Light and Optimal Schrödinger Bridge Matching

Nikita Gushchin [*1] **Sergei Kholkin** [*1] **Evgeny Burnaev** [1 2] **Alexander Korotin** [1 2]

Abstract

Schrödinger Bridges (SB) have recently gained the attention of the ML community as a promising extension of classic diffusion models which is also interconnected to the Entropic Optimal Transport (EOT). Recent solvers for SB exploit the pervasive bridge matching procedures. Such procedures aim to recover a stochastic process transporting the mass between distributions given only a transport plan between them. In particular, given the EOT plan, these procedures can be adapted to solve SB. This fact is heavily exploited by recent works giving rise to matching-based SB solvers. The cornerstone here is recovering the EOT plan: recent works either use heuristical approximations (e.g., the minibatch OT) or establish iterative matching procedures which by the design accumulate the error during the training. We address these limitations and propose a novel procedure to learn SB which we call the **optimal Schrödinger bridge matching**. It exploits the optimal parameterization of the diffusion process and provably recovers the SB process **(a)** with a single bridge matching step and **(b)** with arbitrary transport plan as the input. Furthermore, we show that the optimal bridge matching objective coincides with the recently discovered energy-based modeling (EBM) objectives to learn EOT/SB. Inspired by this observation, we develop a light solver (which we call LightSB-M) to implement optimal matching in practice using the Gaussian mixture parameterization of the adjusted Schrödinger potential. We experimentally showcase the performance of our solver in a range of practical tasks. The code for our solver can be found at `https://github.com/SKholkin/LightSB-Matching`.

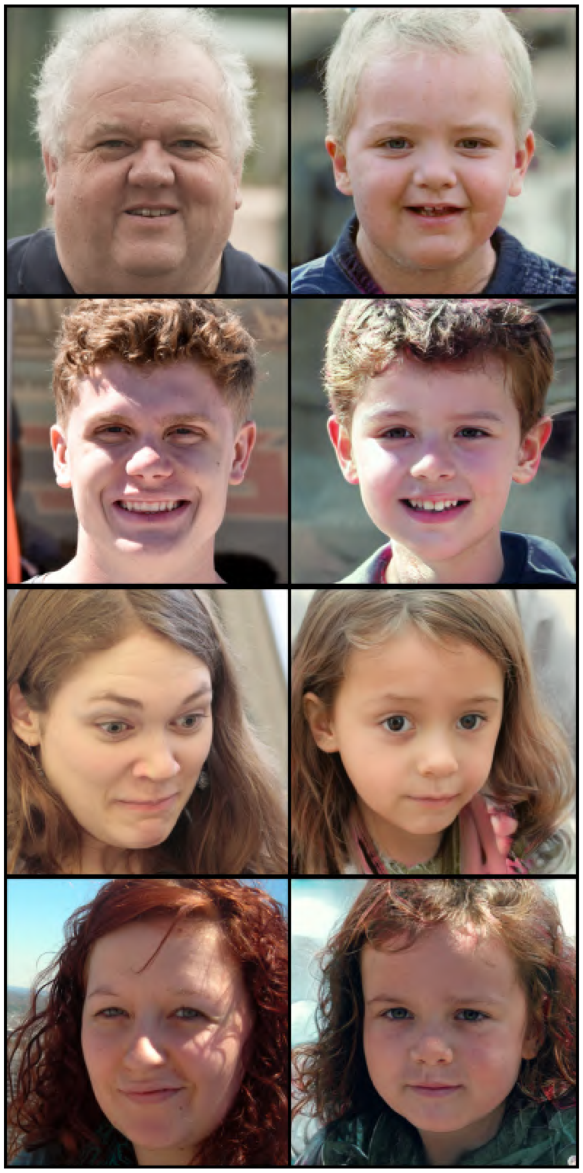


Figure 1: Unpaired *adult* → *child* translation with our LightSB-M solver applied in the latent space of ALAE (Pidhorskyi et al., 2020) for 1024x1024 FFHQ images (Karras et al., 2019). *Our LightSB-M solver converges on 4 cpu cores in several minutes.*

[*]Equal contribution [1]Skolkovo Institute of Science and Technology [2]Artificial Intelligence Research Institute. Correspondence to: Nikita Gushchin <n.gushchin@skoltech.ru>, Sergei Kholkin <s.kholkin@skoltech.ru>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

1. Introduction

Diffusion models are a powerful type of generative models that show an impressive quality of image generation (Ho et al., 2020; Rombach et al., 2022). However, they still have several directions for improvement on which the research community is actively working. Some of these directions are: speeding up the generation (Wang et al., 2022; Song et al., 2023), application to the image-to-image transfer (Liu et al., 2023a) extension to unpaired image transfer (Meng et al., 2021) or domain adaptation (Vargas et al., 2021), including biological tasks with single cell data.

A promising approach to advance these directions is the development of new theoretical frameworks for learning flows and diffusions. Recently proposed novel techniques such as flow (Lipman et al., 2022) and bridge (Shi et al., 2023) matching for flow and diffusion-based models show promising potential for further extending and improving generative and translation models. Furthermore, by exploiting theoretical links between flow and diffusion models with Optimal Transport (Villani, 2008, OT) and Schrödinger Bridge (Léonard, 2013, SB) problems, several new methods have been proposed to speed up the inference (Liu et al., 2023b), to improve the quality of image generation (Liu et al., 2023a), and to solve unpaired image and domain translations (De Bortoli et al., 2021; Shi et al., 2023).

Recent approaches (Tong et al., 2023; Shi et al., 2023; Liu et al., 2022) to OT and SB problems based on flow and bridge matching either use iterative bridge matching-based procedures or employ heuristic approximations (e.g., the minibatch OT) to recover the SB through its relation to the Entropic OT problem. Unfortunately, iterative methods imply solving a sequence of time-consuming optimization problems and experience error accumulation. In turn, minibatch OT approximations can lead to biased solutions.

Contributions. We show that the above-mentioned issues can be eliminated. We do this by proposing a novel bridge matching-based approach to solve the SB in one iteration.

1. We propose a new bridge matching-based approach to solve the SB problem. Our approach exploits the novel "optimal" projection for stochastic processes that projects directly onto the set of SBs (§3.1).
2. Based on the new theoretical results, we develop a new fast solver for the SB problem. We use the "light" parameterization for SBs (Korotin et al., 2024) and our new theory on "optimal" projections to solve the SB problem in one bridge matching iteration (§3.2).
3. We perform extensive comparisons of this new solver on many setups where SB solvers are widely used, including the SB benchmark (§5.2), single-cell data (§5.3) and unpaired image translation (§5.4).

Notations. The notations of our paper mostly follow those used by the LightSB's authors in their work (Korotin et al., 2024). We work in $\mathbb{R}^D$, which is the D-dimensional Euclidean space equipped with the Euclidean norm $\|\cdot\|$. We use $\mathcal{P}(\mathbb{R}^D)$ to denote the absolutely continuous Borel probability distributions whose variance and differential entropy are finite. To denote the density of $p \in \mathcal{P}(\mathbb{R}^D)$ at a point $x \in \mathbb{R}^D$, we use $p(x)$. We use $\mathcal{N}(x|\mu, \Sigma)$ to denote the density at a point $x \in \mathbb{R}^D$ of the normal distribution with mean $\mu \in \mathbb{R}^D$ and covariance $0 \prec \Sigma \in \mathbb{R}^{D\times D}$. We write $\text{KL}(\cdot\|\cdot)$ to denote the Kullback-Leibler divergence between two distributions. In turn, $H(\cdot)$ denotes the differential entropy of a distribution. We use Ω to denote the space of trajectories, i.e., continuous $\mathbb{R}^D$-valued functions of $t \in [0, 1]$. We write $\mathcal{P}(\Omega)$ to denote the probability distributions on the trajectories Ω whose marginals at $t = 0$ and $t = 1$ belong to $\mathcal{P}(\mathbb{R}^D)$; this is the set of stochastic processes. We use dW_t to denote the differential of the standard Wiener process $W \in \mathcal{P}(\Omega)$. For a process $T \in \mathcal{P}(\Omega)$, we denote its joint distribution at $t = 0, 1$ by $\pi^T \in \mathcal{P}(\mathbb{R}^D \times \mathbb{R}^D)$. In turn, we use $T_{|x_0,x_1}$ to denote the distribution of T for $t \in (0, 1)$ conditioned on T's values x_0, x_1 at $t = 0, 1$.

2. Preliminaries

We start with recalling the main concepts of the Schrödinger Bridge problem (§2.1). Next, we discuss the SB solvers which are the most relevant to our study (§2.2, 2.3).

2.1. Background on Schrödinger Bridges

To begin with, we recall the SB problem with the Wiener prior and its equivalent Entropic Optimal Transport problem with the quadratic cost. We start from the latter as it is easier to introduce and interpret. For a detailed discussion of both these problems, we refer to (Léonard, 2013; Chen et al., 2016). Next, we describe the computational setup for learning SBs, which we consider in the paper.

Entropic Optimal Transport (EOT) with the quadratic cost. Consider distributions $p_0 \in \mathcal{P}(\mathbb{R}^D)$, $p_1 \in \mathcal{P}(\mathbb{R}^D)$. For $\epsilon > 0$, the EOT problem with the quadratic cost is to find the minimizer of

$$\min_{\pi\in\Pi(p_0,p_1)} \int\limits_{\mathbb{R}^D}\int\limits_{\mathbb{R}^D} \frac{\|x_0 - x_1\|^2}{2}\pi(x_0, x_1)dx_0dx_1 - \epsilon H(\pi), \quad (1)$$

where $\Pi(p_0, p_1)$ is the set of the transport plans, i.e., probability distributions on $\mathbb{R}^D \times \mathbb{R}^D$ whose marginals are p_0 and p_1, respectively. The minimizer π^* of (1) exists, is unique, and is absolutely continuous; it is called the *EOT plan*.

Schrödinger Bridge with the Wiener Prior. Consider the Wiener process $W^\epsilon \in \mathcal{P}(\Omega)$ with volatility $\epsilon > 0$ which starts at p_0 at $t = 0$. Its differential satisfies the stochastic differential equation (SDE): $dW^\epsilon_t = \sqrt{\epsilon}dW_t$. The SB problem with the Wiener prior W^ϵ between p_0, p_1 is

$$\min_{T\in\mathcal{F}(p_0,p_1)} \text{KL}\left(T\|W^\epsilon\right), \tag{2}$$

where $\mathcal{F}(p_0,p_1) \subset \mathcal{P}(\Omega)$ is the subset of stochastic processes which start at distribution p_0 (at $t=0$) and end at p_1 (at $t=1$). There exists a unique minimizer T^*. Furthermore, it is a diffusion process described by the SDE: $dx_t = g^*(x_t,t)dt + dW_t^\epsilon$ (Léonard, 2013, Prop. 2.3). The optimal process T^* is called the Schrödinger Bridge and $g^*: \mathbb{R}^D \times [0,1] \to \mathbb{R}^D$ is the optimal drift.

Relation of EOT and SB. EOT (1) and SB (2) are closely related to each other. It holds that the joint marginal distribution π^{T^*} of T^* at times $0,1$ coincides with the EOT plan π^* solving (1), i.e., $\pi^{T^*} = \pi^*$. Hence, the solution π^* of the EOT problem (1) can be recovered from T^*. Thus, SB can be viewed as a dynamic extension of EOT: a user is interested not only in the optimal mass transport plan π^*, but in the entire time-dependent mass transport process T^*.

Given just the optimal plan π^*, one may also complete it to get the full process T^*. It suffices to consider a process whose join marginal distribution at $t=0,1$ is π^* and the trajectory distribution $T^*_{|x_0,x_1}$ at $t\in(0,1)$ conditioned on the ends (x_0,x_1) coincides with the Wiener Prior's, i.e., $T^*_{|x_0,x_1} = W^\epsilon_{|x_0,x_1}$. The latter is known as the Brownian Bridge (Pinsky & Karlin, 2011, Sec. 8.3.3). Thus, we obtain $T^* = \int_{\mathbb{R}^D\times\mathbb{R}^D} W^\epsilon_{|x_0,x_1} d\pi^*(x_0,x_1)$. This strategy does not directly give the optimal drift g^*, but it can recovered by other means, e.g., with the bridge matching (§2.3).

Characterization for EOT and SB solutions. It is known that the EOT plan π^* can be represented through the input density p_0 and a function $v^*: \mathbb{R}^D \to \mathbb{R}_+$:

$$\pi^*(x_0,x_1) = \underbrace{p_0(x_0)}_{=\pi^*(x_0)} \cdot \underbrace{\exp\left(\langle x_0,x_1\rangle/\epsilon\right) v^*(x_1)/c_{v^*}(x_0)}_{=\pi^*(x_1|x_0)}, \tag{3}$$

where $c_{v^*}(x_0) \overset{\text{def}}{=} \int_{\mathbb{R}^D} \exp\left(\langle x_0,x_1\rangle/\epsilon\right) v^*(x_1)dy$. Following the notation of (Korotin et al., 2024), we call v^* the adjusted Schrödinger potential. The optimal drift of T^* can also be expressed using v^*. Namely,

$$g^*(x_t,t) = \epsilon\nabla_{x_t} \log\Big(\int_{\mathbb{R}^D} \mathcal{N}(x'|x_t,(1-t)\epsilon I_D) \exp\big(\frac{\|x'\|^2}{2\epsilon}\big) v^*(x')dx'\Big), \tag{4}$$

see (Korotin et al., 2024, §2, 3) for a deeper discussion. Note that v^* is defined up to the multiplicative constant.

Computational SB/EOT setup. In practice, distributions p_0 and p_1 are usually not available explicitly but only through their empirical samples $\{x_0^1,\dots,x_0^N\} \sim p_0$ and $\{x_1^1,\dots,x_1^M\} \sim p_1$. The typical task is to obtain a good approximation $\widehat{g} \approx g^*$ of the drift of SB process T^* or explicitly/implicitly approximate the EOT plan's conditional distributions $\widehat{\pi}(\cdot|x_0) \approx \pi^*(\cdot|x_0)$ for *all* $x_0 \in \mathbb{R}^D$. This is needed to do the *out-of-sample estimation*, i.e., for new (test) points $x_0^{new} \sim p_0$ sample $x_1 \sim \pi^*(\cdot|x_0^{new})$ or simulate T^*'s trajectories staring at a point x_0^{new} at time $t=0$. This setup widely appears in generative modeling (De Bortoli et al., 2021; Gushchin et al., 2023a) and analysis of biological single cell data (Vargas et al., 2021; Koshizuka & Sato, 2022; Tong et al., 2023).

The setup above is usually called the *continuous* EOT or SB and should not be confused with the *discrete* setup, which is widely studied in the discrete OT literature (Peyré et al., 2019; Cuturi, 2013). There one is mostly interested in computing the EOT plan directly between the empirical samples (probably weighted), i.e., match them with each other. There is usually no need in the out-of-sample estimation.

2.2. Energy-based EOT/SB Solvers

Given a good approximation of the optimal potential v^*, one may approximate the conditional EOT plans and the optimal drift via (3) and (4), respectively (using v^*'s approximation). Inspired by the idea above, papers (Korotin et al., 2024; Mokrov et al., 2024) provide related approaches to learn this potential. They show that v^* can be learned via solving

$$\mathcal{L}_0(v) \overset{\text{def}}{=} \min_v \Big\{\int_{\mathbb{R}^D} \log c_v(x_0)p_0(x_0)dx_0 - \int_{\mathbb{R}^D} \log v(x_1)p_1(x_1)dy\Big\}, \tag{5}$$

where $c_v(x) \overset{\text{def}}{=} \int_{\mathbb{R}^D} \exp\left(\langle x,y\rangle/\epsilon\right) v(y)dy$. This objective magically turns to be equal up to an additive v-independent constant to $\text{KL}\left(\pi^*\|\pi_v\right) = \text{KL}\left(T^*\|S_v\right)$, where

$$\pi_v(x_0,x_1) \overset{\text{def}}{=} p_0(x_0) \underbrace{\frac{\exp\left(\langle x_0,x_1\rangle/\epsilon\right) v(x_1)}{c_v(x_0)}}_{=\pi_v(x_1|x_0)}, \tag{6}$$

is an approximation of the optimal plan constructed by v instead of v^* in (3). In turn, $S_v \in \mathcal{P}(\Omega)$ is a process with joint marginal (at $t=0,1$) is π_v and $S_{|x_0,x_1} = W^\epsilon_{|x_0,x_1}$. Its drift g_v can be recovered by using (4) with v instead of v^*.

Here we use the letter S instead of T to denote the process, and this is for a reason. With mild assumptions on v, the process S_v is the Schrödinger bridge between p_0 and $p_v(x_1) \overset{\text{def}}{=} \int_{\mathbb{R}^D} \pi_v(x_0,x_1)dx_0$, i.e., its marginal at $t=1$. This follows from the EOT benchmark constructor theorem (Gushchin et al., 2023b, Theorem 3.2). Hence, minimization (5) can be viewed as the optimization over processes S_v, which are SBs determined by their potential v.

Unfortunately, the optimization of (5) is tricky. While the potential v can be directly parameterized, e.g., with a neural network v_θ, the key challenge is to compute c_v, which is a non-trivial integral. Note that due to (3), one has $\pi^*(x_1|x_0=0) \propto v^*(y)$, i.e., v^* is an unnormalized density of some distribution. This fact is exploited in (Mokrov et al., 2024; Korotin et al., 2024) to establish ways to optimize (5).

Energy-guided EOT solver (EgNOT). In (Mokrov et al., 2024), the authors find out that, informally, objective (5) aims to find an unnormalized density v^* by optimizing KL divergence. Therefore, it resembles the objectives of *Energy-based Models* (LeCun et al., 2006, EBM). Inspired by this discovery, the authors show how the standard EBM approaches can be modified to optimize (5) and later sample from the learned plan π_v. The limitation of the approach is the necessity to use time-consuming MCMC techniques.

Light Schrödinger Bridge solver (LightSB). In (Korotin et al., 2024), they use the fact from (Gushchin et al., 2023b) that the Gaussian parameterization

$$v_\theta(x_1) = \sum_{k=1}^{K} \alpha_k \mathcal{N}(x_1|\mu_k, \epsilon\Sigma_k) \tag{7}$$

for v provides a closed form analytic expression for c_θ. This removes the necessity to use time-consuming MCMC approaches at both the training and the inference. Furthermore, Gaussian parameterization provides *the closed form* expression for the drift of S_v and allows lightspeed sampling from conditional distributions $\pi_v(x_1|x_0)$, see (Korotin et al., 2024, Propositions 3.2, 3.3).

2.3. Bridge matching Procedures for EOT/SB

Recovering SB process from EOT plan (OT-CFM). Since every SB solution is given by the EOT plan π^* and the Brownian Bridges $W^\epsilon_{|x_0,x_1}$, i.e., $T^* = \int_{\mathbb{R}^D\times\mathbb{R}^D} W^\epsilon_{|x_0,x_1} d\pi^*(x_0,x_1)$, solution of the EOT problem π^* already provides a way to sample from marginal distributions $p_{T^*}(x_t,t)$ of T^* at each time $t\in[0,1]$. The authors of (Tong et al., 2023) propose to use this property to recover the drift $g^*(x_t,t)$ of the process T^* using flow (Lipman et al., 2022) and score matching techniques. They use the flow matching to fit the drift $g^\circ(x_t,t)$ of the probability flow ODE for marginals $p_{T^*}(x_t,t)$ (at time t) of the process T^*, i.e., $g^\circ(x_t,t)$ for which the continuity equation $\frac{\partial p_{T^*}(x_t,t)}{\partial t} = -\nabla\cdot(p_{T^*}(x_t,t)g^\circ(x_t,t))$ holds. In turn, score matching is used to fit the score functions $\nabla \log p_{T^*}(x_t,t)$ of marginal distributions. Then they recover the Schrödinger bridge drift by using the relationship between the probability flow ODE and the SDE representation of stochastic processes: $g^\circ(x_t,t) + \frac{\epsilon}{2}\nabla\log p_{T^*}(x_t,t) = g^*(x_t,t)$.

Unfortunately, the solution of the EOT problem π^* for two arbitrary distributions p_0 and p_1 is unknown. The authors use the discrete (minibatch) OT between empirical distributions $\widehat{p}_0 \overset{\text{def}}{=} \sum_{n=1}^N \delta_{x_n}$ and $\widehat{p}_1 \overset{\text{def}}{=} \sum_{m=1}^M \delta_{y_m}$ constructed by available samples instead. However, the empirical EOT plan $\widehat{\pi}$ may be highly biased from the true π^*. This potentially leads to undesirable errors in approximating SB.

Learning SB process without EOT solution (DSBM). Another matching method has been proposed by (Shi et al., 2023) to get the SB without knowing the EOT plan π^*. To begin with, for any $\pi\in\Pi(p_0,p_1)$, define T_π (called the *reciprocal* process of π) as a mixture of Brownian Bridges with weights given by π, i.e., $T_\pi = \int_{\mathbb{R}^D\times\mathbb{R}^D} W^\epsilon_{|x_0,x_1} d\pi(x_0,x_1)$.

To get π^* and T^*, the authors alternate between two projections of stochastic processes: the *reciprocal* and the *Markovian*. For a process $T\in\mathcal{P}(\Omega)$, its reciprocal projection is a mixture of Brownian bridges given by the plan π^T:

$$\text{proj}_{\mathcal{R}}(T) \overset{\text{def}}{=} \int_{\mathbb{R}^D\times\mathbb{R}^D} W^\epsilon_{|x_0,x_1} d\pi^T(x,y). \tag{8}$$

This is a reciprocal process with the same joint marginal π^T at times $t=0,1$ as T (one may write $\text{proj}_{\mathcal{R}}(T) = T_{\pi^T}$).

Consider any reciprocal process T_π. Its Markovian projection $\text{proj}_{\mathcal{M}}(T_\pi)$ is a diffusion process defined by an SDE $dx_t = g(x_t,t)dt + \sqrt{\epsilon}dW_t$, that preserves all time marginals of T_π. Its drift function is analytically given by:

$$g(x_t,t) = \int_{\mathbb{R}^D} \frac{x_1-x_t}{1-t} p_{T_\pi}(x_1|x_t)dx_1, \tag{9}$$

where p_{T_π} denotes the distribution of T_π. Drift (9) is a solution to the following optimization problem:

$$\min_g \int_0^1 \int_{\mathbb{R}^D\times\mathbb{R}^D} ||g(x_t,t) - \frac{x_1-x_t}{1-t}||^2 dp_{T_\pi}(x_t,x_1)dt \tag{10}$$

and can be learned by sampling $(x_0,x_1)\sim\pi$, $x_t\sim W^\epsilon_{|x_0,x_1}$ and parametrizing g by a neural network. This procedure is the so-called **bridge matching** procedure.

The authors prove (Shi et al., 2023, Theorem 8) that a sequence $(T^l)_{l\in\mathbb{N}}$ constructed by alternating the projections

$$T^{2l+1} = \text{proj}_{\mathcal{M}}(T^{2l+2}), \quad T^{2l} = \text{proj}_{\mathcal{R}}(T^{2l+1}), \tag{11}$$

with $T^0 = T_\pi$ and any $\pi\in\Pi(p_0,p_1)$ converges to the SB solution T^* between p_0 and p_1. When $\epsilon\to 0$, the Markovian projection transforms into the well-known flow matching procedure (Lipman et al., 2022), and the whole iterative procedure becomes the Rectified Flow (Liu et al., 2022).

Markovian projection (9) is the bottleneck of the iterative procedure. In practice, the method uses a neural net to learn the drift of the projection. This introduces approximation errors at each iteration. The errors lead to differences between the process T^n's marginal distribution at time $t=1$ and the actual p_1. These errors accumulate after each iteration and affect convergence, motivating the search for a bridge matching procedure that converges in a single iteration.

3. Light and Optimal SB Matching Solver

In §3.1, we present the main theoretical development of our paper – the optimal Schrödinger bridge matching method. Next, in §3.2, we propose our novel **LightSB-M** solver, which implements the method in practice. In §3.3, we discuss its connections with the related EOT/SB solvers. In Appendix A we provide proofs of all theorems.

3.1. Theory. Optimal Schrödinger Bridge Matching

Our algorithm is based on the properties of KL projections of stochastic processes on the set $\mathcal{S}$ of Schrödinger Bridges:

$$\mathcal{S} \overset{\text{def}}{=} \big\{S \in \mathcal{P}(\Omega) \text{ such that } \exists p_0^S, p_1^S \in \mathcal{P}(\mathbb{R}^D) \\ \text{for which } S = \operatorname*{arg\,min}_{T\in\mathcal{F}(p_0^S,p_1^S)} \text{KL}\left(T\|W^\epsilon\right) \big\}. \tag{12}$$

In addition to reciprocal and Markovian projections, we define a new "optimal projection" (OP). Consider **any** plan $\pi \in \Pi(p_0, p_1)$, e.g., independent, minibatch, optimal, etc. Given a **reciprocal** process T_π, its projection is the process

$$\text{proj}_{\mathcal{S}}(T_\pi) \overset{\text{def}}{=} \operatorname*{arg\,min}_{S\in\mathcal{S}} \text{KL}\left(T_\pi\|S\right). \tag{13}$$

We prove that optimal projection allows to obtain the solution of Schrödinger Bridge in just one projection step.

Theorem 3.1 (OP of a reciprocal process)**.** *The optimal projection of a reciprocal process T_π, given by a joint distribution $\pi \in \Pi(p_0, p_1)$ leads to the Schrödinger Bridge T^* between the distributions p_0 and p_1, i.e.:*

$$proj_{\mathcal{S}}(T_\pi) = \operatorname*{arg\,min}_{S\in\mathcal{S}} KL\left(T_\pi\|S\right) = T^*. \tag{14}$$

To implement this in practice, we need to **(a)** have a tractable estimator of $\text{KL}\left(T_\pi\|S\right)$ and **(b)** be able to optimize over $\mathcal{S}$. We denote $\mathcal{S}(p_0)$ as the subset of $\mathcal{S}$ of processes which start at p_0 at $t = 0$. Since $T^* \in \mathcal{S}(p_0)$, it suffices to optimize over $\mathcal{S}(p_0)$ in (14). As it was noted in the background §2.2, processes $S \in \mathcal{S}(p_0)$ are determined by their adjusted Schrödinger potential v. We will write S_v instead of S for convenience.

Theorem 3.2 (Tractable objective for the OP)**.** *For the SB $S_v \in \mathcal{S}(p_0)$ and a reciprocal process T_π with $\pi \in \Pi(p_0, p_1)$ the optimal projection objective* (13) *is*

$$KL\left(T_\pi\|S_v\right) = C(\pi) + \\ \frac{1}{2\epsilon}\int_0^1\int_{\mathbb{R}^D\times\mathbb{R}^D} ||g_v(x_t,t) - \frac{x_1 - x_t}{1-t}||^2 dp_{T_\pi}(x_t,x_1)dt, \tag{15}$$

where g_v is the drift of S_v given by (4) *(with v instead of v^*). Here the constant $C(\pi)$ does not depend on S_v.*

This result provides an opportunity to optimize S_v via fitting its drift g_v. Indeed, we can estimate $\text{KL}\left(T_\pi\|S_v\right)$ up to a constant by sampling from T_π. To sample from T_π, it is sufficient to sample a pair $(x_0, x_1) \sim \pi$ and then to sample x_t from the Brownian bridge $W^\epsilon_{|x_0,x_1}$. The natural remaining question is how to parameterize the drifts of the SB processes $S_v \in \mathcal{S}$. We explain this in the section below.

3.2. Practice. LightSB-M Optimization Procedure

To solve the Schrödinger Bridge between two distributions p_0 and p_1 by using optimal projection (13) and its tractable objective (15), we use **any** plan $\pi \in \Pi(p_0, p_1)$ accessible by samples. It can be the independent plan, i.e., just independent samples from p_0, p_1, any minibatch OT plan, i.e., the one obtained by solving discrete OT on minibatch from p_0 and p_1, etc. To optimize over Schrödinger Bridges $S_v \in \mathcal{S}$, we use the parametrization of v as a Gaussian mixture (7) from LightSB (§2.2), which for every v_θ provides $g_\theta \overset{\text{def}}{=} g_{v_\theta}$ (4) in a closed form (Korotin et al., 2024, Proposition 3.3):

$$g_\theta(x,t) = \epsilon\nabla_x \log\big(\mathcal{N}(x|0,\epsilon(1-t)I_D) \\ \sum_{k=1}^K \big\{\alpha_k\mathcal{N}(r_k|0,\epsilon\Sigma_k)\mathcal{N}(h(x,t)|0,A_k^t)\big\}\big) \tag{16}$$

with $A_k^t \overset{\text{def}}{=} \frac{t}{\epsilon(1-t)}I_D + \frac{\Sigma_k^{-1}}{\epsilon}$ and $h_k(x,t) \overset{\text{def}}{=} \frac{x}{\epsilon(1-t)} + \frac{1}{\epsilon}\Sigma_k^{-1}r_k$. Using this parametrization and **any** $\pi \in \Pi(p_0, p_1)$, we optimize objective (15) with the stochastic gradient descent.

Algorithm 1 Light SB Matching (LightSB-M)

```
Input : plan $\pi \in \Pi(p_0, p_1)$ accessible by samples; adjusted Schrödinger potential $v_\theta$ parametrized by a gaussian mixture ($\theta = \{\alpha_k, \mu_k, \Sigma_k\}_{k=1}^K$).
Output : learned drift $g_\theta$ approximating the optimal $g^*$.
repeat
    Sample batch of pairs $\{x_0^n, x_1^n\}_{n=0}^N \sim \pi$;
    Sample batch $\{t_n\}_{n=0}^N \sim U[0, 1]$;
    Sample batch $\{x_t^n\}_{n=0}^N \sim W^\epsilon_{|x_0,x_1}$;
    $\mathcal{L}_\theta \leftarrow \frac{1}{N}\sum_{n=1}^N ||g_\theta(x_t^n, t_n) - \frac{1}{1-t_n}(x_1^n - x_t^n)||^2$;
    Update $\theta$ using $\frac{\partial\mathcal{L}_\theta}{\partial\theta}$;
until converged;
```

The **training** procedure is described in Algorithm 1. We recall that the Brownian bridge $W^\epsilon_{|x_0,x_1}$ has time marginals $p_{BB}(x_t|x_0,x_1) \overset{\text{def}}{=} \mathcal{N}(x_t|tx_1 + (1-t)x_0, \epsilon t(1-t)I_D)$, i.e. has a normal distribution with a scalar covariance matrix.

After learning the drift $g_v(x,t)$ of the Schrödinger Bridge SDE $dx_t = g_v(x_t,t)dt + \sqrt{\epsilon}dW_t$, one can use any SDE solver to **infer** trajectories. For example, one can use the simplest and most popular Euler-Maruyama scheme (Kloeden et al., 1992, §9.2). However, SDE solvers introduce some errors due to discrete approximations. Using the

LightSB parameterization of Schrödinger bridges from (Korotin et al., 2024), we can sample trajectories without having to solve the learned SDE numerically. To do so, we first sample from the learned plan $\pi_v(x_1|x_0)$ given by (6) and then sample the trajectory of the Brownian bridge $W^{\epsilon}_{|x_0,x_1}$ using it's self-similarity property (Korotin et al., 2024, §3.2). We recall that the self-similarity of the Brownian bridge means that if we have a trajectory $x_0, x_{t_1}, ..., x_{t_L}, x_1$, we can sample a new point at time $t_l < t < t_{l+1}$ by using the following property of the Brownian bridge:

$$x_t \sim \mathcal{N}\big(x_t|x_{t_l} + \frac{t'-t_l}{t_{l+1}-t_l}(x_{t_{l+1}} - x_{t_l}), \epsilon\frac{(t'-t_l)(t_{l+1}-t')}{t_{l+1}-t_l}\big).$$

3.3. Connections to the Most Related Prior Works

DSBM (Shi et al., 2023). Schrödinger Bridge T^* between p_0 and p_1 is the only process that simultaneously is Markovian and reciprocal (Léonard, 2013, Proposition 2.3). This fact lies at the core of DSBM's iterative approach of alternating Markovian and reciprocal projections. In turn, our optimal projection (13) provides the SB in **one step**, projecting a process on the set of processes that are both reciprocal and Markovian, i.e., Schrödinger Bridges.

OT-CFM (Tong et al., 2023). Our optimal projection (13) of a reciprocal process of T_π with **any** $\pi \in \Pi(p_0, p_1)$ is the same Schrödinger Bridge between p_0 and p_1. Thus, optimal projection does not depend on the choice of the plan π. In turn, OT-CFM provides theoretical guarantees of finding the Schrödinger Bridge only if one chooses as plan π the EOT plan π^*, which is unknown for arbitrary distributions p_0, p_1.
EgNOT/LightSB (Mokrov et al., 2024; Korotin et al., 2024). Our main objective (13) resembles objective (5) of EgNOT and LightSB as the latter equals KL $(T^*\|S_v)$ up to a constant. At the same time, our objective allows to use **any** reciprocal process T_π instead of $T^* = T_{\pi^*}$. Interestingly, our obtained tractable bridge matching objective turns out to be closely related to the EgNOT/LightSB objective (5).

Theorem 3.3 (Equivalence to EgNOT/LightSB objective)**.** *The OP objective* (15) *for a reciprocal process T_π and $\pi \in \Pi(p_0, p_1)$ is equivalent to LightSB objective $\mathcal{L}_0$* (5)*:*

$$\frac{1}{2\epsilon}\int_0^1\int_{\mathbb{R}^D\times\mathbb{R}^D}||g_v(x_t,t) - \frac{x_1-x_t}{1-t}||^2 dp_{T_\pi}(x_t,x_1)dt = \widetilde{C}(\pi) + \mathcal{L}_0(v).$$

One interesting conclusion from this equivalence is that our LightSB-M solver automatically inherits the theoretical generalization and approximation properties of the LightSB solver; see (Korotin et al., 2024, §3) for details about them.

4. Other Related Works

Here, we overview other existing works related to solving SB/EOT. Unlike the works described above, these are less relevant to our study. Still, we want to highlight some aspects of other solvers related to our solver.

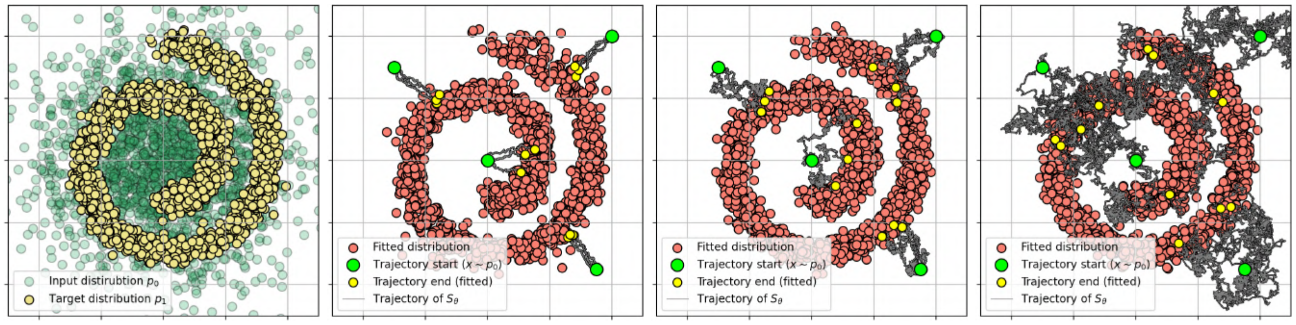
4.1. Iterative proportional fitting (IPF) solvers.

There are several Schrödinger Bridge solvers (Vargas et al., 2021; De Bortoli et al., 2021; Chen et al., 2021a) for continuous probability distributions based on the Iterative Proportional Fitting (IPF) procedure (Fortet, 1940; Kullback, 1968; Ruschendorf, 1995). The IPF procedure is related to the Sinkhorn algorithm (Cuturi, 2013) and, as was recently shown in work (Vargas & Nüsken, 2023), coincides with the expectation-maximization (EM) algorithm (Dempster et al., 1977). All these three IPF-based SB solvers consist of iterative reversing of Markovian processes and differ only in particular methods to fit a reversion of a process by a neural network. The first two (Vargas et al., 2021; De Bortoli et al., 2021) methods use similar mean-matching procedures, while the last (Chen et al., 2021a) utilizes a different approach which includes the estimation of a divergence.

In (Shi et al., 2023) the authors show, that due to iterative nature of one of these solvers (De Bortoli et al., 2021) it can diverge, due to errors accumulation on each iteration. Furthermore, the authors of (Vargas & Nüsken, 2023) show that these solvers tend to lose the information of Wiener Prior of Schrödinger Bridge and converge to the Markovian process that does not solve the SB problem. In turn, *our approach eliminates the need for iterative learning* of a sequence of Markovian processes and is free from the possible issues with divergence or obtaining a biased solution.

4.2. EOT solvers and EOT-based SB solvers.

Recall that EOT and SB problems are closely related: SB solutions can be recovered from EOT solutions by using Brownian Bridge $W^{\epsilon}_{|x_0,x_1}$ or recovering the drift $g(x_t, t)$, e.g., as in (Tong et al., 2023). Due to this, we also give a quick overview of EOT solvers for continuous distributions. Several works (Genevay et al., 2016; Seguy et al., 2018; Daniels et al., 2021) consider solving the EOT problem by utilizing the classic dual EOT problem (Genevay et al., 2019). Classic dual EOT problem for continuous p_0 and p_1 is an unconstrained maximization problem over dual variables, also called potentials, which can be parameterized by neural networks and trained. After training, these potentials can be used to directly sample from distribution $\pi^*(x_1|x_0)$ by using additional score model for $\nabla_x \log p_1(x)$ (Daniels et al., 2021) or to train neural network model to predict conditional expectation $\mathbb{E}_{\pi^*(x_1|x_0)}x_1$, i.e., the barycentric projection. However, the main disadvantage of these methods is that in practice, dual EOT problem cannot be solved by neural networks for practically meaningful (small) coefficients ϵ due to numerical errors of calculating dual EOT objective since it includes terms in form $\mathbb{E}_{x_0\sim p_0, x_1\sim p_1}\exp(\frac{f(x_0,x_1)}{\epsilon})$.



(a) $x \sim p_0, y \sim p_1$. (b) $\epsilon = 0.01$. (c) $\epsilon = 0.1$. (d) $\epsilon = 1$.

Figure 2: The process S_θ learned with LightSB-M **(ours)** in *Gaussian→Swiss roll* example (§5.1).

There is also one SB solver based on the theory of EOT dual problem (Gushchin et al., 2023a). This solver directly fits the drift g of the Schrödinger Bridge by using a maximin reformulation of the dual EOT problem and its link to the SB problem. This allows to overcome the numerical problems and solve SB for practically meaningful values of ϵ.

Our solver is also based on solving EOT and SB using the theory behind the dual EOT problem. Thanks to using parametrization of adjusted Schrödinger potential as in (Korotin et al., 2024) instead of EOT potentials as in (Seguy et al., 2018; Daniels et al., 2021) and using novel optimization objective based on bridge matching, *our method overcomes numerical issues of the previously developed dual EOT-based methods* without the maximin optimization.

4.3. Other SB solvers.

The authors of (Kim et al., 2024) propose a different minimax SB solver by considering the self-similarity of the SB in learning objectives and an additional consistency regularization. While showing good results, their approach requires using neural estimation of entropy, which involves solving additional optimization problem at every minimization step.

All previously considered solvers are designed to solve SB as a problem of finding the optimal translation between two distributions p_0, p_1 without any paired data from them, but there are also several SB solvers (Liu et al., 2023a; Somnath et al., 2023) for setups with paired trained data such as the super-resolution. In fact, the concept of bridge matching was introduced in (Liu et al., 2023a) but for the paired setup. The authors work under the assumption that the available paired data is a good approximation of the EOT plan and propose using Bridge matchi to recover the SB from this data, which makes their method related to (Tong et al., 2023). As noted earlier, *our solver provably recovers SB using data provided by arbitrary plan π between p_0 and p_1.*

5. Experimental Illustrations

To evaluate our new LightSB-M solver, we considered several setups from related works. The code for our solver is written in PyTorch and available at https://github.com/SKholkin/LightSB-Matching. For each experiment, we present a separate self-explaining Jupyter notebook, which can be used to reproduce the results of our solver. We provide the technical details in Appendix B.

5.1. Qualitative 2D Example

We start our evaluation with an illustrative 2D setup. We solve the SB between a Gaussian distribution p_0 and a Swiss roll p_1. We run our LightSB-M solver with mini-batch (MB) discrete OT as plan π for different values of the coefficient ϵ and present the results in Figure 2. As expected, we see that the amount of noise in the trajectories and the stochasticity of the learned map are proportional to coefficient ϵ. The technical details of this setup are given in Appendix B.1.

5.2. Quantitative Evaluation on the SB Benchmark

We use the SB mixtures benchmark proposed by (Gushchin et al., 2023b, §4) to experimentally verify that our approach based on the optimal projection is indeed able to solve the Schrödinger Bridge between p_0 and p_1 *by using any reciprocal process T_π, $\pi \in \Pi(p_0, p_1)$*. The benchmark provides continuous probability distribution pairs p_0, p_1 for dimensions $D \in \{2, 16, 64, 128\}$ with the known EOT plan $\pi^*(x_0, x_1)$ for parameter $\epsilon \in \{0.1, 1.10\}$. To evaluate the quality of the SB solution (EOT plan) we use cB$\mathbb{W}_2^2$-UVP metric as suggested by the authors (Gushchin et al., 2023b, §5). Additionally, we study how well the solvers restore the target distribution p_1 in Appendix B.3.

We provide results of our LightSB-M solver with independent (ID) and mini-batch discrete OT (MB) as π in T_π for mixture benchmark pairs in Table 1. Since the benchmark provides the ground truth EOT plan π^* (GT), we also run our solver with it. Note that we have access to the GT EOT plan thanks to the benchmark, and in regular setups there is, of course, no access to it. As shown in the Table 1, our solver demonstrates comparable performance to the best among other solvers for all considered plans π. As noted in (Korotin et al., 2024, §5.2), the mixture parameterization used by LightSB and which we adapt in our LightSB-M

		$\epsilon = 0.1$				$\epsilon = 1$				$\epsilon = 10$			
	Solver Type	$D\!=\!2$	$D\!=\!16$	$D\!=\!64$	$D\!=\!128$	$D\!=\!2$	$D\!=\!16$	$D\!=\!64$	$D\!=\!128$	$D\!=\!2$	$D\!=\!16$	$D\!=\!64$	$D\!=\!128$
Best solver on benchmark†	Varies	1.94	13.67	11.74	11.4	1.04	9.08	18.05	15.23	1.40	1.27	2.36	1.31
LightSB†	KL minimization	0.03	0.08	0.28	0.60	0.05	0.09	0.24	0.62	0.07	0.11	0.21	0.37
DSBM		5.2	16.8	37.3	35	0.3	1.1	9.7	31	3.7	105	3557	15000
SF^2M-Sink		0.54	3.7	9.5	10.9	0.2	1.1	9	23	0.31	4.9	319	819
LightSB-M (ID, **ours**)	Bridge matching	0.04	0.18	0.77	1.66	0.09	**0.18**	0.47	1.2	**0.12**	0.19	**0.36**	0.71
LightSB-M (MB, **ours**)		**0.02**	**0.1**	0.56	1.32	0.09	**0.18**	**0.46**	**1.2**	0.13	**0.18**	**0.36**	0.71
LightSB-M (GT, **ours**)		**0.02**	**0.1**	**0.49**	**1.16**	**0.09**	**0.18**	0.47	**1.2**	0.13	**0.18**	**0.36**	**0.69**

Table 1: Comparisons of $\text{cB}\mathbb{W}_2^2$-UVP $\downarrow$ (%) between the optimal plan π^* and the learned plan π_θ on the EOT/SB benchmark (§5.2). The best metric over *bridge matching* solvers is **bolded**. Results marked with † are taken from (Korotin et al., 2024).

Solver type	**DIM** **Solver**	**50**	**100**	**1000**
Langevin-based	(Mokrov et al., 2024)† [1 GPU V100]	2.39 ± 0.06 (19 m)	2.32 ± 0.15 (19 m)	1.46 ± 0.20 (15 m)
Minimax	(Gushchin et al., 2023a)† [1 GPU V100]	2.44 ± 0.13 (43 m)	2.24 ± 0.13 (45 m)	1.32 ± 0.06 (71 m)
IPF	(Vargas et al., 2021)† [1 GPU V100]	3.14 ± 0.27 (8 m)	2.86 ± 0.26 (8 m)	2.05 ± 0.19 (11 m)
KL minimization	LightSB (Korotin et al., 2024)† [4 CPU cores]	2.31 ± 0.27 (65 s)	2.16 ± 0.26 (66 s)	1.27 ± 0.19 (146 s)
Bridge matching	DSBM (Shi et al., 2023) [1 GPU V100]	2.46 ± 0.1 (6.6 m)	2.35 ± 0.1 (6.6 m)	1.36 ± 0.04 (8.9 m)
	SF^2M-Sink (Tong et al., 2023) [1 GPU V100]	2.66 ± 0.18 (8.4 m)	2.52 ± 0.17 (8.4 m)	1.38 ± 0.05 (13.8 m)
	LightSB-M (ID, **ours**) [4 CPU cores]	2.347 ± 0.11 (58 s)	2.174 ± 0.08 (60 s)	1.35 ± 0.05 (147 s)
	LightSB-M (MB, **ours**) [4 CPU cores]	$\mathbf{2.33} \pm 0.09$ (80 s)	$\mathbf{2.172} \pm 0.08$ (80 s)	$\mathbf{1.33} \pm 0.05$ (176 s)

Table 2: Energy distance (averaged for two setups and 5 random seeds) on the MSCI dataset (§5.3) along with 95%-confidence interval ($\pm$ intervals) and average training times (s - seconds, m - minutes). The best *bridge matching* solver according to the mean value is **bolded**. Results marked with † are taken from (Korotin et al., 2024).

solver may introduce some inductive bias, since it uses the analogous principles used to construct the benchmark.

We empirically see that our LightSB-M solver finds the same (optimal) solution for all considered plans π.

Baselines. We present results for other bridge matching methods such as DSBM (Shi et al., 2023), which uses Markovian and reciprocal projections, and SF^2M-Sink (Tong et al., 2023), which uses an approximation of the EOT plan by the Sinkhorn algorithm (Cuturi, 2013). On the setups with $\epsilon = 10$ both methods exibits difficulties due to the necessity to learn SDE with high magnitude. On the setups with $\epsilon = 0.1$ and $\epsilon = 1$, SF^2M-Sink works better than DSBM. This result may seem counterintuitive at first, since DSBM methods should find the true SB solution, while SF^2M-Sink should find some approximation to it based on how close the minibatch discrete EOT approximates the GT EOT plan. One possible reason is that DSBM simply requires more iterations of Markovian/reciprocal projections. However, in our experiments we observe that increasing the number of iterations does not improve the quality.

We provide an additional study of dynamic metrics and the inference speed of our solver in Appendix B.3.

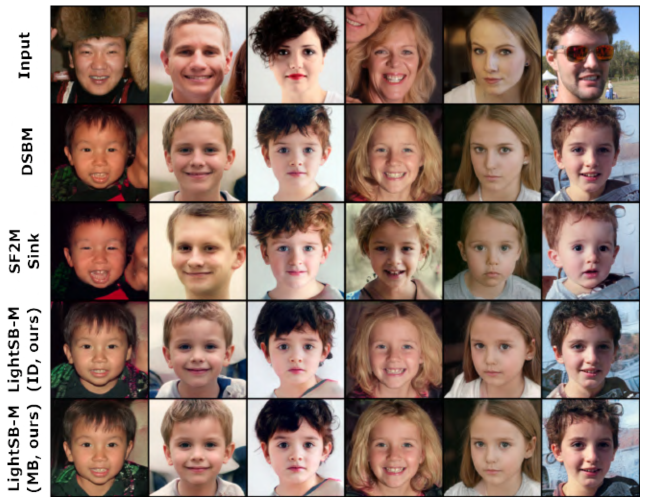
5.3. Quantitative Evaluation on Biological Data

We evaluate our algorithm on the inference of cell trajectories from unpaired single-cell data problem, where OT/SB is widely used (Vargas et al., 2021; Tong et al., 2023; Koshizuka & Sato, 2022). We consider the recent high-dimensional single-cell setup provided by (Tong et al., 2023) based on the dataset from the Kaggle competition "Open Problems - Multimodal Single-Cell Integration." This dataset provides single-cell data from four human donors on days $2, 3, 4$ and 7 and describes the gene expression levels of distinct cells. The task of this setup is to learn a trajectory model for the cell dynamics, given only unpaired samples at two time points, representing distributions p_0 and p_1. As in related works (Tong et al., 2023; Korotin et al., 2024), we use PCA projections of the original data with $\text{DIM} \in \{50, 100, 1000\}$ components.

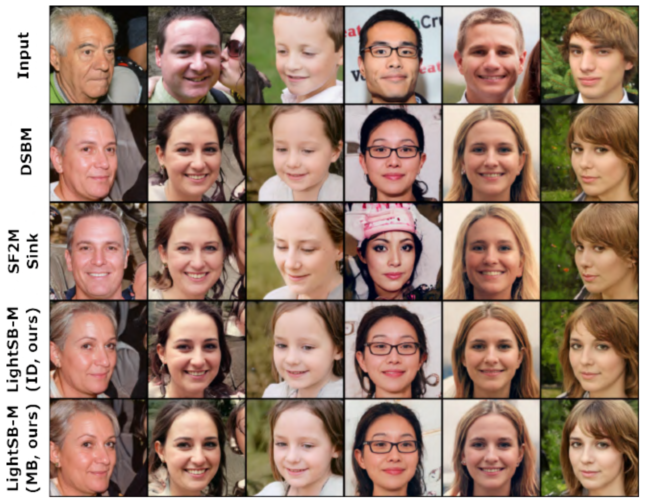
In our experiments, we consider two setups by taking data from two different days as p_0, p_1 to solve the Shrödinger Bridge and one intermediate day for evaluation. The first setup includes data from day 2 as p_0, data from day 4 as p_1, and data from day 3 for evaluation, while the second setup includes data from day 3 as p_0, data from day 7 as p_1, and data from day 4 for evaluation. At evaluation, we use learned models to sample one trajectory for each cell from the initial distribution p_0 and then compare the predicted distribution at the intermediate time point with the ground truth data distribution. For comparison, we use energy distance (Rizzo & Székely, 2016) and present results in Table 2.

We see that our LightSB-M's solution with independent (ID) and minibatch discrete OT (MB) plans for T_π provides the same metrics since it learns the same solution, as follows from the developed theory. It also shows performance on the same level as other neural network-based matching methods such as DSBM and SF^2M-Sink, but converges faster even without using GPU similar to the LightSB solver.

In Appendix B.2, we provide the technical details for this setup and additional results for different values of ϵ.



(a) Adult → Child



(b) Man → Woman

Figure 3: Unpaired translation between subsets of FFHQ dataset (1024x1024) performed by various SB solvers (§5.4) in the latent space of ALAE (Pidhorskyi et al., 2020).

5.4. Comparison on Unpaired Image-to-image Transfer

Another popular setup that involves learning a translation between two distributions without paired data is image-to-image translation (Zhu et al., 2017). Methods based on SB show promising results in solving this problem thanks to the perfect theoretical agreement of this setup with the SB formulation (Shi et al., 2023). Due to the used parameterization based on Gaussian mixture, learning the translation between low-dimensional image manifolds is difficult for LightSB-M. Fortunately, many approaches use autoencoders (Rombach et al., 2022) for more efficient generation and translation. We follow the setup of (Korotin et al., 2024) with the pre-trained ALAE autoencoder (Pidhorskyi et al., 2020) on 1024×1024 FFHQ dataset (Karras et al., 2019).

We present the qualitative results of our solver with discrete minibatch OT plan (MB) and independent plan (ID) in Fig 3. For comparison, we also provide results of DSBM and SF^{2}M-Sink. Our LightSB-M solver converges to nearly the same solution for both ID and MB plans and demonstrates good results. The samples provided by DSBM are close to the samples of LightSB-M, which is expected since both methods provide theoretical guarantees for solving the SB problem. Samples obtained by SF^{2}M-Sink slightly differ, probably due to the bias of the discrete EOT plans. We provide additional examples of translation in Appendix B.4. The details of the baselines are given in Appendix B.5.

6. Discussion

Potential impact. Our main contribution is methodological: we show that one may perform just a single (but *optimal*) bridge matching step to learn SB. This finding helps us eliminate limitations of existing bridge matching-based approaches, such as heuristical minibatch OT approximations or error accumulation during training. We believe that this insight is a significant step towards developing novel efficient computational approaches for SB/EOT tasks.

Limitations. Given an adjusted Schrödinger potential v, it may be not easy to compute the drift g_v (4) of S_v needed to perform the optimal SB matching. We employ the Gaussian mixture parameterization for v for which this drift g_v is analytically known (16). This allows to easily implement our optimal SB matching in practice and obtain a fast bridge matching based solver. Still such a parameterization sometimes may be not sufficient, e.g., for large-scale generative modeling tasks. We point to developing ways to use more general parameterization of v to our optimal SB matching, e.g., neural-network-based, as a promising research avenue. We show possible steps in this direction in Appendix C.

One other limitation of our LightSB-M solver is that it is applicable to a limited set of priors. In this paper, we only consider the Wiener prior, which is one of the most popular priors used for SB. However, our method can be applied to other priors by changing the variables. These include Arithmetic Brownian Motion and Geometric Brownian Motion, also known as the Black-Scholes model, which is widely used in mathematical finance. Developing light solvers for Scrödinger Bridges with more general priors is a promising direction for the future research.

Acknowledgements

The work was supported by the Analytical center under the RF Government (subsidy agreement 000000D730321P5Q0002, Grant No. 70-2021-00145 02.11.2021).

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Chen, T., Liu, G.-H., and Theodorou, E. Likelihood training of schrödinger bridge using forward-backward sdes theory. In *International Conference on Learning Representations*, 2021a.

Chen, Y., Georgiou, T. T., and Pavon, M. On the relation between optimal transport and schrödinger bridges: A stochastic control viewpoint. *Journal of Optimization Theory and Applications*, 169:671–691, 2016.

Chen, Y., Georgiou, T. T., and Pavon, M. Stochastic control liaisons: Richard sinkhorn meets gaspard monge on a schrodinger bridge. *SIAM Review*, 63(2):249–313, 2021b.

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

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

De Bortoli, V., Thornton, J., Heng, J., and Doucet, A. Diffusion schrödinger bridge with applications to score-based generative modeling. *Advances in Neural Information Processing Systems*, 34:17695–17709, 2021.

Dempster, A. P., Laird, N. M., and Rubin, D. B. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society: series B (methodological)*, 39(1):1–22, 1977.

Flamary, R., Courty, N., Gramfort, A., Alaya, M. Z., Boisbunon, A., Chambon, S., Chapel, L., Corenflos, A., Fatras, K., Fournier, N., Gautheron, L., Gayraud, N. T., Janati, H., Rakotomamonjy, A., Redko, I., Rolet, A., Schutz, A., Seguy, V., Sutherland, D. J., Tavenard, R., Tong, A., and Vayer, T. Pot: Python optimal transport. *Journal of Machine Learning Research*, 22(78):1–8, 2021. URL http://jmlr.org/papers/v22/20-451.html.

Fortet, R. Résolution d'un système d'équations de m. schrödinger. *Journal de Mathématiques Pures et Appliquées*, 19(1-4):83–105, 1940.

Genevay, A., Cuturi, M., Peyré, G., and Bach, F. Stochastic optimization for large-scale optimal transport. In *Advances in neural information processing systems*, pp. 3440–3448, 2016.

Genevay, A., Chizat, L., Bach, F., Cuturi, M., and Peyré, G. Sample complexity of sinkhorn divergences. In *The 22nd international conference on artificial intelligence and statistics*, pp. 1574–1583. PMLR, 2019.

Gushchin, N., Kolesov, A., Korotin, A., Vetrov, D., and Burnaev, E. Entropic neural optimal transport via diffusion processes. In *Advances in Neural Information Processing Systems*, 2023a.

Gushchin, N., Kolesov, A., Mokrov, P., Karpikova, P., Spiridonov, A., Burnaev, E., and Korotin, A. Building the bridge of schr\" odinger: A continuous entropic optimal transport benchmark. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023b.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Karras, T., Laine, S., and Aila, T. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4401–4410, 2019.

Kim, B., Kwon, G., Kim, K., and Ye, J. C. Unpaired image-to-image translation via neural schrödinger bridge. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=uQBW7ELXfO.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kloeden, P. E., Platen, E., Kloeden, P. E., and Platen, E. *Stochastic differential equations*. Springer, 1992.

Korotin, A., Gushchin, N., and Burnaev, E. Light schr\" odinger bridge. In *International Conference on Learning Representations*, 2024.

Koshizuka, T. and Sato, I. Neural lagrangian schr\"{o} dinger bridge: Diffusion modeling for population dynamics. In *The Eleventh International Conference on Learning Representations*, 2022.

Kullback, S. Probability densities with given marginals. *The Annals of Mathematical Statistics*, 39(4):1236–1243, 1968.

LeCun, Y., Chopra, S., Hadsell, R., Ranzato, M., and Huang, F. A tutorial on energy-based learning. *Predicting structured data*, 1(0), 2006.

Léonard, C. A survey of the schr\" odinger problem and some of its connections with optimal transport. *arXiv preprint arXiv:1308.0215*, 2013.

Lipman, Y., Chen, R. T., Ben-Hamu, H., Nickel, M., and Le, M. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2022.

Liu, G.-H., Vahdat, A., Huang, D.-A., Theodorou, E. A., Nie, W., and Anandkumar, A. I^2 sb: Image-to-image schr\" odinger bridge. *arXiv preprint arXiv:2302.05872*, 2023a.

Liu, X., Gong, C., et al. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *The Eleventh International Conference on Learning Representations*, 2022.

Liu, X., Zhang, X., Ma, J., Peng, J., and Liu, Q. Instaflow: One step is enough for high-quality diffusion-based text-to-image generation. *arXiv preprint arXiv:2309.06380*, 2023b.

Meng, C., He, Y., Song, Y., Song, J., Wu, J., Zhu, J.-Y., and Ermon, S. Sdedit: Guided image synthesis and editing with stochastic differential equations. *arXiv preprint arXiv:2108.01073*, 2021.

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

Pavon, M. and Wakolbinger, A. On free energy, stochastic control, and schrödinger processes. In *Modeling, Estimation and Control of Systems with Uncertainty: Proceedings of a Conference held in Sopron, Hungary, September 1990*, pp. 334–348. Springer, 1991.

Peyré, G., Cuturi, M., et al. Computational optimal transport. *Foundations and Trends® in Machine Learning*, 11(5-6): 355–607, 2019.

Pidhorskyi, S., Adjeroh, D. A., and Doretto, G. Adversarial latent autoencoders. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14104–14113, 2020.

Pinsky, M. A. and Karlin, S. 8 - brownian motion and related processes. In Pinsky, M. A. and Karlin, S. (eds.), *An Introduction to Stochastic Modeling (Fourth Edition)*, pp. 391–446. Academic Press, Boston, fourth edition edition, 2011. ISBN 978-0-12-381416-6. doi: https://doi.org/10.1016/B978-0-12-381416-6.00008-3. URL https://www.sciencedirect.com/science/article/pii/B9780123814166000083.

Rizzo, M. L. and Székely, G. J. Energy distance. *wiley interdisciplinary reviews: Computational statistics*, 8(1): 27–38, 2016.

Roberts, G. O. and Tweedie, R. L. Exponential convergence of Langevin distributions and their discrete approximations. *Bernoulli*, 2(4):341 – 363, 1996.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10684–10695, 2022.

Ruschendorf, L. Convergence of the iterative proportional fitting procedure. *The Annals of Statistics*, pp. 1160–1174, 1995.

Seguy, V., Damodaran, B. B., Flamary, R., Courty, N., Rolet, A., and Blondel, M. Large scale optimal transport and mapping estimation. In *International Conference on Learning Representations*, 2018.

Shi, Y., Bortoli, V. D., Campbell, A., and Doucet, A. Diffusion schrödinger bridge matching. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=qy07OHsJT5.

Somnath, V. R., Pariset, M., Hsieh, Y.-P., Martinez, M. R., Krause, A., and Bunne, C. Aligned diffusion schr\" odinger bridges. *arXiv preprint arXiv:2302.11419*, 2023.

Song, Y., Dhariwal, P., Chen, M., and Sutskever, I. Consistency models. *arXiv preprint arXiv:2303.01469*, 2023.

Tong, A., Malkin, N., Fatras, K., Atanackovic, L., Zhang, Y., Huguet, G., Wolf, G., and Bengio, Y. Simulation-free schr\" odinger bridges via score and flow matching. *arXiv preprint arXiv:2307.03672*, 2023.

Vargas, F. and Nüsken, N. Transport, variational inference and diffusions: with applications to annealed flows and schr\" odinger bridges. *arXiv preprint arXiv:2307.01050*, 2023.

Vargas, F., Thodoroff, P., Lamacraft, A., and Lawrence, N. Solving schrödinger bridges via maximum likelihood. *Entropy*, 23(9):1134, 2021.

Villani, C. *Optimal transport: old and new*, volume 338. Springer Science & Business Media, 2008.

Wang, Z., Zheng, H., He, P., Chen, W., and Zhou, M. Diffusion-gan: Training gans with diffusion. In *The Eleventh International Conference on Learning Representations*, 2022.

Zhu, J.-Y., Park, T., Isola, P., and Efros, A. A. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pp. 2223–2232, 2017.

A. Proofs

Proof of Theorem 3.1. Let p_0, p_1 denote the marginals of π. Let π^* be the EOT plan between p_0, p_1. Let p_0^S, p_1^S denote the distribution of S at $t=0$ and $t=1$, respectively. We use the fact that each element S of $\mathcal{S}$ is a reciprocal process with some EOT plan $\pi^S \in \Pi(p_0^S, p_1^S)$, i.e. $S = \int W^{\epsilon}_{|x_0,x_1} d\pi^S(x_0,x_1)$, recall §2.1. In turn, π^S can be represented through the input density p_0^S and the potential v^S as in (6), i.e.:

$$\pi^S(x_0,x_1) = p_0^S(x_0)\frac{\exp\left(\langle x_0, x_1\rangle/\epsilon\right)v^S(x_1)}{c_{v^S}(x_0)} \tag{17}$$

We derive:

$$\text{KL}\left(T_\pi \| S\right) = \text{KL}\left(\pi\|\pi^S\right) + \int_{\mathbb{R}^D\times\mathbb{R}^D} \text{KL}\left(T_{\pi|x_0,x_1}\|S_{|x_0,x_1}\right)\pi(x_0,x_1)dx_0dx_1 = \tag{18}$$

$$\text{KL}\left(\pi\|\pi^S\right) + \int_{\mathbb{R}^D\times\mathbb{R}^D} \underbrace{\text{KL}\left(W^{\epsilon}_{|x_0,x_1}\|W^{\epsilon}_{|x_0,x_1}\right)}_{=0}\pi(x_0,x_1)dx_0dx_1 = \tag{19}$$

$$\int_{\mathbb{R}^D\times\mathbb{R}^D} \frac{\log \pi(x_0,x_1)}{\log \pi^S(x_0,x_1)}\pi(x_0,x_1)dx_0dx_1 =$$

$$\int_{\mathbb{R}^D\times\mathbb{R}^D} \log \pi(x_0,x_1)\pi(x_0,x_1)dx_0dx_1 - \int_{\mathbb{R}^D\times\mathbb{R}^D} \log \pi^S(x_0,x_1)\pi(x_0,x_1)dx_0dx_1 =$$

$$-H(\pi) - \int_{\mathbb{R}^D\times\mathbb{R}^D} \log \pi^S(x_0,x_1)\pi(x_0,x_1)dx_0dx_1 = \tag{20}$$

$$-H(\pi) - \int_{\mathbb{R}^D\times\mathbb{R}^D} \log\left(p_0^S(x_0)\frac{\exp\left(\langle x_0,x_1\rangle/\epsilon\right)v^S(x_1)}{c_{v^S}(x_0)}\right)\pi(x_0,x_1)dx_0dx_1 = \tag{21}$$

$$-H(\pi) - \int_{\mathbb{R}^D\times\mathbb{R}^D}\Big(\log p_0^S(x_0) + \langle x_0,x_1\rangle + \log v^S(x_1) - \log c_{v^S}(x_0)\Big)\pi(x_0,x_1)dx_0dx_1 =$$

$$-H(\pi) - \int_{\mathbb{R}^D\times\mathbb{R}^D}\langle x_0,x_1\rangle\pi(x_0,x_1)dx_0dx_1$$

$$-\int_{\mathbb{R}^D\times\mathbb{R}^D}\Big(\log p_0^S(x_0) - \log c_{v^S}(x_0)\Big)\pi(x_0,x_1)dx_0dx_1 - \int_{\mathbb{R}^D\times\mathbb{R}^D}\log v^S(x_1)\pi(x_0,x_1)dx_0dx_1 =$$

$$-H(\pi) - \int_{\mathbb{R}^D\times\mathbb{R}^D}\langle x_0,x_1\rangle\pi(x_0,x_1)dx_0dx_1 - \int_{\mathbb{R}^D}\log v^S(x_1)\underbrace{\Big(\int_{\mathbb{R}^D}\pi(x_0|x_1)dx_0\Big)}_{=1=\int_{\mathbb{R}^D}\pi^*(x_0|x_1)dx_0}\underbrace{\pi(x_1)}_{=\pi^*(x_1)}dx_1$$

$$-\int_{\mathbb{R}^D}\Big\{\left(\log p_0^S(x_0) - \log c_{v^S}(x_0)\right)\underbrace{\int_{\mathbb{R}^D}\pi(x_1|x_0)dx_1}_{=1=\int_{\mathbb{R}^D}\pi^*(x_1|x_0)dx_1}\Big\}\underbrace{\pi(x_0)}_{=\pi^*(x_0)}dx_0 =$$

$$-H(\pi) - \int_{\mathbb{R}^D\times\mathbb{R}^D}\langle x_0,x_1\rangle\pi(x_0,x_1)dx_0dx_1 - \int_{\mathbb{R}^D}\log v^S(x_1)\Big(\int_{\mathbb{R}^D}\pi^*(x_0|x_1)dx_0\Big)\pi^*(x_1)dx_1$$

$$-\int_{\mathbb{R}^D}\Big\{\left(\log p_0^S(x_0) - \log c_{v^S}(x_0)\right)\int_{\mathbb{R}^D}\pi^*(x_1|x_0)dx_1\Big\}\pi^*(x_0)dx_0 =$$

$$-H(\pi) - \int_{\mathbb{R}^D\times\mathbb{R}^D}\langle x_0,x_1\rangle\pi(x_0,x_1)dx_0dx_1 - \int_{\mathbb{R}^D\times\mathbb{R}^D}\log v^S(x_1)\pi^*(x_0,x_1)dx_0dx_1$$

$$-\int_{\mathbb{R}^D\times\mathbb{R}^D}\Big(\log p_0^S(x_0) - \log c_{v^S}(x_0)\Big)\pi^*(x_0,x_1)dx_0dx_1 =$$

$$-H(\pi) - \int_{\mathbb{R}^D\times\mathbb{R}^D}\langle x_0,x_1\rangle\pi(x_0,x_1)dx_0dx_1$$

$$
\begin{aligned}
&+\underbrace{\int_{\mathbb{R}^D\times\mathbb{R}^D}\langle x_0,x_1\rangle\pi^*(x_0,x_1)dx_0dx_1-\int_{\mathbb{R}^D\times\mathbb{R}^D}\langle x_0,x_1\rangle\pi^*(x_0,x_1)dx_0dx_1}_{=0}\\
&-\int_{\mathbb{R}^D\times\mathbb{R}^D}\Big(\log p_0^S(x_0)+\log v^S(x_1)-\log c_{v^S}(x_0)\Big)\pi^*(x_0,x_1)dx_0dx_1=\\
&-H(\pi)-\int_{\mathbb{R}^D\times\mathbb{R}^D}\langle x_0,x_1\rangle\pi(x_0,x_1)dx_0dx_1+\int_{\mathbb{R}^D\times\mathbb{R}^D}\langle x_0,x_1\rangle\pi^*(x_0,x_1)dx_0dx_1\\
&-\int_{\mathbb{R}^D\times\mathbb{R}^D}\Big(\log p_0^S(x_0)+\langle x_0,x_1\rangle+\log v^S(x_1)-\log c_{v^S}(x_0)\Big)\pi^*(x_0,x_1)dx_0dx_1=\\
&-H(\pi)-\int_{\mathbb{R}^D\times\mathbb{R}^D}\langle x_0,x_1\rangle\pi(x_0,x_1)dx_0dx_1+\int_{\mathbb{R}^D\times\mathbb{R}^D}\langle x_0,x_1\rangle\pi^*(x_0,x_1)dx_0dx_1\\
&-\int_{\mathbb{R}^D\times\mathbb{R}^D}\log\Big(p_0^S(x_0)\underbrace{\frac{\exp\big(\langle x_0,x_1\rangle/\epsilon\big)v^S(x_1)}{c_{v^S}(x_0)}}_{\pi^S(x_1|x_0)}\Big)\pi^*(x_0,x_1)dx_0dx_1=\\
&-H(\pi)-\int_{\mathbb{R}^D\times\mathbb{R}^D}\langle x_0,x_1\rangle(\pi(x_0,x_1)-\pi^*(x_0,x_1))dx_0dx_1-\int_{\mathbb{R}^D\times\mathbb{R}^D}\log\pi^S(x_0,x_1)\pi^*(x_0,x_1)dx_0dx_1\\
&+\underbrace{\int_{\mathbb{R}^D\times\mathbb{R}^D}\log\pi^*(x_0,x_1)\pi^*(x_0,x_1)dx_0dx_1-\int_{\mathbb{R}^D\times\mathbb{R}^D}\log\pi^*(x_0,x_1)\pi^*(x_0,x_1)dx_0dx_1}_{=0}=\\
&\underbrace{-H(\pi)-\int_{\mathbb{R}^D\times\mathbb{R}^D}\langle x_0,x_1\rangle(\pi(x_0,x_1)-\pi^*(x_0,x_1))dx_0dx_1-\int_{\mathbb{R}^D\times\mathbb{R}^D}\log\pi^*(x_0,x_1)\pi^*(x_0,x_1)dx_0dx_1}_{\stackrel{\text{def}}{=}\widehat{C}(\pi)}\\
&+\underbrace{\int_{\mathbb{R}^D\times\mathbb{R}^D}\log\frac{\pi^*(x_0,x_1)}{\pi^S(x_0,x_1)}\pi^*(x_0,x_1)dx_0dx_1}_{=\mathrm{KL}(\pi^*\|\pi^S)}=\\
&\widehat{C}(\pi)+\mathrm{KL}\left(\pi^*\|\pi^S\right).
\end{aligned}
$$

In (18) we use disintegration theorem for KL divergence to distinguish process plan π^S and "inner part" (Vargas et al., 2021, Appendix C, D). In transition from (18) to (19) we notice, that $T_{\pi|x_0,x_1}=W^\epsilon_{|x_0,x_1}$ and $S_{|x_0,x_1}=W^\epsilon_{|x_0,x_1}$, since T_π is a reciprocal process as well as Schrödinger Bridge S. In transition from (20) to (21) we use the fact, that π^S is given by (17). Since

$$\mathrm{KL}\left(T_\pi\|S\right)=\widehat{C}(\pi)+\mathrm{KL}\left(\pi^*\|\pi^S\right),$$

the minimum of $\mathrm{KL}\left(T_\pi\|S\right)$ is achieved for S such that $\pi^S=\pi^*$, i.e., when S is the SB between p_0 and p_1.

□

Proof of Theorem 3.2. We start by using a Pythagorean theorem for Markovian projection (Shi et al., 2023, Lemma 6)

$$\mathrm{KL}\left(T_\pi\|S_v\right)=\mathrm{KL}\left(T_\pi\|\mathrm{proj}_{\mathcal{M}}(T_\pi)\right)+\mathrm{KL}\left(\mathrm{proj}_{\mathcal{M}}(T_\pi)\|S_v\right), \tag{22}$$

where the drift $g_{\mathcal{M}}$ of the Markoian projection $\mathrm{proj}_{\mathcal{M}}(T_\pi)$ is given by (9):

$$g_{\mathcal{M}}(x_t,t)=\int_{\mathbb{R}^D}\frac{x_1-x_t}{1-t}dp_{T_\pi}(x_1|x_t). \tag{23}$$

We use the expression of KL between Markovian processes starting from the same distribution p_0 through their drifts (Pavon & Wakolbinger, 1991) and note that Markovian projection preserve the time marginals $p_{T_\pi}(x_t)$:

$$\mathrm{KL}\left(\mathrm{proj}_{\mathcal{M}}(T_\pi)\|S_v\right)=\frac{1}{2\epsilon}\int_0^1\int_{\mathbb{R}^D}||g_v(x_t,t)-g_{\mathcal{M}}(x_t,t)||^2dp_{T_\pi}(x_t)dt \tag{24}$$

Then we substitute $g_{\mathcal{M}}$ by (23):

$$\text{KL}\left(\text{proj}_{\mathcal{M}}(T_\pi)\|S_v\right) = \frac{1}{2\epsilon}\int_0^1\int_{\mathbb{R}^D}||g_v(x_t,t) - g_{\mathcal{M}}(x_t,t)||^2 dp_{T_\pi}(x_t)dt =$$

$$\frac{1}{2\epsilon}\int_0^1\int_{\mathbb{R}^D}||g_v(x_t,t) - \int_{\mathbb{R}^D}\frac{x_1-x_t}{1-t}dp_{T_\pi}(x_1|x_t)||^2 dp_{T_\pi}(x_t)dt =$$

$$\frac{1}{2\epsilon}\int_0^1\int_{\mathbb{R}^D}\Big\{||g_v(x_t,t)||^2 - 2\langle g_v(x_t,t), \int_{\mathbb{R}^D}\frac{x_1-x_t}{1-t}dp_{T_\pi}(x_1|x_t)\rangle\Big\}dp_{T_\pi}(x_t)dt +$$

$$\underbrace{\frac{1}{2\epsilon}\int_0^1\int_{\mathbb{R}^D}||\int_{\mathbb{R}^D}\frac{x_1-x_t}{1-t}dp_{T_\pi}(x_1|x_t)||^2 dp_{T_\pi}(x_t)dt}_{\overset{\text{def}}{=}C'(\pi)} =$$

$$\frac{1}{2\epsilon}\int_0^1\int_{\mathbb{R}^D}\Big\{||g_v(x_t,t)||^2 - 2\langle g_v(x_t,t), \int_{\mathbb{R}^D}\frac{x_1-x_t}{1-t}dp_{T_\pi}(x_1|x_t)\rangle\Big\}dp_{T_\pi}(x_t)dt + C'(\pi) =$$

$$\frac{1}{2\epsilon}\int_0^1\int_{\mathbb{R}^D}\int_{\mathbb{R}^D}\Big\{||g_v(x_t,t)||^2 - 2\langle g_v(x_t,t), \frac{x_1-x_t}{1-t}\rangle\Big\}\underbrace{dp_{T_\pi}(x_1|x_t)dp_{T_\pi}(x_t)}_{dp_{T_\pi}(x_t,x_1)}dt + C'(\pi) =$$

$$\frac{1}{2\epsilon}\int_0^1\int_{\mathbb{R}^D}\int_{\mathbb{R}^D}\Big\{||g_v(x_t,t)||^2 - 2\langle g_v(x_t,t), \frac{x_1-x_t}{1-t}\rangle\Big\}dp_{T_\pi}(x_t,x_1)dt + C'(\pi) =$$

$$\frac{1}{2\epsilon}\int_0^1\int_{\mathbb{R}^D}\int_{\mathbb{R}^D}\Big\{||g_v(x_t,t) - \frac{x_1-x_t}{1-t}||^2\Big\}dp_{T_\pi}(x_t,x_1)dt -$$

$$\underbrace{\frac{1}{2\epsilon}\int_0^1\int_{\mathbb{R}^D}\int_{\mathbb{R}^D}||\frac{x_1-x_t}{1-t}||^2 dp_{T_\pi}(x_t,x_1)dt + C'(\pi)}_{\overset{\text{def}}{=}C''(\pi)} =$$

$$\frac{1}{2\epsilon}\int_0^1\int_{\mathbb{R}^D}\int_{\mathbb{R}^D}\Big\{||g_v(x_t,t) - \frac{x_1-x_t}{1-t}||^2\Big\}dp_{T_\pi}(x_t,x_1)dt + C''(\pi)$$

Thus,

$$\text{KL}\left(T_\pi\|S_v\right) = \underbrace{\text{KL}\left(T_\pi\|\text{proj}_{\mathcal{M}}(T_\pi)\right) + C'(T_\pi)}_{\overset{\text{def}}{=}C(\pi)} +$$

$$\frac{1}{2\epsilon}\int_0^1\int_{\mathbb{R}^D}\int_{\mathbb{R}^D}\Big\{||g_v(x_t,t) - \frac{x_1-x_t}{1-t}||^2\Big\}dp_{T_\pi}(x_t,x_1)dt =$$

$$C(\pi) + \frac{1}{2\epsilon}\int_0^1\int_{\mathbb{R}^D}\int_{\mathbb{R}^D}\Big\{||g_v(x_t,t) - \frac{x_1-x_t}{1-t}||^2\Big\}dp_{T_\pi}(x_t,x_1)dt. \tag{25}$$

Proof of Theorem 3.3. From Theorem 3.2 it follows that:

$$\text{KL}\left(T_\pi\|S_v\right) = C(\pi) + \frac{1}{2\epsilon}\int_0^1\int_{\mathbb{R}^D}\int_{\mathbb{R}^D}\Big\{||g_v(x_t,t) - \frac{x_1-x_t}{1-t}||^2\Big\}dp_{T_\pi}(x_t,x_1)dt. \tag{26}$$

In turn, from the proof of Theorem (3.1) it holds that:

$$\text{KL}\left(T_\pi\|S_v\right) = \widehat{C}(\pi) + \text{KL}\left(\pi^*\|\pi^{S_v}\right). \tag{27}$$

From the (Korotin et al., 2024, Propositon 3.1) it follows that $\text{KL}\left(\pi^*\|\pi^S_v\right) = \mathcal{L}_0(v) - \mathcal{L}^*$, where $\mathcal{L}^*$ is a constant depending on distirbutions p_0, p_1 and value ϵ. Hence, we combine these two expressions and get

$$\frac{1}{2\epsilon}\int_0^1\int_{\mathbb{R}^D}\int_{\mathbb{R}^D}\Big\{||g_v(x_t,t) - \frac{x_1-x_t}{1-t}||^2\Big\}dp_{T_\pi}(x_t,x_1)dt = \widetilde{C}(\pi) + \mathcal{L}_0(v),$$

where $\widetilde{C}(\pi) \overset{\text{def}}{=} \widehat{C}(\pi) - \mathcal{L}^* - C(\pi)$.

□

B. Experiments details and extra results

We build our LightSB-M implementation upon LightSB official implementation `https://github.com/ngushchin/LightSB`. All the parametrization, optimization and initialization details are the same as (Korotin et al., 2024) if not stated otherwise. In the Mini-batch (MB) setting, discrete OT algorithm `ot.emd` is taken from POT library (Flamary et al., 2021). The batch size is always 128.

B.1. Qualitative 2D setup hyperparameters

We use $K = 250$ potentials and Adam optimizer with $lr = 10^{-3}$ in all the cases to train LightSB-M.

B.2. Evaluation on Biological Single-cell Data.

We follow the same setup as (Korotin et al., 2024) and use their code and data from `https://github.com/ngushchin/LightSB`. All models are trained with $\epsilon = 0.1$ if not stated otherwise. For completeness, we provide additional results of our solver trained with the independent plan with different values of the parameter ϵ, see Table 3.

ϵ \ **DIM**	**50**	**100**	**1000**
0.3	2.37 ± 0.11	2.169 ± 0.11	1.310 ± 0.06
0.1	2.347 ± 0.11	2.174 ± 0.08	1.35 ± 0.05
0.03	2.349 ± 0.09	2.32 ± 0.09	1.279 ± 0.05
0.01	2.404 ± 0.12	2.28 ± 0.07	1.309 ± 0.04

Table 3: Energy distance (averaged for two setups and 5 random seeds) on the MSCI dataset (§5.3) along with 95%-confidence interval ($\pm$ intervals) for LightSB-M (ID).

B.3. Evaluation on the Schrodinger Bridge Benchmark

Here we first provide an additional evaluation of solvers using target matching and dynamic metrics. Then we study the speed of inference in our LightSB-M solver using the Brownian bridge vs. using the Euler–Maruyama simulation.

Target metric evaluation. We additionally study how well each solver map initial distribution p_0 into p_1 by measuring the metric B$\mathbb{W}_2^2$-UVP also proposed by the authors of the benchmark (Gushchin et al., 2023b, §4). We present the results in Table 4. We observe that our method performs better than other bridge-matching approaches.

		$\epsilon = 0.1$				$\epsilon = 1$				$\epsilon = 10$			
	Solver Type	$D=2$	$D=16$	$D=64$	$D=128$	$D=2$	$D=16$	$D=64$	$D=128$	$D=2$	$D=16$	$D=64$	$D=128$
Best solver on benchmark†	Varies	0.016	0.05	0.25	0.22	0.005	0.09	0.56	0.12	0.01	0.02	0.15	0.23
LightSB†	KL minimization	0.005	0.017	0.037	0.069	0.004	0.01	0.03	0.07	0.03	0.04	0.17	0.30
DSBM	Bridge matching	0.03	0.18	0.7	2.26	0.04	0.09	1.9	7.3	0.26	102	3563	15000
SF^{2}M-Sink	Bridge matching	0.04	0.18	0.39	1.1	0.07	0.3	4.5	17.7	0.17	4.7	316	812
LightSB-M (ID, **ours**)	Bridge matching	0.02	**0.03**	**0.2**	**0.46**	0.005	0.04	**0.11**	0.27	0.07	**0.03**	0.11	**0.21**
LightSB-M (MB, **ours**)	Bridge matching	**0.005**	0.07	0.27	0.63	**0.002**	0.04	0.12	0.36	0.04	0.07	0.11	0.23
LightSB-M (GT, **ours**)	Bridge matching	0.02	**0.03**	0.21	0.55	0.011	**0.03**	**0.11**	**0.26**	**0.016**	0.04	**0.09**	**0.21**

Table 4: Comparisons of B$\mathbb{W}_2^2$-UVP $\downarrow$ (%) between the ground truth target distribution p_1 and learned target distribution $\pi_\theta(x_1)$. The best metric over *bridge matching* solvers is **bolded**. Results marked with † are taken from (Korotin et al., 2024).

Dynamic metrics evaluation. Following the authors of the benchmark paper (Gushchin et al., 2023b, Appendix F), we provide additional metrics for the learned dynamic of the Schrödinger Bridge. The authors of the benchmark measure forward KL$(T^*||S)$ and reversed KL$(S||T^*)$ divergences between the ground-truth process T^* and the learned process S. To do so, they define two auxiliary values:

$$\mathcal{L}^2_{\text{fwd}}[t] = \mathbb{E}_{x_t \sim T^*} \|g^*(x_t, t) - g_S(x_t, t)\|^2, \quad \mathcal{L}^2_{\text{rev}}[t] = \mathbb{E}_{x_t \sim S} \|g^*(x_t, t) - g_S(x_t, t)\|^2,$$

and use the fact, that KL$(T^*||S) = \frac{1}{2\epsilon}\int_0^1 \mathcal{L}^2_{\text{fwd}}[t]dt$ and KL$(S||T^*) = \frac{1}{2\epsilon}\int_0^1 \mathcal{L}^2_{\text{rev}}[t]dt$. The values of $\mathcal{L}^2_{\text{fwd}}[t]$ and $\mathcal{L}^2_{\text{rev}}[t]$ for the range of $t \in [0, 1]$ are plotted in Figure 4. We observe that LightSB-M has a lower error $\mathcal{L}^2_{\text{fwd}}[t]$ and $\mathcal{L}^2_{\text{rev}}[t]$ as well as KL$(T^*||S)$ and KL$(S||T^*)$ in approximating the ground-truth optimal drift $g^*(X_t, t)$ than other algorithm including DSBM (Shi et al., 2023) and SF2M (Tong et al., 2023). Values of KL$(T^*||S)$ and KL$(S||T^*)$ are given in the Table 5. Since $\mathcal{L}^2_{\text{fwd}}[t]$ and $\mathcal{L}^2_{\text{rev}}[t]$ are lower for most times for our algorithm, KL$(T^*||S)$ and KL$(S||T^*)$ are lower for our **LightSB-M** algorithm.

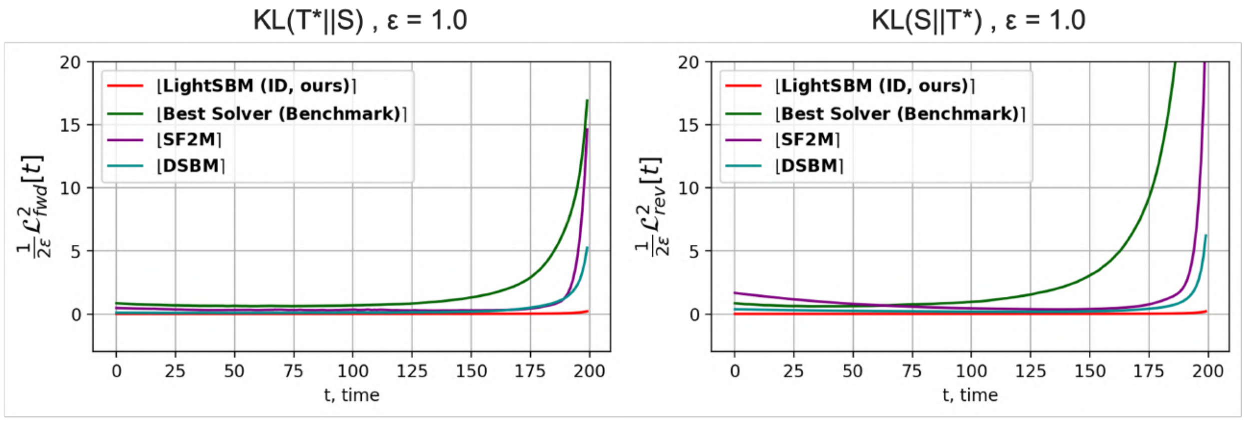


Figure 4: Dynamic KL evaluation. $\mathcal{L}^2_{\text{fwd}}[t]$ and $\mathcal{L}^2_{\text{bwd}}[t]$ values w.r.t. time for different algorithms. Results denoted as "Best solver (benchmark)" are taken from the benchmark paper (Gushchin et al., 2023b)

Solver	LightSBM (ID, ours)	Best solver (benchmark)	SF2M	DSBM
KL($T^*\|\|S$)	0.0093	1.64	0.6422	0.2950
KL($S\|\|T^*$)	0.0099	49.65	1.0765	0.39

Table 5: Dynamic KL values for different algorithms. Results denoted as "Best solver (benchmark)" are taken from the benchmark paper (Gushchin et al., 2023b).

Study of the efficiency of the sampling. Here we measure the performance of sampling (time and cB$\mathbb{W}_2^2$-UVP $\downarrow$) directly from the learned plan $\pi_\theta(x_1|x_0)$ versus sampling by the Euler-Maruyama algorithm (Kloeden et al., 1992, §9.2) and using the drift function $g_\theta(x_t, t)$. We conduct our experiments on the benchmark setup with $\epsilon = 0.1$ and $D = 16$. We present our results in the Table 6 and Table 7:

Inference type	**Time**
Euler-Maruyama, 3 steps	0.046 ± 0.053 sec
Euler-Maruyama, 10 steps	0.19 ± 0.14 sec
Euler-Maruyama, 30 steps	0.365 ± 0.08 sec
Euler-Maruyama, 100 steps	1.268 ± 0.3 sec
Euler-Maruyama, 300 steps	3.931 ± 0.34 sec
Euler-Maruyama, 1000 steps	12.61 ± 1.32 sec
Sampling from the plan π_θ	0.00058 ± 0.0001 sec

Table 6: Time measurements for LightSB-M sampling using the SDE approach (Euler-Maruyama) and direct sampling from the plan π_θ on SB Benchmark (Gushchin et al., 2023b) with $\epsilon = 0.1$ and $D = 16$. The number of steps for Euler-Maruyama is the number of SDE solver discretization steps. Results are averaged over 5 runs with std provided after $\pm$.

As we can see from the obtained results, the Euler-Maruyama approach requires up to 500 steps to accurately solve the Schrödinger Bridge SDE. Thanks to the special form of the SDE provided by the used parametrization, we can directly sample from $\pi_\theta(x_1|x_0)$. This is orders of magnitude faster than the full simulation of the trajectories.

B.4. Evaluation on unpaired image-to-image translation.

We follow the same setup as (Korotin et al., 2024) and use their code and data from `https://github.com/ngushchin/LightSB`. All models are trained with $\epsilon = 0.1$ if not stated otherwise.

According to (Korotin et al., 2024) we first split the FFHQ data into train (first 60k) and test (last 10k) images. Then we create subsets of *males*, *females*, *children* and *adults* in both train and test subsets. For training we first use the ALAE encoder to extract 512 dimensional latent vectors for each image and then train our solver on the extracted latent vectors. At

Inference type	$\text{cB}\mathbb{W}_2^2$-UVP $\downarrow$
Euler-Maruyama, 10 steps	1.53
Euler-Maruyama, 50 steps	0.22
Euler-Maruyama, 100 steps	0.126
Euler-Maruyama, 200 steps	0.102
Euler-Maruyama, 500 steps	0.09
Sampling from the plan π_θ	0.09

Table 7: $\text{cB}\mathbb{W}_2^2$-UVP $\downarrow$ measurements for LightSB-M sampling using SDE approach (Euler-Maruyama) and sampling from the plan π_θ on SB Benchmark (Gushchin et al., 2023b) with $\epsilon = 0.1$ and $D = 16$. Number of steps for Euler-Maruyama means number of SDE solver discretization steps.

the inference stage, we first extract the latent vector from the image, translate it by LightSB-M, and then decode the mapped vector to produce the mapped image. In Figure 7, we provide extra examples for our LigthSB-M and other baselines.

Test FID values. The FID values for our *man* $\rightarrow$ *woman* FFHQ image translation setup are provided in Table 8. We measure FID values between decoded translated latents and encoded-decoded true images from the FFHQ dataset. For all considered solvers, we use the same value of the coefficient $\epsilon = 0.1$, which produces moderate diversity in the generated images. The FID values are similar for all methods, which align with the good quality of images given in Figure 3.

Solver	LightSBM (ID, **ours**)	LightSBM (MB, **ours**)	DSBM	SF^{2}M-Sink
FID	0.852	0.859	0.859	0.8613

Table 8: FID values on unpaired *man* $\rightarrow$ *woman* translation for different solvers applied in the latent space of ALAE (Pidhorskyi et al., 2020) for 1024x1024 FFHQ images. (Karras et al., 2019)

Different values of ϵ. We provide extra *male* $\rightarrow$ *female* results for a wide range of values $\epsilon \in \{0.01, 0.1, 1, 10\}$) in the Figure 8 below. We observe that our solver shows the expected behavior by providing more diversity for larger ϵ.

B.5. Baselines

DSBM (Shi et al., 2023). Implementation is taken from official repo

https://github.com/yuyang-shi/dsbm-pytorch

For forward and backward drift approximations, instead of those used in the official repository, we use MLP neural networks with positional encoding as they give better results. Number of inner gradient steps for Markovian Fitting Iteration is 10000, number of Markovian Fitting Iterations is 10. Adam optimizer (Kingma & Ba, 2014) with $lr = 10^{-4}$ is used for optimization.

SF^{2}M-Sink (Tong et al., 2023). Implementation is taken from official repo

https://github.com/atong01/conditional-flow-matching

For drift and score function approximation, we use MLP neural networks with positional encoding instead of those used in the official repository, as they give better results. Number of gradient updates 50000 for SB benchmark and Single-cell Data experiments and 20000 for unpaired image-to-image translation. Adam optimizer (Kingma & Ba, 2014) with $lr = 10^{-4}$ is used for optimization.

LightSB's results are taken from the paper (Korotin et al., 2024).

C. Neural Network parametrization

Our LightSB-M is based on the Gaussian mixture parametrization. However, it is not the only way to implement optimal projection in practice. Here, we additionally propose a method to use neural network parametrization. We call this modification of our algorithm Hard Schrödinger Bridge Matching, or **HardSB-M**.

To begin with, we discuss another type of Schrödinger potential for better clarity. In the main text, we utilize the more convenient adjusted Schrödinger potential v since it simplifies the usage of Gaussian Mixture approximation. However, in the literature, the more popular way is the usage of the Schrödinger potential φ (Chen et al., 2021b, Eq. 4.11), which has a one-to-one correspondence with the adjusted Schrödinger potential v:

$$\varphi(x, t=1) = \varphi(x) = v(x)\exp(\frac{||x||^2}{2\epsilon}). \tag{28}$$

Furthermore, the drift $g(x,t)$ of the Schrödinger Bridge with potential $\varphi(x)$ is given by:

$$\begin{aligned} g(x_t,t) = \epsilon\nabla_{x_t}\log\int_{\mathbb{R}^D}\mathcal{N}(x'|x_t,(1-t)\epsilon I_D)\exp(\frac{\|x'\|^2}{2\epsilon})v(x')dx' = \\ \epsilon\nabla_{x_t}\log\int_{\mathbb{R}^D}\mathcal{N}(x'|x_t,(1-t)\epsilon I_D)\varphi(x')dx'. \end{aligned} \tag{29}$$

Below we use this non adjusted Schrödinger potential and denote $\varphi(x,t=1)$ as $\varphi(x)$ to make derivations more concise. We parametrize this potential by a neural network $\varphi_\theta(x)$ and use (29) to derive the drift g_θ given by $\varphi_\theta(x)$.

C.1. Drift Estimation

In the case of neural parametrization of φ_θ (or v_θ), computation of drift $g_\theta(x_t,t)$ (29) becomes a non-trivial task since it is no longer a convolution of a Gaussian mixture with a Gaussian distribution. We propose two ways to tackle this issue.

Variant 1. Monte Carlo (MC) estimator. First, we recap SB drift expression (29) which for parametrized Schrödinger potential φ_θ states that the drift $g_\theta(x_t,t)$ is given by:

$$g_\theta(x_t,t) = \epsilon\nabla_{x_t}\log\int_{\mathbb{R}^D}\mathcal{N}(x'|x_t,(1-t)\epsilon I_D)\varphi_\theta(x')dx'$$

By using the reparametrization trick (introducing $x' \overset{\text{def}}{=} z\sqrt{(1-t)\epsilon}+x_t$), we get

$$\begin{aligned} g_\theta(x_t,t) = \epsilon\nabla_{x_t}\log\int_{\mathbb{R}^D}\mathcal{N}(z|0,I_D)\varphi_\theta(z\sqrt{(1-t)\epsilon}+x_t)dz = \\ \epsilon\frac{\int_{\mathbb{R}^D}\nabla_{x_t}\big(\varphi_\theta(z\sqrt{(1-t)\epsilon}+x_t)\big)\mathcal{N}(z|0,I_D)dz}{\int_{\mathbb{R}^D}\big(\varphi_\theta(z\sqrt{(1-t)\epsilon}+x_t)\big)\mathcal{N}(z|0,I_D)dz} = \epsilon\frac{\mathbb{E}_{z\sim\mathcal{N}(z|0,I_D)}\big[\nabla_{x_t}\big(\varphi_\theta(z\sqrt{(1-t)\epsilon}+x_t)\big)\big]}{\mathbb{E}_{z\sim\mathcal{N}(z|0,I_D)}\big[\big(\varphi_\theta(z\sqrt{(1-t)\epsilon}+x_t)\big)\big]}. \end{aligned}$$

Then we can estimate $g_\theta(x_t,t)$ just by drawing samples $\{z\}_{n=1}^N$ and $\{z\}_{m=1}^M$ from $\mathcal{N}(z|0,I_D)$ and using:

$$g_\theta(x_t,t) \approx \frac{\frac{1}{N}\sum_{n=1}^N\nabla_{x_t}\big(\varphi_\theta(z_n\sqrt{(1-t)\epsilon}+x_t)\big)}{\frac{1}{M}\sum_{m=1}^M\big(\varphi_\theta(z_n\sqrt{(1-t)\epsilon}+x_t)\big)}$$

Calculation of the gradient of loss $\text{KL}\left(T_\pi\|S_{\varphi_\theta}\right)$ given by (15) w.r.t. the parameters is straightforward using auto-differentiation software.

Variant 2. Monte Carlo Markov Chain (MCMC) estimator. The MC estimator proposed above is biased. We also suggest a non-biased estimator based on sampling from the unnormalized density below.

Theorem C.1 (**HardSB-M drift expression**). *The drift $g(x,t)$ for the Schrodinger potential $\varphi(x)$ is given by:*

$$g(x_t,t) = \frac{1}{1-t}\big(\mathbb{E}_{x'\sim p^\varphi(x'|x_t)}[x'] - x_t\big), \tag{30}$$

where $p^\varphi(x'|x_t) \propto \exp\left(-\frac{\|x'-x_t\|^2}{2\epsilon(1-t)}\right)\varphi(x')$.

To estimate drift by Theorem C.1, one needs to sample from unnormalized density $p^\varphi(x'|x_t)$. To do this, one may use the standard Unadjusted Langevin Algorithm (ULA), also known as just Langevin Dynamics (Roberts & Tweedie, 1996).

C.2. Loss Gradient Estimation

To optimize the objective (15), one needs to compute its gradient w.r.t. the parameters θ which also involves $\nabla_\theta g_\theta(x_t, t)$. With the MC estimator for $g_\theta(x_t, t)$, the gradient $\nabla_\theta g_\theta(x_t, t)$ is trivially computed using automatic differentiation. However, with the MCMC estimator proposed in Theorem C.1 the way to compute $\nabla_\theta g_\theta(x_t, t)$ is not trivial. We propose an unbiased gradient of loss estimator via sampling from unnormalized density. For this, we need the following theorem.

Theorem C.2 (HardSB-M loss gradient expression). *The gradient of* (15) *for the Schrodinger potential* $\varphi_\theta(x)$ *is given by:*

$$\frac{1}{\epsilon}\int_0^1 \int_{\mathbb{R}^D\times\mathbb{R}^D} \Big\{(\nabla_\theta g_\theta(x_t,t))^\top (g_\theta(x_t,t) - \frac{x_1 - x_t}{1-t})\Big\} dp_{T_\pi}(x_t, x_1)dt,$$

where

$$\nabla_\theta g_\theta(x_t,t) = \frac{1}{1-t}\nabla_\theta \mathbb{E}_{p^{\varphi_\theta}(x'|x_t)}[x'].$$

In turn, $\nabla_\theta \mathbb{E}_{p^{\varphi_\theta}(x_t)}[x']$ *can be computed via*

$$\nabla_\theta \mathbb{E}_{p^{\varphi_\theta}(x'|x_t)}[x'] = \mathbb{E}_{p^{\varphi_\theta}(x'|x_t)}\Big[x'\{\nabla_\theta \log\varphi_\theta(x') - \mathbb{E}_{x''\sim p^{\varphi_\theta}(x''|x_t)}[\nabla_\theta \log\varphi_\theta(x'')]\}\Big].$$

To use this theorem in practice, we first estimate $g_\theta(x_t, t)$ by MCMC using (30) from Theorem C.1. Then, we estimate the gradient of the objective by using the Theorem C.2. At both stages, samples from $p^{\varphi_\theta}(x'|x_t)$ can be drawn, e.g., using the Unadjusted Langevin Algorithm (Roberts & Tweedie, 1996, ULA).

C.3. Inference after model training

There are several inference approaches, e.g., **Energy-Based** and **SDE Based**.

Energy based inference. We can sample directly from the EOT plan (3) using generic MCMC samplers similar to EgNOT (Mokrov et al., 2024). After sampling the end point x_1 given the start point x_0, the trajectories can be infered using self similarity property (Korotin et al., 2024, §3.2) of the Brownian Bridge $W^\epsilon_{|x_0,x_1}$.

SDE based inference. Given the way to estimate drift g_θ of the Schrödinger Bridge e.g. by MC (Appendix C.1) or MCMC (Appendix C.1) approaches, one can use any SDE solver to simulate trajectories. For example, one can use the simplest and most popular Euler-Maruyama scheme (Kloeden et al., 1992, §9.2).

One can combine these approaches by sampling an MCMC proposal used for Energy Based inference via SDE simulation.

C.4. Toy 2D experimental illustration

We use the same setups as in §5.1 with $\epsilon \in \{0.03, 0.1, 1\}$ and provide results for MC estimation in Figure 5 and for MCMC estimation in Figure 6. The Schrödinger potential $\varphi_\theta(x) : \mathbb{R}^D \to \mathbb{R}_+$ is parametrized using $\exp(\text{NN}_\theta)$, where NN_θ is a MLP. We test both MC and MCMC approaches.

Hyperparameters. For both MC and MCMC estimators, we use MLP with two hidden layers of widths $[256, 256]$ with `torch.nn.SiLU` activations as NN_θ. During training, Adam optimizer (Kingma & Ba, 2014) with $lr = 10^{-4}$ is used, batch size is 128, model is trained for 10^5 loss gradient updates.

MC estimator. During training and inference, we use 1000 MC samples. Inference is held with SDE simulation using 1000 Euler-Maruyama discretization steps for $\epsilon = 1$ and 100 Euler-Maruyama discretization steps for other ϵ. Due to the necessity to compute $\exp(\text{NN}_\theta)$ which may have very high values, we use double precision `torch.DoubleTensor` for all MC-related calculations.

MCMC estimator. During training and inference to estimate g_θ by (30) we use 100 samples drawn using Unadjusted Langevin Algorithm (ULA) with 50 steps and step size $\eta = 0.001$. The inference was performed in two steps: first, the SDE simulation was performed with g_θ estimation by ULA, and then the result was used as a proposal for energy-based sampling from the EOT plan, (3). SDE simulaion was held using 100 Euler-Maruyama discretization steps (ULA settings are the same as for training) and Energy Based sampling from EOT plan using ULA with with 1000 steps and step size $\eta = 10^{-4}$.

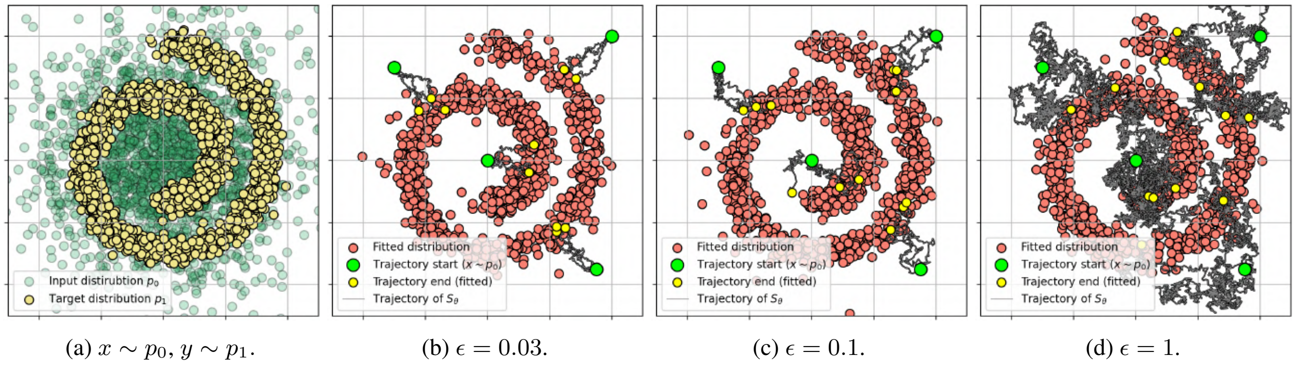


Figure 5: The process S_θ learned by HardSB-M (**ours**) with MC drift estimator *Gaussian→Swiss roll* example.

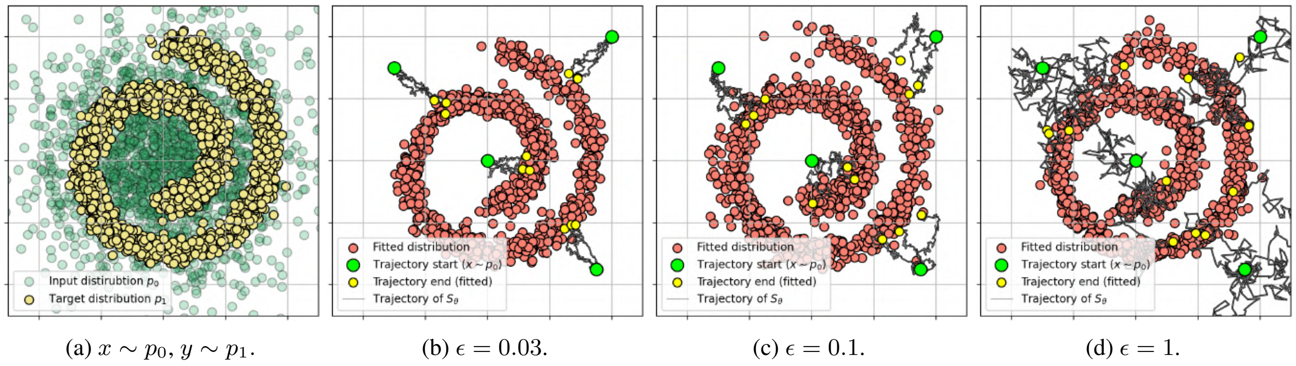


Figure 6: The process S_θ learned by HardSB-M (**ours**) with MCMC drift estimator *Gaussian→Swiss roll* example.

C.5. Proofs

Proof of Theorem C.1. We denote by $Z_{x_t,(1-t)\epsilon} \overset{\text{def}}{=} \int_{\mathbb{R}^D} \exp\big(-\frac{\|x'-x_t\|^2}{2(1-t)\epsilon}\big)dx'$ the normalization constant of the normal distribution $\mathcal{N}(x'|x_t,(1-t)\epsilon I_D)$. For a potential $\varphi_\theta(x)$, the corresponding drift $g(x_t,t)$ is given by (29):

$$g(x_t,t) = \epsilon\nabla_{x_t}\log\int_{\mathbb{R}^D}\mathcal{N}(x'|x_t,(1-t)\epsilon I_D)\varphi(x')dx'$$

We will proceed with this equality to obtain an unbiased estimator. We proceed as follows:

$$g(x_t,t) = \epsilon\nabla_{x_t}\log\int_{\mathbb{R}^D}\mathcal{N}(x'|x_t,(1-t))\varphi(x')dx' =$$
$$\epsilon\nabla_{x_t}\log\int_{\mathbb{R}^D}\frac{1}{Z_{x_t,(1-t)\epsilon}}\exp\big(-\frac{\|x'-x_t\|^2}{2(1-t)\epsilon}\big)\varphi(x')dx' =$$
$$\epsilon\frac{\nabla_{x_t}\int_{\mathbb{R}^D}\frac{1}{Z_{x_t,(1-t)\epsilon}}\exp\big(-\frac{\|x'-x_t\|^2}{2(1-t)\epsilon}\big)\varphi(x')dx'}{\int_{\mathbb{R}^D}\frac{1}{Z_{x_t,(1-t)\epsilon}}\exp\big(-\frac{\|x''-x_t\|^2}{2(1-t)\epsilon}\big)\varphi(x'')dx''} =$$
$$\epsilon\frac{\nabla_{x_t}\int_{\mathbb{R}^D}\exp\big(-\frac{\|x'-x_t\|^2}{2(1-t)\epsilon}\big)\varphi(x')dx'}{\int_{\mathbb{R}^D}\exp\big(-\frac{\|x''-x_t\|^2}{2(1-t)\epsilon}\big)\varphi(x'')dx''} =$$
$$\epsilon\frac{\int_{\mathbb{R}^D}\nabla_{x_t}\Big\{\exp\big(-\frac{\|x'-x_t\|^2}{2(1-t)\epsilon}\big)\Big\}\varphi(x')dx'}{\int_{\mathbb{R}^D}\exp\big(-\frac{\|x''-x_t\|^2}{2(1-t)\epsilon}\big)\varphi(x'')dx''} = \Big[\nabla_x f(x) = f(x)\nabla_x\log f(x)\Big] = \tag{31}$$
$$\epsilon\int_{\mathbb{R}^D}\nabla_{x_t}\{\frac{-\|x'-x_t\|^2}{2(1-t)\epsilon}\}\frac{\exp\big(-\frac{\|x'-x_t\|^2}{2(1-t)\epsilon}\big)\varphi(x')dx'}{\int_{\mathbb{R}^D}\exp\big(-\frac{\|x''-x_t\|^2}{2(1-t)\epsilon}\big)\varphi(x'')dx''} =$$
$$\Big[p^\varphi(x'|x_t) \propto \exp\big(-\frac{\|x'-x_t\|^2}{2\epsilon(1-t)}\big)\varphi(x')\Big] =$$

$$\epsilon\int_{\mathbb{R}^D}\nabla_{x_t}\{\frac{-\|x'-x_t\|^2}{2(1-t)\epsilon}\}p^{\varphi}(x'|x_t)dx' =$$
$$\epsilon\mathbb{E}_{x'\sim p^{\varphi}(x'|x_t)}\big[\frac{x'-x_t}{(1-t)\epsilon}\big] = \frac{1}{(1-t)}\big(\mathbb{E}_{x'\sim p^{\varphi}(x'|x_t)}[x'] - x_t\big). \tag{32}$$

In line (31), we use log-derivatve trick. □

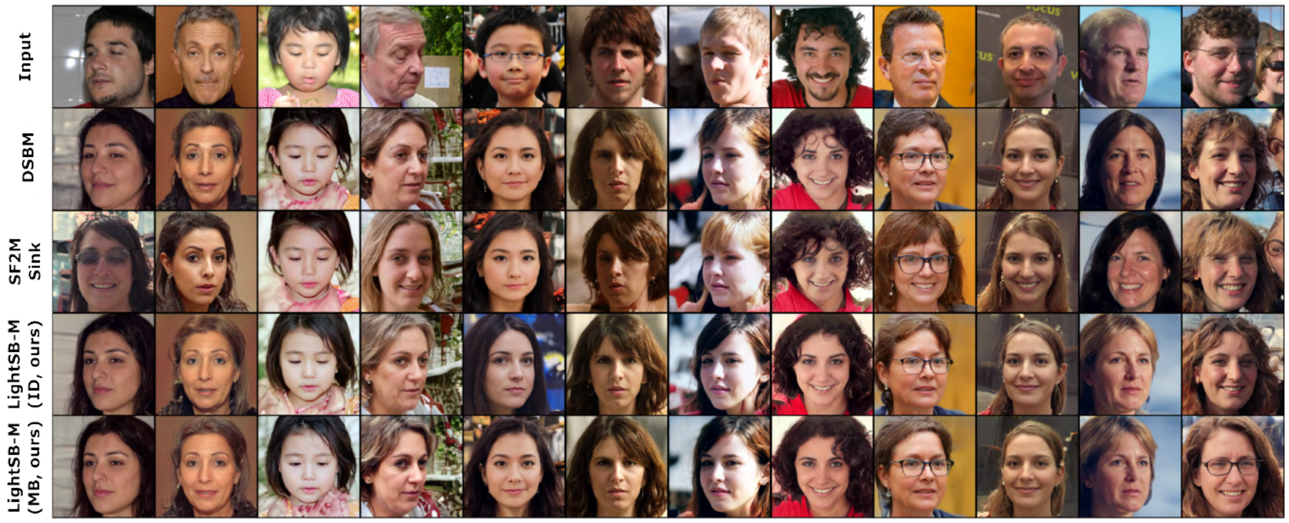
Proof of Theorem C.2. We derive

$$\nabla_\theta\frac{1}{\epsilon}\int_0^1\int_{\mathbb{R}^D\times\mathbb{R}^D}\|g_\theta(x_t,t)-\frac{x_1-x_t}{1-t}\|^2dp_{T_\pi(x_t,x_1)}dt =$$
$$\frac{1}{\epsilon}\int_0^1\int_{\mathbb{R}^D\times\mathbb{R}^D}\Big\{(\nabla_\theta g_\theta(x_t,t))^\top(g_\theta(x_t,t)-\frac{x_1-x_t}{1-t})\Big\}dp_{T_\pi(x_t,x_1)}dt. \tag{33}$$

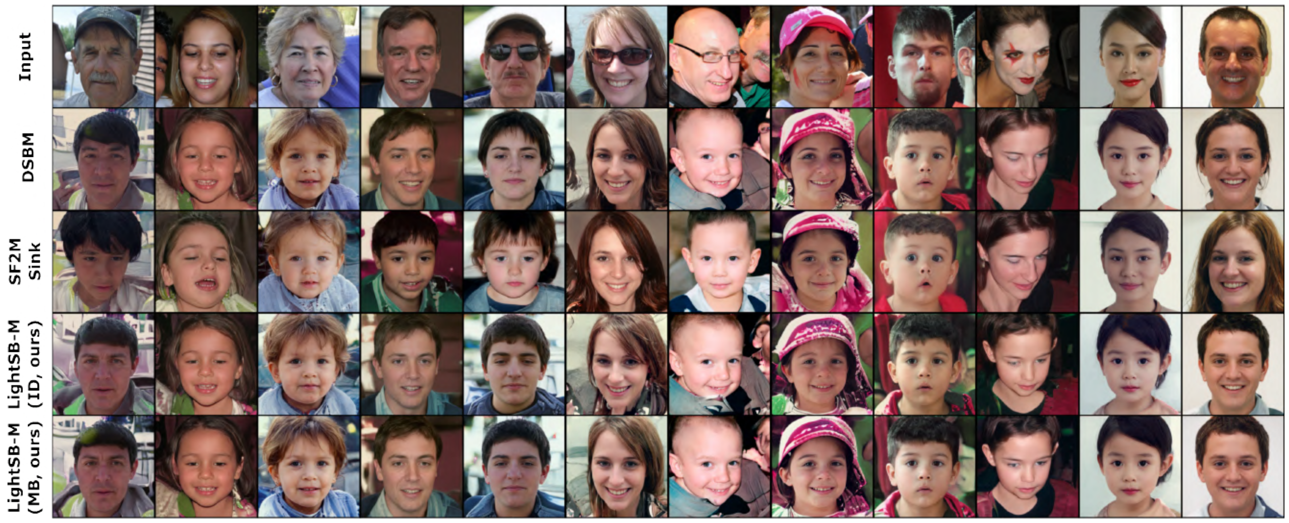
Now we recap the result of Theorem C.1: $g_\theta(x,t)=\frac{1}{(1-t)}\big(\mathbb{E}_{x'\sim p^{\varphi_\theta}(x'|x_t)}[x']-x_t\big)$. Now we derive $\nabla_\theta g_\theta(x,t)$:

$$\nabla_\theta g_\theta(x_t,t) = \frac{1}{(1-t)}\big(\nabla_\theta\mathbb{E}_{x'\sim p^{\varphi_\theta}(x'|x_t)}[x']\big) = \big[Z(x_t,\varphi_\theta)\overset{\text{def}}{=}\int_{\mathbb{R}^D}\exp\big(-\frac{\|x'-x_t\|^2}{2(1-t)\epsilon}\big)\varphi_\theta(x')dx\big] =$$
$$= \frac{1}{(1-t)}\big(\nabla_\theta\int_{\mathbb{R}^D}x'\frac{\exp\big(-\frac{\|x'-x_t\|^2}{2(1-t)\epsilon}\big)\varphi_\theta(x')}{Z(x_t,\varphi_\theta)}dx'\big) =$$
$$= \frac{1}{(1-t)}\big(\int_{\mathbb{R}^D}x'\nabla_\theta\{\frac{\exp\big(-\frac{\|x'-x_t\|^2}{2(1-t)\epsilon}\big)\varphi_\theta(x')}{Z(x_t,\varphi_\theta)}\}dx'\big) = \Big[\nabla_\theta f_\theta(\cdot)=f_\theta(\cdot)\nabla_\theta\log f_\theta(\cdot)\Big] = \tag{34}$$
$$= \frac{1}{(1-t)}\big(\nabla_\theta\mathbb{E}_{x'\sim p^{\varphi_\theta}(x'|x_t)}\big[x'\nabla_\theta(-\frac{\|x'-x_t\|^2}{2(1-t)\epsilon}+\log\varphi_\theta(x')-\log Z(x_t,\varphi_\theta))\big]\big) =$$
$$= \frac{1}{(1-t)}\big(\nabla_\theta\mathbb{E}_{x'\sim p^{\varphi_\theta}(x'|x_t)}\big[x'(\nabla_\theta\{\log\varphi_\theta(x')\}-\nabla_\theta\log Z(x_t,\varphi_\theta))\big]\big) =$$
$$= \frac{1}{(1-t)}\big(\nabla_\theta\mathbb{E}_{x'\sim p^{\varphi_\theta}(x'|x_t)}\big[x'\nabla_\theta\big(\{\log\varphi_\theta(x')\}-\frac{\int_{\mathbb{R}^D}\nabla_\theta\big(\exp\{-\frac{\|x''-x_t\|^2}{2\epsilon(1-t)}\}\varphi_\theta(x'')\big)dx''}{Z(x_t,\varphi_\theta)})\big]\big) =$$
$$= \Big[\nabla_\theta f_\theta(\cdot)=f_\theta(\cdot)\nabla_\theta\log f_\theta(\cdot)\Big] = \tag{35}$$
$$= \frac{1}{(1-t)}\big(\nabla_\theta\mathbb{E}_{x'\sim p^{\varphi_\theta}(x'|x_t)}\big[x'\big(\nabla_\theta\{\log\varphi_\theta(x')\}-\frac{\int_{\mathbb{R}^D}\exp\{-\frac{\|x''-x_t\|^2}{2\epsilon(1-t)}\}\varphi_\theta(x'')\nabla_\theta\log\varphi_\theta(x'')dx''}{Z(x_t,\varphi_\theta)})\big]\big) =$$
$$= \frac{1}{(1-t)}\big(\nabla_\theta\mathbb{E}_{x'\sim p^{\varphi_\theta}(x'|x_t)}\big[x'\big(\nabla_\theta\log\varphi_\theta(x')-\mathbb{E}_{x''\sim p^{\varphi_\theta}(x''|x_t)}\big[\nabla_\theta\log\varphi_\theta(x'')\big]\big]\big)\big). \tag{36}$$

In lines (34) and (35), we use the log-derivative trick. □



(a) *Man* $\rightarrow$ *Woman*.



(b) *Adult* $\rightarrow$ *Child*.

Figure 7: Additional examples of image-to-image translation.

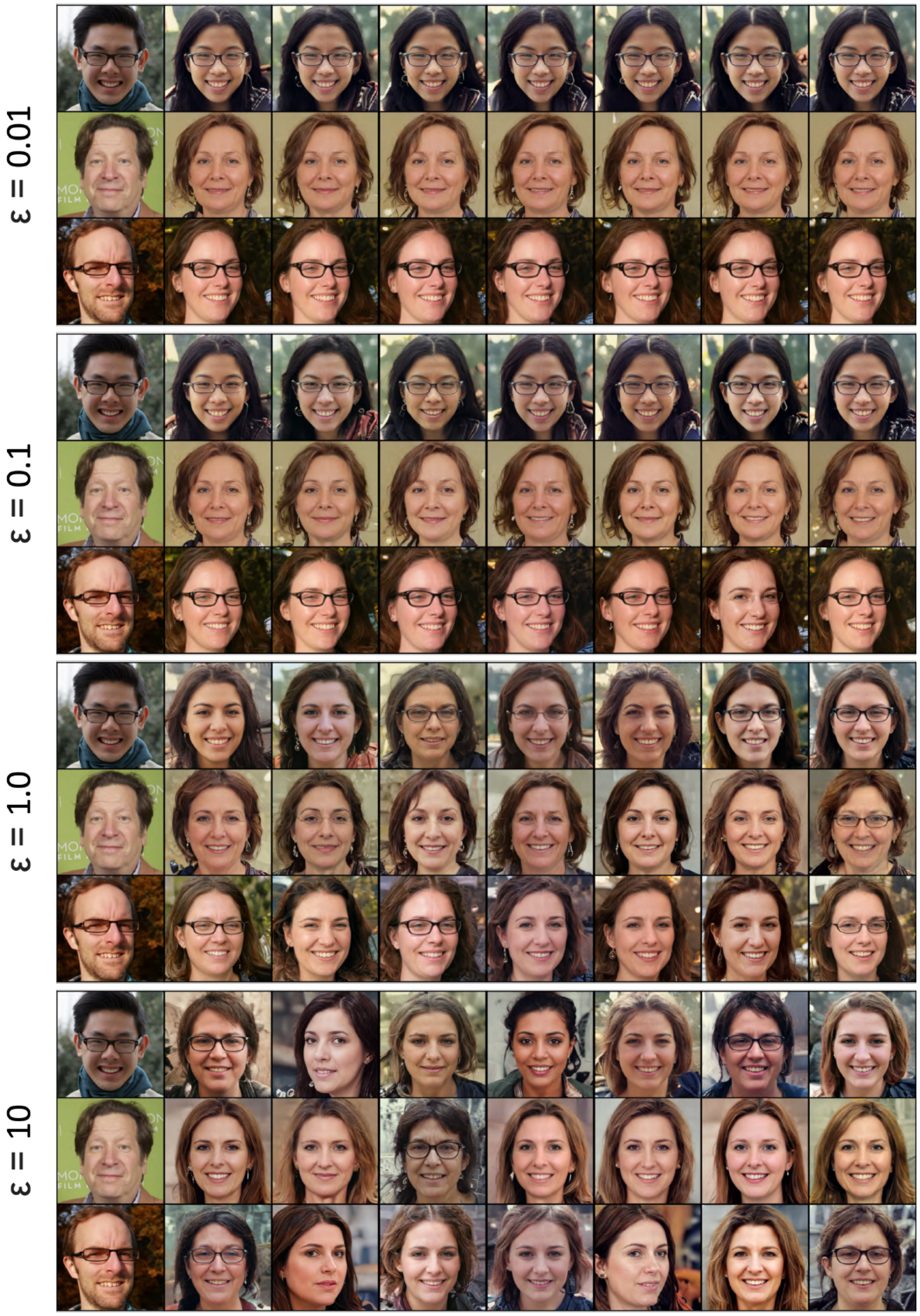


Figure 8: Image-to-image experiments with $\epsilon \in \{0.01, 0.1, 1, 10\}$.