ENTERING THE ERA OF DISCRETE DIFFUSION MODELS: A BENCHMARK FOR SCHRÖDINGER BRIDGES AND ENTROPIC OPTIMAL TRANSPORT

Anonymous authors
Paper under double-blind review

ABSTRACT

The Entropic Optimal Transport (EOT) problem and its dynamic counterpart, the Schrödinger bridge (SB) problem, play an important role in modern machine learning, linking generative modeling with optimal transport theory. While recent advances in discrete diffusion and flow models have sparked growing interest in applying SB methods to discrete domains, there is still no reliable way to evaluate how well these methods actually solve the underlying problem. We address this challenge by introducing a benchmark for SB on discrete spaces. Our construction yields pairs of probability distributions with analytically known SB solutions, enabling rigorous evaluation. As a byproduct of building this benchmark, we obtain two new SB algorithms, DLightSB and DLightSB-M, and additionally extend prior related work to construct the α-CSBM algorithm. We demonstrate the utility of our benchmark by evaluating both existing and new solvers in high-dimensional discrete settings. This work provides the first step toward proper evaluation of SB methods on discrete spaces, paving the way for more reproducible future studies.

1 INTRODUCTION

The Entropic Optimal Transport (Cuturi, 2013, EOT) problem and its dynamic counterpart, the Schrödinger bridge (Schrödinger, 1931, SB), have recently attracted significant attention in the machine learning community due to their relevance for generative modeling and unpaired learning. A variety of methods have been developed to solve these problems in *continuous data spaces* such as (Daniels et al., 2021; Gushchin et al., 2023a; 2024b; Mokrov et al., 2024; Vargas et al., 2021; Chen et al., 2021; Shi et al., 2023; De Bortoli et al., 2024; Korotin et al., 2024; Gushchin et al., 2024a).

At the same time, much real world data are *discrete by nature*, including text (Austin et al., 2021; Gat et al., 2024), molecular graphs (Vignac et al., 2022; Qin et al., 2024; Luo et al., 2024), and protein sequences (Campbell et al., 2024). Others are *discrete by construction*, such as vector-quantized representations of images and audio (Van Den Oord et al., 2017; Esser et al., 2021).

Given the prevalence of such discrete data and the rapid progress in discrete diffusion/flow models (Hoogeboom et al., 2021; Austin et al., 2021; Campbell et al., 2022; Lou et al., 2023; Sahoo et al., 2024; Campbell et al., 2024; Gat et al., 2024), research on SBs has attracted growing attention in recent years. For instance, several recent works have already taken first steps in this direction (Kim et al., 2024, DDSBM;Ksenofontov & Korotin, 2025, CSBM), adapting diffusion methodologies from (Austin et al., 2021, D3PM;Vignac et al., 2022, DiGress), respectively.

Despite the rapid progress in discrete SB research, there is still a lack of evaluation benchmarks. These benchmarks enable us to determine whether SB methods actually solve the intended mathematical problem, separating true algorithmic performance from artifacts of specific parameterizations, regularization schemes, and other implementation choices. While this has recently become possible in the continuous setting of Schrodinger Bridges (Gushchin et al., 2023b), no such approach exists for discrete data, leaving it unclear how closely SB solvers approximate the true solution of the SB problem on discrete domains. To address this gap, we make the following **contributions:**

- **Theory & Methodology.** We present a general methodology to create pairs of discrete probability distributions with known SB solutions (§3.1). To overcome tractability issues of the methodology in discrete spaces, we introduce a CP-based parameterization (§3.2). This parameterization yields a closed-form SB and enables a practically feasible benchmark construction.

- **Algorithms.** The CP-based parameterization of our benchmark allows us to construct two novel discrete SB methods: DLightSB and DLightSB-M (§4.3 and §4.4). Which mirror their continuous-space counterparts LightSB and LightSB-M (Korotin et al., 2024; Gushchin et al., 2024a). Additionally, we introduce α-CSBM (§4.2), a new solver for discrete SB. Which combines the recent discrete-space solver CSBM (Ksenofontov & Korotin, 2025) with the incremental/online update strategy of α-DSBM used in continuous settings (De Bortoli et al., 2024).
- **Practice.** We use these benchmark pairs to evaluate both existing and newly introduced SB solvers in high-dimensional settings

Notation. We consider a discrete state space $\mathcal{X} = \mathbb{S}^D$, where $\mathbb{S} = \{0, 1, \dots, S-1\}$ is the set of S categories and D is the dimensionality. Each $x \in \mathcal{X}$ is a D-dimensional vector $x = (x^1, \dots, x^D)$. Time is discretized as $\{t_n\}_{n=0}^{N+1}$ with $0 = t_0 < t_1 < \cdots < t_N < t_{N+1} = 1$. This gives $N+2$ time points and defines the *path space* $\mathcal{X}^{N+2}$ with the tuple $x_{\text{in}} \overset{\text{def}}{=} (x_{t_1}, \dots, x_{t_N}) \in \mathcal{X}^N$ collecting the intermediate states. The set $\mathcal{P}(\mathcal{X}^{N+2})$ comprises all discrete time stochastic processes on the path space, with $\mathcal{M}(\mathcal{X}^{N+2}) \subset \mathcal{P}(\mathcal{X}^{N+2})$ denoting the subset of *Markov processes*. Any $q \in \mathcal{M}(\mathcal{X}^{N+2})$ admits forward and backward representations: $q(x_0, x_{\text{in}}, x_1) = q(x_0) \prod_{n=1}^{N+1} q(x_{t_n}|x_{t_{n-1}}) = q(x_1) \prod_{n=1}^{N+1} q(x_{t_{n-1}}|x_{t_n})$, where $q(\cdot|\cdot)$ denotes conditional probabilities.

2 Background: Problem Statement

This section provides an overview of the discrete-time Schrödinger Bridge problem. First, we present the dynamic SB and its reduction to a static problem (§2.1). Next, we analyze diffusion-type reference processes (§2.2) that yield practical cost functions, linking to the EOT framework in §2.3. Finally, we introduce our problem setting (§2.4).

2.1 Dynamic and Static Schrödinger Bridges on Discrete Spaces

Dynamic Schrödinger Bridge. The original SB problem (Schrödinger, 1931; 1932; Léonard, 2013) seeks to find a process $q^* \in \mathcal{P}(\mathcal{X}^{N+2})$ interpolating between an initial distribution p_0 at $t_0 = 0$ and a final distribution p_1 at $t_{N+1} = 1$. This distribution is found by minimizing the Kullback-Leibler (KL) divergence with respect to a given Markov reference process $q^{\text{ref}} \in \mathcal{M}(\mathcal{X}^{N+2})$ subject to the marginal constraints $p_0(x_0) = q(x_0)$ and $p_1(x_1) = q(x_1)$. One finds

$$q^* = \operatorname*{arg\,min}_{q \in \Pi_N(p_0, p_1)} \text{KL}\left(q(x_0, x_{\text{in}}, x_1) \| q^{\text{ref}}(x_0, x_{\text{in}}, x_1)\right), \tag{1}$$

where $\Pi_N(p_0, p_1) \subset \mathcal{P}(\mathcal{X}^{N+2})$ denotes the subset of $\mathcal{X}$-valued stochastic processes which have p_0 and p_1 as marginals at times $t_0 = 0$ and $t_{N+1} = 1$, respectively. In other words, the dynamic SB problem seeks the stochastic process q^* that minimally deviates from a reference process q^{ref} while respecting the boundary distributions p_0 and p_1.

Static Schrödinger Bridge. We now introduce the static formulation of the SB. This begins with observing that (1) admits the following decomposition:

$$\min_{q \in \Pi_N(p_0, p_1)} \Big[\text{KL}\left(q(x_0, x_1) \| q^{\text{ref}}(x_0, x_1)\right) + \mathbb{E}_{q(x_0, x_1)} \text{KL}\left(q(x_{\text{in}}|x_0, x_1) \| q^{\text{ref}}(x_{\text{in}}|x_0, x_1)\right)\Big]. \tag{2}$$

We further note that the conditional KL term in (2) vanishes when $q(x_{\text{in}}|x_0, x_1) = q^{\text{ref}}(x_{\text{in}}|x_0, x_1)$. Thus, we restrict q to the set of processes that satisfy this condition. This set is known as *the reciprocal class* of q^{ref} and is denoted by $\mathcal{R}^{\text{ref}}(\mathcal{X}^{N+2}) \subset \mathcal{P}(\mathcal{X}^{N+2})$. Under this restriction, the optimization reduces to the first KL term alone, leading directly to the static SB problem

$$q^*(x_0, x_1) = \operatorname*{arg\,min}_{q \in \Pi(p_0, p_1)} \text{KL}\left(q(x_0, x_1) \| q^{\text{ref}}(x_0, x_1)\right), \tag{3}$$

where $\Pi(p_0, p_1) \in \mathcal{P}(\mathcal{X}^2)$ is the set of joint distributions $q(x_0, x_1)$ whose marginals are p_0 and p_1.

2.2 Formulations of the Reference Process

The key ingredient in both SB formulations is the Markov reference process q^{ref}. In discrete space it is usually modeled as *a discrete-time Markov chain* defined by transition matrices $Q_n^{\text{ref}} \in [0,1]^{|\mathcal{X}|\times|\mathcal{X}|}$, where $q^{\text{ref}}(x_{t_n}^d | x_{t_{n-1}}^d) = [Q_n^{\text{ref}}]_{x_{t_{n-1}}^d, x_{t_n}^d}$. Assuming time-homogeneity ($Q_n^{\text{ref}} = Q^{\text{ref}}$ for all n), the n-step transition probabilities are given by the matrix power $\overline{Q}_n^{\text{ref}} = [Q^{\text{ref}}]^n$. To define Q, we further restrict to $D = 1$ for clarity, noting that for $D > 1$ the transition probabilities are obtained as a product over dimensions.

Remark. The reference process q^{ref} can also be defined in continuous time. In which transitions are characterized by rates instead of probabilities. Since controlling these rates is less direct and not all discrete processes admit a continuous analogue, we restrict our attention to the discrete setting, which is more flexible and well-suited for a benchmark construction.

We now introduce two popular diffusion-like transitions: uniform (Hoogeboom et al., 2021; Campbell et al., 2022) and Gaussian-like (Austin et al., 2021).

The reference process $q^{\text{ref}} \in \mathcal{M}(\mathcal{X}^{N+2})$ is modeled as a *discrete-state diffusion process*, i.e., a discrete-time Markov chain defined by transition matrices $Q_n^{\text{ref}} \in [0,1]^{|\mathcal{X}|\times|\mathcal{X}|}$, where $q^{\text{ref}}(x_{t_n}^d | x_{t_{n-1}}^d) = [Q_n^{\text{ref}}]_{x_{t_{n-1}}^d, x_{t_n}^d}$. Assuming time-homogeneity ($Q_n^{\text{ref}} = Q^{\text{ref}}$ for all n), the n-step transition probabilities are given by the matrix power $\overline{Q}_n^{\text{ref}} = [Q^{\text{ref}}]^n$. To define Q, we further restrict to $D = 1$ for clarity, noting that for $D > 1$ the transition probabilities are obtained as a product over dimensions. We now introduce two diffusion-like transitions: uniform (Hoogeboom et al., 2021; Campbell et al., 2022) and Gaussian-like (Austin et al., 2021).

Uniform Reference Process (q^{unif}). For unordered data, where no relation exists between categories, a natural choice is a so-called uniform transition matrix. For each dimension d, the elements of the transition matrix Q^{ref} are defined by

$$[Q^{\text{ref}}]_{x_{t_{n-1}}^d, x_{t_n}^d} = \begin{cases} 1-\gamma, & \text{if } x_{t_n}^d = x_{t_{n-1}}^d, \\ \frac{\gamma}{S-1}, & \text{if } x_{t_n}^d \neq x_{t_{n-1}}^d, \end{cases} \tag{4}$$

where $\gamma \in [0,1]$ is an stochasticity parameter. This reference process introduces randomness independently of the distance between categories. It assigns equal probability to transitioning into any different category, while having a staying probability $1-\gamma$. This ignores any inherent ordering or relationships among categories.

Gaussian Reference Process (q^{gauss}). For ordered data, where categories are expected to exhibit meaningful relations, a Gaussian-like transition matrix is more appropriate. With the stochasticity parameter $\gamma > 0$ and the maximum category distance $\Delta = S - 1$, the transition probabilities are

$$[Q^{\text{ref}}]_{x_{t_{n-1}}^d, x_{t_n}^d} = \frac{\exp\left(-\frac{4(x_{t_n}^d - x_{t_{n-1}}^d)^2}{(\gamma\Delta)^2}\right)}{\sum\limits_{\delta=-\Delta}^{\Delta} \exp\left(-\frac{4\delta^2}{(\gamma\Delta)^2}\right)}, \qquad x_{t_n}^d \neq x_{t_{n-1}}^d. \tag{5}$$

The diagonal entries take the remaining probability so that each row sums to 1.

2.3 Entropic Optimal Transport on Discrete Spaces

Following the construction of the Markov reference process in §2.2, the static SB problem (§3) takes a form equivalent to the entropic optimal transport (EOT) problem (Cuturi, 2013). Concretely, expressing $q^{\text{ref}}(x_0, x_1) = q^{\text{ref}}(x_0) q^{\text{ref}}(x_1|x_0)$ and setting $q^{\text{ref}}(x_0) = p_0(x_0)$, allows the minimization in equation (3) to be rewritten as

$$\min_{q\in\Pi(p_0,p_1)} \mathrm{KL}\left(q(x_0,x_1) \| q^{\text{ref}}(x_0,x_1)\right) =$$

$$= \min_{q\in\Pi(p_0,p_1)} \sum_{x_0,x_1} q(x_0,x_1) \log \frac{q(x_0,x_1)}{q^{\text{ref}}(x_0) q^{\text{ref}}(x_1|x_0)}$$

$$= \min_{q\in\Pi(p_0,p_1)} -H(q) - \sum_{x_0,x_1} q(x_0,x_1)\log q^{\text{ref}}(x_1|x_0) \underbrace{-\sum_{x_0,x_1} q(x_0,x_1)\log q^{\text{ref}}(x_0)}_{=\sum_{x_0} p_0(x_0)\log p_0(x_0) = -H(p_0)} \tag{6}$$

$$= \min_{q\in\Pi(p_0,p_1)} \mathbb{E}_{q(x_0,x_1)}\big[-\log q^{\text{ref}}(x_1|x_0)\big] - H(q) - \text{const}$$

$$= \min_{q\in\Pi(p_0,p_1)} \mathbb{E}_{(x_0,x_1)\sim q}\big[c(x_0,x_1)\big] - H(q) - \text{const},$$

where $H(q)$ is the entropy of q, while $H(p_0)$ remains constant when minimizing over q. Thus, the static SB formulation becomes equivalent to the entropy-regularized optimal transport problem with cost $c(x_0,x_1) = -\log q^{\text{ref}}(x_1|x_0)$. This perspective establishes a direct correspondence between SB and EOT, which we use in the design of our benchmark and methodological framework in §3.

Since the conditional distribution $q^{\text{ref}}(x_1|x_0)$ is obtained by taking the $(N+1)$-th power of Q^{ref}, it admits the following closed-form expression in the uniform case:

$$\overline{Q}^{\text{ref}}_{N+1} = \left(1-\gamma\frac{S}{S-1}\right)^{N+1}\mathbb{I} + \frac{1-\left(1-\gamma\frac{S}{S-1}\right)^{N+1}}{S}\mathbf{1}\mathbf{1}^\top, \tag{7}$$

where $\mathbf{1} = [1,\dots,1]^\top \in \mathbb{R}^S$ is a vector full of ones. From here it can be seen that $\overline{Q}^{\text{ref}}_{N+1}$ converges to $(1/S)\mathbf{1}\mathbf{1}^\top$ when $(N+1)\to\infty$, that is a uniform distribution over the number of categories S, the derivation of (7) can be found in Appendix A. In the case of the Gaussian reference process, the closed-form expression can also be obtained, but it is much more complex.

2.4 Problem Setup for Discrete Schrödinger Bridges

In this section, we recall the *generative SB task on discrete spaces*, a well-established problem in the SB and OT literature (Kim et al., 2024; Ksenofontov & Korotin, 2025). In short, the goal is to learn an SB process or coupling that performs transport between probability distributions on discrete spaces using available empirical data samples. Formally, we consider the following learning setup:

We assume the learner is given empirical datasets $\{x_0^{(i)}\}_{i\in I_0}$ and $\{x_1^{(j)}\}_{j\in I_1}$, $x_0^{(i)}, x_1^{(i)} \in \mathcal{X}$, consisting of i.i.d. samples from the unknown distributions $p_0, p_1 \in \mathcal{P}(\mathcal{X})$ where $\mathcal{X}$ is a discrete state space. Then, the task is to use these samples to find a solution q^* to the SB problem (1) or (3) between p_0 and p_1 for a given reference q^{ref}. Moreover, the solution should support out-of-sample generation so that for any new (x_0^{new}) one can generate $x_1^{\text{new}} \sim q^*(x_1|x_0^{\text{new}})$.

Despite recent progress in the development of SB methods that solve this task, there remains no standard methodology for performance evaluation, mainly due to the absence of ground-truth distribution pairs (p_0, p_1). In this work, we propose a benchmark construction, inspired by (Gushchin et al., 2023b), that enables standard evaluation of such methods on datasets built from SB pairs (x_0, x_1) with known $q^*(x_1|x_0)$. Such datasets provide more informative metrics and offer a consistent framework for assessing the performance of SB methods on discrete spaces.

Remark. Our paper is not related to the discrete EOT, which includes solvers such as the Sinkhorn algorithm (Cuturi, 2013) or gradient-based methods (Dvurechensky et al., 2018). These approaches are designed for a non-generative problem setting, see (Ksenofontov & Korotin, 2025, §2.3). They treat samples as empirical distributions $p_0(x_0) = \frac{1}{|I_0|}\sum_{i\in I_0}\delta_{x_0^{(i)}}$, $p_1(x_1) = \frac{1}{|I_1|}\sum_{j\in I_1}\delta_{x_1^{(j)}}$. The resulting coupling is then a bi-stochastic $|I_0| \times |I_1|$ matrix, which does not support out-of-sample generation. While some extensions attempt to provide inference for unseen data (Hütter & Rigollet, 2021; Pooladian & Niles-Weed, 2021; Manole et al., 2024; Deb et al., 2021), they are designed for continuous spaces ($\mathcal{X} = \mathbb{R}^D$) rather than the discrete spaces ($\mathcal{X} = \mathbb{S}^D$) considered in our work.

3 Benchmark

This section outlines our theoretical and practical foundations necessary for constructing the benchmark for the SB. We introduce our benchmark construction in §3.1. Our benchmark construction can benefit from a specific parameterization which we explore in §3.2. This construction and parameterization are later used to build our high-dimensional Gaussian mixture benchmark §3.3.

3.1 Main Theorem for Benchmark Construction

Given an initial distribution $p_0 \in \mathcal{P}(\mathcal{X})$, we aim to construct a target distribution $p_1 \in \mathcal{P}(\mathcal{X})$ such that the static SB $q^*(x_0, x_1)$ between them is known by our construction. The resulting pair (p_0, p_1) together with q^* can then be used as benchmark data for evaluating SB methods. Our following theorem plays the key role in the construction of benchmark pairs.

Theorem 3.1 (Benchmark pair construction for SB on discrete Spaces)**.** *Let $p_0 \in \mathcal{P}(\mathcal{X})$ be a given source distribution on a discrete space $\mathcal{X}$ and $v^* : \mathcal{X} \to \mathbb{R}$ be a given scalar-valued function. Let $q^* \in \mathcal{P}(\mathcal{X}^2)$ be a joint distribution for which for all $x_0 \in \mathcal{X}$ it holds that $q^*(x_0) = p_0(x_0)$ and*

$$q^*(x_1|x_0) \propto v^*(x_1)q^{\text{ref}}(x_1|x_0), \tag{8}$$

Let $p_1 \in \mathcal{P}(\mathcal{X})$ be the second marginal of q^*, i.e., $q^*(x_1) \overset{def}{=} p_1(x_1)$. Then $q^*(x_0, x_1)$ is the static SB (3) between p_0 and p_1. In turn, $q^*(x_0, x_{\text{in}}, x_1) \overset{def}{=} q^*(x_0, x_1)q^{\text{ref}}(x_{\text{in}}|x_0, x_1)$ is the dynamic SB (1).

Theorem 3.1 establishes that any pair (p_0, v^*) can be used to construct (p_0, p_1) for the SB problem, thereby yielding a known solution q^*. The construction considers conditional distributions $q^*(x_1|x_0)$ in an unnormalized form, so we further write

$$q^*(x_1|x_0) = \frac{1}{c^*(x_0)} v^*(x_1)q^{\text{ref}}(x_1|x_0), \tag{9}$$

where $c^*(x_0) \overset{\text{def}}{=} \sum_{x_1 \in \mathcal{X}} v^*(x_1)q^{\text{ref}}(x_1|x_0)$ is the normalization constant.

Our benchmark construction idea may be non-trivial to implement in practice. Specifically, working in the high-dimensional space $\mathcal{X} = \mathbb{S}^D$ makes computing the normalization constant and sampling from q^* computationally expensive. To address this, we introduce a parameterization that enables efficient computation and sampling, as detailed in the next section.

3.2 Practical Parameterization of the Scalar-valued Function v^*

We parameterize the scalar-valued function v^* using a rank-1 Canonical Polyadic (CP) decomposition, which captures interactions across dimensions and provides a compact yet expressive representation. Such decompositions act as universal approximators, capable of modeling complex functions when the rank is sufficiently large (Cohen et al., 2016; Basharin et al., 2025). Thus, v^* is given by

$$v^*(x_1) \;=\; \sum_{k=1}^{K} \beta_k \prod_{d=1}^{D} r_k^d[x_1^d]. \tag{10}$$

Expression (10) defines a mixture of K *factorizable distributions*, each with weight $\beta_k \geq 0$. For each mixture component k and dimension d, probabilities are given by non-negative vectors $r_k^d \in \mathbb{R}_+^S$, referred to as *CP cores*, where $r_k^d[x_1^d]$ denotes the probability of state x_1^d. The key advantage of this parameterization is that the factorization across dimensions makes both the normalizing constant $c(x_0)$ and the conditional distribution $q^*(x_1|x_0)$ computationally tractable. Specifically, the product structure allows efficient ancestral sampling by drawing each dimension independently.

Proposition 3.1 (Tractable Parameterization of Conditional Distributions)**.** *Given the CP decomposition of the scalar-valued function $v(x_1) = \sum_{k=1}^{K} \beta_k \prod_{d=1}^{D} r_k^d[x_1^d]$ and a factorizable reference process $q^{\text{ref}}(x_1|x_0) = \prod_{d=1}^{D} q^{\text{ref}}(x_1^d|x_0)$, the optimal conditional distribution satisfies:*

$$q^*(x_1|x_0) = \tfrac{1}{c(x_0)} \sum_{k=1}^{K} \beta_k \prod_{d=1}^{D} \Big[r_k^d[x_1^d] q^{\text{ref}}(x_1^d|x_0) \Big]; \tag{11}$$

$$c(x_0) = \sum_{k=1}^{K} \beta_k \prod_{d=1}^{D} \left(\sum_{x_1^d=0}^{S-1} r_k^d[x_1^d] q^{\text{ref}}(x_1^d|x_0) \right) \tag{12}$$

where $c(x_0)$ is the normalization constant. This formulation expresses $q^*(x_1|x_0)$ as a mixture of K factorizable distributions, each weighted by a scalar coefficient β_k.

Note that the considered reference processes (§2.2) q^{gauss} and q^{unif} are both factorizable by construction. Consequently, the normalization constant is tractable, as the combination of the factorized reference and the CP decomposition reduces the high-dimensional sum to a product of independent one-dimensional sums.

3.3 High-dimensional Gaussian Mixtures Benchmark Construction

We set p_0 as a noise distribution (uniform or discretized Gaussian) on $D \in \{2, 16, 64\}$ dimensions with $S = 50$ categories. For v^*, we use $K = 4$ components with uniformly initialized weights $\beta \in \mathbb{R}^K$, and the CP cores are initialized by setting their logarithms to the log-density of discretized Gaussians with varying means and fixed variance. Given p_0 and v^*, we then construct p_1 (Theorem 3.1). This initialization produces a target p_1 resembling a discretized Gaussian mixture with a clear visual structure. Moreover, our benchmark formulation further allows the generation of an unlimited number of samples for training.

We construct pairs under different reference processes q^{ref}: Gaussian q^{gauss} with $\gamma \in \{0.02, 0.05\}$ and uniform q^{unif} with $\gamma \in \{0.005, 0.01\}$, using $N + 1 = 128$ for both, see Figure 1b to visualize ground truth benchmark pairs.

4 Solvers for Evaluation

The field of discrete SB solvers remains in early development, with limited methods available for evaluation. We assess four approaches: the *Categorical Schrödinger Bridge Matching (CSBM)* method (Ksenofontov & Korotin, 2025), designed specifically for categorical distributions; our *α-CSBM* extension, which applies the online methodology of (De Bortoli et al., 2024) to CSBM; new *Discrete Light Schrödinger Bridge (DLightSB)* solver, constructed using our benchmark framework (§3) and adapting ideas from (Korotin et al., 2024) to discrete settings; and finally new *DLightSB-M*, which extends DLightSB to dynamic setups following (Gushchin et al., 2024a). Further details about methods can be found in Appendix B.

4.1 Categorical Schrödinger Bridge Matching (CSBM)

In (Ksenofontov & Korotin, 2025, Theorem 3.1), the discrete space SB problem is addressed by extending the discrete-time existence theorem of (Gushchin et al., 2024b, Theorem 3.6) to the discrete space/time setting, thereby establishing convergence of the *discrete time Iterative Markovian Fitting (D-IMF) procedure*. This constructive method uses the fact that the dynamic SB q^* is both reciprocal ($q^* \in \mathcal{R}^{\text{ref}}(\mathcal{X}^{N+2})$) and Markov ($q^* \in \mathcal{M}(\mathcal{X}^{N+2})$). The D-IMF algorithm alternates between projections onto these two sets, starting from an initial process $q^0(x_0, x_1)q^{\text{ref}}(x_{\text{in}}|x_0, x_1)$, where $q^0(x_0, x_1) \in \Pi(p_0, p_1)$, e.g., $p_0(x_0)p_1(x_1)$, and converges to the SB q^* in KL. Namely,

$$\begin{tikzcd} q^{2l} \arrow[r, shift left, "\text{proj}_{\mathcal{M}}"] & q^{2l+2} \arrow[l, shift left, "\text{proj}_{\mathcal{R}^{\text{ref}}}"] & l = 0, 1, \ldots \end{tikzcd}$$

where

$$[\text{proj}_{\mathcal{R}^{\text{ref}}}(q)](x_0, x_{\text{in}}, x_1) = \underset{r \in \mathcal{R}^{\text{ref}}(\mathcal{X}^{N+2})}{\arg\min}\, \text{KL}\left(q(x_0, x_{\text{in}}, x_1) \| r(x_0, x_{\text{in}}, x_1)\right), \quad \forall q \in \mathcal{P}(\mathcal{X}^{N+2}), \tag{13}$$

$$[\text{proj}_{\mathcal{M}}(q)](x_0, x_{\text{in}}, x_1) = \underset{m \in \mathcal{M}(\mathcal{X}^{N+2})}{\arg\min}\, \text{KL}\left(q(x_0, x_{\text{in}}, x_1) \| m(x_0, x_{\text{in}}, x_1)\right), \quad \forall q \in \mathcal{R}^{\text{ref}}(\mathcal{X}^{N+2}). \tag{14}$$

Loss. Because ancestral sampling makes the reciprocal part straightforward, the challenge lies in the Markov projection, for which the authors propose minimizing an alternative objective function.

$$\begin{aligned} \text{KL}\left(q(x_0, x_{\text{in}}, x_1) \| m(x_0, x_{\text{in}}, x_1)\right) = \mathbb{E}_{q(x_0,x_1)} \Bigg[\sum_{n=1}^{N} \mathbb{E}_{q^{\text{ref}}(x_{t_{n-1}}|x_0,x_1)} \\ \text{KL}\big(q^{\text{ref}}(x_{t_n}|x_{t_{n-1}}, x_1) \,\|\, m(x_{t_n}|x_{t_{n-1}})\big) - \mathbb{E}_{q^{\text{ref}}(x_{t_N}|x_0,x_1)}[\log m(x_1|x_{t_N})] \Bigg]. \end{aligned} \tag{15}$$

In practice, the D-IMF procedure is implemented in a bidirectional manner (see Ksenofontov & Korotin (2025, §3.2.5)). That is, it first applies the Markovian projection using both forward and backward representations. Notably, the KL loss can be replaced by any divergence from the Bregman family, introducing additional hyperparameters for this and several subsequent methods. For details on this equivalence, see (Ksenofontov & Korotin, 2025, Appendix C.1).

Remark. A continuous-time IMF was introduced in the Discrete Diffusion Schrödinger Bridge Matching (Kim et al., 2024, DDSBM) paper, which performs the Markovian projection (14) by matching the generator matrices of continuous-time Markov chains. As it reduces to the same loss and inference process due to the neccesity to discretize time, we report results only for CSBM.

4.2 α-Categorical Schrödinger Bridge Matching (α-CSBM)

Recently, an online alternative to the IMF procedure, called α-IMF, was proposed (De Bortoli et al., 2024; Peluchetti, 2024). In this approach, the exact projections in (13) and (14) are replaced by partial updates (De Bortoli et al., 2024, Eq. 9), and the resulting iteration is proven to converge to the SB. This means that instead of running each projection until full convergence, only a single optimization step is performed at each iteration, still guiding the distribution toward the double projection $\text{proj}_{\mathcal{R}^{\text{ref}}}(\text{proj}_{\mathcal{M}}(\cdot))$. Since those works address the continuous setting, we extend the same ideas to CSBM §4.1, interpreting the discrete formulation of α-IMF as a heuristic analogue of the original procedure.

Loss. Since the approach does not require each projection to reach full convergence, a single optimization step can be performed for both representation directions at once. This allows us to extend the CSBM bidirectional setup (§4.1) by updating the forward and backward models jointly, with a shared loss computed for both representations as:

$$L(\overrightarrow{m}, \overleftarrow{m}) = \frac{1}{2}\Big(\text{KL}\left(\overrightarrow{r_{\text{sg}}}(x_0, x_{\text{in}}, x_1) \| \overleftarrow{m}(x_0, x_{\text{in}}, x_1)\right) \\ + \text{KL}\left(\overleftarrow{r_{\text{sg}}}(x_0, x_{\text{in}}, x_1) \| \overrightarrow{m}(x_0, x_{\text{in}}, x_1)\right)\Big), \quad (16)$$

where $\rightarrow$ and $\leftarrow$ denote the direction of representations (forward and backward, respectively), and r_{sg} denotes $\text{proj}_{\mathcal{R}^{\text{ref}}}(m)$ evaluated with the stop-gradient operation.

4.3 Discrete Light Schrödinger Bridge (DLightSB)

Below we introduce DLightSB, a solver for discrete spaces derived from our benchmark construction in §3.2

Loss. Following (Korotin et al., 2024), we derive a discrete surrogate objective $\text{KL}\left(q^* \| q_\theta\right)$.

Proposition 4.1 (Feasible Discrete Reformulation of the KL Minimization.). *For the characterization (9) of $q(x_1|x_0)$, it holds that the alternative KL objective KL $(q^*\|q)$ admits the representation KL $(q^*\|q_\theta) = \mathcal{L}(\theta) - \mathcal{L}^*$ where*

$$\mathcal{L}(\theta) = \sum_{x_0 \in \mathcal{X}} \log c_\theta(x_0) p_0(x_0) - \sum_{x_1 \in \mathcal{X}} \log v_\theta(x_1) p_1(x_1), \quad (17)$$

and $\mathcal{L}^* \in \mathbb{R}$ is a constant value not depending on θ, therefore, it can be omitted.

4.4 Discrete Light Schrödinger Bridge Matching (DLightSB-M)

Inspired by (Gushchin et al., 2024a), we propose a matching approach for solving the SB problem in discrete settings. This approach enables obtaining the SB in a single projection, which is referred to as the *optimal projection*. Specifically, its idea lies in restating the Markovian projection (14) as the projection of a reciprocal process $r \in \mathcal{R}^{\text{ref}}(\mathcal{X}^{N+2})$ onto the set of all SBs:

$$\mathcal{S}(\mathcal{X}^{N+2}) \overset{\text{def}}{=} \Big\{ q^{\text{SB}} \in \mathcal{P}(\mathcal{X}^{N+2}) \text{ such that } \exists\, q_0^{\text{SB}}, q_1^{\text{SB}} \in \mathcal{P}(\mathcal{X}) \\ q^{\text{SB}} = \underset{q \in \Pi_N(q_0^{\text{SB}}, q_1^{\text{SB}})}{\arg\min} \text{KL}\left(q \| q^{\text{ref}}\right) \Big\}, \quad (18)$$

We show that (Gushchin et al., 2024a, Theorem 3.1) can be generalized to an arbitrary reference process q^{ref}, thereby enabling the application of the optimal projection in discrete space settings under our CP parametrization (10).

Proposition 4.2 (Optimal Projection with an Arbitrary Reference Process). *Let $r \in \mathcal{R}^{\text{ref}}(\mathcal{X}^{N+2})$ be a reciprocal process defined with a reference process $q^{\text{ref}} \in \mathcal{M}(\mathcal{X}^{N+2})$ and a joint distribution $r(x_0, x_1) \in \Pi(p_0, p_1)$. Then, the optimal projection of r onto the set of all SBs $\mathcal{S}(\mathcal{X}^{N+2})$ is the SB q^* between the desired marginals p_0 and p_1, i.e.,*

$$q^* = \underset{q^{\text{SB}} \in \mathcal{S}(\mathcal{X}^{N+2})}{\arg\min} KL\left(r \| q^{\text{SB}}\right). \quad (19)$$

The main requirement is to define q^{SB} such that the minimization is restricted to $q^{\text{SB}} \in \mathcal{S}(\mathcal{X}^{N+2})$. The following proposition establishes this characterization of SB transitions and, through its CP cores r_k^d, directly connects this approach to DLightSB (§4.3).

Proposition 4.3 (The SB's Transition Distributions with CP Decomposition)**.** *Let q^{ref} be a reference Markov process on a discrete space $\mathcal{X}$ with transition matrix Q^{ref}. Using the CP decomposition of the scalar-valued function v^* (10), the marginal transition distributions of the SB are given by*

$$q^{\text{SB}}(x^d_{t_n}|x_{t_{n-1}}) = q^{\text{ref}}(x^d_{t_n}|x_{t_{n-1}}) \sum_{k=1}^{K} \beta_k u^d_{k,t_n}\left[x^d_{t_n}\right] \prod_{\substack{j=1\\ j\neq d}}^{D} u^j_{k,t_{n-1}}\left[x^j_{t_{n-1}}\right], \tag{20}$$

where $u^d_{k,t_n}\left[x^d_{t_n}\right] = \sum_{x^d_1} [\overline{Q}^{\text{ref}}_{N+1-n}]_{x^d_{t_n}, x^d_1} r^d_k\left[x^d_1\right]$. Sampling is done via ancestral sampling.

Loss. The loss (15) could be applied directly to train the SB transitions q^{SB}.

5 Evaluation

We first present our evaluation metrics (§5.1), given the analogous problem structure, we adopt metrics from tabular data analysis (Zhang et al., 2024). Then we use them to assess the experimental setups from §3.3, and report the results in §5.2. It is important to highlight that DLightSB and DLightSB-M methods have some inductive bias as they use a similar construction as the benchmark (e.g., CP parameterization and factorizable reference process).

5.1 Metrics for Evaluation

Evaluating generative models on discrete data is challenging since common metrics (e.g., generative perplexity for text, FID for images (Heusel et al., 2017)) are domain-specific. Following work on tabular data evaluation (Zhang et al., 2024; Shi et al., 2025), we adopt the **Shape Score** and **Trend Score** metrics. Which are used to measure the quality of the resulting SB for each method.

Shape Score. This metric measures how well the predicted data preserves the marginal (per-dimension) distributions of the real data. We consider a dataset with $|I_R|$ real samples x and corresponding predicted samples $\tilde{x}$. We compute a per-dimension score for the empirical distributions (expressed in δ-delta notation) and report the average across all dimensions:

$$\text{SSM}_d = 1 - \frac{1}{2}\sum_{s=0}^{S-1}\left|\frac{1}{|I_R|}\sum_{i=1}^{|I_R|}\delta(s - x_d^{(i)}) - \frac{1}{|I_R|}\sum_{j=1}^{|I_R|}\delta(s - \tilde{x}_d^{(j)})\right|, \quad \text{SSM} = \frac{1}{D}\sum_{d=1}^{D}\text{SSM}_d.$$

Trend Score. This metric evaluates whether pairwise dimension dependencies in the real data are preserved in the predictions. For a dataset with $|I_R|$ real samples $x^{(k)}$ and corresponding predicted samples $\tilde{x}^{(k)}$. We compute a trend score and report the average across all dimension pairs:

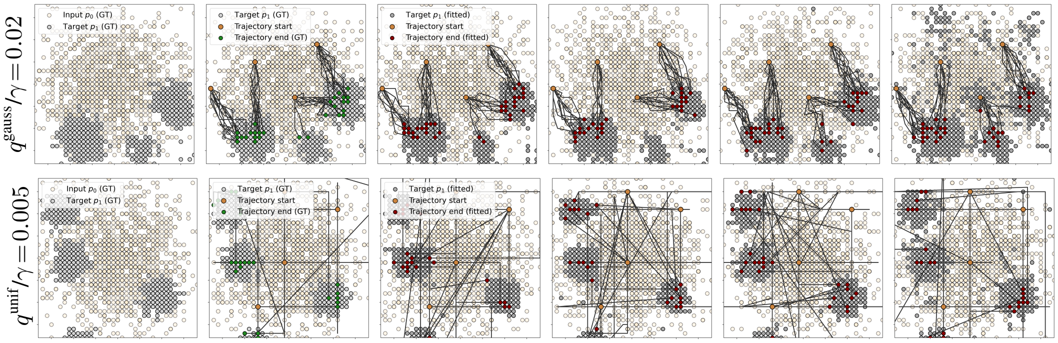
$$\text{TSM}_{d_i,d_j} = 1 - \frac{1}{2}\sum_{s_i=0}^{S-1}\sum_{s_j=0}^{S-1}\left|\frac{1}{|I_R|}\sum_{k=1}^{|I_R|}\delta(s_i - x^{(k)}_{d_i})\delta(s_j - x^{(k)}_{d_j}) - \frac{1}{|I_R|}\sum_{k=1}^{|I_R|}\delta(s_i - \tilde{x}^{(k)}_{d_i})\delta(s_j - \tilde{x}^{(k)}_{d_j})\right|,$$

where $x^{(k)}_{d_i}$ represents the d_i-th dimension of the k-th sample in this case.

Conditional Metrics. In our evaluation, we primarily report conditional variants of the aforementioned metrics. These are computed by generating multiple samples of x_1 for each $x_0 \sim p_0$. This approach provides a direct measure of the fidelity of the learned conditional distribution $q(x_1|x_0)$ and quantifies how well the SB solver captures the underlying stochastic transport.

5.2 Results

We use our benchmark pair constructor differently for training and testing. For *training*, we randomly sample $x_0^{\text{train}} \sim p_0$ and generate $x_1^{\text{train}} \sim p_1$ via our benchmark theorem, allowing infinite sample generation. Training is performed in an unpaired manner. For *testing*, we use a fixed set of 20 000 precomputed sample benchmark pairs (x_0, x_1), which we provide to facilitate benchmarking new discrete SB solvers. We also use different training setups, first by varying N across CSBM, α-CSBM, and DLightSB-M. For the same set of methods, we experiment with two loss functions: KL and MSE. We compare all methods to an *Independent* baseline. This approach assumes x_1 is independent of x_0, so we simply sample from the target distribution. In the main text, we report only the conditional metrics, as they more accurately reflect the performance of the SB solvers, in Appendix D.2 we provide experiments to validate conditional metrics against the unconditional ones. Further experimental details are provided in Appendix C.



(a) Input/Target (b) Benchmark (c) CSBM (d) α-CSBM (e) DLightSB (f) DLightSB-M

Figure 1: Samples from all methods on two high-dimensional Gaussian mixture benchmarks. **Top row**: q^{unif} ($\gamma = 0.005$). **Bottom row**: Gaussian benchmark ($\gamma = 0.02$). CSBM, α-CSBM, and DLightSB-M were trained with KL loss ($N+1 = 64$).

			$D=2$				$D=16$				$D=64$			
			gaussian		uniform		gaussian		uniform		gaussian		uniform	
Method	Loss	$N+1$	0.02	0.05	0.005	0.01	0.02	0.05	0.005	0.01	0.02	0.05	0.005	0.01
Independent	–	–	0.369	0.646	0.577	0.700	0.359	0.555	0.466	0.515	0.374	0.519	0.424	0.503
DLightSB	–	–	**0.979**	**0.976**	**0.974**	**0.983**	**0.972**	**0.980**	**0.970**	**0.981**	**0.966**	**0.980**	**0.980**	**0.973**
CSBM	KL	16	0.849	0.733	0.919	0.892	0.884	0.806	0.841	0.810	0.929	0.938	0.918	0.922
		64	$\underline{0.934}$	0.888	0.958	0.958	$\underline{0.944}$	0.933	0.933	0.927	$\underline{0.934}$	0.963	0.926	$\underline{0.949}$
	MSE	16	0.721	0.700	0.824	0.846	0.854	0.783	0.839	0.745	0.915	0.932	0.893	0.896
		64	0.444	0.841	0.818	0.780	0.885	0.902	0.890	0.894	0.854	0.942	0.867	0.928
α-CSBM	KL	16	0.829	0.738	0.927	0.918	0.881	0.836	0.873	0.825	0.930	$\underline{0.972}$	0.929	0.943
		64	0.902	0.896	0.952	0.958	0.936	$\underline{0.963}$	0.932	0.941	0.927	0.959	0.924	0.942
	MSE	16	0.803	0.695	0.841	0.890	0.865	0.820	0.861	0.815	0.908	0.943	0.884	0.910
		64	0.908	0.896	0.858	0.875	0.908	0.924	0.881	0.911	0.883	0.925	0.859	0.913
DLightSB-M	KL	16	0.926	$\underline{0.956}$	$\underline{0.969}$	$\underline{0.970}$	0.894	0.930	0.961	0.952	0.931	0.929	$\underline{0.954}$	0.905
		64	0.907	0.954	0.967	0.968	0.878	0.953	$\underline{0.962}$	$\underline{0.967}$	0.910	0.942	0.950	0.942
	MSE	16	0.782	0.951	0.881	0.926	0.726	0.921	0.942	0.951	0.718	0.918	0.891	0.850
		64	0.717	0.942	0.892	0.914	0.685	0.914	0.953	0.943	0.632	0.906	0.730	0.879

Table 1: Conditional Shape Score metric (↑) on the high-dimensional Gaussian mixture benchmark. The best-performing method is highlighted in bold, and the second is underlined. Color code threshold: red for < 0.7, yellow for $[0.7, 0.9)$, and green for ≥ 0.9.

High-Dimensional Gaussian Mixtures. In this section, we report results on the high-dimensional Gaussian mixture benchmark constructed as in §3.3 using the methods from §4. Visual results are shown in Figure 1 for q^{gauss} ($\gamma=0.02$) and q^{unif} ($\gamma=0.005$). See Appendix D.2 for additional plots. Tables 1 and 2 show that DLightSB consistently achieves the best performance on Conditional Shape Score and Trend Score metrics, respectively. We attribute this to the benchmark pairs being built on the same principle used by the DLightSB solver. DLightSB-M, which incorporates this inductive bias as well, achieves similar results with a slight drop in metrics, likely due to error accumulation in the iterative sampling. Interestingly, our results resemble those on continuous data (Korotin et al., 2024, Table 2; Gushchin et al., 2024a, Table 1), showing comparable performance with a slight drop for the DLightSB-M. Unconditional metrics are reported in Tables 4 and 5.

On the other hand, CSBM and α-CSBM perform noticeably worse than DLight methods. Notably, α-CSBM achieves similar quality to CSBM while halving computational cost, making it a more efficient alternative. Regarding N and the loss function, increasing N mostly improves metrics. For the loss function, KL consistently outperforms MSE, likely because MSE minimizes pointwise squared error and produces over-smoothed solutions that blur modes (see Figure 2).

6 Discussion

Our work fills a key gap in discrete SB research by introducing the first standardized benchmark for these methods. This contribution provides the community with ground truth data and standard evaluation metrics. The benchmark reveals fundamental limitations of current approaches: CP-based solvers (DLightSB, DLightSB-M) face severe memory constraints in high dimensions, while

			$D=2$				$D=16$				$D=64$			
			gaussian		uniform		gaussian		uniform		gaussian		uniform	
Method	Loss	$N+1$	0.02	0.05	0.005	0.01	0.02	0.05	0.005	0.01	0.02	0.05	0.005	0.01
Independent	–	–	0.315	0.611	0.491	0.609	0.202	0.480	0.334	0.404	0.172	0.362	0.248	0.329
DLightSB	–	–	**0.968**	**0.970**	**0.967**	**0.975**	**0.943**	**0.967**	**0.956**	**0.967**	**0.919**	**0.956**	**0.955**	**0.950**
CSBM	KL	16	0.793	0.654	0.884	0.856	0.803	0.694	0.732	0.676	0.853	0.895	0.830	0.861
		64	$\underline{0.911}$	0.854	0.932	0.923	$\underline{0.886}$	0.890	0.874	0.874	0.859	0.936	0.848	$\underline{0.901}$
	MSE	16	0.611	0.631	0.752	0.781	0.739	0.653	0.725	0.612	0.835	0.883	0.799	0.823
		64	0.331	0.775	0.735	0.729	0.808	0.831	0.812	0.821	0.767	0.891	0.777	0.863
α-CSBM	KL	16	0.773	0.651	0.898	0.876	0.810	0.744	0.783	0.724	0.854	$\underline{0.945}$	0.847	0.891
		64	0.874	0.855	0.921	0.913	0.878	$\underline{0.934}$	0.877	0.903	0.852	0.929	0.845	0.896
	MSE	16	0.728	0.603	0.756	0.829	0.771	0.716	0.769	0.710	0.818	0.883	0.781	0.821
		64	0.861	0.855	0.797	0.807	0.829	0.863	0.795	0.846	0.798	0.848	0.747	0.817
DLightSB-M	KL	16	0.878	$\underline{0.943}$	$\underline{0.952}$	$\underline{0.956}$	0.738	0.914	$\underline{0.932}$	0.930	$\underline{0.862}$	0.900	$\underline{0.920}$	0.674
		64	0.856	0.940	0.951	0.953	0.716	0.923	0.928	$\underline{0.936}$	0.833	0.901	0.648	0.820
	MSE	16	0.701	0.933	0.838	0.904	0.551	0.877	0.897	0.917	0.575	0.853	0.773	0.555
		64	0.640	0.922	0.852	0.889	0.503	0.856	0.903	0.910	0.464	0.818	0.498	0.700

Table 2: Conditional Trend Score (↑) on the high-dimensional Gaussian mixture benchmark. The best-performing method is highlighted in bold, and the second is underlined. Color code threshold: red for < 0.7, yellow for $[0.7, 0.9)$, and green for ≥ 0.9.

matching-based methods (CSBM, α-CSBM) struggle with parameter sensitivity and long training times. Our experiments show that DLightSB(-M) solvers may be viewed as oracle-like methods on this benchmark: their inductive bias makes them less informative as indicators of pure performance. See Appendix D.1 for an analysis of the reverse benchmark setting designed to probe this inductive bias. This behavior is expected, and it does not diminish the overall usefulness of the benchmark. The benchmark still faithfully captures the strengths and weaknesses of other unbiased methods. Moreover, the CP-parameterization limits DLightSB(-M) to simpler tasks, as complex settings require an impractical number of components.

Reproducibility. We provide the experimental details in Appendix C and the code to reproduce the conducted experiments in the supplementary materials (see readme.md).

LLM Usage. Large Language Models (LLMs) were used only to assist with rephrasing sentences and improving the clarity of the text. All scientific content, results, and interpretations in this paper were developed solely by the authors.

REFERENCES

Jacob Austin, Daniel D Johnson, Jonathan Ho, Daniel Tarlow, and Rianne Van Den Berg. Structured denoising diffusion models in discrete state-spaces. *Advances in Neural Information Processing Systems*, 34:17981–17993, 2021.

Artem Basharin, Andrei Chertkov, and Ivan Oseledets. Faster language models with better multi-token prediction using tensor decomposition, 2025. URL https://arxiv.org/abs/2410.17765.

Andrew Campbell, Joe Benton, Valentin De Bortoli, Thomas Rainforth, George Deligiannidis, and Arnaud Doucet. A continuous time framework for discrete denoising models. *Advances in Neural Information Processing Systems*, 35:28266–28279, 2022.

Andrew Campbell, Jason Yim, Regina Barzilay, Tom Rainforth, and Tommi Jaakkola. Generative flows on discrete state-spaces: Enabling multimodal flows with applications to protein co-design. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 5453–5512. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/campbell24a.html.

Tianrong Chen, Guan-Horng Liu, and Evangelos Theodorou. Likelihood training of Schrödinger bridge using forward-backward SDEs theory. In *International Conference on Learning Representations*, 2021.

Nadav Cohen, Or Sharir, and Amnon Shashua. On the expressive power of deep learning: A tensor analysis, 2016. URL https://arxiv.org/abs/1509.05009.

Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, 26, 2013.

Max Daniels, Tyler Maunu, and Paul Hand. Score-based generative neural networks for large-scale optimal transport. *Advances in neural information processing systems*, 34:12955–12965, 2021.

Valentin De Bortoli, Iryna Korshunova, Andriy Mnih, and Arnaud Doucet. Schrödinger bridge flow for unpaired data translation. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=1F32iCJFfa.

Nabarun Deb, Promit Ghosal, and Bodhisattva Sen. Rates of estimation of optimal transport maps using plug-in estimators via barycentric projections. *Advances in Neural Information Processing Systems*, 34:29736–29753, 2021.

Pavel Dvurechensky, Alexander Gasnikov, and Alexey Kroshnin. Computational optimal transport: Complexity by accelerated gradient descent is better than by Sinkhorn's algorithm. In *International conference on machine learning*, pp. 1367–1376. PMLR, 2018.

Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 12873–12883, 2021.

Itai Gat, Tal Remez, Neta Shaul, Felix Kreuk, Ricky TQ Chen, Gabriel Synnaeve, Yossi Adi, and Yaron Lipman. Discrete flow matching. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

Nikita Gushchin, Alexander Kolesov, Alexander Korotin, Dmitry Vetrov, and Evgeny Burnaev. Entropic neural optimal transport via diffusion processes. In *Advances in Neural Information Processing Systems*, 2023a.

Nikita Gushchin, Alexander Kolesov, Petr Mokrov, Polina Karpikova, Andrey Spiridonov, Evgeny Burnaev, and Alexander Korotin. Building the bridge of Schrödinger: A continuous entropic optimal transport benchmark. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023b.

Nikita Gushchin, Sergei Kholkin, Evgeny Burnaev, and Alexander Korotin. Light and optimal Schrödinger bridge matching. In *Forty-first International Conference on Machine Learning*, 2024a.

Nikita Gushchin, Daniil Selikhanovych, Sergei Kholkin, Evgeny Burnaev, and Alexander Korotin. Adversarial Schrödinger bridge matching. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024b. URL https://openreview.net/forum?id=L3Knnigicu.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In *Advances in neural information processing systems*, pp. 6626–6637, 2017.

Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. Argmax flows and multinomial diffusion: Learning categorical distributions. *Advances in Neural Information Processing Systems*, 34:12454–12465, 2021.

Jan-Christian Hütter and Philippe Rigollet. Minimax estimation of smooth optimal transport maps. 2021.

Sergei Kholkin, Grigoriy Ksenofontov, David Li, Nikita Kornilov, Nikita Gushchin, Evgeny Burnaev, and Alexander Korotin. Diffusion & adversarial Schrödinger bridges via iterative proportional Markovian fitting. *arXiv preprint arXiv:2410.02601*, 2024.

Jun Hyeong Kim, Seonghwan Kim, Seokhyun Moon, Hyeongwoo Kim, Jeheon Woo, and Woo Youn Kim. Discrete diffusion Schrödinger bridge matching for graph transformation. *arXiv preprint arXiv:2410.01500*, 2024.

Alexander Korotin, Nikita Gushchin, and Evgeny Burnaev. Light Schrödinger bridge. In *The Twelfth International Conference on Learning Representations*, 2024.

Grigoriy Ksenofontov and Alexander Korotin. Categorical Schrödinger bridge matching. In *Forty-second International Conference on Machine Learning*, 2025. URL `https://openreview.net/forum?id=RBly0nOr2h`.

Christian Léonard. A survey of the Schrödinger problem and some of its connections with optimal transport. *arXiv preprint arXiv:1308.0215*, 2013.

David Lopez-Paz and Maxime Oquab. Revisiting classifier two-sample tests. In *International Conference on Learning Representations*, 2017. URL `https://openreview.net/forum?id=SJkXfE5xx`.

Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion language modeling by estimating the ratios of the data distribution. 2023.

Xiaoshan Luo, Zhenyu Wang, Jian Lv, Lei Wang, Yanchao Wang, and Yanming Ma. CrystalFlow: A flow-based generative model for crystalline materials. *arXiv preprint arXiv:2412.11693*, 2024.

Tudor Manole, Sivaraman Balakrishnan, Jonathan Niles-Weed, and Larry Wasserman. Plugin estimation of smooth optimal transport maps. *The Annals of Statistics*, 52(3):966–998, 2024.

Petr Mokrov, Alexander Korotin, Alexander Kolesov, Nikita Gushchin, and Evgeny Burnaev. Energy-guided entropic neural optimal transport. In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=d6tUsZeVs7`.

Stefano Peluchetti. Bm^2: Coupled Schrödinger bridge matching. *arXiv preprint arXiv:2409.09376*, 2024.

Aram-Alexandre Pooladian and Jonathan Niles-Weed. Entropic estimation of optimal transport maps. *arXiv preprint arXiv:2109.12004*, 2021.

Yiming Qin, Manuel Madeira, Dorina Thanou, and Pascal Frossard. DeFoG: Discrete flow matching for graph generation. *arXiv preprint arXiv:2410.04263*, 2024.

Subham Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. *Advances in Neural Information Processing Systems*, 37:130136–130184, 2024.

Erwin Schrödinger. *Über die Umkehrung der Naturgesetze*. Verlag der Akademie der Wissenschaften in Kommission bei Walter De Gruyter u. Company, 1931.

Erwin Schrödinger. Sur la théorie relativiste de l'électron et l'interprétation de la mécanique quantique. In *Annales de l'institut Henri Poincaré*, volume 2, pp. 269–310, 1932.

Juntong Shi, Minkai Xu, Harper Hua, Hengrui Zhang, Stefano Ermon, and Jure Leskovec. Tabdiff: a mixed-type diffusion model for tabular data generation. In *The Thirteenth International Conference on Learning Representations*, 2025. URL `https://openreview.net/forum?id=swvURjrt8z`.

Yuyang Shi, Valentin De Bortoli, Andrew Campbell, and Arnaud Doucet. Diffusion Schrödinger bridge matching. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL `https://openreview.net/forum?id=qy07OHsJT5`.

Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

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

Clement Vignac, Igor Krawczuk, Antoine Siraudin, Bohan Wang, Volkan Cevher, and Pascal Frossard. Digress: Discrete denoising diffusion for graph generation. *arXiv preprint arXiv:2209.14734*, 2022.

Hengrui Zhang, Jiani Zhang, Balasubramaniam Srinivasan, Zhengyuan Shen, Xiao Qin, Christos Faloutsos, Huzefa Rangwala, and George Karypis. Mixed-type tabular data synthesis with score-based diffusion in latent space, 2024. URL `https://arxiv.org/abs/2310.09656`.

A Proofs

Proof of Theorem 3.1. We start from the expression of the static EOT minimization problem in (8)

$$\begin{aligned}\min_{q\in\Pi(p_0,p_1)} &\mathrm{KL}\left(q(x_0,x_1)\|q^{\mathrm{ref}}(x_0,x_1)\right) = \\ &= \min_{q\in\Pi(p_0,p_1)} -H(q) - \sum_{x_0,x_1} q(x_0,x_1)\log q^{\mathrm{ref}}(x_1|x_0) - \mathrm{const} \\ &= \min_{q\in\Pi(p_0,p_1)} \sum_{x_0,x_1} q(x_0,x_1)\log q(x_0,x_1) - \sum_{x_0,x_1} q(x_0,x_1)\log q^{\mathrm{ref}}(x_1|x_0) - \mathrm{const}\end{aligned} \tag{21}$$

Noting that the joint distribution factorizes as $q(x_0,x_1) = q(x_0)q(x_1|x_0) = p_0(x_0)q(x_1|x_0)$, and enforcing the marginal constraints $\sum_{x_0} p_0(x_0)q(x_1|x_0) = p_1(x_1)$ and $\sum_{x_1} q(x_1|x_0) = 1$ (equivalently $q(x_0) = p_0(x_0)$), the corresponding Lagrangian can be formulated as

$$\begin{aligned}\mathcal{L}(q) = &\sum_{x_0,x_1} p_0(x_0)q(x_1|x_0)\log\left(p_0(x_0)q(x_1|x_0)\right) - \sum_{x_0,x_1} p_0(x_0)q(x_1|x_0)\log q^{\mathrm{ref}}(x_1|x_0) + \\ &+ \sum_{x_1}\lambda(x_1)\left(\sum_{x_0} q(x_1|x_0)p_0(x_0) - p_1(x_1)\right) + \sum_{x_0}\tau(x_0)\left(\sum_{x_1} q(x_1|x_0) - p_0(x_0)\right) \\ &= \underbrace{\sum_{x_0,x_1} p_0(x_0)q(x_1|x_0)\log p_0(x_0))}_{=\sum_{x_0} p_0(x_0)\log p_0(x_0))} + \sum_{x_0,x_1} p_0(x_0)q(x_1|x_0)\log q(x_1|x_0) - \\ &- \sum_{x_0,x_1} p_0(x_0)q(x_1|x_0)\log q^{\mathrm{ref}}(x_1|x_0) + \sum_{x_1}\lambda(x_1)\left(\sum_{x_0} q(x_1|x_0)p_0(x_0) - p_1(x_1)\right) \\ &+ \sum_{x_0}\tau(x_1)\left(\sum_{x_1} q(x_1|x_0) - 1\right)\end{aligned} \tag{22}$$

where $\lambda(x_1)$ and $\tau(x_0)$ denote the Lagrange multipliers associated with the marginal constraints on x_1 and x_0, respectively. Taking the pointwise partial derivative of $\mathcal{L}(q)$ with respect to $q(x_1|x_0)$ then yields

$$\frac{\partial\mathcal{L}}{\partial q} = p_0(x_0)\left(\log q(x_1|x_0) + 1\right) - p_0(x_0)\log q^{\mathrm{ref}}(x_0,x_1) + \lambda(x_1)p_0(x_0) + \tau(x_1) = 0 \tag{23}$$

Therefore, the optimal process q^* can be written as

$$q^*(x_1|x_0) = \exp(-\lambda(x_1) - 1)q^{\mathrm{ref}}(x_1|x_0)p_0\exp\left(-\frac{\tau(x_0)}{p_0(x_0)}\right) \tag{24}$$

Setting $v^*(x_1) = \exp(-\lambda(x_1) - 1)$ concludes the proof. □

Proof of Proposition 3.1. Assuming the CP parameterization introduced in (10), and further assuming that the reference process factorizes across dimensions as $q^{\mathrm{ref}}(x_1|x_0) = \prod_{d=1}^{D} q^{\mathrm{ref}}(x_1^d|x_0)$, the normalized conditional distribution $q^*(x_1|x_0)$ in (9) can be rewritten as

$$\begin{aligned}q^*(x_1|x_0) &= \frac{1}{c(x_0)}\left(\sum_{k=1}^{K}\beta_k\prod_{d=1}^{D} r_k^d[x_1^d]\right)\prod_{d=1}^{D} q^{\mathrm{ref}}(x_1^d|x_0) \\ &= \frac{1}{c(x_0)}\sum_{k=1}^{K}\beta_k\prod_{d=1}^{D} r_k^d[x_1^d]\, q^{\mathrm{ref}}(x_1^d|x_0),\end{aligned} \tag{25}$$

where the reference factors can be merged with the rank-1 components because they are independent of the mixture index k and factorize over dimensions. From here, it is possible to obtain the normalizing constant $c(x_0)$ by summing over all possible values of $x_1 \in \mathcal{X} = \mathbb{S}^D$, where $x_1^d \in \{0, \dots, S-1\}$. The normalizing constant can then be rewritten as

$$
\begin{aligned}
c(x_0) &= \sum_{x_1 \in \mathbb{S}^D} \sum_{k=1}^{K} \beta_k \prod_{d=1}^{D} r_k^d[x_1^d]\, q^{\text{ref}}(x_1^d|x_0) \\
&= \sum_{k=1}^{K} \beta_k \sum_{x_1 \in \mathbb{S}^D} \prod_{d=1}^{D} r_k^d[x_1^d]\, q^{\text{ref}}(x_1^d|x_0) \\
&= \sum_{k=1}^{K} \beta_k \prod_{d=1}^{D} \sum_{x_1^d=0}^{S-1} r_k^d[x_1^d]\, q^{\text{ref}}(x_1^d|x_0),
\end{aligned}
\tag{26}
$$

where $\sum_{x_1 \in \mathbb{S}^D} = \sum_{x_1^1=0}^{S-1} \sum_{x_1^2=0}^{S-1} \cdots \sum_{x_1^D=0}^{S-1}$. The exchange between the product and the sum is valid here because the summation is separable across dimensions, i.e., each factor depends only on its corresponding coordinate x_1^d. □

Proof of Proposition 4.1. We start from the standard KL minimization problem from the LightSB paper (Korotin et al., 2024) and define it in discrete space.

$$
\begin{aligned}
\mathrm{KL}\left(q^* \| q\right) &= \sum_{x_0, x_1} q^*(x_0, x_1) \log\left(\frac{q^*(x_0, x_1)}{q(x_0, x_1)}\right) = \sum_{x_0, x_1} q^* \log q^*(x_0, x_1) - \sum_{x_0, x_1} q^* \log q(x_0, x_1) = \\
&= -H(q^*) - \sum_{x_0, x_1} q^*(x_0, x_1) \log q(x_0, x_1) = -H(q^*) - \sum_{x_0, x_1} q^*(x_0, x_1) \log\left(q(x_0) q(x_1|x_0)\right) \\
&= -H(q^*) - \sum_{x_0, x_1} q^*(x_0, x_1) \log \underbrace{q(x_0)}_{=p_0(x_0)} - \sum_{x_0, x_1} q^*(x_0, x_1) \log q(x_1|x_0) = \\
&= -H(q^*) - \sum_{x_0} \log p_0(x_0) \underbrace{\sum_{x_1} q^*(x_0, x_1)}_{=q^*(x_0) = p_0(x_0)} - \sum_{x_0, x_1} q^*(x_0, x_1) \log q(x_1|x_0)
\end{aligned}
$$

Now using (9) on $q(x_1|x_0)$ we can get

$$
\begin{aligned}
\mathrm{KL}\left(q^* \| q\right) &= -H(q^*) - \sum_{x_0} \log p_0(x_0) p_0(x_0) - \sum_{x_0, x_1} q^*(x_0, x_1) \log\left(\frac{v^*(x_1)}{c^*(x_0)} q^{\text{ref}}(x_1|x_0)\right) = \\
&= \underbrace{-H(q^*) - \sum_{x_0} \log p_0(x_0) p_0(x_0) - \sum_{x_0, x_1} q^*(x_0, x_1) \log q^{\text{ref}}(x_1|x_0) -}_{=-\mathcal{L}^*} \\
&- \sum_{x_0, x_1} q^*(x_0, x_1) \log\left(\frac{v^*(x_1)}{c^*(x_0)}\right) = \\
&= -\mathcal{L}^* + \sum_{x_0, x_1} q^*(x_0, x_1) \log c^*(x_0) - \sum_{x_0, x_1} q^*(x_0, x_1) \log v^*(x_1) = \\
&= \sum_{x_0} p_0^*(x_0) \log c^*(x_0) - \sum_{x_1} q^*(x_1) \log v^*(x_1) - \mathcal{L}^*,
\end{aligned}
$$

That concludes the proof. □

Proof of expression 7. Let Q be the transition matrix in (4), rewritten as

$$
Q = (1-\gamma) I + \frac{\gamma}{S-1} (\mathbf{1}\mathbf{1}^\top - I)
$$

$$= \left(1 - \gamma \frac{S}{S-1}\right) I + \frac{\gamma}{S-1} \mathbf{1}\mathbf{1}^\top,$$

where I is the identity matrix and $\mathbf{1}\mathbf{1}^\top$ is the all-ones matrix. Let

$$a = 1 - \gamma \frac{S}{S-1}, \quad b = \frac{\gamma}{S-1},$$

so that $Q = aI + b\mathbf{1}\mathbf{1}^\top$ and note that $a + Sb = 1$. We compute Q^{N+1} using the binomial expansion. Since I and $\mathbf{1}\mathbf{1}^\top$ commute:

$$\begin{aligned} Q^n &= (aI + b\mathbf{1}\mathbf{1}^\top)^n \\ &= \sum_{k=0}^{n} \binom{n}{k} a^{n-k} b^k (\mathbf{1}\mathbf{1}^\top)^k. \end{aligned}$$

Using $(\mathbf{1}\mathbf{1}^\top)^k = S^{k-1}\mathbf{1}\mathbf{1}^\top$ for $k \geq 1$ and separating the $k = 0$ term:

$$\begin{aligned} Q^n &= a^n I + \sum_{k=1}^{n} \binom{n}{k} a^{n-k} b^k S^{k-1} \mathbf{1}\mathbf{1}^\top \\ &= a^n I + \frac{1}{S} \left(\sum_{k=1}^{n} \binom{n}{k} a^{n-k} (bS)^k \right) \mathbf{1}\mathbf{1}^\top. \end{aligned}$$

The binomial expansion gives:

$$(a + bS)^n = \sum_{k=0}^{n} \binom{n}{k} a^{n-k} (bS)^k = a^n + \sum_{k=1}^{n} \binom{n}{k} a^{n-k} (bS)^k.$$

Since $a + bS = 1$, we have $(a + bS)^n = 1$, so $\sum_{k=1}^{n} \binom{n}{k} a^{n-k} (bS)^k = 1 - a^n$. Thus,

$$Q^n = a^n I + \frac{1 - a^n}{S} \mathbf{1}\mathbf{1}^\top.$$

Substituting $n = N + 1$ and $a = 1 - \gamma \frac{S}{S-1}$ yields

$$q^{\text{ref}}(x_1|x_0) = Q^{N+1} = \left(1 - \gamma \frac{S}{S-1}\right)^{N+1} I + \frac{1 - \left(1 - \gamma \frac{S}{S-1}\right)^{N+1}}{S} \mathbf{1}\mathbf{1}^\top.$$

This completes the proof. □

Proof of Proposition 4.2.

$$\begin{aligned} &\text{KL}\left(q(x_0, x_{\text{in}}, x_1) \| q^{\text{SB}}(x_0, x_{\text{in}}, x_1)\right) = \\ &= \text{KL}\left(q(x_0, x_1) \| q^{\text{SB}}(x_0, x_1)\right) + \underbrace{\text{KL}\left(q^{\text{ref}}(x_{\text{in}}|x_0, x_1) \| q^{\text{ref}}(x_{\text{in}}|x_0, x_1)\right)}_{=0} = \quad (27) \\ &= \underbrace{\sum_{x_0, x_1} q(x_0, x_1) \log q(x_0, x_1)}_{= -H(q(x_0, x_1))} - \sum_{x_0, x_1} q(x_0, x_1) \log q^{\text{SB}}(x_0, x_1) = \\ &= -H(q(x_0, x_1)) - \sum_{x_0, x_1} q(x_0, x_1) \log \frac{v^{\text{SB}}(x_1) q^{\text{ref}}(x_1|x_0)}{c^{\text{SB}}(x_0)} = \quad (28) \\ &= -H(q(x_0, x_1)) - \sum_{x_0, x_1} q(x_0, x_1) \log v^{\text{SB}}(x_1) - \\ &- \sum_{x_0, x_1} q(x_0, x_1) \log q^{\text{ref}}(x_1|x_0) + \sum_{x_0, x_1} q(x_0, x_1) \log c^{\text{SB}}(x_0) = \end{aligned}$$

$$= -H(q(x_0, x_1)) - \sum_{x_1} \log v^{\text{SB}}(x_1) \underbrace{q(x_1)}_{=p(x_1)=q^*(x_1)} \underbrace{\sum_{x_0} q(x_0|x_1)}_{=1=\sum_{x_0} q^*(x_0|x_1)} - \tag{29}$$

$$- \sum_{x_0, x_1} q(x_0, x_1) \log q^{\text{ref}}(x_1|x_0) + \sum_{x_0} \log c^{\text{SB}}(x_0) \underbrace{q(x_0)}_{=p(x_0)=q^*(x_0)} \underbrace{\sum_{x_1} q(x_1|x_0)}_{=1=\sum_{x_1} q^*(x_1|x_0)} = \tag{30}$$

$$= \underbrace{-H(q(x_0, x_1)) - \sum_{x_0, x_1} q(x_0, x_1) \log q^{\text{ref}}(x_1|x_0)}_{=C_1} - \sum_{x_0, x_1} q^*(x_0, x_1) \log \frac{v^{\text{SB}}(x_1)}{c^{\text{SB}}(x_0)} =$$

$$= C_1 - \sum_{x_0, x_1} q^*(x_0, x_1) \log \frac{v^{\text{SB}}(x_1)}{c^{\text{SB}}(x_0)} -$$

$$\underbrace{- \sum_{x_0, x_1} q^*(x_0, x_1) \log q^{\text{ref}}(x_1|x_0) + \sum_{x_0, x_1} q^*(x_0, x_1) \log q^{\text{ref}}(x_1|x_0)}_{=0} = \tag{31}$$

$$= - \sum_{x_0, x_1} q^*(x_0, x_1) \log \frac{v^{\text{SB}}(x_1) q^{\text{ref}}(x_1|x_0)}{c^{\text{SB}}(x_0)} + \underbrace{C_1 + \sum_{x_0, x_1} q^*(x_0, x_1) \log q^{\text{ref}}(x_1|x_0)}_{=C_2} =$$

$$= C_2 - \sum_{x_0, x_1} q^*(x_0, x_1) \log q^{\text{SB}}(x_0, x_1) +$$

$$\underbrace{+ \sum_{x_0, x_1} q^*(x_0, x_1) \log q^*(x_0, x_1) - \sum_{x_0, x_1} q^*(x_0, x_1) \log q^*(x_0, x_1)}_{=0} = \tag{32}$$

$$= \sum_{x_0, x_1} q^*(x_0, x_1) \log \frac{q^*(x_0, x_1)}{q^{\text{SB}}(x_0, x_1)} + \underbrace{C_2 - \sum_{x_0, x_1} q^*(x_0, x_1) \log q^*(x_0, x_1)}_{C_3} =$$

$$= \text{KL}\left(q^*(x_0, x_1) \| q^{\text{SB}}(x_0, x_1)\right) + C_3$$

In (27), we use the disintegration of the KL divergence to transition from the dynamic to the static formulation. In (28), we apply our parameterization from (9). Next, in (29) and (30), we use the properties of the reciprocal process q, which has the true marginals at $t = 0$ and $t = 1$. In (31), we add a zero term to introduce $q^{\text{ref}}(x_1|x_0)$ with the expectation taken over the optimal coupling $q^*(x_0, x_1)$. Finally, in (32), we obtain the entropy term, completing the expression for the desired KL divergence. □

Proof of Proposition 4.3. We first derive the transitional distributions of the SB by recalling its well-known characterization (Léonard, 2013, Prop. 4.2):

$$q^{\text{SB}}\left(x_{t_n}|x_{t_{n-1}}\right) = q^{\text{ref}}\left(x_{t_n}|x_{t_{n-1}}\right) \frac{\phi_{t_n}^{\text{SB}}(x_{t_n})}{\phi_{t_{n-1}}^{\text{SB}}(x_{t_{n-1}})}, \qquad \phi_{t_n}^{\text{SB}}(x_{t_n}) = \mathbb{E}_{q^{\text{ref}}(x_1|x_{t_n})}\left[v^{\text{SB}}(x_1)\right].$$

Using the CP parametrization of v^{SB} from (10) and exploiting the conditional independence of dimensions under q^{ref}, the scalar-valued functions ϕ_{t_n} can be written as:

$$\phi_{t_n}^{\text{SB}}(x_{t_n}) = \sum_{k=1}^{K} \beta_k \prod_{d=1}^{D} \mathbb{E}_{q^{\text{ref}}(x_1^d|x_{t_n}^d)}[r_k^d(x_1^d)] = \sum_{k=1}^{K} \beta_k \prod_{d=1}^{D} \underbrace{\sum_{x_1^d=0}^{S-1} [\overline{Q}_{N+1-n}^{\text{ref}}]_{x_{t_n}^d, x_1^d} r_k^d[x_1^d]}_{u_{k,t_n}^d[x_{t_n}^d]},$$

where u^d_{k,t_n} satisfy the following recursive relation:

$$u^d_{k,t_n}[x^d_{t_n}] = \sum_{x^d_{t_{n+1}}=0}^{S-1} [Q^{\text{ref}}]_{x^d_{t_n}, x^d_{t_{n+1}}} u^d_{k,t_{n+1}}[x^d_{t_{n+1}}], \qquad u^d_{k,t_1} = r^d_k.$$

Thus, we obtain the following transition distributions:

$$q^{\text{SB}}(x_{t_n}|x_{t_{n-1}}) \propto q^{\text{ref}}(x_{t_n}|x_{t_{n-1}}) \sum_{k=1}^{K} \beta_k \prod_{j=1}^{D} u^j_{k,t_n}[x^j_{t_n}]. \tag{33}$$

To obtain the d-th marginal transition distribution, we marginalize over $x^{-d}_{t_n} \overset{\text{def}}{=} \{x^j_{t_n}\}_{j \neq d}$ as follows:

$$q^{\text{SB}}\left(x^d_{t_n}|x_{t_{n-1}}\right) \propto \sum_{x^{-d}_{t_n}} \left(\prod_{j=1}^{D} [Q^{\text{ref}}]_{x^j_{t_{n-1}}, x^j_{t_n}} \right) \left(\sum_{k=1}^{K} \beta_k \prod_{j=1}^{D} u^j_{k,t_n}[x^j_{t_n}] \right) =$$

$$= [Q^{\text{ref}}]_{x^d_{t_{n-1}}, x^d_{t_n}} \sum_{k=1}^{K} \beta_k u^d_{k,t_n}[x^d_{t_n}] \prod_{\substack{j=1 \\ j \neq d}}^{D} \underbrace{\sum_{x^j_{t_n}} [Q^{\text{ref}}]_{x^j_{t_{n-1}}, x^j_{t_n}} u^j_{k,t_n}[x^j_{t_n}]}_{u^j_{k,t_{n-1}}[x^j_{t_{n-1}}] \text{ (by recursion)}}.$$

Finally, we obtain the desired expression up to normalization:

$$q^{\text{SB}}\left(x^d_{t_n}|x_{t_{n-1}}\right) \propto [Q^{\text{ref}}]_{x^d_{t_{n-1}}, x^d_{t_n}} \sum_{k=1}^{K} \beta_k u^d_{k,t_n}[x^d_{t_n}] \prod_{\substack{j=1 \\ j \neq d}}^{D} u^j_{k,t_{n-1}}[x^j_{t_{n-1}}].$$

Now we derive the sampling procedure. Sampling from the SB transitional distributions is based on the following factorization:

$$q^{\text{SB}}(x_{t_n}|x_{t_{n-1}}) = \sum_{k=1}^{K} p(k|x_{t_{n-1}}) \prod_{d=1}^{D} q^{\text{SB}}(x^d_{t_n}|x_{t_{n-1}}, k).$$

Using the full joint SB transition distribution (33), the probability of k is

$$p(k|x_{t_{n-1}}) \propto \beta_k \prod_{d=1}^{D} u^d_{k,t_{n-1}}[x^d_{t_{n-1}}].$$

Using the marginal distributions conditioned on k, the factors with $j \neq d$ are independent of $x^d_{t_n}$ and absorbed into normalization, yielding

$$q^{\text{SB}}(x^d_{t_n}|x_{t_{n-1}}, k) \propto [Q^{\text{ref}}]_{x^d_{t_{n-1}}, x^d_{t_n}} u^d_{k,t_n}[x^d_{t_n}].$$

Thus, sampling proceeds by first drawing

$$k^* \sim p(k|x_{t_n}),$$

and then sampling each coordinate independently as

$$x^d_{t_{n+1}} \sim q^{\text{SB}}(\cdot|x_{t_n}, k^*) \propto [Q^{\text{ref}}]_{x^d_{t_n}, \cdot} \, u^d_{k^*,t_{n+1}}[\cdot], \qquad d = 1, \ldots, D.$$

□

B Methods Details

This section provides additional theoretical and implementation details complementing §4, focusing on the methods used to evaluate our benchmark pairs.

CSBM. In practice, the D-IMF procedure is usually implemented bidirectionally: the Markovian projection is applied using both forward and backward representations (see "Notation"), which is the approach we adopt in our experiments. This design has two advantages. First, it mitigates error accumulation caused by imperfect model fitting, as shown in (De Bortoli et al., 2024, Appendix F). Second, it enables the use of alternative starting couplings, as proposed in (Kholkin et al., 2024).

Limitations. To reduce the computational overhead of evaluating the full probability state space of size S^D, the authors propose factorizing transition probabilities across dimensions, reducing the space to $D \times S$. However, this parametrization constitutes a key limitation of CSBM, as it introduces approximation error.

DLightSB. We optimize over the logarithm of the mixture weights $\beta \in \mathbb{K}$ and the logarithm of the CP cores r_k^d. This allows computation of the `log` terms in the surrogate loss (17) by stable `log-sum-exp` operations.

Limitations. (1) In spite of being numerically stable, the `log-sum-exp` operations allocate extra memory, which can become a bottleneck when applied repeatedly in high-dimensional settings. (2) The CP parameterization requires an impractically large number of components to capture complex data, making it infeasible under memory constraints. Additionally, we approximate the summations using Monte Carlo samples from p_0 and p_1.

DLightSB-M. Limitations. The practical implementation requires storing or recomputing $u_{k,t}^d$ at each iteration, which scales as $\mathcal{O}(B \times S^2 \times K)$ in memory and computation. This quickly becomes prohibitive for high-dimensional data, limiting scalability to small state spaces.

C Experiment Details

This section provides detailed descriptions of all methods and their configurations.

Shared Aspects. Across all experiments, we use the AdamW optimizer with fixed `beta` values of 0.95 and 0.99. For the high-dimensional Gaussian benchmark (§5.2). Notably, for diffusion-based methods, we fully sample the Markov chain, in contrast to Austin et al. (2021), which applies an `argmax` operation at the final timestep. To evaluate the methods on the high-dimensional Gaussian mixture benchmark (§5.2), we use 20 000 samples. Conditional metrics are computed using 156 instances of x_0, with 1 000 samples of x_1 generated for each x_0.

CSBM and α-CSBM. For CSBM and α-CSBM, we use the official implementation from Ksenofontov & Korotin (2025):

`https://github.com/gregkseno/csbm`.

To stabilize training and improve final performance, we apply Exponential Moving Average (EMA) parameter updates with a decay rate of 0.999, tuned consistently across all experiments. Unlike Austin et al. (2021), we omit the L_{simple} loss during training. We employ a simple MLP with three hidden layers of size $[128, 128, 128]$ and ReLU activations. Time conditioning is implemented via an embedding layer of the same size as dimensions, D. Both methods are trained for 5 D-IMF iterations, using 120 000 gradient updates in the first iteration and 40 000 in each subsequent iteration. For α-CSBM, we use a learning rate of 10^{-3} and halve the batch size for training a single model, following De Bortoli et al. (2024). For CSBM, we use a learning rate of 10^{-4}.

DLightSB and DLightSB-M. For all benchmark experiments, both methods use $K = 1000$ components initialized from data samples and are trained for 100 000 gradient updates. The learning rate is set to 10^{-2} for both, with DLightSB-M using independent coupling ($q^0(x_0, x_1) = p_0(x_0)p_1(x_1)$).

Computational Resources and Training Time. All high-dimensional Gaussian mixture benchmark experiments were conducted on 1 A100 GPU unless otherwise specified, with training times reported inclusive of evaluation. For $D = 2$, training is relatively short: CSBM and α-CSBM each complete within about 5 hours, DLightSB-M within 4 hours, and DLightSB in roughly 20 minutes.

For $D = 64$, CSBM completes in under 14 hours, α-CSBM in under 9 hours, DLightSB-M in just under 2 days (on 2 A100 GPUs), and DLightSB in under 7 hours.

D Additional Experiments

D.1 Reverse benchmark

In this section we try to overcome inherited inductive bias of DLightSB(-M) solvers. By construction, the forward conditional distribution $q^*(x_1|x_0)$ admits a CP decomposition, while the reverse distribution $q^*(x_0|x_1)$ does not. As a result, when the benchmark is used in the reverse direction with the same marginals p_0 and p_1, DLightSB(-M) methods can no longer rely on the inductive bias that benefits them in the forward setup.

Unfortunately, in this setup, the true conditional distributions are not available, so we cannot compute conditional metrics. To overcome this restriction, we decided to compute the Classifier Two Sample Test (Lopez-Paz & Oquab, 2017, C2ST) metric, ROC AUC of classifier between pairs $(x_0, x_1) \sim p_1(x_1)q^*(x_0|x_1)$ and $(\hat{x}_0, x_1) \sim p_1(x_1)q_\theta(x_0|x_1)$. As the classifier, we used two layer MLP with ReLU activations that takes as input the concatenation of one-hot vectors of x_0 and x_1. We present C2ST scores in following table.

			$D=2$				$D=16$				$D=64$			
			gaussian		uniform		gaussian		uniform		gaussian		uniform	
Method	Loss	$N+1$	0.02	0.05	0.005	0.01	0.02	0.05	0.005	0.01	0.02	0.05	0.005	0.01
DLightSB	–	–	0.926	0.998	0.996	0.985	0.961	0.971	0.993	0.996	0.972	0.993	0.985	0.990
CSBM	KL	16	0.990	0.991	1.000	0.996	0.979	0.990	0.999	0.988	0.990	0.990	0.991	0.997
		64	0.995	1.000	0.992	0.998	0.991	0.982	0.986	0.981	0.999	0.999	0.994	0.999
	MSE	16	0.952	0.996	0.987	0.997	0.998	0.976	0.995	0.985	0.987	0.997	0.983	0.999
		64	0.900	0.990	0.993	0.981	0.985	0.992	0.998	0.973	0.987	0.997	1.000	0.999

Table 3: C2ST metric (↑) on the high-dimensional Gaussian mixture benchmark. Color code threshold: red for < 0.7, yellow for $[0.7, 0.9)$, and green for ≥ 0.9.

As can be seen from Table 3, computed metric values are not informative. Across all methods the metric values are nearly identical, indicating that such a simple classifier is already capable of distinguishing generated samples from real ones. As a result, we decided to discard this setup.

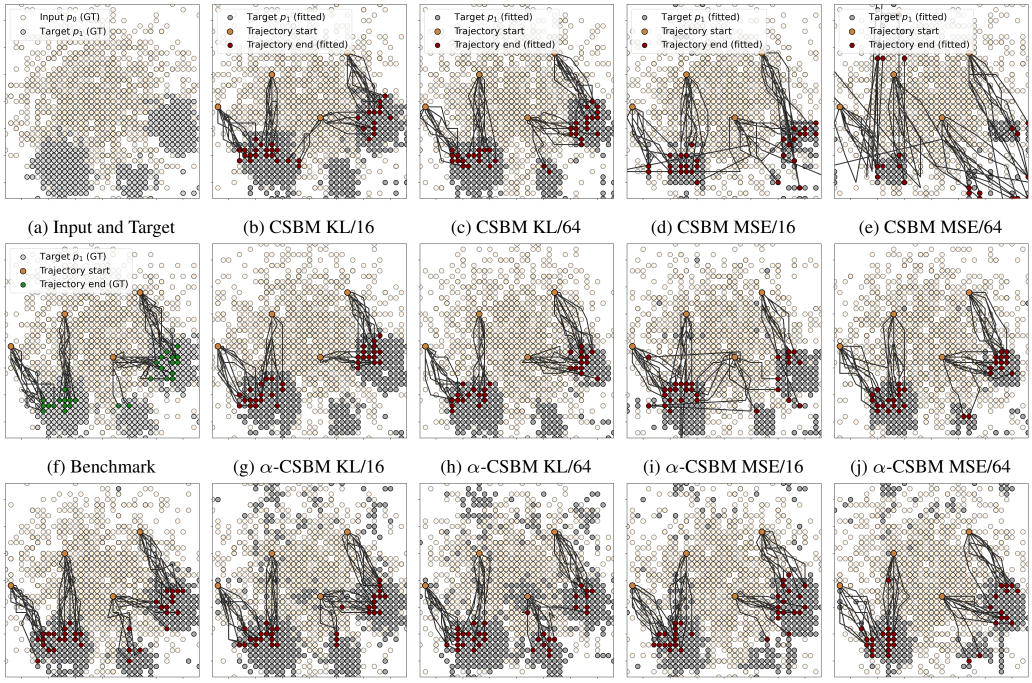
D.2 Additional Metrics and Plots

			$D=2$				$D=16$				$D=64$			
			gaussian		uniform		gaussian		uniform		gaussian		uniform	
Method	Loss	$N+1$	0.02	0.05	0.005	0.01	0.02	0.05	0.005	0.01	0.02	0.05	0.005	0.01
Independent	–	–	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
DLightSB	–	–	0.975	0.969	0.969	0.980	0.973	0.976	0.968	0.975	0.971	0.971	0.974	0.970
CSBM	KL	16	0.855	0.739	0.914	0.893	0.890	0.807	0.855	0.806	0.953	0.934	0.951	0.937
		64	0.936	0.893	0.955	0.952	0.959	0.934	0.962	0.940	0.966	0.967	0.963	0.969
	MSE	16	0.726	0.704	0.814	0.850	0.852	0.782	0.845	0.754	0.935	0.936	0.913	0.903
		64	0.449	0.843	0.783	0.774	0.879	0.903	0.918	0.915	0.860	0.944	0.881	0.949
α-CSBM	KL	16	0.829	0.749	0.925	0.914	0.887	0.836	0.888	0.827	0.965	0.968	0.959	0.965
		64	0.902	0.900	0.965	0.961	0.963	0.955	0.954	0.963	0.964	0.960	0.953	0.961
	MSE	16	0.810	0.712	0.841	0.887	0.877	0.821	0.854	0.819	0.951	0.947	0.912	0.930
		64	0.909	0.903	0.867	0.883	0.934	0.914	0.883	0.929	0.895	0.925	0.878	0.930
DLightSB-M	KL	16	0.924	0.952	0.961	0.960	0.919	0.931	0.957	0.948	0.935	0.921	0.947	0.905
		64	0.909	0.951	0.964	0.964	0.905	0.949	0.960	0.962	0.922	0.937	0.962	0.941
	MSE	16	0.787	0.944	0.870	0.920	0.743	0.921	0.944	0.950	0.723	0.914	0.890	0.850
		64	0.712	0.942	0.886	0.908	0.686	0.915	0.950	0.937	0.639	0.903	0.731	0.879

Table 4: Shape Score metric (↑) on the high-dimensional Gaussian mixture benchmark. The best-performing method is highlighted in bold, and the second is underlined. Color code threshold: red for < 0.7, yellow for $[0.7, 0.9)$, and green for ≥ 0.9.

			$D\!=\!2$				$D\!=\!16$				$D\!=\!64$			
			gaussian		uniform		gaussian		uniform		gaussian		uniform	
Method	Loss	$N\!+\!1$	0.02	0.05	0.005	0.01	0.02	0.05	0.005	0.01	0.02	0.05	0.005	0.01
Independent	–	–	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
DLightSB	–	–	0.949	0.952	0.950	0.960	0.914	0.943	0.933	0.939	0.888	0.909	0.907	0.906
CSBM	KL	16	0.797	0.662	0.876	0.854	0.809	0.691	0.747	0.670	0.877	0.868	0.881	0.867
		64	0.907	0.856	0.926	0.914	0.899	0.884	0.909	0.883	0.888	0.905	0.897	0.904
	MSE	16	0.620	0.633	0.739	0.781	0.743	0.651	0.735	0.618	0.861	0.864	0.840	0.824
		64	0.337	0.774	0.705	0.724	0.809	0.826	0.845	0.838	0.776	0.865	0.807	0.867
α-CSBM	KL	16	0.772	0.662	0.887	0.868	0.821	0.740	0.798	0.721	0.888	0.905	0.895	0.899
		64	0.872	0.853	0.929	0.914	0.907	0.916	0.905	0.918	0.886	0.899	0.891	0.900
	MSE	16	0.733	0.621	0.759	0.822	0.790	0.714	0.771	0.715	0.864	0.857	0.824	0.829
		64	0.860	0.854	0.803	0.811	0.855	0.846	0.802	0.855	0.816	0.825	0.777	0.820
DLightSB-M	KL	16	0.874	0.933	0.934	0.935	0.762	0.902	0.909	0.908	0.842	0.864	0.874	0.665
		64	0.857	0.928	0.935	0.937	0.747	0.906	0.909	0.911	0.828	0.865	0.650	0.792
	MSE	16	0.703	0.920	0.821	0.892	0.572	0.865	0.888	0.902	0.577	0.822	0.760	0.548
		64	0.629	0.915	0.838	0.879	0.511	0.845	0.889	0.890	0.470	0.791	0.497	0.685

Table 5: Trend Score (↑) on the high-dimensional Gaussian mixture benchmark. The best-performing method is highlighted in bold, and the second is underlined. Color code threshold: red for < 0.7, yellow for $[0.7, 0.9)$, and green for ≥ 0.9.

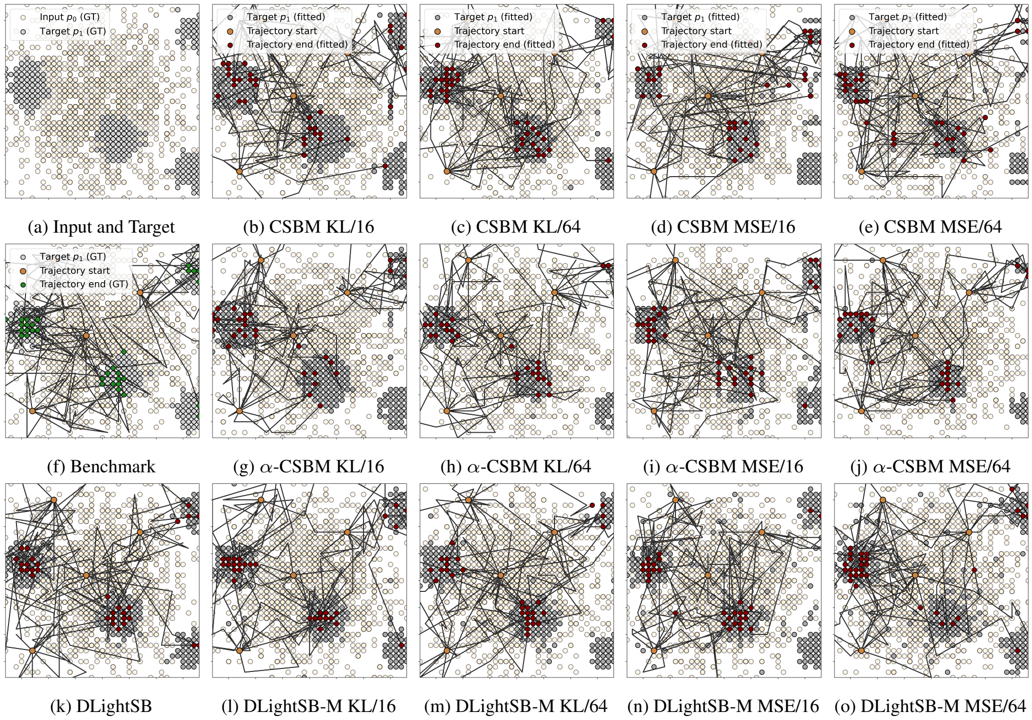


(a) Input and Target (b) CSBM KL/16 (c) CSBM KL/64 (d) CSBM MSE/16 (e) CSBM MSE/64

(f) Benchmark (g) α-CSBM KL/16 (h) α-CSBM KL/64 (i) α-CSBM MSE/16 (j) α-CSBM MSE/64

(k) DLightSB (l) DLightSB-M KL/16 (m) DLightSB-M KL/64 (n) DLightSB-M MSE/16 (o) DLightSB-M MSE/64

Figure 2: Samples from all methods on the high-dimensional Gaussian mixture benchmark using the Gaussian reference process q^{gauss} with $\gamma = 0.02$.

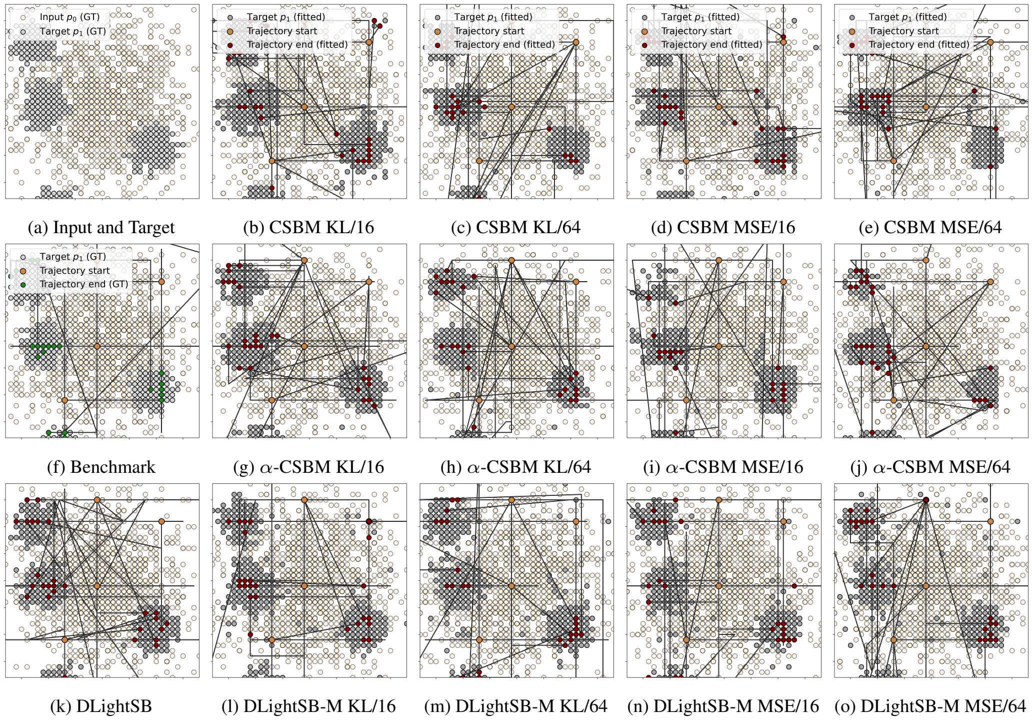


(a) Input and Target (b) CSBM KL/16 (c) CSBM KL/64 (d) CSBM MSE/16 (e) CSBM MSE/64

(f) Benchmark (g) α-CSBM KL/16 (h) α-CSBM KL/64 (i) α-CSBM MSE/16 (j) α-CSBM MSE/64

(k) DLightSB (l) DLightSB-M KL/16 (m) DLightSB-M KL/64 (n) DLightSB-M MSE/16 (o) DLightSB-M MSE/64

Figure 3: Samples from all methods on the high-dimensional Gaussian mixture benchmark using the Gaussian reference process q^{gauss} with $\gamma = 0.05$.

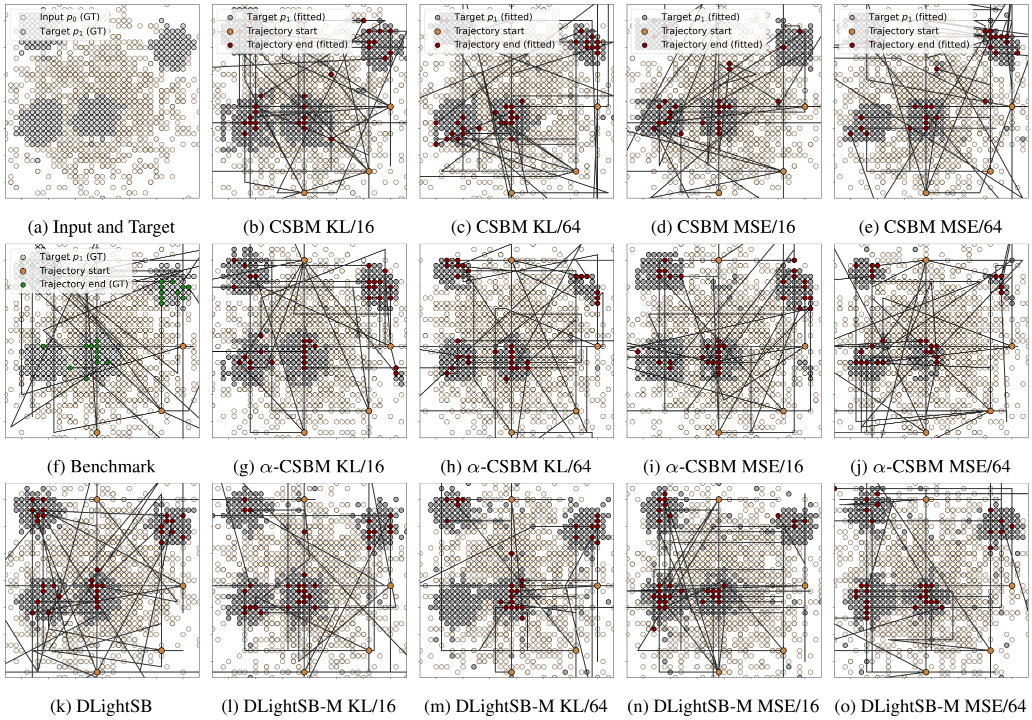


(a) Input and Target (b) CSBM KL/16 (c) CSBM KL/64 (d) CSBM MSE/16 (e) CSBM MSE/64

(f) Benchmark (g) α-CSBM KL/16 (h) α-CSBM KL/64 (i) α-CSBM MSE/16 (j) α-CSBM MSE/64

(k) DLightSB (l) DLightSB-M KL/16 (m) DLightSB-M KL/64 (n) DLightSB-M MSE/16 (o) DLightSB-M MSE/64

Figure 4: Samples from all methods on the high-dimensional Gaussian mixture benchmark using the uniform reference process q^{unif} with $\gamma = 0.005$.



(a) Input and Target (b) CSBM KL/16 (c) CSBM KL/64 (d) CSBM MSE/16 (e) CSBM MSE/64

(f) Benchmark (g) α-CSBM KL/16 (h) α-CSBM KL/64 (i) α-CSBM MSE/16 (j) α-CSBM MSE/64

(k) DLightSB (l) DLightSB-M KL/16 (m) DLightSB-M KL/64 (n) DLightSB-M MSE/16 (o) DLightSB-M MSE/64

Figure 5: Samples from all methods on the high-dimensional Gaussian mixture benchmark using the uniform reference process q^{unif} with $\gamma = 0.01$.