STOCHASTIC FLOW MAP FOR COUNT DATA

Anonymous authors
Paper under double-blind review

ABSTRACT

High-dimensional count data are common in scientific applications, but most diffusion and flow models are designed for continuous or categorical data, and generation often requires many sequential model evaluations. We propose Count Flow Map, a generative model that learns finite-time transitions directly in count space for one- or few-step generation. Our model directly learns stochastic transitions over finite time intervals, using Poisson births and Binomial deaths to preserve nonnegative integer counts without a predefined maximum. These transition models are trained to match the underlying local birth–death dynamics and to maintain consistency across step sizes. We characterize the connection between local dynamics and finite-time transition consistency and derive a bound on the generation error. After validating Count Flow Map in several simulations, including a high-dimensional, high-count setting, we apply it to single-cell drug perturbation prediction and neural population forecasting, where it captures perturbation responses and supports forecasts of high-activity events with only one or a few model evaluations. Together, these experiments demonstrate that Count Flow Map enables high-quality generation directly in count space across inference budgets, from one-step to few-step generation, using a single trained model.

1 INTRODUCTION

High-dimensional count data arise in many scientific applications, including single-cell RNA sequencing and neural spike trains. Their discreteness, sparsity, and complex dependence structure make flexible joint modeling challenging, particularly when the count range is large. Recent advances in diffusion models and flow matching provide powerful tools for generative modeling (Ho et al., 2020; Lipman et al., 2023; Albergo et al., 2025), but their application to count data has two challenges. First, most existing methods are designed for continuous or categorical data. Applying continuous models to counts requires a continuous representation of the data, while categorical models treat each possible count as a separate category (Austin et al., 2021; Campbell et al., 2022; Gat et al., 2024). This categorical representation typically requires specifying a maximum count, and the model's output dimension grows with the supported count range. Second, generation typically requires many sequential evaluations of the learned dynamics, making sampling costly even when training is efficient.

Count-specific generative models have begun to address the representation problem, including count-FM, Count Bridges, and CountsDiff (Wei & Pearson, 2026; Fishman et al., 2026; Soatto et al., 2026). In particular, count-FM models transport between arbitrary source and target count distributions directly on $\mathbb{N}_0^d$ through local birth–death dynamics. Its conditional binomial bridge enables simulation-free training of the marginal transition rates, while its rate parameterization avoids enumerating possible count levels. Count-FM therefore provides a direct and parameter-efficient formulation for count-valued transport. However, it learns infinitesimal transition rates, and generation still requires repeatedly simulating short-time transitions of the learned jump process. The remaining challenge is to retain this count-valued formulation while reducing the number of model evaluations needed for generation. Flow Maps provide a way to address this sampling cost by learning finite-time evolution directly (Boffi et al., 2025b;a). For continuous dynamics, a two-time flow map predicts the state reached over a specified interval, allowing one- or few-step generation from the same trained model. This perspective connects to consistency models, consistency trajectory models, progressive distillation, and shortcut models (Song et al., 2023; Kim et al., 2024; Salimans & Ho, 2022; Frans et al., 2025). Recent Discrete Flow Maps and Categorical Flow Maps extend accelerated generation

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.

to categorical data through probability-simplex representations (Potaptchik et al., 2026; Roos et al., 2026). These developments motivate extending count-FM from learning local birth–death rates to learning the corresponding finite-time transitions, while preserving its unbounded count space.

We propose *Count Flow Map*, which extends count-FM from local birth–death dynamics to finite-time transitions, enabling one- or few-step generation directly in count space. Count Flow Map models stochastic transitions between pairs of time points using Poisson births and Binomial deaths, ensuring that samples remain nonnegative integers without imposing a predefined maximum count. The transition models are trained to recover count-FM's local birth–death dynamics in the infinitesimal-time limit and to remain consistent when longer transitions are composed from shorter ones. This allows Count Flow Map to support different inference budgets without retraining.

We evaluate Count Flow Map in 2-D and 32-D simulations and in two biological applications, single-cell drug perturbation prediction using Tahoe-100M (Zhang et al., 2025) and neural population forecasting using the recordings of Steinmetz et al. (2019). Overall, these experiments show that Count Flow Map achieves strong distributional quality at low sampling cost, requiring only one or a few model evaluations. Our main contributions are as follows:

- We formulate Count Flow Map using stochastic transition kernels on an unbounded count space and characterize their connection to local birth–death dynamics through the forward, backward, and Chapman–Kolmogorov consistency conditions.
- We develop a tractable transition model with Poisson births and Binomial deaths, with finite-time corrections that preserve the local birth–death dynamics. The training objective combines count-FM rate matching with transition consistency across time intervals.
- We derive a terminal-error bound that separates endpoint truncation, generator estimation, and finite-time approximation, and quantify how local approximation errors and transition inconsistencies affect finite-time accuracy.

2 METHOD

In this section, we formulate Count Flow Map, introduce a finite-time transition model for count data, derive the training objective, and describe one- and few-step generation. Proofs and additional derivations are given in Appendix A. A Python implementation of Count Flow Map is provided in the Supplementary Material.

2.1 COUNT FLOW MAP

Let $\mathcal{X} := \mathbb{N}_0^d$ denote the space of d-dimensional nonnegative count vectors, and let p_0 and p_1 be source and target distributions on $\mathcal{X}$. We consider a probability path $(p_t)_{t\in[0,1]}$ connecting them, with endpoints p_0 and p_1. For $0 \le s \le t \le 1$, a *Count Flow Map* is a Markov transition kernel

$$K_{s,t}(y \mid x) := \mathbb{P}(X_t = y \mid X_s = x), \qquad x, y \in \mathcal{X}, \tag{1}$$

that transports the path marginals, $p_t(y) = \sum_{x\in\mathcal{X}} p_s(x)K_{s,t}(y \mid x)$. Thus, for a fixed state x at time s, the finite-time map returns the distribution of the state at time t. For a function $f : \mathcal{X} \to \mathbb{R}$, we write $(K_{s,t}f)(x) := \sum_y K_{s,t}(y \mid x)f(y)$ and let I denote the identity kernel. The infinitesimal generator of the transition family is

$$Q_t := \lim_{h\downarrow 0} \frac{K_{t,t+h} - I}{h}. \tag{2}$$

We refer to Q_t as the diagonal generator, as it is determined by $K_{s,t}$ near the diagonal $s = t$. For each coordinate $i \in \{1, \dots, d\}$, we allow local unit transitions, i.e., a birth increases the ith count from x_i to $x_i + 1$, while a death decreases it from x_i to $x_i - 1$. Let e_i denote the ith standard basis vector. The resulting birth–death generator is

$$(Q_t f)(x) = \sum_{i=1}^{d} \lambda_{t,i}(x)\big[f(x + e_i) - f(x)\big] + \sum_{i=1}^{d} \mu_{t,i}(x)\big[f(x - e_i) - f(x)\big], \tag{3}$$

where $\lambda_{t,i}(x) \ge 0$ and $\mu_{t,i}(x) \ge 0$ are the coordinate-wise birth and death rates, with $\mu_{t,i}(x) = 0$ whenever $x_i = 0$.

For two kernels K and M, define their composition by $(KM)(y \mid x) = \sum_z K(z \mid x)M(y \mid z)$. The consistency conditions for deterministic Flow Maps (Boffi et al., 2025a) have the following stochastic counterparts:

$$\begin{aligned} \partial_t K_{s,t} &= K_{s,t}Q_t, && \text{Lagrangian (forward) consistency,} \\ \partial_s K_{s,t} &= -Q_s K_{s,t}, && \text{Eulerian (backward) consistency,} \\ K_{s,t} &= K_{s,u}K_{u,t}, \qquad s \le u \le t, && \text{semigroup consistency.} \end{aligned} \tag{4}$$

The first two are the forward and backward Kolmogorov equations, and the third is the Chapman–Kolmogorov identity (Norris, 1997; Feinberg et al., 2014). Appendix A.1 derives all three identities for the exact transition family and proves the converse characterization used below.

Let $P^Q_{s,t}(y \mid x)$ denote the transition kernel of the continuous-time Markov process generated by Q_t, i.e., the probability of reaching y at time t given x at time s.

Proposition 1 (Characterization of a Count Flow Map)**.** *Suppose that Q_t generates a unique nonexplosive transition family $P^Q_{s,t}$. Under the regularity conditions in Appendix A.1, a transition family $K_{s,t}$ with $K_{t,t} = I$ equals $P^Q_{s,t}$ if it satisfies either the forward equation, the backward equation, or the Chapman–Kolmogorov identity with infinitesimal generator Q_t.*

Thus, the diagonal generator Q_t specifies the local dynamics, while any one of the three consistency conditions identifies the corresponding finite-time transition family. For training, we use Chapman–Kolmogorov consistency $K_{s,t} = K_{s,u}K_{u,t}$ because it can be enforced directly through kernel composition.

2.2 Transition kernel for count data

We next introduce a parametric model $K_{\theta,s,t}$ for the Count Flow Map $K_{s,t}$. Its diagonal generator Q^θ_t has the same birth–death form as Q_t in Equation equation 3, with birth and death rates $\lambda_{\theta,i}(x,t)$ and $\mu_{\theta,i}(x,t)$, respectively. We parameterize the death rate as $\mu_{\theta,i}(x,t) = x_i\beta_{\theta,i}(x,t)$, where $\beta_{\theta,i}(x,t) \ge 0$ is the per-count death rate. For a finite interval from s to t, let $\Delta = t - s$. Given the current state x at time s, we model coordinate i by removing a random number of existing counts and adding a random number of new counts,

$$Y_i = x_i - D_i + B_i, \qquad B_i \sim \text{Poisson}(a_i), \qquad D_i \sim \text{Binomial}(x_i, q_i). \tag{5}$$

Because $0 \le D_i \le x_i$ and $B_i \ge 0$, this transition remains in $\mathbb{N}_0$ without imposing an upper bound on the count. This construction is related to classical integer-valued autoregressive models, which combine binomial thinning of existing counts with new arrivals (Al-Osh & Alzaid, 1987), providing a natural count-valued form for modeling finite-time births and deaths.

In Equation equation 5, $a_i(x,s,t) \ge 0$ is the Poisson mean for the number of births, while $q_i(x,s,t) \in [0,1]$ is the Binomial probability that each count present at time s is removed by time t. These finite-time parameters are constructed so that, as $t \downarrow s$, a_i and q_i match the local birth rate $\lambda_{\theta,i}(x,s)$ and per-count death rate $\beta_{\theta,i}(x,s)$ to first order, while finite-time corrections provide flexibility to learn departures from this local approximation over longer intervals. To capture dependence across coordinates, we use a finite mixture of these Poisson–Binomial kernels, with the mixture component shared across coordinates. The precise parameterization and the resulting probability mass function are given in Appendix A.2.

Proposition 2 (Diagonal generator consistency)**.** *Suppose that the finite-time birth and death parameters satisfy $a_i(x,t,t+h) = h\lambda_{\theta,i}(x,t) + O(h^2)$ and $q_i(x,t,t+h) = h\beta_{\theta,i}(x,t) + O(h^2)$ as $h \downarrow 0$. Then $K_{\theta,t,t} = I$ and*

$$K_{\theta,t,t+h}(x + e_i \mid x) = h\lambda_{\theta,i}(x,t) + O(h^2), \qquad K_{\theta,t,t+h}(x - e_i \mid x) = h\mu_{\theta,i}(x,t) + O(h^2).$$

All transitions involving two or more unit events have probability $O(h^2)$. Hence

$$\lim_{h\downarrow 0} \frac{K_{\theta,t,t+h} - I}{h} = Q^\theta_t,$$

where Q^θ_t is the birth–death generator introduced above.

The proof is provided in Appendix A.3. Thus, the finite-time kernel $K_{\theta,s,t}$ retains Q^θ_t as its infinitesimal generator, preserving the intended local birth–death dynamics.

2.3 TRAINING OBJECTIVE

We train the diagonal generator and the off-diagonal finite-time transitions jointly. The diagonal objective coincides with count-FM (Wei & Pearson, 2026), and the off-diagonal objective enforces Chapman–Kolmogorov consistency.

Diagonal rate matching. Let π be a coupling of p_0 and p_1, and draw $(X_0, X_1) \sim \pi$. Conditional on $(X_0 = x_0, X_1 = x_1)$, we use the signed-binomial bridge from count-FM. For each coordinate i and time $t \in [0, 1]$,

$$X_t^{(i)} = x_{0,i} + \mathrm{sgn}(x_{1,i} - x_{0,i})B_t^{(i)}, \qquad B_t^{(i)} \sim \mathrm{Binomial}(|x_{1,i} - x_{0,i}|, t)\,, \tag{6}$$

independently across coordinates given the endpoints. As shown in count-FM (Wei & Pearson, 2026), substituting this bridge into the one-dimensional mass-preservation equation for local birth–death dynamics gives the conditional target rates

$$\bar{\lambda}_{t,i} = \frac{(x_{1,i} - X_t^{(i)})_+}{1-t}, \qquad \bar{\mu}_{t,i} = \frac{(X_t^{(i)} - x_{1,i})_+}{1-t}, \tag{7}$$

where $(\cdot)_+ = \max(\cdot, 0)$. These rates are singular at $t = 1$, therefore we set $\tau = 1 - \varepsilon$ for a small $\varepsilon > 0$ and train on $t \in [0, \tau]$. The diagonal loss is

$$\mathcal{L}_{\mathrm{diag}}(\theta) = \mathbb{E}_{\substack{(X_0,X_1)\sim\pi,\\ t\sim\mathrm{Unif}[0,\tau],\\ X_t\sim p_t(\cdot|X_0,X_1)}} \left[\sum_{i=1}^{d} \ell\big(\bar{\lambda}_{t,i}, \lambda_{\theta,i}(X_t, t)\big) + \sum_{i=1}^{d} \ell(\bar{\mu}_{t,i}, \mu_{\theta,i}(X_t, t))\right]. \tag{8}$$

where $\ell(a, b) := b - a \log b$. A small numerical floor inside the logarithm is used in implementation.

The corresponding marginal rates are $\lambda_i^\star(x, t) = \mathbb{E}[\bar{\lambda}_{t,i} \mid X_t = x]$ and $\mu_i^\star(x, t) = \mathbb{E}[\bar{\mu}_{t,i} \mid X_t = x]$. Although the conditional bridge is independent across coordinates given (X_0, X_1), the marginal rates depend on the full state x and can therefore capture cross-coordinate dependence. Generator matching shows that these rates generate the marginal probability path (Holderrieth et al., 2025). Moreover, the conditional objective in Equation equation 8 has the same population optimum as directly matching $\lambda_i^\star$ and $\mu_i^\star$. Thus the diagonal part of Count Flow Map training is exactly count-FM and identifies the infinitesimal generator $Q_t^\star$ of the prescribed count path. The derivation is provided in Appendix A.4.

Off-diagonal Chapman–Kolmogorov consistency. For any $0 \le s < u < t \le \tau$, Chapman–Kolmogorov consistency requires the direct transition over $[s, t]$ to agree with the composition of transitions over $[s, u]$ and $[u, t]$. To construct the self-distillation target, let $\widetilde{K}_\theta$ denote K_θ with gradients stopped. Starting from $X_s \sim p_s$, draw $Z \sim \widetilde{K}_{\theta,s,u}(\cdot \mid X_s)$ and then $Y \sim \widetilde{K}_{\theta,u,t}(\cdot \mid Z)$. Equivalently, Y is drawn from the composed target kernel $R_{\theta;s,u,t} = \widetilde{K}_{\theta,s,u}\widetilde{K}_{\theta,u,t}$. We fit the direct transition with

$$\mathcal{L}_{\mathrm{CK}}(\theta) = -\mathbb{E}_{\substack{X_s\sim p_s,\\ Y\sim R_{\theta;s,u,t}(\cdot|X_s)}} \left[\log K_{\theta,s,t}(Y \mid X_s)\right]. \tag{9}$$

Because gradients are stopped through $R_{\theta;s,u,t}$, the CK loss differs from the expected KL divergence $\mathbb{E}_{X_s\sim p_s}[D_{\mathrm{KL}}(R_{\theta;s,u,t}(\cdot \mid X_s) \| K_{\theta,s,t}(\cdot \mid X_s))]$ only by the target entropy, so minimizing $\mathcal{L}_{\mathrm{CK}}$ is equivalent to minimizing this expected KL divergence. Since $\widetilde{K}_\theta$ and K_θ have identical values, zero KL discrepancy implies $K_{\theta,s,t} = K_{\theta,s,u}K_{\theta,u,t}$.

The complete objective is

$$\mathcal{L}(\theta) = \mathcal{L}_{\mathrm{diag}}(\theta) + \alpha\mathcal{L}_{\mathrm{CK}}(\theta), \tag{10}$$

where $\alpha \ge 0$ controls the off-diagonal consistency term. In this paper, we set $\alpha = 1$ in all experiments. The diagonal loss identifies the local generator, while the Chapman–Kolmogorov loss penalizes deviations between direct and composed finite-time transitions.

Corollary 1 (Identification by diagonal and Chapman–Kolmogorov consistency)**.** *Suppose that $Q_t^\star$ generates a unique nonexplosive transition family $P_{s,t}^\star$. If a Count Flow Map $K_{s,t}$ has diagonal generator $Q_t^\star$ and satisfies $K_{s,t} = K_{s,u}K_{u,t}$ for every $s \le u \le t$, then $K_{s,t} = P_{s,t}^\star$.*

Corollary 1 follows directly from Proposition 1. In training, we set $u = (s+t)/2$ in the Chapman–Kolmogorov loss. Repeated midpoint composition recursively reduces a finite interval to shorter subintervals, while Proposition 2 anchors the transition kernel to the learned generator in the short-interval limit. Appendix A.6.3 shows how these two properties control the finite-time transition error when midpoint consistency is approximate.

Terminal error decomposition. Let $P^{\theta}_{s,t}$ denote the exact transition family generated by the learned diagonal generator Q^{θ}_t. For an inference partition $\Pi = \{0 = t_0 < \cdots < t_L = \tau\}$, write $K^{\Pi}_{\theta} := K_{\theta,t_0,t_1} \cdots K_{\theta,t_{L-1},t_L}$. Appendix A.6 shows

$$D_{\mathrm{TV}}(p_1, p_0 K^{\Pi}_{\theta}) \le \underbrace{\varepsilon\, \mathbb{E}_{\pi}\|X_1 - X_0\|_1}_{\text{endpoint truncation}} + \underbrace{\sqrt{\frac{\tau}{2}\left[\mathcal{L}_{\mathrm{diag}}(\theta) - \mathcal{L}^{\star}_{\mathrm{diag}}\right]}}_{\text{generator estimation}} + \underbrace{\mathcal{E}_{\mathrm{map}}(\Pi)}_{\text{finite-time map}}, \tag{11}$$

where $\mathcal{L}^{\star}_{\mathrm{diag}}$ is the population diagonal loss at the exact marginal rates and $\mathcal{E}_{\mathrm{map}}(\Pi) := D_{\mathrm{TV}}(p_0 P^{\theta}_{0,\tau}, p_0 K^{\Pi}_{\theta})$.

The endpoint and generator-estimation terms follow the count-FM error analysis and the path-space relative-entropy argument for jump processes (Wei & Pearson, 2026; Arampatzis et al., 2015). The finite-time term is controlled by local transition errors and midpoint Chapman–Kolmogorov residuals in Appendix A.6.3.

2.4 Sample generation

After training, the Count Flow Map supports different inference budgets without retraining. Given a partition $0 = t_0 < t_1 < \cdots < t_L = \tau$, we sample $X_{t_0} \sim p_0$ and sequentially apply $K_{\theta,t_k,t_{k+1}}$ for $k = 0, \ldots, L-1$. Thus, $L = 1$ gives direct one-step generation with $K_{\theta,0,\tau}$, while $L > 1$ performs multi-step generation over shorter time intervals. Under exact Chapman–Kolmogorov consistency, these choices give the same finite-time transition. For conditional generation, covariates c are provided as additional model inputs, giving $K_{\theta,s,t}(\cdot \mid x, c)$, with the same sampling procedure. All experiments were conducted on a single NVIDIA GeForce RTX 2080 Ti GPU.

3 Simulation

Here, we consider two simulation settings to evaluate endpoint quality and inference efficiency across different numbers of function evaluations (NFE). We begin with a 2-D example, which is simple and easy to visualize. We then consider a 32-dimensional high-count setting to assess performance on a more challenging count distribution.

We compare Count Flow Map with Count-FM using its original unit-jump sampler (Wei & Pearson, 2026) and a binomial τ-leap variant motivated by binomial leap methods (Tian & Burrage, 2004), CountsDiff (Soatto et al., 2026), D3PM with ordinal Gaussian transitions (Austin et al., 2021), and Discrete Flow Maps (Potaptchik et al., 2026). The two Count-FM samplers use the same trained rate network and differ only at inference. All learned models are trained for 10,000 updates over five random seeds, using MLPs with hidden width 256 and depth 4 for a fair comparison. We evaluate NFE in $\{1, 2, 4, 16, 64, 128, 256\}$. Additional implementation and evaluation details are given in Appendix B.1.

2-D simulation. The target is an equal-weight mixture of two Gamma–Poisson (GP) distributions, $\mathrm{GP}((60, 5), (160, 80))$ and $\mathrm{GP}((60, 40), (160, 140))$, where $\mathrm{GP}(m, \alpha)$ denotes a Gamma–Poisson distribution with mean m and concentration α. For Count Flow Map and Count-FM, the source is $p_0 = \mathrm{Unif}\{0, \ldots, 72\}^2$, and the signed-binomial bridge is truncated at $\tau = 0.98$. The remaining baselines follow their original reference distributions, with details given in Appendix B.1. We report total variation (TV) distance and empirical W_2. Figure 1 shows the quality–efficiency curves, with detailed values at selected operating points reported in Appendix B.2.

Count Flow Map achieves the best one-step endpoint quality (NFE=1), with TV 0.153 ± 0.005 and $W_2 = 2.50 \pm 0.34$. Its performance is also stable across inference budgets, with TV between 0.126 and 0.153 and W_2 between 2.50 and 2.74 from NFE=1 to 256. In contrast, Count-FM, CountsDiff,

and D3PM require substantially more inference steps to reach comparable quality, while Discrete Flow Maps is competitive at low NFE, particularly at NFE=4. These results show that Count Flow Map retains high endpoint quality with one or a few evaluations and is robust to the number of inference steps. This stability is consistent with the learned Chapman–Kolmogorov constraint.

Endpoint samples and generation paths are shown in Appendix B.3. The endpoint samples again show that Count Flow Map captures the two target modes at NFE=1, while Count-FM, CountsDiff, and D3PM require more steps to approach the target. The intermediate paths further show that Count Flow Map, similar to Count-FM, evolves smoothly through count space. In contrast, categorical-state methods exhibit more abrupt transitions, reflecting the mismatch between categorical representations and the ordered geometry of count data.

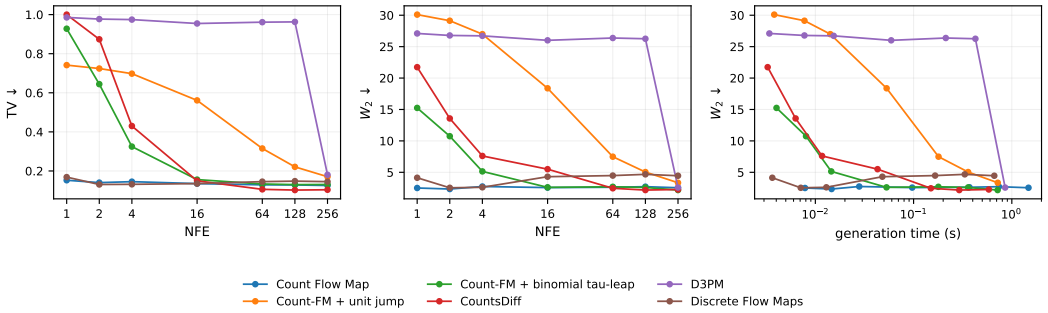

Figure 1: 2-D quality–efficiency trade-offs over the full NFE grid. Left and center show endpoint TV and W_2 versus NFE. Right shows W_2 versus measured generation time. Count Flow Map gives the strongest one-step endpoint quality and remains stable across the NFE grid. Discrete Flow Maps becomes competitive by NFE=4 but is worse at NFE=1, while Count-FM, CountsDiff, and D3PM require more inference steps to approach their best endpoint quality.

32-D simulation. We next consider a 32-dimensional high-count setting with a three-component Gamma–Poisson factor target and four latent factors per component. For Count Flow Map and Count-FM, the source is $p_0 = \mathrm{Unif}\{0, \ldots, 50\}^{32}$. The same pattern persists in this more challenging setting. At NFE=1, Count Flow Map achieves $\mathrm{SW}_2 = 0.556 \pm 0.134$, compared with 4.511 ± 0.148 for the next-best method, and it maintains the lowest SW_2 across all evaluated NFEs. It also achieves the lowest $\mathrm{MMD}^2_{\mathrm{RBF}}$ through NFE=128. Overall, the 32-D results show the strong low-NFE performance of Count Flow Map in a more challenging high-dimensional, high-count setting. Full results are given in Appendix B.4.

4 Applications

4.1 Single-Cell Drug Perturbation Prediction

We evaluate held-out drug responses in Tahoe-100M (Zhang et al., 2025), generating gene-expression counts conditioned on cell line, drug, and dose. The panel contains 70,064 treated cells, five cell lines, eight drugs, three doses, and 2,000 training-selected highly variable genes. We split the 120 conditions into 78 training, 18 validation, and 24 test conditions, withholding all treated cells from each held-out condition. Every held-out cell-line–drug pair is observed at another dose during training. Sources are cell-line- and plate-matched DMSO controls. Full split and preprocessing details are in Appendix C.1.

We compare Count Flow Map with Count-FM's unit-jump and binomial τ-leap samplers (Wei & Pearson, 2026; Tian & Burrage, 2004), a conditional NB-VAE (Sohn et al., 2015; Lopez et al., 2018), CPA (Lotfollahi et al., 2023), scGen (Lotfollahi et al., 2019), scVIDR (Kana et al., 2023), Sinkhorn OT with dose interpolation (Cuturi, 2013), and linear dose response. Both count-flow models use either independent or OT training couplings between matched DMSO and treated cells, following OT-CFM and count-FM (Tong et al., 2024; Wei & Pearson, 2026). For OT coupling, treated endpoints are permuted within each perturbation condition and experimental plate to minimize the

Method	Coupling	NFE	$SW_2 \downarrow$	logFC $r \uparrow$	Top overlap↑	Time (s)↓
Count Flow Map	Indep.	1	0.448 ± 0.006	0.779 ± 0.003	0.362 ± 0.011	0.21 ± 0.00
Count Flow Map		16	0.429 ± 0.013	0.771 ± 0.006	0.386 ± 0.008	1.62 ± 0.00
Count-FM (unit jump)		128–512	0.424 ± 0.006	0.782 ± 0.005	$\mathbf{0.396 \pm 0.003}$	10.85 ± 7.05
Count-FM (τ-leap)		128	0.430 ± 0.007	$\mathbf{0.786 \pm 0.006}$	0.394 ± 0.005	5.47 ± 0.08
Count Flow Map	OT	1	$\mathbf{0.394 \pm 0.002}$	0.779 ± 0.006	0.389 ± 0.006	0.20 ± 0.00
Count Flow Map		16	0.423 ± 0.003	0.764 ± 0.006	0.381 ± 0.006	1.60 ± 0.01
Count-FM (unit jump)		256–512	0.430 ± 0.017	0.760 ± 0.009	0.383 ± 0.006	15.26 ± 5.21
Count-FM (τ-leap)		128–256	0.434 ± 0.015	0.762 ± 0.004	0.376 ± 0.007	6.90 ± 2.94
Conditional NB-VAE	–	1-pass	0.451 ± 0.018	0.773 ± 0.012	0.393 ± 0.014	0.16 ± 0.01
CPA	–	1-pass	2.594 ± 0.120	0.342 ± 0.020	0.191 ± 0.006	0.48 ± 0.00
scGen	–	1-pass	1.787 ± 0.016	0.317 ± 0.008	0.159 ± 0.001	0.19 ± 0.01
scVIDR	–	1-pass	1.950 ± 0.026	0.274 ± 0.003	0.159 ± 0.002	0.19 ± 0.01
Sinkhorn OT	–	1-pass	1.540	0.336	0.196	0.71
Linear dose-response	–	1-pass	1.113	0.264	0.160	0.17

Table 1: Held-out perturbation prediction. Quality metrics are averaged equally across 24 test conditions. For neural models, we report mean ± sample standard deviation across three training seeds. These repeated runs assess neural-network training variability, while the non-neural Linear and Sinkhorn baselines are evaluated once. Count-FM's NFE is selected on validation for each seed. Ranges show the lowest and highest selected NFE. Time is the number of seconds needed to generate 9,600 cells, excluding setup and metric computation. Evaluation and timing details can be found in Appendix C.1.

total raw-count L_1 distance. We report Count Flow Map at fixed 1 and 16 NFE, while Count-FM's NFE is selected separately on validation for each sampler, coupling, and seed. Neural checkpoints are selected by validation sliced W_2.

Held-out perturbation prediction. Table 1 reports distributional accuracy via sliced W_2 in a training-only PCA space (Bonneel et al., 2015), response recovery via log fold-change (logFC) correlation and top-200 gene overlap (Liang et al., 2025), and generation time. Metric definitions are in Appendix C.1.

With OT coupling, one-step Count Flow Map achieves the lowest mean sliced W_2 (0.394 ± 0.002), generating 9,600 cells in approximately 0.20 seconds. Its logFC correlation (0.779) and top-gene overlap (0.389) are close to the best reported means (0.786 and 0.396). OT improves its one-step distributional accuracy and top-gene overlap while leaving mean logFC correlation approximately unchanged. Increasing from 1 to 16 NFE improves sliced W_2 under independence coupling but worsens it under OT. Approximate Chapman–Kolmogorov consistency may allow transition errors to accumulate differently under these couplings. Overall, Count Flow Map combines strong distributional accuracy and competitive response recovery with very few model evaluations.

Inference efficiency and effect recovery. Figure 2A–B shows distributional accuracy versus sampling cost. Panel C compares perturbation strengths relative to matched DMSO controls across all 24 test conditions in one fixed run. Observed and generated strengths correlate at $r = 0.65$ under independence coupling and $r = 0.81$ under OT. Fitted slopes of 0.88 and 0.75 indicate underestimation, greater under OT despite its higher correlation. Appendix C.2 examines gene-response patterns by condition at both sampling budgets.

4.2 Neural Population Forecasting

We next study temporal prediction of neural population activity using six 50 ms-binned recordings from the SpikeProphecy release (Steinmetz et al., 2019; Anonymous, 2026). The recordings contain 60,569–68,927 population count vectors and 490–1,104 neurons each, with 385,019 count vectors in total. The task is to forecast the joint population count vector in the next 50 ms bin from the preceding 500 ms of spike-count history. Within each recording, let $Y_b \in \mathbb{N}_0^D$ denote the population vector at bin b, where D is the recording-specific number of neurons, and let $H_b = (Y_{b-9}, \ldots, Y_b)$ denote the 10-bin history window. We model the next-bin forecast as $\widehat{Y}_{b+1} \sim K_{\theta,0,\tau}(\cdot \mid Y_b, H_b)$,

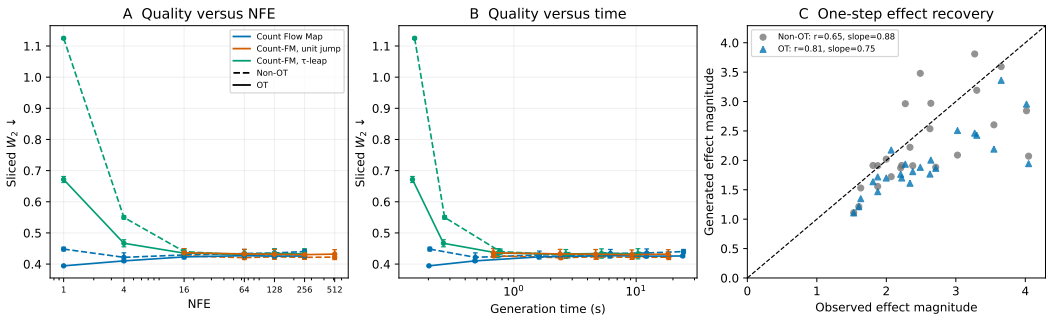

Figure 2: Held-out single-cell prediction and generation cost. A–B: sliced W_2 versus NFE and generation time for 9,600 cells. Error bars show one sample standard deviation across three training seeds. Colors identify the model or sampler. Solid and dashed lines indicate OT and independence coupling, respectively. C: observed versus generated perturbation strengths at 1 NFE under both couplings, using one fixed training seed. Each point represents one of the 24 test conditions. The dashed line indicates perfect agreement. r is Pearson correlation across condition magnitudes, and the slope is fitted through the origin. Effect definitions are provided in Appendix C.2.

where $[0, \tau]$ denotes flow time rather than physical time. Because the neuron dimension differs across recordings, we train a separate model for each recording using a chronological 70%/15%/15% train/validation/test split. The models condition only on spike-count history. We evaluate 4,096 test histories per recording, with 64 generated draws for each history. Metrics are computed separately for each recording and averaged equally across the six recordings within each training seed. We report the mean and standard deviation of these metrics across three seeds.

Population forecasts and high-activity events. Figure 3A–B illustrates how one-step Count Flow Map tracks fluctuations in observed population activity over a fixed test segment. Beyond predicting the mean, joint samples estimate the probability of high-activity events, defined by a training-only 95th-percentile threshold on total population counts. To assess the contribution of dependence, we independently shuffle each neuron's generated draws within each history, preserving its empirical marginal distribution. Shuffling increases the mean event Brier score (Brier, 1950) by 0.00085 at 1 NFE and 0.00127 at 16 NFE (Figure 3C). Retaining dependence therefore improves this population-level forecast, with the size of the benefit varying across recordings.

Forecast quality and computational cost. We compare sampling budgets within the count-flow family, using the same selected checkpoint at every budget. Joint forecast quality is assessed by dimension-normalized energy score, while the continuous ranked probability score (CRPS) evaluates the distribution of mean count per neuron (Gneiting & Raftery, 2007). At 1 NFE, Count Flow Map achieves an energy score of 0.3933 ± 0.0004 and population CRPS of 0.02292 ± 0.00034, taking 2.93 ms per 64-draw forecast. Count-FM at 8 NFE attains population CRPS of 0.02248 with unit jumps and 0.02242 with binomial τ-leaping, taking 6.83 and 8.12 ms, respectively. Count Flow Map is therefore 2.3–2.8$\times$ faster, at the cost of approximately 2% higher population CRPS. Its energy score is comparable to the corresponding Count-FM scores of 0.3967 and 0.3934. Increasing Count Flow Map to 16 NFE improves population CRPS to 0.02246, while its energy score changes little to 0.3938. The full comparison and efficiency curves appear in Table 4 and Figure 8 in Appendix D.

5 Conclusion

In this work, we introduced Count Flow Map, a finite-time extension of count-FM for efficient generation directly in count space. By learning transitions that replace many local birth–death updates, Count Flow Map supports one- or few-step generation while keeping samples nonnegative and integer-valued without a predefined maximum count. We developed a Poisson–Binomial transition model trained through diagonal rate matching and off-diagonal self-distillation. Across simulations, single-cell drug perturbation prediction, and neural population forecasting, Count Flow Map achieved competitive distributional quality with one or a few model evaluations. The single-cell results

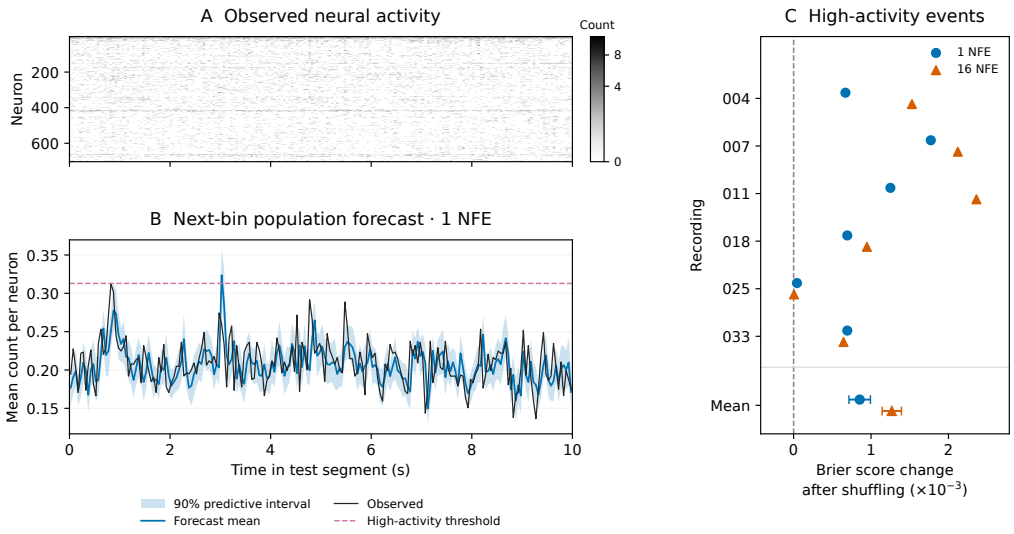

Figure 3: Neural population forecasting. A shows observed counts in a fixed 10-second test segment. B shows successive next-bin forecasts from observed histories, with the 1-NFE predictive mean and central 90% sample interval. A–B use recording 004 and one fixed training run. C shows the change in high-activity event Brier score after marginal-preserving neuron shuffling. Positive values favor joint samples. Recording points average three seeds. Aggregate bars show one sample standard deviation across seed means.

further demonstrated generation for held-out dose conditions with strong distributional accuracy and competitive perturbation-response recovery. The neural results demonstrated efficient one-step population forecasting and showed that retaining dependence among generated neuron counts improves prediction of high-activity events.

Although Count Flow Map performed well across our experiments, the current formulation still has several limitations. First, the Poisson–Binomial mixture provides a tractable finite-time transition family, but richer count-valued kernels may better capture complex transitions over long intervals while preserving the same local birth–death dynamics. Second, Chapman–Kolmogorov consistency is enforced approximately through self-distillation, so improving this consistency could reduce discrepancies between direct and multi-step generation. Self-conditioning (Chen et al., 2023) and fixed-point flow formulations (Yoo et al., 2026) suggest possible extensions, but their adaptation to count-valued transition kernels and their effects on consistency remain to be investigated. Finally, the signed-binomial bridge inherits the endpoint singularity of count-FM (Wei & Pearson, 2026), so the current sampler truncates before $t = 1$. Count bridges with better endpoint behavior (Fishman et al., 2026) could reduce the resulting truncation error.

In summary, Count Flow Map provides a direct and efficient approach to count-valued generation. By extending local birth–death dynamics to finite-time transitions, it supports one- or few-step sampling while preserving nonnegative integer counts.

AI USE STATEMENT

Generative AI tools were used to assist with literature discovery, research ideation, drafting and editing, code and synthetic-data generation, and the development and checking of mathematical arguments. All AI-assisted content was reviewed and verified by the authors, who take responsibility for the final content of the paper.

REFERENCES

M. A. Al-Osh and A. A. Alzaid. First-order integer-valued autoregressive (INAR(1)) process. *Journal of Time Series Analysis*, 8(3):261–275, 1987. doi: 10.1111/j.1467-9892.1987.tb00438.x.

Michael S. Albergo, Nicholas M. Boffi, and Eric Vanden-Eijnden. Stochastic interpolants: A unifying framework for flows and diffusions. *Journal of Machine Learning Research*, 26(209):1–80, 2025. URL https://jmlr.org/papers/v26/23-1605.html.

Anonymous. SpikeProphecy / Steinmetz 2019 (processed). Hugging Face dataset, 2026. URL https://huggingface.co/datasets/mysteriousauthor/spikeprophecy-steinmetz. 50 ms-binned spike-count release.

Georgios Arampatzis, Markos A. Katsoulakis, and Yannis Pantazis. Pathwise sensitivity analysis in transient regimes. In Stefan Heinz and Hakima Bessaih (eds.), *Stochastic Equations for Complex Systems: Theoretical and Computational Topics*, Mathematical Engineering, pp. 105–124. Springer International Publishing, Cham, 2015. ISBN 978-3-319-18206-3. doi: 10.1007/978-3-319-18206-3_5. URL https://doi.org/10.1007/978-3-319-18206-3_5.

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. In *Advances in Neural Information Processing Systems*, volume 34, pp. 17981–17993, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/958c530554f78bcd8e97125b70e6973d-Abstract.html.

Nicholas M. Boffi, Michael S. Albergo, and Eric Vanden-Eijnden. How to build a consistency model: Learning flow maps via self-distillation. In *Advances in Neural Information Processing Systems*, volume 38, pp. 37559–37595, 2025a. doi: 10.52202/085713-1121. URL https://papers.nips.cc/paper_files/paper/2025/hash/2ff3605c9e442d55156d99fe4f5f228d-Abstract-Conference.html.

Nicholas Matthew Boffi, Michael Samuel Albergo, and Eric Vanden-Eijnden. Flow map matching with stochastic interpolants: A mathematical framework for consistency models. *Transactions on Machine Learning Research*, 2025b. URL https://openreview.net/forum?id=cqDH0e6ak2.

Nicolas Bonneel, Julien Rabin, Gabriel Peyré, and Hanspeter Pfister. Sliced and Radon Wasserstein barycenters of measures. *Journal of Mathematical Imaging and Vision*, 51(1):22–45, 2015. doi: 10.1007/s10851-014-0506-3. URL https://doi.org/10.1007/s10851-014-0506-3.

Glenn W. Brier. Verification of forecasts expressed in terms of probability. *Monthly Weather Review*, 78(1):1–3, 1950. doi: 10.1175/1520-0493(1950)078<0001:VOFEIT>2.0.CO;2. URL https://journals.ametsoc.org/view/journals/mwre/78/1/1520-0493_1950_078_0001_vofeit_2_0_co_2.xml.

Andrew Campbell, Joe Benton, Valentin De Bortoli, Thomas Rainforth, George Deligiannidis, and Arnaud Doucet. A continuous time framework for discrete denoising models. In *Advances in Neural Information Processing Systems*, volume 35, pp. 28266–28279, 2022. doi: 10.52202/068431-2049. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/b5b528767aa35f5b1a60fe0aaeca0563-Abstract-Conference.html.

Ting Chen, Ruixiang Zhang, and Geoffrey Hinton. Analog Bits: Generating discrete data using diffusion models with self-conditioning. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=3itjR9QxFw.

Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In *Advances in Neural Information Processing Systems*, volume 26, 2013. URL https://papers.nips.cc/paper/4927-sinkhorn-distances-lightspeed-computation-of-optimal-transport.

Eugene A. Feinberg, Manasa Mandava, and Albert N. Shiryaev. On solutions of Kolmogorov's equations for nonhomogeneous jump Markov processes. *Journal of Mathematical Analysis and Applications*, 411(1):261–270, 2014. doi: 10.1016/j.jmaa.2013.09.043. URL https://doi.org/10.1016/j.jmaa.2013.09.043.

Christopher A. T. Ferro. Fair scores for ensemble forecasts. *Quarterly Journal of the Royal Meteorological Society*, 140(683):1917–1923, 2014. doi: 10.1002/qj.2270. URL https://doi.org/10.1002/qj.2270.

Nic Fishman, Gokul Gowri, Tanush Kumar, Jiaqi Lu, Valentin de Bortoli, Jonathan S. Gootenberg, and Omar Abudayyeh. Count Bridges enable modeling and deconvolving transcriptomic data. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2603.04730.

Kevin Frans, Danijar Hafner, Sergey Levine, and Pieter Abbeel. One step diffusion via shortcut models. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=OlzB6LnXcS.

Itai Gat, Tal Remez, Neta Shaul, Felix Kreuk, Ricky T. Q. Chen, Gabriel Synnaeve, Yossi Adi, and Yaron Lipman. Discrete flow matching. In *Advances in Neural Information Processing Systems*, volume 37, pp. 133345–133385, 2024. doi: 10.52202/079017-4239. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/f0d629a734b56a642701bba7bc8bb3ed-Abstract-Conference.html.

Tilmann Gneiting and Adrian E. Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007. doi: 10.1198/016214506000001437. URL https://doi.org/10.1198/016214506000001437.

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

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, pp. 6840–6851, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/4c5bcfec8584af0d967f1ab10179ca4b-Abstract.html.

Peter Holderrieth, Marton Havasi, Jason Yim, Neta Shaul, Itai Gat, Tommi Jaakkola, Brian Karrer, Ricky T. Q. Chen, and Yaron Lipman. Generator matching: Generative modeling with arbitrary markov processes. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/819aaee144cb40e887a4aa9e781b1547-Abstract-Conference.html.

Omar Kana, Rance Nault, David Filipovic, Daniel Marri, Tim Zacharewski, and Sudin Bhattacharya. Generative modeling of single-cell gene expression for dose-dependent chemical perturbations. *Patterns*, 4(8):100817, 2023. doi: 10.1016/j.patter.2023.100817. URL https://doi.org/10.1016/j.patter.2023.100817.

Dongjun Kim, Chieh-Hsin Lai, Wei-Hsiang Liao, Naoki Murata, Yuhta Takida, Toshimitsu Uesaka, Yutong He, Yuki Mitsufuji, and Stefano Ermon. Consistency trajectory models: Learning probability flow ODE trajectory of diffusion. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/c204d12afa0175285e5aac65188808b4-Abstract-Conference.html.

Zhaokang Liang, Shuyang Zhuang, Xiaoran Jiao, Weian Mao, Hao Chen, and Chunhua Shen. scPPDM: A diffusion model for single-cell drug-response prediction, 2025. URL https://arxiv.org/abs/2510.11726.

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

Romain Lopez, Jeffrey Regier, Michael B. Cole, Michael I. Jordan, and Nir Yosef. Deep generative modeling for single-cell transcriptomics. *Nature Methods*, 15(12):1053–1058, 2018. doi: 10.1038/s41592-018-0229-2. URL https://www.nature.com/articles/s41592-018-0229-2.

Mohammad Lotfollahi, F. Alexander Wolf, and Fabian J. Theis. scGen predicts single-cell perturbation responses. *Nature Methods*, 16(8):715–721, 2019. doi: 10.1038/s41592-019-0494-8. URL https://www.nature.com/articles/s41592-019-0494-8.

Mohammad Lotfollahi, Anna Klimovskaia Susmelj, Carlo De Donno, Leon Hetzel, Yuge Ji, Ignacio L. Ibarra, Sanjay R. Srivatsan, Mohsen Naghipourfar, Riza M. Daza, Beth Martin, Jay Shendure, Jose L. McFaline-Figueroa, Pierre Boyeau, F. Alexander Wolf, Nafissa Yakubova, Stephan Günnemann, Cole Trapnell, David Lopez-Paz, and Fabian J. Theis. Predicting cellular responses to complex perturbations in high-throughput screens. *Molecular Systems Biology*, 19(6):e11517, 2023. doi: 10.15252/msb.202211517. URL https://link.springer.com/article/10.15252/msb.202211517.

J. R. Norris. *Markov Chains*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, Cambridge, 1997. ISBN 978-0-521-48181-6. doi: 10.1017/CBO9780511810633. URL https://doi.org/10.1017/CBO9780511810633.

Peter Potaptchik, Jason Yim, Adhi Saravanan, Peter Holderrieth, Eric Vanden-Eijnden, and Michael S. Albergo. Discrete flow maps. arXiv preprint arXiv:2604.09784, 2026. URL https://arxiv.org/abs/2604.09784.

Daan Roos, Oscar Davis, Floor Eijkelboom, Michael Bronstein, Max Welling, İsmail İlkan Ceylan, Luca Ambrogioni, and Jan-Willem van de Meent. Categorical flow maps. In *International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2602.12233.

Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=TIdIXIpzhoI.

Renzo G. Soatto, Anders Hoel, Greycen Ren, Shorna Alam, Stephen Bates, Nikolaos P. Daskalakis, Caroline Uhler, and Maria Skoularidou. CountsDiff: A diffusion model on the natural numbers for generation and imputation of count-based data. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2604.03779.

Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. In *Advances in Neural Information Processing Systems*, volume 28, 2015. URL https://proceedings.neurips.cc/paper_files/paper/2015/file/8d55a249e6baa5c06772297520da2051-Paper.pdf.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 32211–32252. PMLR, 2023. URL https://proceedings.mlr.press/v202/song23a.html.

Nicholas A. Steinmetz, Peter Zatka-Haas, Matteo Carandini, and Kenneth D. Harris. Distributed coding of choice, action and engagement across the mouse brain. *Nature*, 576(7786):266–273, 2019. doi: 10.1038/s41586-019-1787-x. URL https://doi.org/10.1038/s41586-019-1787-x.

Tianhai Tian and Kevin Burrage. Binomial leap methods for simulating stochastic chemical kinetics. *The Journal of Chemical Physics*, 121(21):10356–10364, 2004. doi: 10.1063/1.1810475. URL https://doi.org/10.1063/1.1810475.

Alexander Tong, Kilian Fatras, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Guy Wolf, and Yoshua Bengio. Improving and generalizing flow-based generative models with minibatch optimal transport. *Transactions on Machine Learning Research*, 2024. URL https://arxiv.org/abs/2302.00482.

Ganchao Wei and John Pearson. Flow matching for count data. arXiv preprint arXiv:2605.07746, 2026. URL https://arxiv.org/abs/2605.07746.

Jaehoon Yoo, Wonjung Kim, Floor Eijkelboom, Chanhyuk Lee, Nicholas M. Boffi, Seunghoon Hong, and Jinwoo Kim. Self-conditioned flow map language models via fixed-point flows. arXiv preprint arXiv:2607.00714, 2026. URL https://arxiv.org/abs/2607.00714.

Jesse Zhang, Airol A. Ubas, Richard de Borja, Valentine Svensson, Nicole Thomas, Neha Thakar, et al. Tahoe-100M: A giga-scale single-cell perturbation atlas for context-dependent gene function and cellular modeling. *bioRxiv*, 2025. doi: 10.1101/2025.02.20.639398. URL https://www.biorxiv.org/content/10.1101/2025.02.20.639398v3. Preprint.

A Additional Method Details and Theory

A.1 Consistency rules for Count Flow Map

We justify the three consistency rules in Equation equation 4 and Proposition 1. Let Q_t be a time-dependent Markov jump generator on $\mathcal{X} = \mathbb{N}_0^d$. Assume that the associated process is nonexplosive, admits a unique transition family $P^Q_{s,t}$, and satisfies the regularity needed for the derivatives below. See Norris (1997) for continuous-time Markov-chain background and Feinberg et al. (2014) for Kolmogorov equations and uniqueness results for nonhomogeneous jump Markov processes.

Chapman–Kolmogorov consistency. For $s \le u \le t$, the Markov property gives

$$P^Q_{s,t}(y \mid x) = \sum_{z \in \mathcal{X}} P^Q_{s,u}(z \mid x) P^Q_{u,t}(y \mid z),$$

which is exactly $P^Q_{s,t} = P^Q_{s,u} P^Q_{u,t}$.

Lagrangian (forward) consistency. The diagonal generator satisfies $P^Q_{t,t+h} = I + hQ_t + o(h)$. Combining this expansion with Chapman–Kolmogorov consistency,

$$P^Q_{s,t+h} = P^Q_{s,t} P^Q_{t,t+h},$$

gives, for every test function f,

$$\frac{P^Q_{s,t+h} f - P^Q_{s,t} f}{h} = P^Q_{s,t} \frac{P^Q_{t,t+h} f - f}{h} \longrightarrow P^Q_{s,t} Q_t f.$$

Hence $\partial_t P^Q_{s,t} = P^Q_{s,t} Q_t$.

Eulerian (backward) consistency. Similarly,

$$P^Q_{s,t} = P^Q_{s,s+h} P^Q_{s+h,t}.$$

Using $P^Q_{s,s+h} = I + hQ_s + o(h)$ and rearranging yields

$$\frac{P^Q_{s+h,t} - P^Q_{s,t}}{h} \longrightarrow -Q_s P^Q_{s,t},$$

so $\partial_s P^Q_{s,t} = -Q_s P^Q_{s,t}$.

These arguments show that the exact transition family satisfies all three consistency rules. Conversely, the forward equation with $K_{s,s} = I$ uniquely determines $P^Q_{s,t}$, and the same is true for the backward equation with $K_{t,t} = I$. Finally, suppose that $K_{s,t}$ satisfies Chapman–Kolmogorov consistency and has diagonal generator Q_t. Then

$$K_{s,t+h} = K_{s,t} K_{t,t+h}$$

and the same difference-quotient argument used above gives $\partial_t K_{s,t} = K_{s,t} Q_t$. Uniqueness of the forward equation therefore implies $K_{s,t} = P^Q_{s,t}$. This proves Proposition 1.

A.2 Exact likelihood of the count transition kernel

We first specify the finite-time parameterization used in our model. Let $\Delta = t - s$. Using the local rates $\lambda_{\theta,i}(x,s)$ and $\mu_{\theta,i}(x,s) = x_i \beta_{\theta,i}(x,s)$ introduced in Section 2.2, for mixture component j we parameterize the finite-time birth mean and death probability as

$$\begin{aligned} a_{j,i}(x,s,t) &= \Delta\, \lambda_{\theta,i}(x,s) \exp\Big\{ \Delta c^+_{\theta,j,i}(x,s,t) \Big\}, \\ q_{j,i}(x,s,t) &= 1 - \exp\Big[-\Delta\, \beta_{\theta,i}(x,s) \exp\Big\{ \Delta c^-_{\theta,j,i}(x,s,t) \Big\} \Big]. \end{aligned} \tag{12}$$

The functions $c^+_{\theta,j,i}$ and $c^-_{\theta,j,i}$ modify the finite-time birth and death behavior. Multiplication by Δ inside the exponential makes these corrections vanish to first order near the diagonal, so that $a_{j,i} = h\lambda_{\theta,i} + O(h^2)$ and $q_{j,i} = h\beta_{\theta,i} + O(h^2)$ over an interval of length h.

We use a finite mixture to increase the flexibility of the transition kernel. Let J denote the number of components and let $w_{\theta,j}(x, s, t) \geq 0$, with $\sum_{j=1}^{J} w_{\theta,j}(x, s, t) = 1$, denote their weights. Conditional on component j, the coordinates follow the Poisson–Binomial transition described in Equation equation 5, with parameters $a_{j,i}$ and $q_{j,i}$ defined above.

For fixed x, component j, and coordinate i, the relation $Y_i = x_i - D_i + B_i$ implies that, if $D_i = d$, then $B_i = y_i - x_i + d$. Hence

$$k_{\theta,j,i}(y_i \mid x_i, s, t) = \sum_{d=\max(0,x_i-y_i)}^{x_i} \binom{x_i}{d} q_{j,i}^d (1 - q_{j,i})^{x_i - d} \times \exp(-a_{j,i}) \frac{a_{j,i}^{y_i - x_i + d}}{(y_i - x_i + d)!}. \tag{13}$$

Conditional on a mixture component, the coordinates are sampled independently. The full kernel is therefore

$$K_{\theta,s,t}(y \mid x) = \sum_{j=1}^{J} w_{\theta,j}(x, s, t) \prod_{i=1}^{d} k_{\theta,j,i}(y_i \mid x_i, s, t). \tag{14}$$

Each coordinate distribution is normalized, the mixture weights sum to one, and $D_i \leq x_i$ almost surely. Thus Equation equation 14 defines a valid transition kernel on $\mathbb{N}_0^d$ without a predefined maximum count. Setting $J = 1$ recovers the kernel described in the main text.

A.3 Proof of diagonal generator consistency

We prove Proposition 2. Fix x and t, and consider the interval $[t, t + h]$. From Equation equation 12, local boundedness of $c^+_{\theta,j,i}$ and $c^-_{\theta,j,i}$ gives, for every mixture component j,

$$a_{j,i} = h\lambda_{\theta,i}(x, t) + O(h^2), \qquad q_{j,i} = h\beta_{\theta,i}(x, t) + O(h^2).$$

Therefore

$$\mathbb{P}(B_{j,i} = 1) = h\lambda_{\theta,i}(x, t) + O(h^2), \qquad \mathbb{P}(D_{j,i} = 1) = h\mu_{\theta,i}(x, t) + O(h^2),$$

while the probabilities of two or more births, two or more deaths, simultaneous birth and death, or events in more than one coordinate are all $O(h^2)$. Consequently,

$$\mathbb{P}(Y = x + e_i \mid x, M = j) = h\lambda_{\theta,i}(x, t) + O(h^2),$$

$$\mathbb{P}(Y = x - e_i \mid x, M = j) = h\mu_{\theta,i}(x, t) + O(h^2),$$

and

$$\mathbb{P}(Y = x \mid x, M = j) = 1 - h \sum_i \big[\lambda_{\theta,i}(x, t) + \mu_{\theta,i}(x, t)\big] + O(h^2).$$

The first-order terms do not depend on the mixture component, so averaging over the mixture weights leaves them unchanged. At $h = 0$, the Poisson means and death probabilities are zero, hence $Y = x$ almost surely. This proves the proposition.

Under local boundedness and sufficient regularity of the jump rates, the exact transition family generated by Q_t^θ also satisfies, for each fixed (x, t),

$$P^\theta_{t,t+h}(\cdot \mid x) = \delta_x + hQ_t^\theta(x, \cdot) + O_{\mathrm{TV}}(h^2).$$

Since $K_{\theta,t,t+h}$ has the same expansion,

$$D_{\mathrm{TV}}\big(K_{\theta,t,t+h}(\cdot \mid x), P^\theta_{t,t+h}(\cdot \mid x)\big) = O(h^2).$$

A.4 DIAGONAL MATCHING AND ITS RELATION TO COUNT-FM

For completeness, we derive the target rates and the population optimum used in the diagonal loss. Given an endpoint pair (x_0, x_1), let $n_i := |x_{1,i} - x_{0,i}|$ and $\sigma_i := \mathrm{sgn}(x_{1,i} - x_{0,i})$. The conditional bridge is $X_t^{(i)} = x_{0,i} + \sigma_i B_t^{(i)}$ with $B_t^{(i)} \sim \mathrm{Binomial}(n_i, t)$.

For a fixed coordinate i, let $p_t^{(i)}(x_i \mid x_0, x_1)$ denote its conditional bridge marginal. Local mass preservation gives

$$\begin{aligned}\partial_t p_t^{(i)}(x_i \mid x_0, x_1) = \bar{\lambda}_{t,i}(x_i - 1)p_t^{(i)}(x_i - 1 \mid x_0, x_1) \\ + \bar{\mu}_{t,i}(x_i + 1)p_t^{(i)}(x_i + 1 \mid x_0, x_1) \\ - \big[\bar{\lambda}_{t,i}(x_i) + \bar{\mu}_{t,i}(x_i)\big]p_t^{(i)}(x_i \mid x_0, x_1).\end{aligned}$$

The first two terms describe birth and death influx into x_i, while the last term describes outflux from x_i. Substituting the Binomial bridge gives the rates in Equation equation 7. Equivalently, the n_i required unit changes can be viewed as occurring independently over $[0, 1]$; conditional on not having occurred before time t, each remaining event has instantaneous hazard $(1 - t)^{-1}$.

Let $p_t(x)$ be the marginal bridge after averaging over $(X_0, X_1) \sim \pi$. The marginal rates

$$\lambda_i^\star(x, t) = \mathbb{E}[\bar{\lambda}_{t,i} \mid X_t = x], \qquad \mu_i^\star(x, t) = \mathbb{E}[\bar{\mu}_{t,i} \mid X_t = x]$$

generate p_t by conditional-to-marginal generator matching (Holderrieth et al., 2025).

It remains to verify that Equation equation 8 identifies these marginal rates. For any nonnegative target A and any prediction $b(X) > 0$ measurable with respect to X,

$$\mathbb{E}[\ell(A, b(X)) \mid X] = b(X) - \mathbb{E}[A \mid X] \log b(X).$$

When $\mathbb{E}[A \mid X] > 0$, this is minimized at $b(X) = \mathbb{E}[A \mid X]$; when $\mathbb{E}[A \mid X] = 0$, the optimum is approached as $b(X) \downarrow 0$. Applying this separately to the birth and death rates gives $\lambda_i^\star$ and $\mu_i^\star$. Moreover, using $0 \log 0 := 0$,

$$\ell(a, b) - \ell(a, a) = a \log \frac{a}{b} - a + b =: D_{\mathrm{GKL}}(a, b),$$

so the excess diagonal loss is an integrated generalized KL divergence between the exact and model marginal rates, exactly as in count-FM (Wei & Pearson, 2026).

A.5 OFF-DIAGONAL CHAPMAN–KOLMOGOROV SELF-DISTILLATION

For fixed x and arbitrary $s < u < t$, define the composed target kernel

$$R_{\theta;s,u,t}(y \mid x) := \sum_z \widetilde{K}_{\theta,s,u}(z \mid x)\widetilde{K}_{\theta,u,t}(y \mid z),$$

where the target kernels are treated as fixed when optimizing the direct transition. The conditional cross-entropy is

$$\mathcal{C}_\theta(x, s, u, t) := \mathbb{E}_{Y \sim R_{\theta;s,u,t}}[-\log K_{\theta,s,t}(Y \mid x)].$$

Adding and subtracting $\log R_{\theta;s,u,t}(Y \mid x)$ gives

$$\mathcal{C}_\theta(x, s, u, t) = H(R_{\theta;s,u,t}) + D_{\mathrm{KL}}(R_{\theta;s,u,t} \| K_{\theta,s,t}(\cdot \mid x)). \tag{15}$$

Thus, with gradients stopped through the target, minimizing the cross-entropy is equivalent to minimizing the conditional KL divergence. Since $\widetilde{K}_\theta$ and K_θ have identical values, zero KL divergence implies the Chapman–Kolmogorov identity.

The same divergence controls the corresponding marginal discrepancy. For any starting distribution ν, data processing followed by Pinsker's inequality gives

$$D_{\mathrm{TV}}(\nu K_{\theta,s,t}, \nu K_{\theta,s,u} K_{\theta,u,t}) \le \sqrt{\frac{1}{2}\mathbb{E}_{X \sim \nu} D_{\mathrm{KL}}((K_{\theta,s,u} K_{\theta,u,t})(\cdot \mid X) \| K_{\theta,s,t}(\cdot \mid X))}.$$

A.6 TERMINAL ERROR ANALYSIS

We separate the terminal error into endpoint truncation, estimation of the diagonal generator, and approximation of its finite-time transition family. Let $p_t^\star$ denote the marginal distribution of the exact signed-binomial bridge, let $Q_t^\star$ denote its marginal generator, and let Q_t^θ denote the model diagonal generator with exact transition family $P_{s,t}^\theta$. For a partition $\Pi = \{0 = t_0 < \cdots < t_L = \tau\}$, define $K_\theta^\Pi := \prod_{k=0}^{L-1} K_{\theta,t_k,t_{k+1}}$. By the triangle inequality,

$$D_{\mathrm{TV}}(p_1, p_0 K_\theta^\Pi) \le D_{\mathrm{TV}}(p_1, p_\tau^\star) + D_{\mathrm{TV}}(p_\tau^\star, p_0 P_{0,\tau}^\theta) + D_{\mathrm{TV}}(p_0 P_{0,\tau}^\theta, p_0 K_\theta^\Pi).$$

We bound the three terms separately.

A.6.1 ENDPOINT TRUNCATION

For an endpoint pair (x_0, x_1), let $n_i := |x_{1,i} - x_{0,i}|$. At $\tau = 1 - \varepsilon$, coordinate i has completed all n_i required unit changes with probability τ^{n_i}. Conditional independence across coordinates gives

$$\mathbb{P}(X_\tau = X_1 \mid X_0 = x_0, X_1 = x_1) = \tau^{\|x_1 - x_0\|_1}.$$

Using (X_τ, X_1) from the same bridge as a coupling,

$$D_{\mathrm{TV}}(p_\tau^\star, p_1) \le \mathbb{E}_\pi[1 - \tau^{\|X_1 - X_0\|_1}] \le \varepsilon\, \mathbb{E}_\pi \|X_1 - X_0\|_1. \tag{16}$$

This is the endpoint bound from count-FM (Wei & Pearson, 2026).

A.6.2 GENERATOR ESTIMATION ERROR

Define the integrated rate error

$$\mathcal{E}_{\mathrm{rate}}(\theta) := \int_0^\tau \mathbb{E}_{X_t \sim p_t^\star} \sum_{i=1}^d [D_{\mathrm{GKL}}(\lambda_i^\star(X_t, t), \lambda_{\theta,i}(X_t, t)) + D_{\mathrm{GKL}}(\mu_i^\star(X_t, t), \mu_{\theta,i}(X_t, t))]\, dt.$$

Under the usual absolute-continuity and integrability conditions for path-space relative entropy of jump processes (Arampatzis et al., 2015),

$$D_{\mathrm{KL}}(p_\tau^\star \| p_0 P_{0,\tau}^\theta) \le \mathcal{E}_{\mathrm{rate}}(\theta).$$

Pinsker's inequality and uniform sampling of t in Equation equation 8 then give

$$D_{\mathrm{TV}}(p_\tau^\star, p_0 P_{0,\tau}^\theta) \le \sqrt{\frac{\tau}{2}\left[\mathcal{L}_{\mathrm{diag}}(\theta) - \mathcal{L}_{\mathrm{diag}}^\star\right]}. \tag{17}$$

This is the same rate-estimation argument used in count-FM (Wei & Pearson, 2026).

A.6.3 FINITE-TIME FLOW MAP ERROR

We next bound the error from replacing the exact transition family P^θ of the learned generator by the learned finite-time kernels K_θ.

Lemma 1 (Kernel telescoping bound). *Let $A_0, \ldots, A_{L-1}$ and $B_0, \ldots, B_{L-1}$ be Markov kernels and let ν_0 be an initial distribution. Define $\nu_k := \nu_0 A_0 \cdots A_{k-1}$ for $k \ge 1$. Then*

$$D_{\mathrm{TV}}(\nu_0 A_0 \cdots A_{L-1}, \nu_0 B_0 \cdots B_{L-1}) \le \sum_{k=0}^{L-1} \mathbb{E}_{X \sim \nu_k} D_{\mathrm{TV}}(A_k(\cdot \mid X), B_k(\cdot \mid X)). \tag{18}$$

Proof. Insert and subtract one kernel at a time,

$$\nu_0 A_0 \cdots A_{L-1} - \nu_0 B_0 \cdots B_{L-1} = \sum_{k=0}^{L-1} \nu_0 A_0 \cdots A_{k-1}(A_k - B_k) B_{k+1} \cdots B_{L-1}.$$

Markov kernels contract total variation, so the total variation norm of the kth term is bounded by the corresponding expectation in Equation equation 18. Summing proves the result. □

Applying Lemma 1 with $A_k = K_{\theta,t_k,t_{k+1}}$ and $B_k = P^{\theta}_{t_k,t_{k+1}}$ gives

$$\mathcal{E}_{\mathrm{map}}(\Pi) \le \sum_{k=0}^{L-1} \mathbb{E}_{X\sim\nu_k^\theta} D_{\mathrm{TV}}\Big(K_{\theta,t_k,t_{k+1}}(\cdot \mid X), P^{\theta}_{t_k,t_{k+1}}(\cdot \mid X)\Big), \tag{19}$$

where $\nu_k^\theta := p_0 K_{\theta,t_0,t_1} \cdots K_{\theta,t_{k-1},t_k}$. Thus the terminal map error is controlled by the finite-time approximation errors on the intervals actually used at inference.

Midpoint Chapman–Kolmogorov consistency. As described in Section 2.3, for each interval $[a, b]$ we use its midpoint $m = (a + b)/2$ in the Chapman–Kolmogorov loss. Applying the same midpoint relation recursively to the resulting subintervals gives a dyadic partition of any finite interval. For an interval $[a, b]$, define the midpoint residual

$$\rho(a, b) := \sup_x D_{\mathrm{TV}}\left(K_{\theta,a,b}(\cdot \mid x), (K_{\theta,a,m}K_{\theta,m,b})(\cdot \mid x)\right).$$

For a depth-J dyadic partition of $[s, t]$, let δ_J be the maximum total-variation discrepancy between K_θ and P^θ over the 2^J smallest intervals, uniformly over x, and let ρ_ℓ be the maximum midpoint residual among the intervals at level ℓ. Repeated midpoint subdivision and total-variation contraction give

$$\sup_x D_{\mathrm{TV}}\left(K_{\theta,s,t}(\cdot \mid x), P^{\theta}_{s,t}(\cdot \mid x)\right) \le \sum_{\ell=0}^{J-1} 2^\ell \rho_\ell + 2^J \delta_J. \tag{20}$$

If midpoint identity $K_{\theta,a,b} = K_{\theta,a,m}K_{\theta,m,b}$ holds on every interval in the recursive subdivision, then $\rho_\ell = 0$ for all ℓ. Appendix A.3 establishes the local $O(h^2)$ transition error. If this local $O(h^2)$ bound holds uniformly over x and the dyadic subintervals, then $\delta_J = O(h^2)$. Since there are 2^J intervals of length $h = (t - s)/2^J$, we have $2^J\delta_J = O(h) \to 0$, and hence $K_{\theta,s,t} = P^{\theta}_{s,t}$. Thus consistency at recursively chosen midpoints is sufficient under this uniform local condition, while Equation equation 20 also quantifies the error when midpoint consistency is approximate.

A.6.4 Terminal distribution error bound

Combining Equations equation 16, equation 17, and equation 19 yields the bound stated in Equation equation 11. More explicitly,

Theorem 1 (Terminal distribution error)**.** *Assume that the target and learned generators admit unique nonexplosive transition families with common initial distribution p_0, that $\mathbb{E}_\pi\|X_1 - X_0\|_1 < \infty$, and that the path-space absolute-continuity and integrability conditions in Appendix A.6.2 hold. Assume also that the population losses in Equation equation 17 are finite. Then, for any inference partition Π,*

$$\begin{aligned} D_{\mathrm{TV}}(p_1, p_0K_\theta^\Pi) \le{}& \varepsilon\,\mathbb{E}_\pi\|X_1 - X_0\|_1 + \sqrt{\frac{\tau}{2}\left[\mathcal{L}_{\mathrm{diag}}(\theta) - \mathcal{L}^\star_{\mathrm{diag}}\right]} \\ &+ \sum_{k=0}^{L-1} \mathbb{E}_{X\sim\nu_k^\theta} D_{\mathrm{TV}}\Big(K_{\theta,t_k,t_{k+1}}(\cdot \mid X), P^{\theta}_{t_k,t_{k+1}}(\cdot \mid X)\Big). \end{aligned}$$

The three terms correspond respectively to stopping the bridge before its singular endpoint, estimating the infinitesimal count dynamics, and learning their finite-time transition kernels.

B Additional Simulation Details and Results

B.1 Experimental details

All experiments use five random seeds and evaluate NFE $\in \{1, 2, 4, 16, 64, 128, 256\}$. Count Flow Map, Count-FM, CountsDiff, D3PM, and Discrete Flow Maps are trained for 10,000 parameter updates using MLPs with hidden width 256 and depth 4. The effective batch size is 512 and the learning rate is 10^{-3}. The two Count-FM samplers use the same trained rate network and differ only at inference. Count Flow Map uses eight mixture components in its finite-time transition head. Discrete Flow Maps uses 8,000 diagonal updates followed by 2,000 self-distillation updates, and

Table 2: 2-D simulation endpoint quality at selected inference budgets. Entries are mean ± standard deviation over five seeds. Lower is better for all metrics. The best quality metric at each NFE is shown in bold.

Method	NFE	TV↓	W_2 ↓	Time (s)↓
Count Flow Map	1	**0.153±0.005**	**2.50±0.34**	0.008±0.000
	4	0.145±0.007	2.74±0.50	0.028±0.001
	128	0.129±0.003	2.71±0.34	0.739±0.039
	256	0.126±0.002	2.54±0.49	1.478±0.076
Count-FM + unit jump	1	0.742±0.002	30.10±0.34	0.004±0.000
	4	0.698±0.001	26.99±0.43	0.014±0.001
	128	0.221±0.002	5.04±0.27	0.359±0.021
	256	0.171±0.002	3.34±0.28	0.719±0.042
Count-FM + binomial τ-leap	1	0.928±0.003	15.25±0.04	0.004±0.000
	4	0.325±0.001	5.14±0.15	0.015±0.001
	128	0.130±0.002	2.59±0.25	0.359±0.022
	256	0.131±0.003	**2.17±0.35**	0.717±0.043
CountsDiff	1	1.000±0.000	21.74±0.19	0.003±0.000
	4	0.430±0.007	7.60±0.45	0.012±0.001
	128	**0.102±0.003**	**2.17±0.44**	0.292±0.020
	256	**0.104±0.003**	2.26±0.23	0.583±0.042
D3PM	1	0.986±0.009	27.10±0.28	0.003±0.000
	4	0.974±0.009	26.72±0.33	0.015±0.001
	128	0.963±0.013	26.26±0.60	0.426±0.023
	256	0.181±0.008	2.57±0.46	0.855±0.047
Discrete Flow Maps	1	0.169±0.002	4.13±0.77	0.004±0.000
	4	**0.132±0.003**	**2.61±0.61**	0.013±0.001
	128	0.148±0.002	4.68±0.72	0.332±0.025
	256	0.145±0.002	4.45±0.22	0.663±0.051

D3PM uses 256 native diffusion steps. Count Flow Map and Count-FM use the signed-binomial bridge with $\tau = 0.98$. For D3PM and Discrete Flow Maps, counts are represented by categories $0, \ldots, C_{\max}$ plus an overflow category.

For the 2-D simulation, we use 30,000 generated and target samples per seed. Total variation (TV) distance is computed against the analytic target PMF, while empirical W_2 is computed by optimal matching on 1,500 samples and averaged over three subsamples. For the 32-D simulation, we use 5,000 generated and target samples per seed. Sliced W_2 (Bonneel et al., 2015) uses 128 random projections, and $\mathrm{MMD}^2_{\mathrm{RBF}}$ (Gretton et al., 2012) uses an RBF kernel with bandwidth selected by the median-distance heuristic. Generation time is the median of three runs after one warm-up.

B.2 2-D SIMULATION RESULTS

The 2-D target is an equal-weight two-component Gamma–Poisson mixture with component mean vectors $(60, 5)$ and $(60, 40)$ and concentration vectors $(160, 80)$ and $(160, 140)$, respectively. Table 2 reports selected operating points.

B.3 2-D ENDPOINT AND PATH VISUALIZATIONS

Figure 4 shows endpoint samples at the four operating points reported in Table 2. At NFE=1, Count Flow Map already closely recovers the two target modes. Both Count-FM samplers, CountsDiff, and D3PM remain visibly far from the target, while Discrete Flow Maps captures the two-mode geometry but has a larger W_2 error than Count Flow Map. As NFE increases, the iterative methods progressively approach the target. Notably, D3PM remains far from the target through NFE=128 and

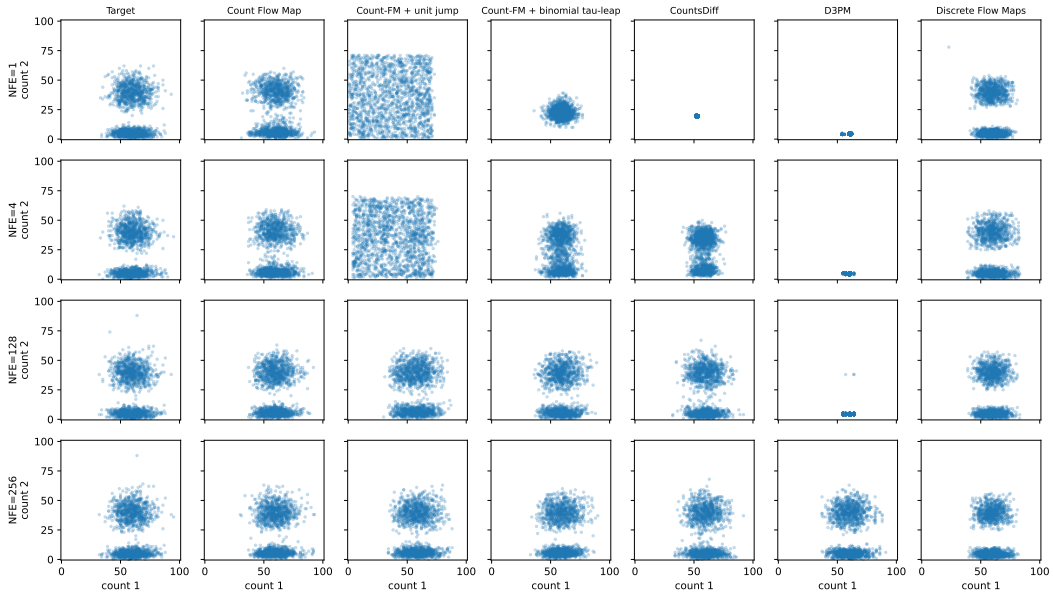

Figure 4: 2-D endpoint samples at NFE $\in \{1, 4, 128, 256\}$. Columns show the target and the six compared methods, and rows correspond to different inference budgets.

only approaches it at NFE=256, likely reflecting its less smooth categorical transition path and the resulting difficulty of coarse inference.

Figure 5 shows generation paths at normalized progress $q \in \{0, 0.25, 0.5, 0.75, 1\}$. For Count Flow Map, the five columns correspond to the initial distribution and the distributions after each of four finite-time map evaluations over $[0, \tau]$. Discrete Flow Maps is also shown using four finite-time compositions, while Count-FM, CountsDiff, and D3PM are shown at matched progress along 256-step sampling paths. Count Flow Map shows a smooth progression through count space, similar to Count-FM, whereas the categorical-state methods exhibit more abrupt transitions, reflecting the mismatch between categorical representations and the ordered geometry of count data.

B.4 32-D SIMULATION

We consider a 32-dimensional Gamma–Poisson factor mixture with three mixture components. Within each component, four latent Gamma factors induce dependence across the 32 count variables. The resulting distribution is multimodal and overdispersed, with a nominal count scale of 20. For Count Flow Map and Count-FM, the source is uniform on $\{0, \dots, 50\}^{32}$. We evaluate sample quality using sliced W_2 and RBF MMD^2. Table 3 reports selected operating points.

Count Flow Map shows a clear advantage at low NFE and remains stable across inference budgets. It achieves the lowest sliced W_2 at all selected NFEs and the lowest $\mathrm{MMD}^2_{\mathrm{RBF}}$ through NFE=128. Figure 6 shows the quality–efficiency curves.

C ADDITIONAL SINGLE-CELL RESULTS

C.1 DATA, SPLIT, AND EVALUATION

We construct a Tahoe-100M panel containing all combinations of five cell lines, eight drugs, and three doses (0.05, 0.5, and 5), giving 120 conditions $c = (\ell, d, a)$. The cell-line identifiers are `CVCL_0023`, `CVCL_0028`, `CVCL_0069`, `CVCL_0099`, and `CVCL_0131`. The drugs are Berbamine (dihydrochloride), Binimetinib, Cytarabine, Daidzin, Everolimus, Flumatinib (mesylate), Fumaric acid, and Gemcitabine. Each condition contains multiple measured cells; preparation retains up to 600 treated cells per condition, yielding 70,064 treated cells in total. These counts refer to individual cells, not independent experimental replicates.

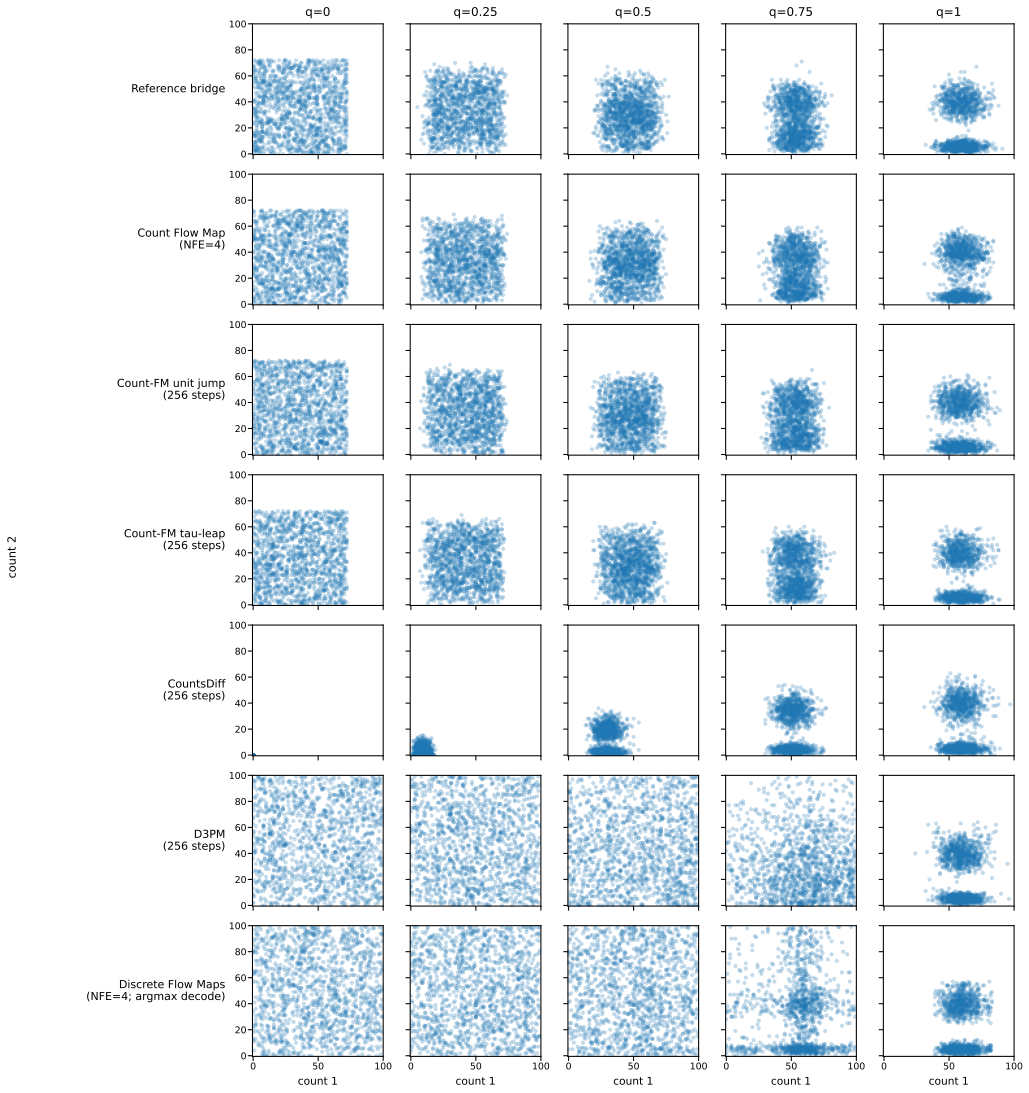

Figure 5: Generation paths for the 2-D simulation at normalized progress q. Count Flow Map and Discrete Flow Maps use NFE=4 finite-time compositions. Count-FM, CountsDiff, and D3PM are shown at matched progress along 256-step sampling paths.

The split is performed at the condition level: all retained treated cells sharing the same cell-line–drug–dose combination belong to exactly one split. Using the fixed split seed 42, a coverage-constrained assignment selects 24 test conditions (20%) and then 18 validation conditions (15%), leaving 78 training conditions (65%). Selection spreads holdouts across cell-line–drug pairs, cell lines, and drugs. A condition can be held out only if at least one other dose of its cell-line–drug pair remains in training. The resulting split is:

			Conditions at each dose		
Split	Conditions	Treated cells	0.05	0.5	5
Training	78	45,024	28	25	25
Validation	18	10,710	4	6	8
Testing	24	14,330	8	9	7
Total	120	70,064	40	40	40

Table 3: 32-D simulation endpoint quality at selected inference budgets. Entries are mean ± standard deviation over five seeds. Lower is better for all metrics. The best quality metric at each NFE is shown in bold.

Method	NFE	$SW_2 \downarrow$	$MMD^2_{RBF} \downarrow$	Time (s)↓
Count Flow Map	1	**0.556±0.134**	**0.0007±0.0004**	0.005±0.000
	4	**0.724±0.134**	**0.0015±0.0009**	0.012±0.001
	128	**0.518±0.048**	**0.0005±0.0001**	0.308±0.005
	256	**0.555±0.087**	0.0005±0.0002	0.617±0.008
Count-FM + unit jump	1	10.898±0.490	0.1717±0.0029	0.003±0.000
	4	9.388±0.284	0.1762±0.0018	0.005±0.000
	128	1.273±0.105	0.0076±0.0014	0.109±0.001
	256	0.804±0.105	0.0025±0.0004	0.215±0.003
Count-FM + binomial τ-leap	1	4.511±0.148	0.1047±0.0035	0.003±0.000
	4	2.522±0.113	0.0237±0.0006	0.005±0.000
	128	0.796±0.069	0.0011±0.0001	0.086±0.001
	256	0.826±0.061	0.0010±0.0002	0.172±0.002
CountsDiff	1	7.577±0.111	0.2551±0.0134	0.002±0.000
	4	4.328±0.135	0.0865±0.0022	0.005±0.000
	128	0.622±0.067	0.0006±0.0001	0.128±0.001
	256	0.599±0.154	**0.0004±0.0001**	0.254±0.002
D3PM	1	7.681±0.331	0.2656±0.0132	0.004±0.000
	4	4.698±0.129	0.0917±0.0027	0.017±0.000
	128	2.545±0.116	0.0196±0.0007	0.582±0.002
	256	1.507±0.084	0.0054±0.0004	1.167±0.003
Discrete Flow Maps	1	4.642±0.081	0.0850±0.0022	0.004±0.000
	4	2.150±0.193	0.0121±0.0010	0.013±0.001
	128	1.303±0.145	0.0027±0.0005	0.351±0.002
	256	1.634±0.172	0.0034±0.0005	0.701±0.004

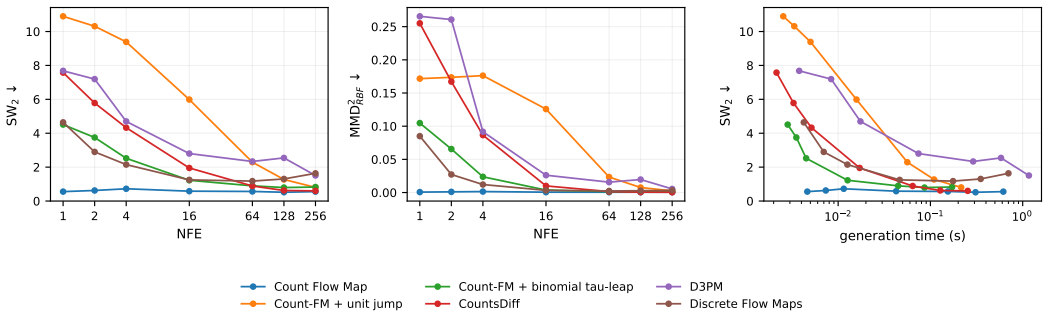

Figure 6: 32-D quality–efficiency curves. Left and center show sliced W_2 and RBF MMD^2 versus NFE. Right shows sliced W_2 versus measured generation time. Count Flow Map achieves strong quality at NFE=1 and retains the lowest sliced W_2 across the full NFE grid. CountsDiff only slightly improves MMD^2_{RBF} at the largest inference budget.

Training therefore covers all five cell lines, all eight drugs, all three dose values, and all 40 cell-line–drug pairs. Of these pairs, 38 have two training doses and two have one. For example, CVCL_0023 with Binimetinib is observed in training at doses 0.05 and 0.5, while dose 5 is held out for testing. The task evaluates prediction of unobserved dose-specific distributions for known cell-line–drug pairs; it includes both dose interpolation and extrapolation relative to the doses observed for each pair. Validation conditions support checkpoint and sampling-budget selection, and test conditions provide the reported predictive evaluation. The same split is used for every method, pairing, and training seed.

Each cell is represented by a 2,000-dimensional nonnegative integer count vector. Highly variable genes are selected using only training treated cells and their matched DMSO controls. The conditioning vector has 14 entries: five cell-line indicators, eight drug indicators, and one standardized $\log(1 + \text{dose})$ value. Dose-standardization statistics are computed from training conditions only. Thus the model conditions on the individual variables rather than a separate categorical label for each of the 120 combinations.

For non-OT training, each treated cell is paired with an independently sampled DMSO cell from the same cell line and matched plate. Control sampling uses replacement, so DMSO cells may be reused across conditions and splits. The 70,064-cell total above counts treated cells and does not include these reused control draws. OT training permutes treated endpoints to minimize the total raw-count L_1 distance within each condition and plate, preserving both empirical endpoint marginals. This is a numerical training coupling, not an inferred pairing of biological cells. Both count models use each coupling; validation and test data remain unchanged.

Evaluation uses the same fixed subset of 400 cells per held-out condition for every method and seed: 7,200 validation cells and 9,600 test cells, drawn from the larger retained pools above. For distributional evaluation, counts are library-size normalized to 10^4, log-transformed, and projected onto 50 principal components fitted using training cells only. This projection is used for evaluation; generation remains in the original 2,000-dimensional count space. Sliced W_2 uses 128 random projections. Metrics are computed within each condition and averaged equally across conditions. For each trained method, we report the mean and sample standard deviation of these averages across training seeds 42, 123, and 2026 on the fixed split. This describes variation across training runs, not biological replicate uncertainty. Linear and Sinkhorn baselines are each evaluated once and have no across-training-seed standard deviation.

For the table, perturbation effects relative to matched DMSO are

$$\Delta_{cg}^{\log \mathrm{FC}} = \log_2 \frac{\bar{x}_{1,cg} + 0.1}{\bar{x}_{0,cg} + 0.1}.$$

Generated effects replace $\bar{x}_{1,cg}$ by the generated mean. LogFC Pearson correlation is computed on the 200 genes with largest observed $|\Delta_{cg}^{\log \mathrm{FC}}|$; top-gene overlap is the fraction of genes shared by the observed and generated top-200 sets. These response genes are ranked by effect magnitude, without a significance-test threshold.

Neural architectures. The main MLP blocks use three hidden layers of 512 units with SiLU activations, without batch normalization or dropout. These settings apply to the count-model rate and transition networks, the conditional NB-VAE prior, posterior, and decoder networks, and the CPA and scGen/scVIDR encoder and decoder networks. Conditional NB-VAE, CPA, and the shared scGen/scVIDR VAE use latent dimension 128. Auxiliary feature and context encoders and method-specific output heads are retained. CPA retains its embeddings, adversaries with two hidden layers of 64 units, and dose networks with two hidden layers of 128 units, using their original activations and normalization.

Total neural parameter counts are approximately 14.52 million for Count Flow Map, 5.36 million for Count-FM, 7.02 million for conditional NB-VAE, 4.42 million for CPA, and 3.30 million for each of scGen and scVIDR. The two Count-FM samplers use the same rate network. Linear and Sinkhorn baselines have no neural network.

Count-model training and selection. Count Flow Map and Count-FM are trained for 50,000 updates with batch size 64, learning rate 2×10^{-4}, and EMA decay 0.9995. Count Flow Map uses four mixture components, CK weight one, a 2,000-update CK warm-up, and a 15,000-update interval-span warm-up. Generation ends at $\tau = 0.98$. Every 5,000 updates, the EMA model is scored by condition-averaged validation sliced W_2, at 1 NFE for Count Flow Map and 128 binomial τ-leap steps for Count-FM. The best checkpoint is selected separately for each model, training coupling, and seed and is held fixed across inference budgets. The full training budget is completed.

The two Count-FM samplers share the selected rate network. Their NFE is subsequently selected on validation separately for each sampler, training coupling, and seed. Unit jumps use $\{16, 64, 128, 256, 512\}$ and binomial τ-leaps use $\{1, 4, 16, 64, 128, 256\}$. Count Flow Map is reported at fixed 1 and 16 NFE, with its full curve evaluated on the latter grid.

Baseline training and selection. Baseline results use the repository implementations with the architecture settings above. Conditional NB-VAE and CPA are trained for 50,000 updates, with validation every 5,000 updates. Conditional NB-VAE uses AdamW with batch size 64 and learning rate 2×10^{-4}. CPA uses Adam with batch size 256, learning rate 3×10^{-4} for its autoencoder and adversaries, and 4×10^{-3} for its dose networks.

The shared scGen/scVIDR VAE uses Adam with batch size 64 and learning rate 10^{-3}. It is trained for 100 epochs, with validation every 10 epochs. Each epoch comprises 1,407 updates over the training treated cells and their matched controls, giving 140,700 updates in total. scGen uses nearest-dose latent arithmetic and scVIDR uses continuous log-dose scaling.

Each neural baseline selects the checkpoint with the lowest condition-averaged validation sliced W_2 among ten evaluated checkpoints, using the same 7,200 validation cells and fixed training-only PCA as the count models. The full training budget is completed. scGen and scVIDR share one VAE training trajectory per seed but select checkpoints separately using their respective downstream predictions, so their selected weights can differ. All neural methods use training seeds 42, 123, and 2026. Validation treated expression is used only for model selection, and test treated expression is used only for final evaluation. Parameter fitting and response-statistic estimation use training data only.

Generation timing. Generation time is measured on the full 9,600-cell test subset, using batches of 256 where applicable. Each seed's time is the median of three runs after one untimed warm-up, including generation transfers and excluding model loading, response-model preparation, PCA, and metric computation. Training-only response statistics for scGen/scVIDR and library-size summaries for CPA are prepared before timing. The table summarizes these times across training seeds. CountFM's selected NFE can differ between seeds.

C.2 Condition-level perturbation recovery

We examine perturbation strength and response patterns across genes. For condition c and gene g, let $\bar{x}_{0,cg}$, $\bar{x}_{1,cg}$, and $\bar{x}_{\mathrm{gen},cg}$ denote mean raw counts in matched DMSO controls, observed treated cells, and generated cells, respectively. We define

$$\delta_{cg} = \log(1 + \bar{x}_{1,cg}) - \log(1 + \bar{x}_{0,cg}),$$
$$\widehat{\delta}_{cg} = \log(1 + \bar{x}_{\mathrm{gen},cg}) - \log(1 + \bar{x}_{0,cg}).$$

These effects use all 2,000 retained genes and differ from the table's top-200-gene logFC metric. Panel C of Figure 2 compares observed strength $\|\delta_c\|_2$ with generated strength $\|\widehat{\delta}_c\|_2$ across the 24 test conditions. We report Pearson correlation across these magnitudes and fit generated magnitudes against observed magnitudes through the origin. A slope below one indicates overall underestimation.

Figure 7 examines gene-response patterns through a common PCA fitted using training-condition effects and the correlation $\mathrm{Corr}_g(\delta_{cg}, \widehat{\delta}_{cg})$ for each test condition. All effect illustrations use one training run per coupling, with the same OT model evaluated at 1 and 16 NFE. The plots show response-pattern recovery across multiple conditions, alongside varying errors in projected direction and magnitude. Consistent with Figure 2C, these results illustrate that Count Flow Map captures variation in perturbation responses with very few model evaluations.

D Additional Neural Forecasting Details and Results

D.1 Data and Forecasting Protocol

We use the 50 ms-binned SpikeProphecy release of the Steinmetz recordings (Steinmetz et al., 2019; Anonymous, 2026), pinned to dataset revision `10da560900af2b7a5475faca63cbce4be8f860b4`. The six recording indices are 004, 007, 011, 018, 025, and 033, containing 703, 1,104, 613, 490, 677, and 538 neurons, respectively. Their respective lengths are 60,887, 65,200, 68,927, 64,586, 60,569, and 64,850 bins, giving 385,019 population count vectors in total.

We retain the release's chronological 70%/15%/15% train/validation/test boundaries. Each input consists of ten consecutive bins, followed by one target bin. Histories and targets must lie entirely

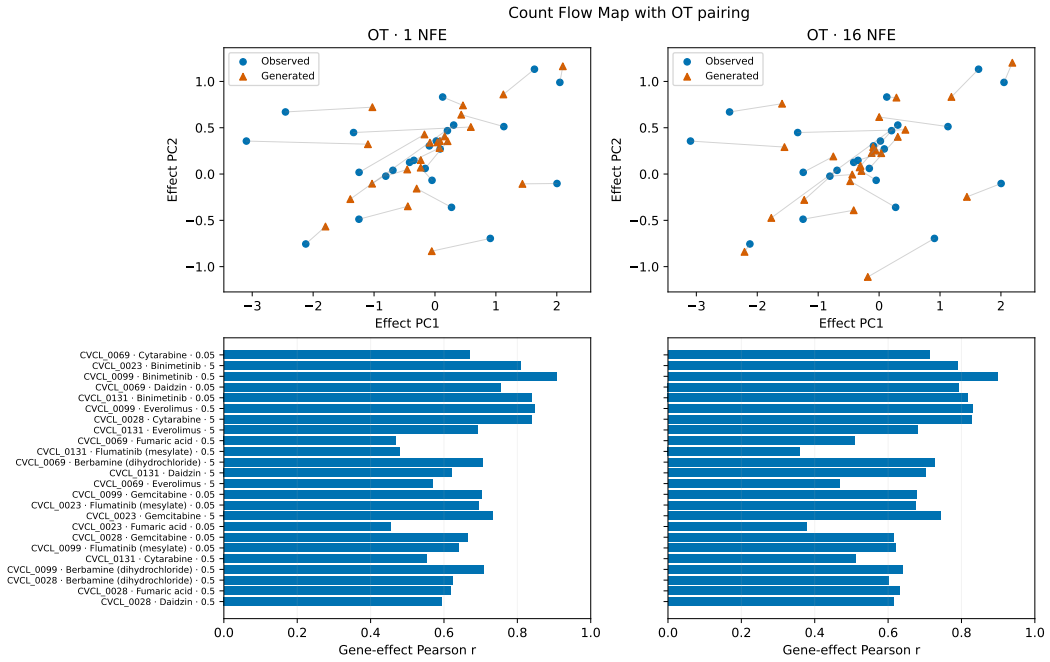

Figure 7: Condition-level perturbation recovery by Count Flow Map with OT coupling at 1 and 16 NFE. Both budgets use the same model from one fixed training run. Top: observed and generated effects projected onto common training-only PCA axes. Lines connect the same condition. Bottom: Pearson correlations between observed and generated effects across all 2,000 retained genes for each test condition. The 24 conditions have the same order in both panels, sorted by decreasing observed effect magnitude. These correlations differ from the table's top-200-gene logFC correlation.

within the same split, giving 269,451 eligible training windows across the six recordings. These overlapping windows are not independent biological replicates. Models are fitted separately to each recording. Neither neurons nor count vectors are aligned across recordings. The task conditions only on spike-count history, without behavioral or stimulus covariates.

Training pairs are the naturally observed consecutive-bin pairs $(X_0, X_1) = (Y_b, Y_{b+1})$, with context $H_b = (Y_{b-9}, \ldots, Y_b)$. We use the signed-binomial bridge and conditional kernels described in the main text, with context fixed throughout each flow trajectory. Inference starts from the last observed count vector Y_b and applies a uniform flow-time grid on $[0, 0.98]$. NFE counts kernel or rate-network evaluations within one predicted physical bin. History encoding is performed once per forecast. Count-FM's unit-jump and binomial τ-leap samplers use the same trained rate network.

Architecture and training. Both count-flow models use a two-layer GRU history encoder with width 256 and MLP heads with three hidden layers of width 384 and SiLU activations. Count Flow Map uses four mixture components. Count features concatenate $x/4$ and $\log(1+x)/\log 5$. These are network inputs, while the stochastic state and generated outputs remain nonnegative integer counts.

Each learning-rate candidate is trained for 30,000 AdamW updates, with batch size 128, weight decay 10^{-5}, gradient-norm clipping at 5, and learning rate in $\{10^{-4}, 3 \times 10^{-4}\}$. We use three training seeds (42, 123, and 2026) and an exponential moving average of weights with decay 0.9995 for validation and evaluation.

Count Flow Map uses the diagonal rate-matching and off-diagonal Chapman–Kolmogorov losses described in the main text, without an auxiliary endpoint loss. Both losses are divided by the number of neurons. The consistency weight increases linearly from zero to one over 2,000 updates. Its maximum sampled interval length increases linearly from 0.1 to 0.98 over 10,000 updates, after which the full interval range is used. The intermediate time bisects the sampled interval. The two-step consistency target is sampled with stop-gradient at the current online weights. The moving-average

model is used for validation and evaluation. Count-FM uses the same diagonal rate-matching objective and optimizer settings.

We validate every 3,000 updates using 512 validation histories and 32 draws per history. The checkpoint and learning rate are chosen by validation energy score, averaged over 1 and 16 NFE for Count Flow Map and evaluated at 64 NFE with the binomial τ-leap sampler for Count-FM. Selected checkpoints are then held fixed across all reported budgets and both Count-FM samplers. The additional NFE sweep evaluates these frozen models without retraining or test-based checkpoint selection. All formal training candidates complete the prescribed 30,000 updates.

D.2 Metrics, Aggregation, and Timing

We evaluate 4,096 evenly spaced eligible test histories per recording, using the same histories for all methods, budgets, and seeds. This gives 24,576 distinct test histories across the six recordings. Repeating training seeds does not create additional test observations. Each history has one observed next-bin count vector $y \in \mathbb{N}_0^D$ and $M = 64$ generated vectors $x^{(1)}, \ldots, x^{(M)}$. We report the energy score and population CRPS (Gneiting & Raftery, 2007). For CRPS, we use the finite-ensemble correction that excludes self-pairs (Ferro, 2014).

The dimension-normalized energy-score estimator is

$$\widehat{\mathrm{ES}} = \frac{1}{\sqrt{D}} \left[\frac{1}{M} \sum_{m=1}^{M} \|x^{(m)} - y\|_2 - \frac{1}{M} \sum_{m=1}^{M/2} \|x^{(m)} - x^{(m+M/2)}\|_2 \right]. \tag{21}$$

The second term uses disjoint pairs of independent draws. Define mean population counts $a^{(m)} = D^{-1} \sum_j x_j^{(m)}$ and $a_y = D^{-1} \sum_j y_j$. Their CRPS estimator is

$$\widehat{\mathrm{CRPS}} = \frac{1}{M} \sum_{m} |a^{(m)} - a_y| - \frac{1}{2M(M-1)} \sum_{m \neq n} |a^{(m)} - a^{(n)}|. \tag{22}$$

We also report neuron-wise RMSE of the predictive mean, subtracting the Monte Carlo variance of that mean before taking the square root. For a recording with C evaluated histories,

$$\widehat{\mathrm{RMSE}} = \sqrt{\max\left\{0, \frac{1}{CD} \sum_{i=1}^{C} \sum_{j=1}^{D} \left[(\bar{x}_{ij} - y_{ij})^2 - \frac{s_{ij}^2}{M} \right] \right\}}, \tag{23}$$

where $\bar{x}_{ij}$ and s_{ij}^2 are the sample mean and unbiased sample variance across generated draws. The correction removes finite-draw bias from the squared-error estimate, not from its square root.

Scores are first computed within each recording, using its own neuron dimension, and then averaged equally over the six recordings for each training seed. We report the mean and sample standard deviation of these three seed-level averages. The uncertainty therefore summarizes variability across repeated training runs on fixed recordings, rather than sampling uncertainty across animals. The dimension normalization and per-neuron population statistic allow comparison of scalar scores without pooling differently sized neuronal populations.

All NFE rows are timed under a common protocol on the NVIDIA GeForce RTX 2080 Ti used throughout the paper. Timing processes 16 test histories sequentially, generating 64 draws for each. After an untimed warm-up pass, we take the median per-history latency over three repetitions, synchronizing CUDA before and after each timed pass. Timing includes history encoding, count sampling, and input/output device transfers, and excludes model loading and metric computation. Thus latency refers to a 64-draw population forecast, not one generated vector. Table 4 reports energy score, population CRPS, RMSE, and latency for all 25 method–budget combinations. Figure 8 shows population CRPS against NFE and measured latency over the shared 1–64 NFE range. Figure 9A additionally shows joint forecast quality, measured by energy score, across the full evaluated NFE grid.

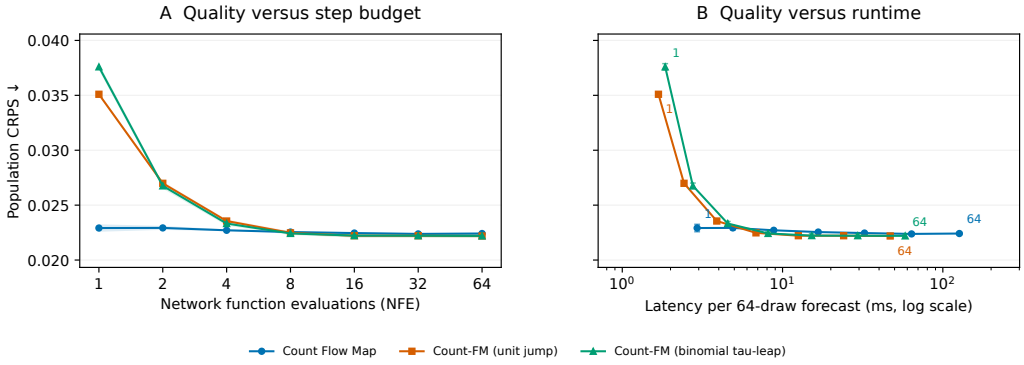

Figure 8: Population forecast quality and sampling cost. Population CRPS versus NFE in A and latency per 64-draw forecast in B, using fixed checkpoints and budgets 1–64. Points average six recordings equally. Error bars show one sample standard deviation across three seed means. Timing includes history encoding and count sampling on an RTX 2080 Ti. Count-FM budgets through 256 NFE are included in Table 4.

D.3 Population Events and Forecast Diagnostics

High-activity events and the dependence control. For each recording, let q be the empirical 95th percentile of total population count over training bins. We predict the event $\sum_j Y_{b+1,j} > q$ using

$$\widehat{p}_b = \frac{1}{M} \sum_{m=1}^{M} \mathbf{1}\left\{ \sum_j x_j^{(m)} > q \right\}.$$

We evaluate the binary Brier score

$$\left(\widehat{p}_b - \mathbf{1}\left\{ \sum_j y_j > q \right\} \right)^2$$

(Brier, 1950), averaged over test histories. Thresholds for recordings 004, 007, 011, 018, 025, and 033 are respectively 220, 437, 226, 273, 183, and 174 total spikes per bin. The corresponding observed test-event frequencies are approximately 4.10%, 9.18%, 10.57%, 3.10%, 0.90%, and 3.93%. The event threshold is not recomputed on the test split.

The shuffled control independently permutes draw indices for every neuron within each conditioning history. This preserves each neuron's finite-ensemble marginal and changes the cross-neuron pairing. We score both ensembles with the same $M = 64$, without a finite-ensemble Brier correction. At 1 NFE, mean Brier score is 0.031788 for joint samples and 0.032642 after shuffling. The paired increase is 0.000854 ± 0.000139. At 16 NFE, the corresponding scores are 0.031626 and 0.032894, with an increase of 0.001268 ± 0.000125. These amount to approximately 2.6% and 3.9% reductions relative to the shuffled control. The benefit varies by recording and is close to zero in recording 025, where test events are rare. This control measures the forecast value of dependence in the generated population distribution. It does not identify synaptic or causal interactions.

Illustrative forecast segment. Figure 3A–B uses the first 200 eligible consecutive test histories from recording 004 and one fixed training run. The displayed interval is the central 90% sample interval of population mean count. Each forecast conditions on the actual preceding ten bins, so the 10-second display consists of successive next-bin predictions rather than an autonomous 10-second forecast. The high-activity threshold in the plot is q/D, consistent with its per-neuron vertical axis. Aggregate scores use all six recordings and all three seeds.

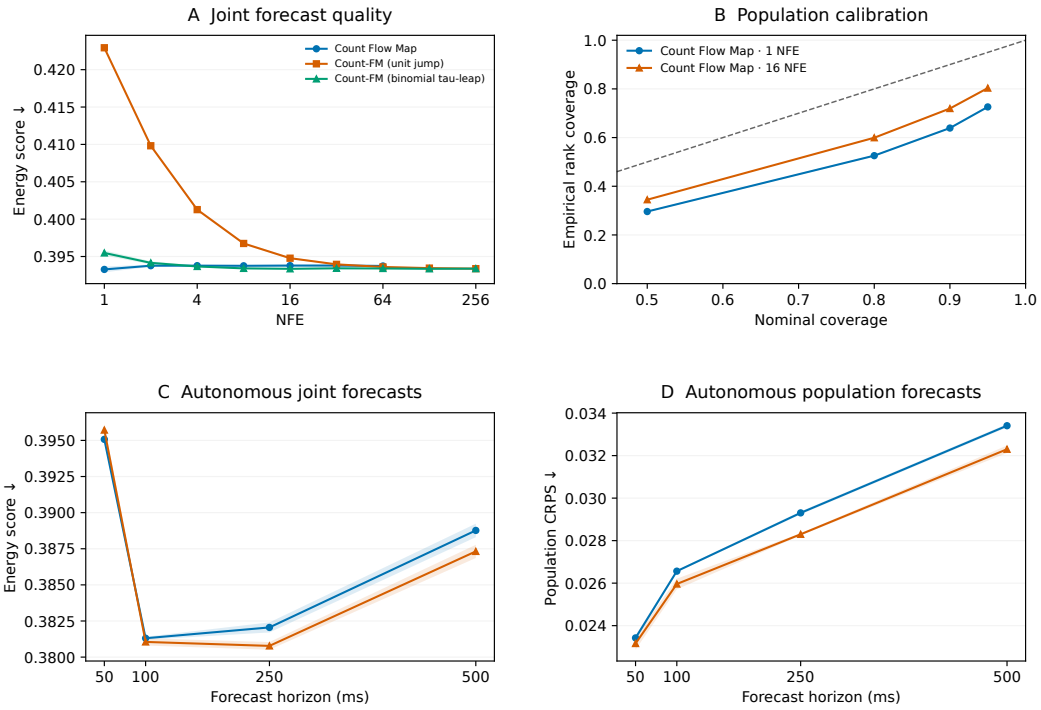

Figure 9: Neural forecasting diagnostics. A shows dimension-normalized energy score across the full NFE grid. B compares randomized population-rank coverage with nominal coverage for Count Flow Map. The dashed line denotes calibration. C–D show energy score and population CRPS under autonomous forecasting, using generated rather than observed future histories. In B–D, orange and blue indicate 16 and 1 NFE per predicted bin. Points average six recordings equally. Bars or bands show one sample standard deviation across three seed means.

Calibration. To handle discrete ties and finite ensembles, define the randomized rank

$$R = \frac{\#\{m : a^{(m)} < a_y\} + U\big(\#\{m : a^{(m)} = a_y\} + 1\big)}{M + 1}, \qquad U \sim \mathrm{Unif}(0, 1). \tag{24}$$

For nominal level c, we report the fraction of histories with $R \in [(1-c)/2, (1+c)/2]$. Figure 9B shows undercoverage. At nominal 90%, empirical rank coverage is approximately 64% at 1 NFE and 72% at 16 NFE. Increasing NFE improves this diagnostic, although predictive uncertainty remains narrower than warranted by the observations.

Autonomous forecasts. Starting from an observed history, we recursively feed generated counts into each sample path's history, without further observations. We evaluate 64 paths for 256 test starting points per recording at horizons 50, 100, 250, and 500 ms. NFE is per predicted bin, so a 500 ms forecast requires ten next-bin predictions. Figure 9C–D shows comparable joint scores at 1 and 16 NFE and increasing population CRPS with horizon. Energy score is not monotonic in horizon. These 256 horizon-eligible starting points differ from the 4,096-history next-bin evaluation.

Method	NFE	Energy score↓	Pop. CRPS↓	RMSE↓	Time (ms)↓
Count Flow Map	1	0.3933 ± 0.0004	0.02292 ± 0.00034	0.5531 ± 0.0043	2.93 ± 0.01
	2	0.3938 ± 0.0002	0.02293 ± 0.00018	0.5522 ± 0.0010	4.90 ± 0.01
	4	0.3938 ± 0.0000	0.02271 ± 0.00008	0.5523 ± 0.0002	8.81 ± 0.01
	8	0.3938 ± 0.0001	0.02255 ± 0.00008	0.5526 ± 0.0001	16.69 ± 0.02
	16	0.3938 ± 0.0001	0.02246 ± 0.00008	0.5528 ± 0.0002	32.45 ± 0.04
	32	0.3938 ± 0.0002	0.02238 ± 0.00009	0.5529 ± 0.0002	63.80 ± 0.06
	64	0.3937 ± 0.0002	0.02241 ± 0.00010	0.5529 ± 0.0002	126.77 ± 0.07
Count-FM (unit jump)	1	0.4229 ± 0.0002	0.03510 ± 0.00012	0.5851 ± 0.0002	1.68 ± 0.00
	2	0.4098 ± 0.0002	0.02699 ± 0.00016	0.5706 ± 0.0003	2.43 ± 0.01
	4	0.4013 ± 0.0002	0.02355 ± 0.00014	0.5609 ± 0.0003	3.89 ± 0.01
	8	0.3967 ± 0.0002	0.02248 ± 0.00015	0.5559 ± 0.0002	6.83 ± 0.04
	16	0.3948 ± 0.0001	0.02222 ± 0.00016	0.5538 ± 0.0002	12.55 ± 0.03
	32	0.3940 ± 0.0001	0.02221 ± 0.00016	0.5530 ± 0.0002	24.05 ± 0.06
	64	0.3936 ± 0.0001	0.02219 ± 0.00017	0.5527 ± 0.0001	46.94 ± 0.04
	128	0.3934 ± 0.0001	0.02224 ± 0.00017	0.5526 ± 0.0002	92.74 ± 0.14
	256	0.3934 ± 0.0001	0.02223 ± 0.00017	0.5525 ± 0.0002	184.44 ± 0.12
Count-FM (τ-leap)	1	0.3955 ± 0.0003	0.03759 ± 0.00029	0.5584 ± 0.0010	1.85 ± 0.00
	2	0.3942 ± 0.0002	0.02676 ± 0.00026	0.5550 ± 0.0004	2.76 ± 0.01
	4	0.3937 ± 0.0001	0.02333 ± 0.00019	0.5535 ± 0.0002	4.55 ± 0.02
	8	0.3934 ± 0.0001	0.02242 ± 0.00015	0.5529 ± 0.0001	8.12 ± 0.03
	16	0.3934 ± 0.0001	0.02222 ± 0.00017	0.5527 ± 0.0001	15.20 ± 0.03
	32	0.3934 ± 0.0001	0.02222 ± 0.00017	0.5526 ± 0.0002	29.45 ± 0.03
	64	0.3934 ± 0.0001	0.02220 ± 0.00017	0.5526 ± 0.0002	58.21 ± 0.14
	128	0.3934 ± 0.0001	0.02224 ± 0.00017	0.5525 ± 0.0002	115.31 ± 0.12
	256	0.3934 ± 0.0001	0.02224 ± 0.00017	0.5525 ± 0.0002	229.72 ± 0.28

Table 4: Full neural forecasting results. Each entry is the mean $\pm$ sample standard deviation of three seed-level means, each averaging the six recordings equally. All rows use 4,096 test histories per recording, 64 draws per history, and the same selected checkpoint within each method, recording, and seed. The two Count-FM samplers share a rate network. Time is latency per 64-draw forecast under the common timing protocol. Values retain the precision of the exported results. Displayed zero standard deviations reflect rounding and should not be interpreted as zero variability.