)}{h^2}$, where the endpoint arguments and $\mathcal{G}$ are omitted for brevity. We minimize

$$\mathcal{L}_{\mathrm{smooth}} = \mathbb{E}\left[\left\|\Delta_h^2 G_\phi(t)\right\|_2^2\right], \tag{11}$$

which suppresses rapid local variations of the conditional velocity.

The complete interpolation objective is

$$\mathcal{L}_{\mathrm{interp}} = \mathcal{L}_{\mathrm{dist}} + \lambda_{\mathrm{ref}}\mathcal{L}_{\mathrm{ref}} + \lambda_{\mathrm{smooth}}\mathcal{L}_{\mathrm{smooth}}. \tag{12}$$

In Algorithm 1 we provide pseudocode for training Graph-Constrained Interpolant.

2.3 GRAPHFM

GraphFM consists of two stages. In the first stage, we learn the graph-constrained interpolant G_ϕ from the observed temporal marginals. In the second stage, we use the temporal derivatives of the learned interpolant to train a graph-aware velocity field v_θ. The interpolant parameters ϕ are fixed throughout the second stage, and only the velocity-field parameters θ are optimized. Algorithm 1 summarizes this first stage. We parameterize the learned velocity field $v_\theta(x, t; \mathcal{G})$ using a message-passing GNN over the same variable graph $\mathcal{G}$. The vector field is trained by minimizing

$$\mathcal{L}_{\mathrm{GraphFM}}(\theta) = \mathbb{E}_{\substack{(x_0,x_1)\sim\pi,\\ t\sim\mathcal{U}[0,1]}}\left[\|v_\theta(x_t, t; \mathcal{G}) - u_t(x_t \mid x_0, x_1)\|_2^2\right], \tag{13}$$

After training, new samples are transported by solving

$$\frac{dx_t}{dt} = v_\theta(x_t, t; \mathcal{G}), \qquad x_0 \sim q_0.$$

3 RELATED WORK

Flow Matching for Temporal Dynamics. FM provides a simulation-free framework for learning transport between probability distributions by regressing a time-dependent vector field (Lipman et al., 2023; Tong et al., 2024a;b). The specification of conditional probability paths is a fundamental modeling choice in FM, particularly for trajectory inference, where the chosen interpolants directly determine the supervision used to learn the underlying vector field. CFM constructs simple Euclidean paths between paired end points, yielding straight-line interpolations (Tong et al., 2024a). However, the underlying dynamics is often nonlinear, and straight Euclidean interpolants may leave the support of p_t and fail to learn trajectories that remain close to an unknown underlying manifold. MFM therefore uses support-aware metrics to construct geodesic interpolants that better follow the data manifold (Kapuśniak et al., 2024). With observations at multiple time points, intermediate marginals provide additional constraints on path construction: Rohbeck et al. (2025) uses

Algorithm 1: Training the Graph-Constrained Interpolant

Input: multi-marginal coupling $\Gamma^{(\mathcal{M})}$, graph $\mathcal{G}$, trainable interpolant R_ϕ, weights $\lambda_{\text{ref}}, \lambda_{\text{smooth}}$
while *Training* **do**
  Sample $\mathbf{x} = \left(x_0, x_{t_{m_1}}, \ldots, x_1\right) \sim \Gamma^{(\mathcal{M})}$ and $t \sim \mathcal{U}(0,1)$;
  $G_\phi(x_0, x_1, t; \mathcal{G}) \leftarrow (1-t)x_0 + tx_1 + t(1-t)R_\phi(x_0, x_1, \mathcal{G}, t)$,
  $\widehat{\mathcal{L}}_{\text{dist}} \leftarrow$ estimate distribution matching loss with $G_\phi(x_0, x_1, t; \mathcal{G})$ and x_t ; `// Level I`
  $\widehat{\mathcal{L}}_{\text{ref}} \leftarrow$ estimate multi-anchor trajectory regularizer ; `// Level II`
  $\widehat{\mathcal{L}}_{\text{smooth}} \leftarrow$ estimate temporal smoothness regularizer ; `// Level III`
  $\widehat{\mathcal{L}}_{\text{interp}} \leftarrow \widehat{\mathcal{L}}_{\text{dist}} + \lambda_{\text{ref}}\widehat{\mathcal{L}}_{\text{ref}} + \lambda_{\text{smooth}}\widehat{\mathcal{L}}_{\text{smooth}}$;
  Update ϕ using gradient $\nabla_\phi \widehat{\mathcal{L}}_{\text{interp}}$;
return *trained graph-constrained interpolant* G_ϕ

natural cubic splines through intermediate samples, and Kviman et al. (2026) adversarially learns interpolants whose induced intermediate marginals match the observed ones.

Graph-Aware Flow Matching. Recent studies have explored different ways of incorporating graph structure into flow-matching models. Graph Flow Matching constructs neighborhood graphs among latent samples at each flow time and introduces a graph-based diffusion term into the velocity field, allowing each velocity prediction to aggregate information from nearby samples and improving image generation quality (Siddiqui et al., 2026). PACE, which addresses the closely related problem of trajectory inference from unpaired temporal snapshots, uses local neighborhoods among samples to estimate the geometry of plausible state changes and incorporates this geometry into both cross-time coupling and transport-path construction (Yu et al., 2026). These methods primarily exploit sample-level neighborhood structure or geometry in the sample space. GiFlow instead considers graphs among the variables of a spatiotemporal signal, using spatial and temporal graph structure to construct a graph-informed source prior and a graph-aware velocity model for spatiotemporal imputation (Zhang et al., 2026). Its conditional flow, however, remains a linear interpolation between the graph-filtered source and the target, without using the graph structure to inform the construction of conditional probability paths that reflect the underlying system dynamics. In contrast, our method incorporates known relations among variables directly into the interpolation process, allowing the graph structure to constrain how variables co-evolve along the transition path and subsequently guide velocity-field learning.

4 EXPERIMENTS

We evaluate GraphFM through four experiments spanning both controlled synthetic dynamics and real-world applications. We first consider a synthetic benchmark in which the ground-truth interaction graph is fully known in § 4.1, allowing us to evaluate GraphFM under an exact structural prior and to systematically examine its sensitivity to graph corruption through ablation studies. Unlike a leave-one-out setting, only a sparse subset of temporal marginals is provided during training, while multiple intermediate time points remain unobserved. We then evaluate GraphFM in three real-world application settings: single-cell trajectory inference (§ 4.2), spatial transcriptomic state interpolation across serial tumour sections (§ 4.3), and distributional traffic-state imputation (§ 4.4).

4.1 SYNTHETIC DATA

First, we evaluate GraphFM on a synthetic directed-graph temporal benchmark. The dataset consists of $N = 5000$ independent trajectories, each represented by $d = 50$ state variables corresponding to the nodes of a directed graph, and observed at $T = 50$ time points. As shown in Figure 2, the underlying interaction graph is sampled from a directed stochastic block model, and the temporal trajectories are generated by a nonlinear, time-dependent ODE (formalised in Appendix A.3.2). We train on 11 of the 50 temporal marginals, $\mathcal{T}_{\text{train}} = \{0, 5, 10, 15, 20, 25, 30, 35, 40, 45, 49\}$, with cross-time correspondences removed, and evaluate the generated held-out marginals using maximum mean discrepanc (MMD) and SWD, where lower values indicate better distributional recovery.

Table 1: Performance on the synthetic directed-graph temporal benchmark. Results are averaged over 35 hidden test time points for each seed and reported as mean ± standard deviation over three random seeds. (smaller is better)

Method	MMD ↓	SWD ↓
OT-MFM	0.09400 ± 0.09040	0.14101 ± 0.12500
OT-MMFM	0.09341 ± 0.00180	0.14393 ± 0.00280
OT-ALI-CFM	0.05757 ± 0.00120	0.08896 ± 0.00138
GraphFM (ours)	$\mathbf{0.04570 \pm 0.00726}$	$\mathbf{0.08199 \pm 0.01110}$

Table 2: Ablation study on graph-structure corruption. Results are averaged over test time points for each seed. Relative SWD degradation is computed with respect to the original graph ($r = 0$).

Graph corruption rate r	SWD ↓	Relative SWD degradation
0.0	$\mathbf{0.02798 \pm 0.00917}$	–
0.3	0.03266 ± 0.00445	$+16.73\%$
0.6	0.03402 ± 0.00340	$+21.58\%$
0.9	0.03078 ± 0.00542	$+10.02\%$
1.0	0.03116 ± 0.00404	$+11.37\%$

As shown in Table 1, under the sparse temporal setting where only 11 of the 50 marginals are available for training, GraphFM achieves the lowest MMD and SWD among all compared methods. One possible explanation is that pairwise MFM relies on local interval-wise geometry, while MMFM constructs intermediate paths from splines anchored at the observed marginals; when temporal observations are sparse, the resulting paths are therefore more strongly determined by local information or the spline prior. In contrast, OT-ALI-CFM and GraphFM directly learn flexible interpolants under distribution-level supervision from the observed marginals, allowing the retained snapshots to jointly constrain a globally defined path. The further improvement of GraphFM over OT-ALI-CFM suggests an additional benefit from incorporating the known variable-interaction graph into both path construction and vector-field learning. Consistently, graph-corruption ablation shows that perturbing the graph degrades SWD by 10.0%–21.6%, supporting the contribution of informative graph structure (see Table 2).

4.2 SINGLE-CELL TRAJECTORY INFERENCE

We next evaluate GraphFM on the reconstruction of cellular dynamics from time-series single-cell RNA sequencing data. Gene regulatory interactions provide a natural structured prior for this problem, as transcriptional dynamics are jointly shaped by interactions among genes (Wang et al., 2026). Following the setup of Cao et al. (2025), we collect external reference data to determine the gene regulatory network, which is then incorporated into GraphFM as the variable interaction structure; details of graph construction are provided in Appendix A.3.3.

We compare GraphFM with baseline methods on the Embryoid Body (EB) (Moon et al., 2019) and Multiome (Multi) datasets (Lance et al., 2021). Both datasets are processed following standard single-cell preprocessing procedures, with full details provided in Appendix A.3.3. We retain the variables represented in the regulatory graph and use their resulting expression profiles as the state space for trajectory inference. We evaluate all methods under a leave-one-out interpolation setting, where each intermediate time point is held out in turn and a separate model is trained on the remaining temporal marginals. The learned vector field is then integrated to the held-out time point, and the predicted cell distribution is compared with the corresponding observed marginal using the Earth Mover's Distance (EMD).

As shown in Table 3, GraphFM achieved the lowest EMD on the Multi dataset, reducing the error by 7.0%, 6.3%, and 9.7% relative to MFM, OT-MMFM, and OT-ALI-CFM, respectively, suggesting that the directed graph prior provides useful structural constraints for modeling complex high-dimensional population dynamics. On EB, GraphFM remained competitive with the best-performing MFM baseline, with a difference of less than 1%, while outperforming OT-MMFM and

Table 3: Performance on the EB and Multi single-cell temporal benchmarks. For each random seed, EMD is averaged over three held-out transitions for EB and two held-out transitions for Multi (smaller is better).

Method	EB EMD ↓	Multi EMD ↓
OT-MFM	$\mathbf{1.3621 \pm 0.0127}$	8.5482 ± 0.2305
OT-MMFM	1.3842 ± 0.0092	8.4844 ± 0.0110
OT-ALI-CFM	1.5248 ± 0.0337	8.8042 ± 0.0866
GraphFM (ours)	1.3711 ± 0.0532	$\mathbf{7.9472 \pm 0.0604}$

Table 4: Tumor interpolation performance averaged over the test tasks (mean ± standard deviation over five seeds; smaller is better).

Method	EMD ↓
OT-MFM	7.98817 ± 0.01482
OT-MMFM	7.86642 ± 0.00336
OT-ALI-CFM	7.63725 ± 0.07477
GraphFM (ours)	$\mathbf{7.51914 \pm 0.08915}$

OT-ALI-CFM. The difference between EB and Multi may partly reflect the fidelity of the supplied regulatory graph. Since GRN inference remains challenging and regulatory interactions can vary across cell states and lineages (Pratapa et al., 2020; Zhang et al., 2023), a single global network may provide only a coarse approximation for the highly branched EB differentiation process (Moon et al., 2019), whereas the graph used for Multi may better capture the regulatory dependencies relevant to its observed transitions.

4.3 Transcriptomic State Interpolation Across Serial Tumour Sections

Spatial transcriptomics (ST) measures gene expression while preserving the spatial organization of tissue, enabling transcriptional heterogeneity to be characterized across different tumour regions Ståhl et al. (2016). When ST is collected from serial tissue sections, each section provides a population-level observation of transcriptional states. However, only a limited number of sections are typically observed, leaving intermediate transcriptional states unresolved. Inferring gene-expression distributions at intermediate sections can therefore provide a more continuous view of transcriptional changes across the tumour.

We consider the breast cancer dataset from Mo et al. (2024), consisting of serial spatial transcriptomic sections sampled from the same tumour tissue block. We focus on reconstructing the distribution of tumour transcriptional states across sections. Specifically, each tumour spot is represented by its gene-expression profile, and the ordered section depth is treated as a continuous interpolation coordinate. Given expression distributions observed at a subset of sections, the task is to recover the transcriptional distribution at an unobserved intermediate section. Averaged across the two held-out-section tasks, GraphFM achieved the lowest EMD (7.519 ± 0.089) as shown in Table 4, outperforming OT-ALI-CFM, OT-MMFM, and OT-MFM. This improvement suggests that externally curated gene interactions provide a useful inductive bias for reconstructing an unobserved tumor section when only a small number of serial sections is available.

4.4 Spatiotemporal Traffic Data Imputation

The imputation of traffic data is a well-established problem in intelligent transportation systems, aiming to recover unobserved traffic measurements from available spatio-temporal observations Shang et al. (2018); Zhang et al. (2024). Of particular interest is the within-day evolution of traffic states, as traffic measurements exhibit systematic temporal patterns across clock time Crawford et al. (2017). We study the evolution of network-wide traffic states during the morning period, considering observations from 06:00 to 12:00 at 30-minute intervals, using the PEMS-BAY traffic speed dataset (Li et al., 2018). At each clock time, observations from different workdays form an empirical time-indexed marginal distribution. Given such marginals at observed clock times, our

Table 5: Traffic-speed interpolation under two missing-timepoint settings. Results are averaged over the 09:00, 10:00, and 11:00 test tasks and reported as mean ± s.d. over five seeds. Lower is better.

Setting	Method	MAE ↓	MSE ↓	EMD ↓
LOO	OT-MFM	3.101 ± 0.054	35.430 ± 0.658	35.806 ± 0.503
	OT-MMFM	3.101 ± 0.070	35.589 ± 0.567	35.976 ± 0.488
	OT-ALI-CFM	3.714 ± 0.421	47.678 ± 8.052	42.891 ± 4.669
	GraphFM	$\mathbf{2.837 \pm 0.010}$	$\mathbf{34.714 \pm 0.237}$	$\mathbf{34.930 \pm 0.137}$
Joint	OT-MFM	3.029 ± 0.044	$\mathbf{34.572 \pm 0.926}$	35.186 ± 0.553
	OT-MMFM	3.036 ± 0.044	35.597 ± 0.265	35.713 ± 0.218
	OT-ALI-CFM	3.435 ± 0.185	42.668 ± 2.579	39.913 ± 2.157
	GraphFM	$\mathbf{2.844 \pm 0.009}$	34.884 ± 0.116	$\mathbf{34.992 \pm 0.034}$

LOO: leave-one-timepoint-out; Joint: jointly held-out.

goal is to recover the traffic-state distribution at an unobserved intermediate time. We construct the sensor graph following Li et al. (2018), where nodes correspond to traffic sensors and directed edges are derived from road-network distances between sensors (details provided in Appendix A.3.5).

We evaluate two settings: (i) leave-one-timepoint-out, where 09:00, 10:00, and 11:00 are held out in turn with separate model training; and (ii) jointly held-out, where all three test times are removed simultaneously. Each target is independently predicted from its immediately preceding observed time point. Mean absolute error (MAE) and mean squared error (MSE) measure element-wise reconstruction accuracy over these date-matched sensor observations. In addition, EMD compare the empirical distributions of the predicted and observed 55-dimensional traffic states. As shown in Table 5, across the two evaluation setting, GraphFM achieved the best performance in five out of the six metric–setting combinations. In the leave-one-timepoint-out setting, GraphFM reduced MAE, MSE, and EMD by 8.48%, 2.02%, and 2.45%, respectively, relative to the strongest baseline for each metric. In the jointly held-out setting, it reduced MAE and EMD by 6.09% and 0.55%, respectively. Overall, these results suggest that incorporating the directed road-network structure provides a useful inductive bias for reconstructing unobserved traffic states under sparse temporal observations, leading to stable interpolation across different missing-timepoint settings.

5 Discussion

We introduced GraphFM, a graph-constrained flow-matching framework for reconstructing continuous population dynamics from sparse temporal marginals. Motivated by the fact that dynamical evolution is often governed by structured interactions among variables, GraphFM incorporates a variable graph into both the conditional interpolant and the learned velocity field through graph-local message passing. In addition, a hierarchical interpolation objective regularizes the learned paths at the levels of marginal consistency, anchor-conditioned trajectory geometry, and local temporal smoothness. Across synthetic and real-world datasets, incorporating graph information consistently improves the recovery of held-out intermediate distributions, suggesting that structural knowledge about variable interactions can provide a useful inductive bias for otherwise underdetermined temporal interpolation. More broadly, GraphFM establishes a general paradigm for augmenting flow matching with external information, enabling domain-specific interaction knowledge to be incorporated directly into both trajectory construction and dynamics learning.

A current limitation is that the quality of the inferred dynamics depends on the reliability of the supplied variable graph. Misspecified or incomplete graph structures may restrict information propagation in ways that are inconsistent with the underlying system. An important direction for future work is therefore to jointly learn or refine the variable graph together with the interpolant and velocity field, yielding a unified framework in which structural dependencies and continuous dynamics are inferred simultaneously from data.

AI USE STATEMENT

Generative AI tools were used as auxiliary tools for manuscript writing. The core conceptual framework, research hypotheses, experimental design, and interpretation of results were developed independently by the authors. All AI-assisted content was reviewed and verified by the authors, who take full responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

The main text and appendix provide the methodological and experimental details required to reproduce our results. Upon acceptance, we will publicly release the implementation of GraphFM together with the scripts used for all experiments.

REFERENCES

Sam Abbott and Sebastian Funk. Estimating epidemiological quantities from repeated cross-sectional prevalence measurements. *MedRxiv*, pp. 2022–03, 2022.

Zhi-Jie Cao, Tianzi Guo, Cheng Li, Fan Zhang, Ping Lu, Chan Gu, Fan Guo, and Ge Gao. Causality-oriented regulatory inference and rational transcriptomic reprogramming design with cascade. *bioRxiv*, pp. 2025–11, 2025.

Fiona Crawford, DP Watling, and RD Connors. A statistical method for estimating predictable differences between daily traffic flow profiles. *Transportation Research Part B: Methodological*, 95:196–213, 2017.

Jun Ding, Nadav Sharon, and Ziv Bar-Joseph. Temporal modelling using single-cell transcriptomics. *Nature Reviews Genetics*, 23(6):355–368, 2022.

Christian LE Franzke, Terence J O'Kane, Judith Berner, Paul D Williams, and Valerio Lucarini. Stochastic climate theory and modeling. *Wiley Interdisciplinary Reviews: Climate Change*, 6(1): 63–78, 2015.

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 (NeurIPS)*, 2024.

Oskar Kviman, Kirill Tamogashev, Nicola Branchini, Víctor Elvira, Jens Lagergren, and Nikolay Malkin. Multi-marginal flow matching with adversarially learnt interpolants. In *International Conference on Learning Representations*, volume 2026, pp. 125197–125217, 2026.

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 *Neural Information Processing Systems (NeurIPS) Competitions and Demonstrations*, 2021.

Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In *International Conference on Learning Representations (ICLR)*, 2018.

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

Chia-Kuei Mo, Jingxian Liu, Siqi Chen, Erik Storrs, Andre Luiz N Targino da Costa, Andrew Houston, Michael C Wendl, Reyka G Jayasinghe, Michael D Iglesia, Cong Ma, et al. Tumour evolution and microenvironment interactions in 2d and 3d space. *Nature*, 634(8036):1178–1186, 2024.

Kevin R Moon, David Van Dijk, Zheng Wang, Scott Gigante, Daniel B Burkhardt, William S Chen, Kristina Yim, Antonia van den Elzen, Matthew J Hirn, Ronald R Coifman, et al. Visualizing structure and transitions in high-dimensional biological data. *Nature biotechnology*, 37(12):1482–1492, 2019.

Aditya Pratapa, Amogh P Jalihal, Jeffrey N Law, Aditya Bharadwaj, and T M Murali. Benchmarking algorithms for gene regulatory network inference from single-cell transcriptomic data. *Nature methods*, 17(2):147–154, 2020.

Martin Rohbeck, Edward De Brouwer, Charlotte Bunne, Jan-Christian Huetter, Anne Biton, Kelvin Chen, Aviv Regev, and Romain Lopez. Modeling complex system dynamics with flow matching across time and conditions. In *International Conference on Learning Representations (ICLR)*, volume 2025, pp. 12164–12188, 2025.

Wouter Saelens, Robrecht Cannoodt, Helena Todorov, and Yvan Saeys. A comparison of single-cell trajectory inference methods. *Nature biotechnology*, 37(5):547–554, 2019.

Qiang Shang, Zhaosheng Yang, Song Gao, and Derong Tan. An imputation method for missing traffic data based on fcm optimized by pso-svr. *Journal of Advanced Transportation*, 2018(1): 2935248, 2018.

Md Shahriar Rahim Siddiqui, Moshe Eliasof, and Eldad Haber. Graph flow matching: Enhancing image generation with neighbor-aware flow fields. In *Association for the Advancement of Artificial Intelligence (AAAI)*, 2026.

Patrik L Ståhl, Fredrik Salmén, Sanja Vickovic, Anna Lundmark, José Fernández Navarro, Jens Magnusson, Stefania Giacomello, Michaela Asp, Jakub O Westholm, Mikael Huss, et al. Visualization and analysis of gene expression in tissue sections by spatial transcriptomics. *Science*, 353 (6294):78–82, 2016.

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*, 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 *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 1279–1287, 2024b.

Weixu Wang, Zhiyuan Hu, Philipp Weiler, Sarah Mayes, Marius Lange, Daniel M Fountain, Julianna O Haug, Jingye Wang, Zhengyuan Xue, Tatjana Sauka-Spengler, et al. Regvelo: gene-regulatory-informed dynamics of single cells. *Cell*, 189(12):3773–3800, 2026.

F Alexander Wolf, Philipp Angerer, and Fabian J Theis. Scanpy: large-scale single-cell gene expression data analysis. *Genome biology*, 19(1):15, 2018.

Chenglei Yu, Chuanrui Wang, Bangyan Liao, and Tailin Wu. Pace: Geometry-aware bridge transport for single-cell trajectory inference. *arXiv preprint arXiv:2605.18587*, 2026.

Shilu Zhang, Saptarshi Pyne, Stefan Pietrzak, Spencer Halberg, Sunnie Grace McCalla, Alireza Fotuhi Siahpirani, Rupa Sridharan, and Sushmita Roy. Inference of cell type-specific gene regulatory networks on cell lineages from single cell omic datasets. *Nature Communications*, 14(1):3064, 2023.

Yimei Zhang, Xiangjie Kong, Wenfeng Zhou, Jin Liu, Yanjie Fu, and Guojiang Shen. A comprehensive survey on traffic missing data imputation. *IEEE Transactions on Intelligent Transportation Systems*, 25(12):19252–19275, 2024.

Zepeng Zhang, Aref Einizade, Jhony H. Giraldo, and Olga Fink. Spatiotemporal imputation with graph-informed flow matching. In *International Conference on Machine Learning (ICML)*, 2026.

A APPENDIX

A.1 PROOF OF PROPOSITION 1

Proof. For node i, the endpoint-constrained interpolant in Eq. equation 3 is

$$G_{\phi,i}(x_0, x_1, t; \mathcal{G}) = (1-t)x_{0,i} + tx_{1,i} + t(1-t)R_{\phi,i}(x_0, x_1, \mathcal{G}, t).$$

For any node $j \notin \mathcal{N}_i^{(L)}$, differentiating with respect to the endpoint features gives

$$\frac{\partial G_{\phi,i}}{\partial x_{0,j}} = (1-t)\delta_{ij} + t(1-t)\frac{\partial R_{\phi,i}}{\partial x_{0,j}}, \qquad \frac{\partial G_{\phi,i}}{\partial x_{1,j}} = t\delta_{ij} + t(1-t)\frac{\partial R_{\phi,i}}{\partial x_{1,j}}, \tag{14}$$

where the arguments of $G_{\phi,i}$ and $R_{\phi,i}$ are omitted for brevity and δ_{ij} denotes the Kronecker delta.

Since R_ϕ is an L-local message-passing GNN, $j \notin \mathcal{N}_i^{(L)}$ implies $\partial R_{\phi,i}/\partial x_{0,j} = 0$ and $\partial R_{\phi,i}/\partial x_{1,j} = 0$. Moreover, because $i \in \mathcal{N}_i^{(L)}$, we have $j \neq i$ and hence $\delta_{ij} = 0$.

Substituting these relations into equation 14 yields

$$\frac{\partial G_{\phi,i}}{\partial x_{0,j}} = \frac{\partial G_{\phi,i}}{\partial x_{1,j}} = 0, \qquad j \notin \mathcal{N}_i^{(L)}.$$

Therefore, the linear endpoint terms introduce no cross-node dependencies, while the residual term inherits the L-local dependency structure of R_ϕ. Hence, the complete interpolant G_ϕ preserves the graph-local dependency specified by $\mathcal{G}$. □

A.2 SYNTHETIC DATA GENERATION

We generated a synthetic temporal dataset on a fixed directed graph. Let $G = (V, E)$ be a directed graph with $|V| = 50$ nodes. The graph was sampled from a three-community directed stochastic block model with community sizes

$$(17, 17, 16).$$

For nodes i and j belonging to communities c_i and c_j, respectively, we sampled the directed edge $i \to j$ with probability

$$\mathbb{P}(A_{ij} = 1) = B_{c_i c_j},$$

where self-loops were removed and

$$B = \begin{pmatrix} 0.25 & 0.12 & 0.02 \\ 0.02 & 0.25 & 0.12 \\ 0.12 & 0.02 & 0.25 \end{pmatrix}.$$

We repeatedly sampled the graph until it was strongly connected. We then defined the incoming-neighbor propagation matrix as $P_{\text{in}} = D_{\text{in}}^{-1} A^\top$, where D_{in} is the diagonal matrix of node in-degrees. Thus, for node states $x(t) \in \mathbb{R}^{50}$, the aggregated incoming-neighbor signal is $x(t)P_{\text{in}}^\top$.

For each trajectory n, we sampled the initial condition from a standard Gaussian distribution,

$$x^{(n)}(0) \sim \mathcal{N}(0, I_{50}), \qquad n = 1, \ldots, 5000.$$

The temporal evolution was generated by a nonlinear, non-autonomous dynamical system defined on the directed graph:

$$\frac{dx}{dt} = \rho\left[\alpha(t)\left(xP_{\text{in}}^\top - x\right) + \gamma \sin(2\pi t)\tanh\left(xP_{\text{in}}^\top\right)\right] + \beta\tanh(x).$$

The dynamics combine a time-varying directed diffusion term, a time-modulated nonlinear graph-coupling term, and a node-wise nonlinear intrinsic-dynamics term, thereby introducing graph-dependent, nonlinear, and explicitly time-varying evolution.

The time-varying diffusion coefficient was defined as

$$\alpha(t) = \alpha_0 + \alpha_1 \sin^2(\pi t). \tag{15}$$

We set

$$\rho = 1.0, \qquad \alpha_0 = 0.2, \qquad \alpha_1 = 1.0, \qquad \beta = 0.2, \qquad \gamma = 0.3.$$

Here, ρ controls the overall scale of graph-mediated interactions, α_0 specifies the baseline diffusion strength, and α_1 determines the amplitude of its temporal variation. The coefficient γ controls the contribution of the nonlinear incoming-neighbor interaction, whereas β determines the strength of the local node-wise nonlinear dynamics. This produced a complete ground-truth tensor $X \in \mathbb{R}^{5000\times 50\times 50}$, where the three axes correspond to trajectories, time points, and graph nodes, respectively.

To construct the unpaired temporal marginal dataset used for training, we retained only 11 visible time marginals,

$$\mathcal{T}_{\text{vis}} = \{0, 5, 10, 15, 20, 25, 30, 35, 40, 45, 49\}.$$

For each visible time t_k, we formed an unordered empirical marginal distribution

$$\mathcal{X}_{t_k} = \left\{ x^{(n)}(t_k) \right\}_{n=1}^{5000}.$$

The complete paired trajectories were stored separately and used only for evaluation and visualization, not for training.

A.3 Additional Experimental Details and Results

A.3.1 Squared sliced-Wasserstein distance

Let

$$\widehat{P} = \frac{1}{B}\sum_{b=1}^{B} \delta_{\widehat{x}_b}, \qquad \widehat{Q} = \frac{1}{B}\sum_{b=1}^{B} \delta_{x_b},$$

denote two empirical distributions with the same number of samples, where $\widehat{x}_b, x_b \in \mathbb{R}^d$. The squared sliced-Wasserstein distance is defined as

$$\mathrm{SW}_2^2(\widehat{P}, \widehat{Q}) = \mathbb{E}_{\omega\sim\mathcal{U}(\mathbb{S}^{d-1})}\left[W_2^2\left((P_\omega)_\# \widehat{P}, (P_\omega)_\# \widehat{Q} \right) \right], \tag{16}$$

where

$$P_\omega(x) = \omega^\top x$$

projects a d-dimensional sample onto the one-dimensional direction $\omega \in \mathbb{S}^{d-1}$.

In practice, we approximate Eq. equation 16 using L randomly sampled projection directions. Specifically, we sample

$$g_\ell \sim \mathcal{N}(0, I_d), \qquad \omega_\ell = \frac{g_\ell}{\|g_\ell\|_2}, \qquad \ell = 1, \dots, L.$$

For each direction, let

$$\widehat{s}_{\ell,(1)} \le \cdots \le \widehat{s}_{\ell,(B)} \quad \text{and} \quad s_{\ell,(1)} \le \cdots \le s_{\ell,(B)}$$

denote the sorted projected values $\{\omega_\ell^\top \widehat{x}_b\}_{b=1}^B$ and $\{\omega_\ell^\top x_b\}_{b=1}^B$, respectively. The empirical squared sliced-Wasserstein distance is then

$$\widehat{\mathrm{SW}}_2^2(\widehat{P}, \widehat{Q}) = \frac{1}{LB}\sum_{\ell=1}^{L}\sum_{b=1}^{B} \left(\widehat{s}_{\ell,(b)} - s_{\ell,(b)} \right)^2. \tag{17}$$

For the marginal-consistency objective at time t_m, we set

$$\widehat{x}_b = G_\phi\left(x_0^{(b)}, x_1^{(b)}, t_m; \mathcal{G} \right), \qquad x_b \sim q_{t_m}.$$

Thus, Eq. equation 17 compares the generated and observed populations after sorting their one-dimensional projections.

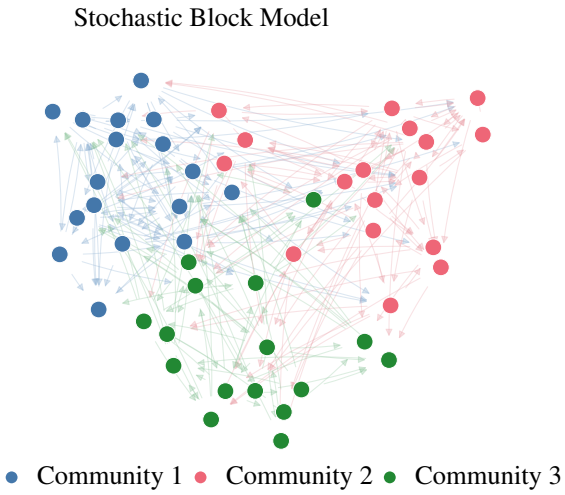


Figure 2: Stochastic block model for synthetic data generation.

A.3.2 Synthetic Data Experiment

Training and evaluation protocol. The complete synthetic dataset contains 50 temporal marginals, of which 11 time points,

$$\{0, 5, 10, 15, 20, 25, 30, 35, 40, 45, 49\},$$

were exposed during training. Four additional hidden time points, $\{8, 18, 28, 38\}$, were used for validation and model selection, while the remaining 35 unobserved internal time points were reserved exclusively for final evaluation. At each visible time point, the 5,000 marginal samples were independently split into 4,500 training samples and 500 validation samples. The split was performed independently for each temporal marginal. All methods were evaluated using three random seeds.

GraphFM. GraphFM was trained in two stages. First, we optimized an endpoint-constrained directed graph neural interpolant. The interpolant used a hidden width of 128 and four residual directed message-passing blocks, with incoming messages aggregated. For each mini-batch, we sampled one intermediate visible time point and constructed pseudo-triplets (x_0, x_t, x_1) using ALI-style MMOT coupling. The interpolant was trained with an SWD marginal-matching loss using 128 random projections, plus a piecewise path regularizer with coefficient 0.5 and a curvature regularizer with coefficient 10^{-3}. The curvature term used a second-order finite difference with $h = 0.02$ and was evaluated every four steps. We used AdamW with batch size 256, learning rate 10^{-3}, weight decay 10^{-4}, gradient clipping at 1.0, 500 warmup steps, and cosine decay to 10^{-5}. Training was run for up to 12,000 steps, with validation every 250 steps using 1,024 fixed SWD projections and early stopping after 12 validation checks without improvement.

The vector field received only the current state and time as input, not the terminal endpoint. Supervision was generated from the frozen interpolant. During inference, we discarded the interpolant and generated target-time distributions by integrating $dx_t/dt = v_\theta(x_t, t; \mathcal{G})$ from the empirical initial marginal.

OT-ALI-CFM. We used the official implementation with its 50-dimensional single-cell configuration. Inputs were min–max normalized, and all models were trained with a batch size of 128. The interpolant network used a hidden width of 1024 and was optimized with the R3GAN adversarial objective. We set the temporal smoothing parameter to $t_{\text{smooth}} = 0.01$, the correction coefficient to 0.5, and the learning rates of both the generator and discriminator to 5×10^{-5}. Multi-marginal OT pretraining was performed for 2,001 steps, followed by 70,001 adversarial training steps and 2,001 conditional flow-matching steps.

OT-MMFM. We used the official MMFM implementation with cubic multi-marginal interpolation. Hyperparameters were selected on seed 1 using the hidden validation marginals and validation SWD. We searched the latent width/time embedding dimension over $\{32, 64, 128\}$, the learning rate over $\{2 \times 10^{-2}, 10^{-2}, 10^{-3}, 10^{-4}\}$, and the flow variance over $\{1.0, 0.1, 0.01\}$. Models were trained for 300 epochs using Adam with a cosine learning-rate scheduler, LeakyReLU activations, and Xavier-normal initialization. The selected configuration was subsequently trained using all three evaluation seeds.

OT-MFM. We used the official metric flow matching implementation with an RBF-based manifold metric and pairwise mini-batch OT coupling between adjacent temporal marginals. Following the high-dimensional single-cell configuration, both the GeoPath network and the velocity network consisted of three hidden layers with width 1024. The manifold metric used 150 RBF centers with $\kappa = 1.5$ and $\rho = -2.75$ and was trained for 200 epochs. The batch size was 128 and the path-noise scale was set to $\sigma = 0.1$. The GeoPath and velocity networks were optimized with learning rates of 10^{-4} and 10^{-3}, respectively, for up to 1,000 epochs. For all three seeds, final predictions were generated from the corresponding trained checkpoints and evaluated on the same 35 hidden test marginals.

A.3.3 SINGLE-CELL TRAJECTORY INFERENCE

All methods were evaluated using the same 100 highly variable genes (HVG) and the same leave-one-time-point-out tasks. For EB, we independently held out the three interior marginals corresponding to Days 06–09, 12–15, and 18–21; for Multi, we held out Days 3 and 4. Each model was trained separately for each held-out marginal and each of five random seeds. During evaluation, trajectories were initialized from the complete empirical marginal immediately preceding the held-out time, rather than from the first time point. For all methods, the learned vector field was integrated using 100 explicit Euler updates over each evaluation interval.

Preprocessing the scRNA-seq data The embryoid body (EB) dataset contained 18,203 cells and 17,789 genes collected over five intervals: Days 00–03, 06–09, 12–15, 18–21 and 24–27. The Multi dataset contained 33,701 cells and 23,418 genes sampled on Days 2, 3, 4 and 7. Gene symbols for the EB dataset were obtained directly from the "gene_name" annotation. For the Multi dataset, version suffixes were removed from Ensembl identifiers before they were mapped to human gene symbols; the original Ensembl identifier was retained when no gene-symbol mapping was available. Cells were grouped according to "sample_labels" for EB and day for Multi, and all cells at each sampling time were retained without downsampling. To obtain a common feature dimension, we identified the 100 HVGs separately for each dataset using the Seurat dispersion procedure implemented in 'scanpy.pp.highly_variable_genes' (Wolf et al., 2018). The resulting gene lists were used in a fixed order to extract the corresponding 100-dimensional expression matrices.

Construction of the prior graph Following CASCADE (Cao et al., 2025), we construct an experiment-specific scaffold by integrating KEGG pathway relations, ENCODE/GeneHancer TF–target links, BioGRID protein–protein interactions, and GTEx-derived gene–gene correlations. For each experiment, the source graphs are aligned and marginalized with respect to the genes included in the dataset. Directed KEGG and TF–target edges retain their orientations, whereas undirected BioGRID and GTEx edges are expanded into both directions. We remove self-loops and take the union of all retained edges, while preserving unmatched genes as isolated nodes. The resulting binary matrix satisfies $A_{ij} = 1$ when $i \to j$ is supported as a candidate connection by at least one source.

GraphFM. Expression values were normalized feature-wise to $[0, 1]$, with the minimum and maximum estimated exclusively from the visible marginals. Separate gene graphs were constructed for EB and Multi by taking the union of TF–target, BioGRID, KEGG, and expression-correlation edges. The resulting graphs contained 365 and 585 directed edges, respectively. An edge $A_{ij} = 1$ represented information flow from gene i to gene j, and the incoming propagation matrix was obtained by transposing A and normalizing each row by the corresponding target in-degree. No self-loops were added.

The model was optimized in two stages. We first trained an endpoint-constrained directed GNN interpolant with four message-passing blocks and a hidden width of 256. At each step, an intermediate visible marginal was sampled and an ALI-style pseudo-triplet was constructed by exact mini-batch optimal transport between the initial, intermediate, and terminal marginals. The interpolant loss combined squared sliced Wasserstein distance, computed using 128 random projections, with piecewise-linear path regularization and finite-difference curvature regularization. The path and curvature terms were weighted by 0.5 and 10^{-3}, respectively, with curvature evaluated every four steps using a finite-difference interval of 0.02. The interpolant was trained for 12,000 steps using AdamW, a batch size of 128, weight decay 10^{-4}, and a learning rate schedule consisting of 500 warm-up steps followed by cosine decay from 10^{-3} to 10^{-5}. We then froze the interpolant and used its states and temporal derivatives as supervision for a graph-aware flow-matching vector field, which also contained four message-passing blocks of width 256. The vector field was trained for 4,000 steps with the mean-squared flow-matching loss, AdamW, a batch size of 128, and a learning rate of 10^{-3}. Final-step checkpoints from both stages were retained.

OT-ALI-CFM. We adapted the official OT-ALI-CFM implementation to the ordered HVG100 expression matrices and trained an independent model for every held-out marginal and random seed. The interpolant, discriminator, and conditional flow-matching networks each used a hidden width of 1024, and all stages used a batch size of 128. We set the temporal smoothing parameter to $t_{\mathrm{smooth}} = 0.01$ and the correction coefficient to 0.5. The piecewise path-regularization coefficients were 10^{-4} for EB and 10^{-3} for Multi.

Training consisted of three consecutive stages. First, the interpolant was initialized by 2,001 steps of full optimal-transport pretraining with a learning rate of 10^{-3}. It was then optimized for 70,001 steps using the R3GAN adversarial objective and ALI-style multi-marginal OT pseudo-couplings; the generator and discriminator learning rates were both 5×10^{-5}. Finally, the conditional flow-matching vector field was trained for 2,001 steps using multi-marginal OT coupling and a learning rate of 10^{-3}. The resulting flow checkpoints were evaluated using the common previous-to-target Euler-100 protocol described above.

OT-MFM. We used the official metric flow matching implementation with the 100-dimensional HVG expression matrices. No additional whitening or feature-wise normalization was applied. Exact pairwise mini-batch OT was used to couple adjacent visible marginals. Both the GeoPath network and the velocity network consisted of three hidden layers of width 1024 with SELU activations, and the conditional path-noise scale was set to $\sigma = 0.1$.

The data-manifold metric was represented by an RBF network with 150 centers, $\kappa = 1.5$, and $\rho = -2.75$. The RBF metric was optimized with Adam at a learning rate of 10^{-2} for up to 200 epochs. The GeoPath network was then trained using Adam with a learning rate of 10^{-4}, whereas the velocity network was trained using AdamW with a learning rate of 10^{-3}. Both networks used a batch size of 128 and were trained for up to 1,000 epochs, with validation-based early stopping.

OT-MMFM. We adapted the official MMFM implementation using cubic multi-marginal interpolation. We used a latent width of 128, a time-embedding dimension of 32, a learning rate of 10^{-3}, and a flow variance of 0.01. Models were trained for 300 epochs using Adam, cosine learning-rate decay, LeakyReLU activations, and Xavier-normal initialization. Unequal-sized visible marginals were padded by sampling with replacement and coupled using the official exact-OT procedure in blocks of 128 cells. We report the flow-rollout EMD averaged over seeds.

A.3.4 TRANSCRIPTOMIC STATE INTERPOLATION

Tumour-region definition and expression preprocessing. We used four serial spatial transcriptomic sections, denoted U2, U3, U4, and U5, from the HT206B1 breast tumour dataset (Mo et al., 2024). Following the spatial-transcriptomic preprocessing convention used by ALI-CFM (Kviman et al., 2026), we defined tumour-enriched regions by retaining spots with `Tumor_purity_estimate` > 0.8. This resulted in 755, 870, 945, and 1,034 tumour spots for U2, U3, U4, and U5, respectively, giving 3,604 spots in total. No additional filtering based on the number of detected genes, total counts, or mitochondrial content was applied.

Gene-expression values were taken directly from the existing SCTransform assay in the original Seurat objects. Specifically, we used the log-transformed, depth-corrected expression values stored in the SCT data matrix. We did not rerun SCTransform, apply an additional log-normalization step, or standardize the combined dataset. We also did not perform batch correction or spatial registration, because the task concerns the interpolation of expression distributions rather than the correspondence between spatial coordinates across sections.

Selection of highly variable genes. Highly variable genes (HVGs) were identified separately within the tumour spots of each section. For each section, gene-wise means and dispersions were calculated from the existing SCT-normalized expression matrix, and the dispersions were normalized using the Seurat-style procedure for log-transformed expression. Genes were then ranked by normalized dispersion within each section.

The top 500 genes were selected independently for U2, U3, U4, and U5. The intersection of these four sets contained 278 genes and was stored as the Boolean annotation `hvg_intersection_500` in the H5AD file. All methods were trained and evaluated using these same 278 genes in the order in which they occurred in the H5AD object. The HVGs were therefore determined without pooling the four sections and without using a method-specific feature-selection procedure.

GraphFM. GraphFM used the same target-specific, visible-only min–max normalization as ALI-CFM. The interpolant and flow vector field were both parameterized by directed graph neural networks with a hidden dimension of 256 and four message-passing blocks. Directed graph propagation was applied inside both the interpolant and vector-field networks rather than as a one-off preprocessing transformation.

The interpolant was trained for 12,000 updates with a batch size of 128. Multi-marginal optimal transport coupling was used to construct the endpoint samples. Training used 128 random projections, an endpoint-correction coefficient of (0.5), and a curvature penalty with weight (10^{-3}), finite-difference interval (0.02), and evaluation frequency of once every four updates. AdamW was used with an initial learning rate of (10^{-3}), a linear warm-up of 500 updates, cosine decay to (10^{-5}), weight decay (10^{-4}), and gradient clipping at 1.

The graph-aware flow vector field was then trained for 4,000 updates using the learned interpolant and the same multi-marginal coupling. It used a batch size of 128, an initial learning rate of (10^{-3}), weight decay (10^{-4}), and gradient clipping at 1. All visible spots were used for training, and the final checkpoint was retained. Predictions were transformed back from the visible-only min–max scale to the SCT expression scale before EMD evaluation.

OT-ALI-CFM. OT-ALI-CFM was trained using the authors' spatial-transcriptomic implementation, with the data interface extended to read the 278-gene mask directly from the H5AD object. For each held-out task, a gene-wise min–max transformation was estimated exclusively from the visible sections. Predictions were transformed back to the original SCT expression scale before evaluation.

The interpolant, discriminator, and conditional flow-matching networks used hidden widths of 1,024, with a batch size of 128. The interpolant was initialized with 2,001 pretraining updates at a learning rate of (10^{-3}), followed by 70,001 adversarial updates using R3GAN and learning rates of (5×10^{-5}) for both the generator and discriminator. The conditional flow-matching vector field was subsequently trained for 2,001 updates with a learning rate of (10^{-3}). Multi-marginal optimal transport coupling was used for both interpolant and flow-matching training, and the endpoint-correction coefficient was fixed to (0.5). The regularization weights and all remaining optimization settings were retained from the original spatial-transcriptomic configuration.

OT-MFM. OT-MFM was trained using the existing metric flow-matching implementation. It operated directly on the SCT expression matrix without additional whitening or standardization. Each held-out experiment used an independent model, and the visible observations were divided into a (90%/10%) training–validation split. The excluded target section did not participate in either partition.

Both the geodesic-path model and the flow model consisted of three hidden layers of width 1,024 with SELU activations. The geodesic-path model was time-conditioned, and exact optimal transport was used to construct the training coupling. The data-dependent RBF metric used 150 centres, with

($\kappa = 1.5$), ($\rho = -2.75$), and metric weight ($\alpha = 1$). The metric was fitted for up to 200 epochs using only the visible training data.

The geodesic-path and flow models were trained for at most 1,000 epochs with early stopping. The geodesic-path model used Adam with a learning rate of (10^{-4}) and weight decay (10^{-5}), whereas the flow model used AdamW with a learning rate of (10^{-3}) and weight decay (10^{-5}). The batch size was 128, and the geodesic regularization coefficient was fixed to (0.2). Checkpoints were selected according to their performance on the visible validation subset.

OT-MMFM. OT-MMFM was implemented by adapting the official MMFM codebase rather than reimplementing its core loss or vector-field architecture. The models were trained directly on the SCT expression values without additional normalization. Because the ST sections contained unequal numbers of spots, we added a data adapter that formed repeated equal-sized, without-replacement coverage batches from consecutive visible sections. This ensured that every visible observation was used while preserving the assumptions of the original MMFM training procedure. Consecutive-section transport pairs were obtained by exact equal-weight optimal assignment under squared Euclidean cost.

We used the official cubic interpolation path, LeakyReLU activations, three output layers, and Xavier-normal initialization with gain 1.4. Optimization was performed with Adam for up to 300 epochs, together with a cosine learning-rate schedule having ($T_{\max} = 1000$) and a minimum learning rate of (10^{-4}). Checkpoints were saved every 25 epochs.

Hyperparameters were selected using seed 42 and only the visible (90%/10%) training–validation split. We searched over hidden widths $(32, 64, 128)$, time-embedding dimensions $(32, 64, 128)$, learning rates $(2 \times 10^{-2}, 10^{-2}, 10^{-3}, 10^{-4})$, and variance parameters (1,0.1,0.01). The selection criterion was the mean exact EMD between predicted and observed consecutive visible validation marginals. The held-out target section was loaded only after hyperparameter and checkpoint selection.

The selected configuration for the U3 task used a hidden width of 64, a time-embedding dimension of 64, a learning rate of (10^{-3}), and a variance parameter of (0.01). For the U4 task, the corresponding values were 128, 32, (2×10^{-2}), and (1). These configurations were then fixed and applied to all five formal random seeds. For consistency with the remaining methods, we evaluated only the learned flow rollout.

Construction of the prior graph. GraphFM used a fixed directed gene graph constructed for the same 278 genes. The graph was derived from the union scaffold provided by CASCADE (Cao et al., 2025), incorporating regulatory and functional evidence from TF–target interactions, protein–protein interactions, pathway annotations, and an externally derived GTEx correlation network. Consequently, no expression values from the held-out HT206B1 sections were used to construct the graph.

The resulting adjacency matrix contained 4,736 directed edges. It was asymmetric, contained no self-loops, and included 11 genes with zero in-degree and 16 genes with zero out-degree. An entry ($A_{ij} = 1$) represented an edge from gene (i) to gene (j). Incoming-neighbour aggregation was implemented using $P = D_{\text{in}}^{-1} A^{\mathsf{T}}$,

where rows corresponding to zero-in-degree genes were kept at zero. The graph gene order was verified to match the H5AD feature order exactly.

A.3.5 SPATIOTEMPORAL TRAFFIC-STATE INTERPOLATION

Traffic data and temporal marginals. We used traffic-speed measurements from the PEMS-BAY dataset (Li et al., 2018). The processed data comprised 130 workdays, 13 clock times from 06:00 to 12:00 at 30-minute intervals, and 55 traffic sensors. The resulting data tensor therefore had dimensions $130 \times 13 \times 55$. Traffic speeds were retained in their original units of miles per hour.

At each clock time, the 130 workday-specific observations were treated as samples from an empirical marginal distribution over the 55 sensors. Clock times were assigned normalized coordinates

$$t_k = \frac{k}{12}, \qquad k = 0, \ldots, 12,$$

with $t_0 = 0$ corresponding to 06:00 and $t_{12} = 1$ corresponding to 12:00. Date identifiers were retained only to evaluate paired prediction errors. They were not provided to the models, transport solvers, or graph-learning components.

Construction of the traffic graph. We constructed the traffic graph from the weighted directed adjacency matrix released with PEMS-BAY (Li et al., 2018). Let $\mathbf{W}^{(0)}$ denote this matrix and define its directed support as

$$B_{ij} = \mathbb{I}\Big[W_{ij}^{(0)} > 0\Big],$$

where $B_{ij} = 1$ indicates a road connection from sensor i to sensor j. Self-loops were removed, while edge directions were preserved without symmetrization or reverse-edge augmentation.

We applied Louvain community detection to the directed binary support using resolution 1.0 and seed 42, and retained the largest community. The data tensor and adjacency matrices were then restricted to these sensors while preserving their original order in the PEMS-BAY data matrix. This produced a graph with 55 sensors and 384 directed edges.

GraphFM. For each random seed and leave-one-timepoint-out task, traffic speeds were normalized sensor-wise to $[0, 1]$, with the minimum and maximum estimated exclusively from the visible data. GraphFM retained the fixed 55-node, 384-edge directed road-network support, while initializing its learnable edge strengths with the corresponding PEMS-BAY weights. An edge from sensor i to sensor j contributed to the incoming message of j, with propagation based on the in-strength-normalized matrix. Before model training, we constructed a single global multi-marginal optimal-transport coupling jointly across all visible time points. Rather than independently pairing consecutive marginals, this coupling matched the 130 samples at every visible time point simultaneously, producing a set of temporally consistent coupled paths. The resulting tensor, $\mathcal{C} \in \mathbb{R}^{130\times12\times55}$, contained 130 coupled paths across 12 visible clock times and 55 sensors.

The model was optimized in two stages. First, we trained an endpoint-constrained directed GNN interpolant with four message-passing blocks and hidden width 256. Each coupled trajectory supplied its first and last visible states as endpoints, while all internal visible states supervised the predicted path through squared sliced Wasserstein distance computed with 128 random projections. The objective additionally included piecewise-linear regularization between every pair of adjacent visible times and finite-difference curvature regularization, weighted by 0.5 and 10^{-3}, respectively. Curvature was evaluated every four updates using an interval of 0.02. The interpolant was trained for 12,000 updates using AdamW, batch size 128, weight decay 10^{-4}, gradient clipping at 1, and 500 warm-up updates followed by cosine learning-rate decay from 10^{-3} to 10^{-5}. We then froze the interpolant and trained a graph-aware flow-matching vector field with the same four-block, width-256 architecture. Its edge weights were initialized from the effective graph learned by the interpolant and remained trainable. Complete endpoint pairs were sampled from the cached coupling, and the interpolant states and temporal derivatives provided the flow-matching inputs and velocity targets. The flow model was trained for 4,000 updates using a mean-squared flow-matching loss, AdamW, batch size 128, learning rate 10^{-3}, weight decay 10^{-4}, and gradient clipping at 1.

OT-ALI-CFM. OT-ALI-CFM was trained using the authors' implementation, with the data interface adapted to read the traffic marginals from the H5AD object. The interpolant, discriminator, and conditional flow-matching networks used hidden widths of 1,024 and a batch size of 128.

The interpolant was initialized using 2,001 pretraining updates with a learning rate of 10^{-3}. This was followed by 70,001 adversarial updates using R3GAN and learning rates of 5×10^{-5} for both the generator and discriminator. The conditional flow-matching vector field was subsequently trained for 2,001 updates with a learning rate of 10^{-3}. Multi-marginal optimal transport coupling was used during interpolant and flow training, and the endpoint-correction coefficient was fixed to 0.5. All remaining optimization settings followed the original configuration.

OT-MFM. OT-MFM was trained using the metric flow-matching implementation. Both the geodesic-path model and flow vector field consisted of three hidden layers of width 1,024. The model operated on the visible-only min–max representation without additional whitening.

Exact optimal transport was used to construct the training coupling. The data-dependent radial-basis-function metric used 150 centres, with $\kappa = 1.5$, $\rho = -2.75$, and metric weight $\alpha = 1$. The metric was fitted for up to 200 epochs using only visible training data.

For methods requiring validation, the 130 dates were divided using a fixed date-level split into 117 training dates and 13 validation dates. The geodesic-path and flow models were trained for at most 1,000 epochs with early stopping. The batch sizes were adapted to the corresponding split sizes, and incomplete batches were retained. The target clock time was excluded from both partitions.

OT-MMFM. OT-MMFM was implemented by adapting the official MMFM codebase. The traffic adapter preserved the common ordering of the 55 sensors and the actual normalized coordinates of all visible clock times. All 130 observations at each visible time were used to construct equal-sized multi-time paths.

Hyperparameter and checkpoint selection used only the fixed visible training–validation split for seed 42. The held-out target was loaded only after model selection. The selected configuration was subsequently fixed for seeds 42–46. For consistency with the other flow-matching methods, the reported predictions were generated using the learned flow rollout.