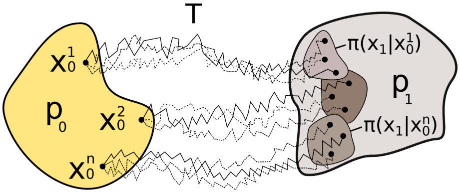



# Building Light Schrödinger Bridges

**Alexander Korotin**
[1]Skolkovo Institute of Science and Technology
[2]Artificial Intelligence Research Institute

Moscow, 2024

# Overview

# Modern Diffusion Models and Their Limitations

# Modern Generative Models for Images

**Text prompt:** woman's transparent futuristic inspired sneakers, glitter, depth of field

KANDINSKY

**Text prompt:** Chicken with potatoes baked in mayonnaise-sour cream sauce

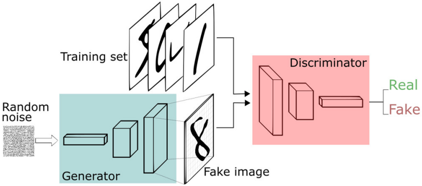
SHEDEVRUM

**Text prompt:** 1967 Dodge Charger, moody lighting, side view, black, front view, lobby of the Louvre ...

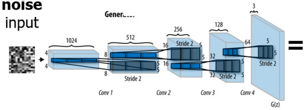
MIDJOURNEY

# Principal Approaches to Generative Modeling

## Adversarial models (GANs, 2014)




Random **noise** input

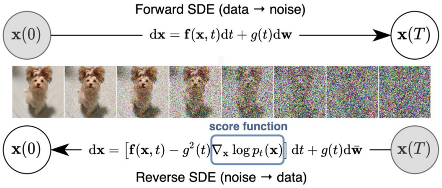



=

Deep **neural network** (generator)

**Generated** images (these people do not exist!)

## Diffusion Models (DM, 2019)




# Diffusion Models[1,2]

MAIN IDEA: reverse the data noising process.

## Forward diffusion (noising SDE)

Take a data distribution $x_0 \sim p_0$ and gradually turn it to noise distribution $x_T \sim p_T = \mathcal{N}(0, \sigma^2 I)$.

$x_0 \rightarrow x_1 \rightarrow x_2 \rightarrow \quad .... \quad \rightarrow x_T$

$$dx_t = f(x_t, t)dt + g(t)dW_t$$

(e.g., $dx_t = -\frac{1}{2}\beta_t dt + \sqrt{\beta_t}dW_t$)

## Reverse diffusion (denoising SDE)

Sample from noise distribution $x_T \sim p_T$ and reverse the diffusion to get $x_0 \sim p_0$:

$x_T \rightarrow x_{T-1} \rightarrow \quad .... \quad x_1 \rightarrow x_0$

$$dx_t = [f(x_t, t) - g^2(t)\nabla_x \log p(x_t, t)]dt + g(t)d\overline{W_t}$$

(or $dx_t = [f(x_t, t) - \frac{1}{2}g^2(t)\nabla_x \log p(x_t, t)]dt$)

[1] Jonathan Ho, Ajay Jain, and Pieter Abbeel (2020). "Denoising diffusion probabilistic models". In: *Advances in neural information processing systems* 33, pp. 6840–6851.

[2] Yang Song et al. (2020). "Score-Based Generative Modeling through Stochastic Differential Equations". In: *International Conference on Learning Representations*.

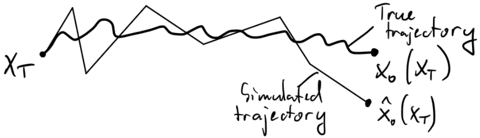
# Limitation 1 of Diffusion Models: Time-Consuming Inference

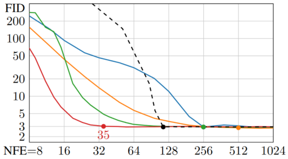
To simulate the denoising process:

$$x_t = \left[f(x_t, t) - g^2(t)\nabla_x \log p(x_t, t)\right] t + g(t)\overline{W}_t$$

one uses the discretization (e.g., Euler-Maruyama simulation):

$$x_{t-\Delta t} = x_t - \left[f(x_t, t) - g^2(t)\nabla_x \log p(x_t, t)\right]\Delta t + g(t)\sqrt{\Delta t}\xi_t \quad , \xi_t \sim \mathcal{N}(0, I).$$




Diff. models performance, CIFAR-10. FID w.r.t NFE.

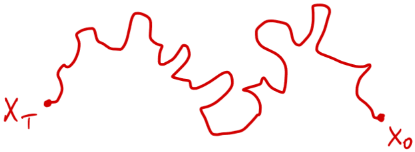
Remark:

NFE (# function evaluations) $\equiv$ (# discretization steps)

# Desire 1: Straightening The Trajectories of Diffusion Models

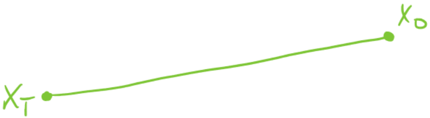
What we have

Not straight (deterministic or stochastic) trajectories, which are HARD to simulate.



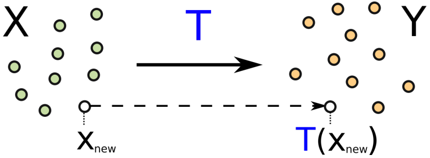

What we want

Straight (deterministic?) trajectories, which are EASY to simulate.



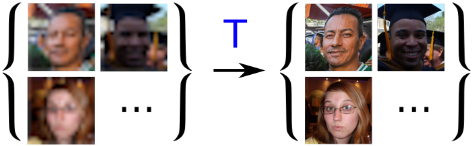
# Limitation 2: Inapplicability to (Unpaired) Domain Translation

**The task**: learn (from samples) a *translation* map between the two given data domains.

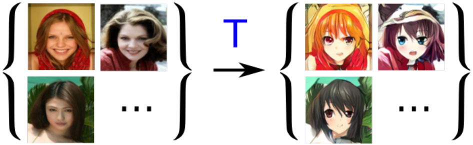



**Important:** the map should generalize to new data (similar to the train set).

**Example 1:** Image Super-Resolution




**Example 2:** Style Translation

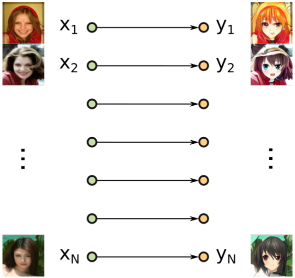


# Desire 2: Be able to Solve Unpaired Domain Translation with DMs[3]

**Supervised**

*Paired* train samples are available:

$$\{(x_1, y_1), \ldots, (x_N, y_N)\}.$$

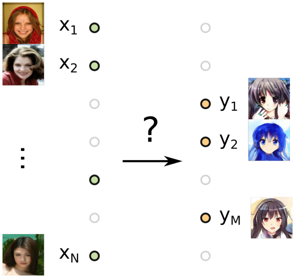



Conditional DMs are applicable.

**Unsupervised** (our interest)

Only *unpaired* train samples are given:

$$\{x_1, \ldots, x_N\}, \{y_1, \ldots, y_M\}.$$



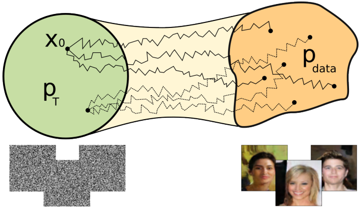

DMs are not applicable.

[3] Jun-Yan Zhu et al. (2017). "Unpaired image-to-image translation using cycle-consistent adversarial networks". In: *Proceedings of the IEEE international conference on computer vision*, pp. 2223–2232.

# Schrödinger Bridges vs. Diffusion Models: Key Differences

## Diffusion models framework (2019)

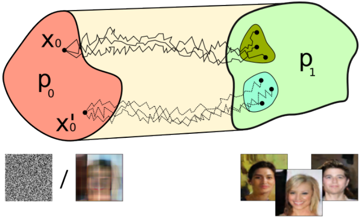
- maps given complex data distribution to the normal distribution.




- uses pre-defined noising process and learns the de-noising process.
- requires infinite time horizon $[0, T]$.

## Schrödinger bridge framework (2021)

- maps arbitrary distribution $p_0$ to arbitrary distribution $p_1$.




- learns a diffusion that is maximally similar to a given prior process.
- finite time horizon $[0, \underset{T}{\underset{\parallel}{1}}]$.

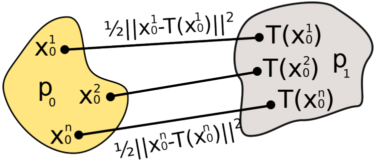
# Optimal Transport and Schrödinger Bridges

# Monge's Formulation of Optimal Transport[4] (with the Quadratic Cost)

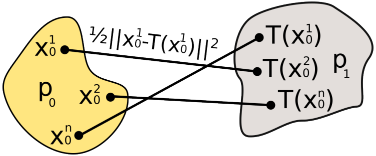
The optimal transport **cost** between distributions $p_0, p_1 \in \mathcal{P}_{2,ac}(\mathbb{R}^D)$ is

$$\mathrm{Cost}(p_0, p_1) = \inf_{T \sharp p_0 = p_1} \int_{\mathcal{X}} \frac{\|x_0 - T(x_0)\|^2}{2} p_0(x_0) dx_0.$$

The map $T^*$ attaining the minimum is called the optimal **transport map**.




Optimal map



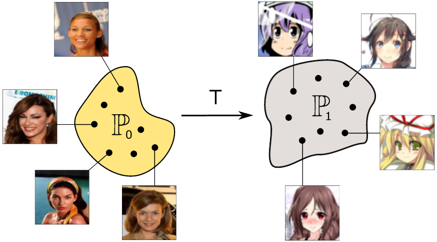

Not optimal map

[4]Cédric Villani (2008). *Optimal transport: old and new*. Vol. 338. Springer Science & Business Media.

# Optimal transport applications

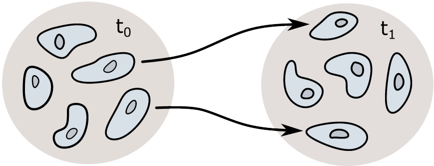
**Domain Translation.**[2]

By considering two unpaired image datasets as samples from $p_0$ and $p_1$, OT learns a map between datasets that preserves content.




**Single-Cell (SC) Biological data.**[3]

SC technology determines the gene expression profile of each measured cell, but destroys all measured cells. OT learns a map between cell populations before and after the perturbation.




[5]Alexander Korotin, Daniil Selikhanovych, and Evgeny Burnaev (2022). "Neural Optimal Transport". In: *The Eleventh International Conference on Learning Representations*.

[6]Charlotte Bunne et al. (2023). "Learning single-cell perturbation responses using neural optimal transport". In: *Nature Methods*, pp. 1–10.

# Entropic Optimal Transport (OT)[7]

Consider two distributions $p_0, p_1 \in \mathcal{P}_{2,ac}(\mathbb{R}^D)$.
**Entropic OT (EOT)** is formulated as follows:

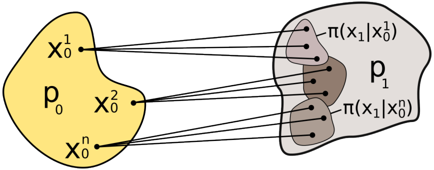
$$\inf_{\pi \in \Pi(p_0,p_1)} \int_{\mathbb{R}^D} C\big(x_0, \pi(\cdot|x_0)\big) p_0(x_0) dx_0.$$

The minimizer $\pi^*$ is called the Entropic OT plan.

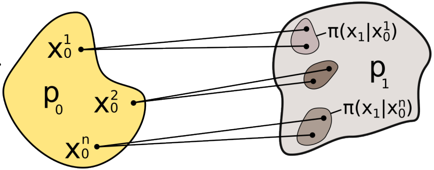
$$C\big(x_0, \pi(\cdot|x_0)\big) \overset{def}{=} \underbrace{\int_{\mathbb{R}^D} \frac{\|x_0 - x_1\|^2}{2}}_{\text{Dissimilarity}} \pi(x_1|x_0) dx_1 - \epsilon \underbrace{H(\pi(\cdot|x_0))}_{\text{Diversity}}.$$

Regularization strength $\epsilon$ controls the diversity.

- $\Pi(p_0, p_1)$ are distributions on $\mathbb{R}^D \times \mathbb{R}^D$ with marginals $p_0$, $p_1$




Stochastic EOT maps for large $\epsilon$.




Stochastic EOT maps for small $\epsilon$.

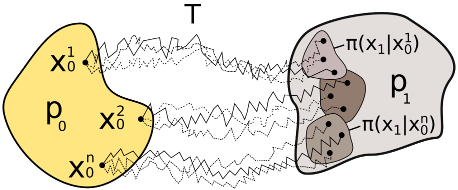
[7] Marco Cuturi (2013). "Sinkhorn distances: Lightspeed computation of optimal transport". In: *Advances in neural information processing systems* 26.

# Schrödinger Bridge (SB) problem[8]

Consider two distributions $p_0, p_1 \in \mathcal{P}_{2,ac}(\mathbb{R}^D)$.
The Schrödinger bridge problem is:

$$\inf_{T \in \mathcal{F}(p_0,p_1)} \mathrm{KL}(T \| W^{\epsilon}),$$

- $\mathcal{F}(p_0, p_1)$ are stochastic processes with marginals $p_0$, $p_1$ at $t = 0$ and $t = 1$ respectively.
- $W^{\epsilon}$ is the Wiener process with the variance $\epsilon$.

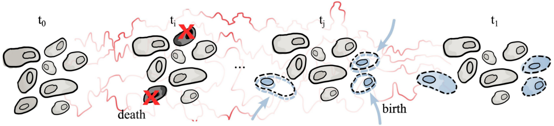



The process $T^*$ attaining the minimum has joint distribution $\pi^{T^*} = \pi^*$ at time moments $t = 0, 1$ which is the solution to the Entropic OT with regularization parameter $\epsilon$.

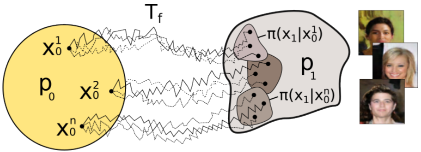
[8] Erwin Schrödinger (1931). *Über die umkehrung der naturgesetze*. Verlag der Akademie der Wissenschaften in Kommission bei Walter De Gruyter u. Company, 1931.

# Applications of Schrödinger Bridge

**Single-cell biological data.**[9]
Solving SB allows to reconstruct the most likely cell trajectories.




**Generation and Domain Translation.**[10]
Solving SB between noise and data with small $\epsilon$ gives diffusion with "straighter" trajectories.



$$T_f : dX_t = f(X_t, t)dt + \sqrt{\epsilon} dW_t, \quad X_0 \sim p_0,$$

[9] Hugo Lavenant et al. (2024). "Toward a mathematical theory of trajectory inference". In: *The Annals of Applied Probability* 34.1A, pp. 428–500. DOI: 10.1214/23-AAP1969.

[10] Valentin De Bortoli et al. (2021). "Diffusion schrödinger bridge with applications to score-based generative modeling". In: *Advances in Neural Information Processing Systems* 34, pp. 17695–17709.

# Part I. Light Schrödinger Bridge (ICLR 2024)

# Light SB outline.

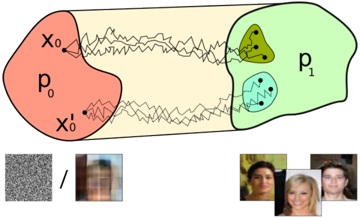
1. Motivation of light SB solvers.
2. Equivalence of SB and EOT problems.
3. Characterisation of SB and EOT solutions.
4. Derivation of the LightSB functional.
5. Gaussian mixture parameterization of Schrödinger Bridges.
6. LightSB training and inference.
7. Experimental Illustrations.

# Motivation of light SB solvers

**Expectation.**

We solve the Scrhödinger Bridge, and it

- maps arbitrary distribution $p_0$ to arbitrary distribution $p_1$.




- provides a diffusion that is maximally similar to a given prior process.

**Reality.**

It is hard to solve the Schrödinger Bridge.

- Many neural-network-based algorithms, almost all of which are poorly scalable and require painful iterative or adversarial learning.
- Absence of simple baseline algorithm, which works fast, provably solves Schrödinger Bridge in moderate dimensions and does not require time-consuming hyperparameter selection.

**With this in mind, we started to search for possible solutions.**

# List of key existing not light SB solvers

See the following **benchmark paper** for a survey of the field in 2023:

- Nikita Gushchin, Alexander Kolesov, Petr Mokrov, et al. (2023). "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*. URL: https://openreview.net/forum?id=OHimIaixXk

A (not comprehensive) list of related works is as follows:

1. **MLE-SB:** Francisco Vargas et al. (2021). "Solving schrödinger bridges via maximum likelihood". In: *Entropy* 23.9, p. 1134
2. **DSB**: Valentin De Bortoli et al. (2021). "Diffusion schrödinger bridge with applications to score-based generative modeling". In: *Advances in Neural Information Processing Systems* 34, pp. 17695–17709
3. **ENOT**: Nikita Gushchin, Alexander Kolesov, Alexander Korotin, et al. (2024). "Entropic neural optimal transport via diffusion processes". In: *Advances in Neural Information Processing Systems* 36
4. **FB-SDE:** Tianrong Chen, Guan-Horng Liu, and Evangelos Theodorou (2022). "Likelihood Training of Schrödinger Bridge using Forward-Backward SDEs Theory". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=nioAdKCEdXB
5. **DSBM:** Yuyang Shi et al. (2023). "Diffusion Schrödinger Bridge Matching". In: *Thirty-seventh Conference on Neural Information Processing Systems*. URL: https://openreview.net/forum?id=qy07OHsJT5
6. **ASBM:** Nikita Gushchin, Daniil Selikhanovych, et al. (2024). "Adversarial Schrödinger Bridge Matching". In: *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. URL: https://openreview.net/forum?id=L3Knnigicu

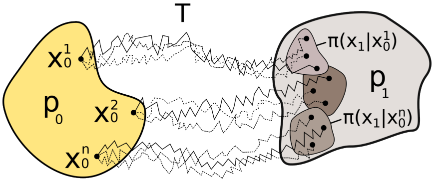
# Schrodinger Bridge formulation[11]

**The Schrödinger Bridge problem**

For two continuous distributions $p_0$ and $p_1$ on $\mathbb{R}^D$, the Schrödinger bridge problem is:

$$\inf_{T \in \mathcal{F}(p_0, p_1)} \mathrm{KL}(T \| W^{\epsilon}).$$

Here $\mathcal{F}(p_0, p_1)$ are stochastic processes with marginals $p_0$, $p_1$ at $t = 0$ and $t = 1$.




Here $W^{\epsilon}$ wiener process with the variance $\epsilon$, i.e., it is a stochastic process with the stochastic differential equation (SDE): $dX_t = \sqrt{\epsilon} dW_t$.

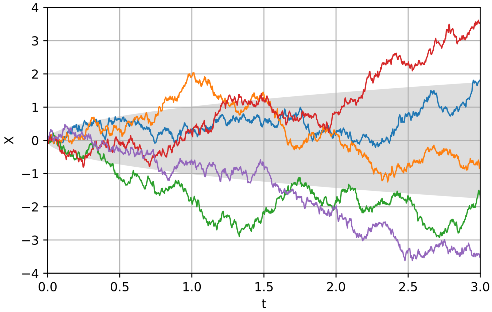



**Figure 1:** Wiener process with $\epsilon = 1$.

[11]Yongxin Chen, Tryphon T Georgiou, and Michele Pavon (2016). "On the relation between optimal transport and Schrödinger bridges: A stochastic control viewpoint". In: *Journal of Optimization Theory and Applications* 169, pp. 671–691.

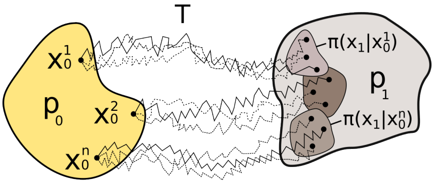
# Decomposition of SB on inner and outer parts

**Schrödinger Bridge formulation.**

For two continuous distributions $p_0$ and $p_1$ on $\mathbb{R}^D$, the Schrödinger bridge problem is:

$$\inf_{T \in \mathcal{F}(p_0, p_1)} \mathrm{KL}(T \| W^\epsilon).$$

Here $\mathcal{F}(p_0, p_1)$ are stochastic processes with marginals $p_0$, $p_1$ at $t = 0$ and $t = 1$. $W^\epsilon$ is a Wiener process with the variance $\epsilon$.




Let $\pi^T$ denote the joint distribution of a stochastic process $T$ at time moments $t = 0, 1$.

Let $T_{|x,y}$ denote the stochastic processes $T$ conditioned on values $x, y$ at times $t = 0, 1$, respectively.

We can expand the functional as follows:

$$\mathrm{KL}(T || W^\epsilon) = \underbrace{\mathrm{KL}(\pi^T || \pi^{W^\epsilon})}_{\text{outer part}} +$$

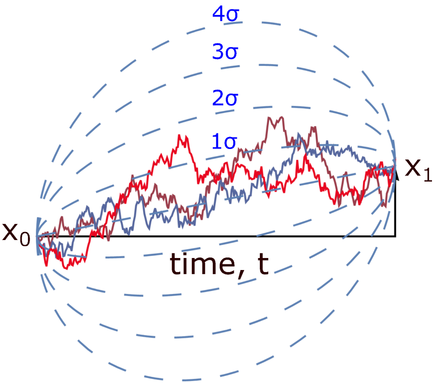
$$\underbrace{\int \mathrm{KL}(T_{|x_0,x_1} || W^\epsilon_{|x_0,x_1}) d\pi^T(x_0, x_1)}_{\text{inner part}}.$$

Here $W^\epsilon_{|x_0,x_1}$ is a Wiener processconditioned on its end and start points. It is known as the **Brownian Bridge**.

# Schrödinger Bridge is a reciprocal process

**Brownian Bridges**

The process $W^{\epsilon}_{|x_0,x_1}$ is a Brownian Bridge. It is a Gaussian process starting at $x_0$ and ending at $x_1$.




We can set to zero the inner part by searching process in the form of a mixture of Brownian Bridges, i.e. $T = \int W^{\epsilon}_{|x_0,x_1} d\pi^T(x_0, x_1)$.

Such processes form reciprocal class, and for brevity, we just call them reciprocal processes.

In this case:

$$\mathrm{KL}(T||W^{\epsilon}) = \underbrace{\mathrm{KL}(\pi^T||\pi^{W^{\epsilon}})}_{\text{outer part}} + \underbrace{\int \mathrm{KL}(T_{|x_0,x_1}||W^{\epsilon}_{|x_0,x_1}) d\pi^T(x_0, x_1)}_{=0, \text{since } T_{|x_0,x_1} = W^{\epsilon}_{|x_0,x_1}}.$$

# Equivalence between EOT and SB

For a reciprocal $T$, the objective is

$$\mathrm{KL}(T||W^{\epsilon}) = \underbrace{\mathrm{KL}(\pi^{T}||\pi^{W^{\epsilon}})}_{\text{outer part}}.$$

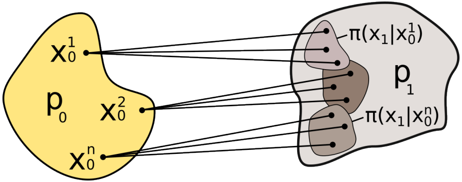
Hence,

$$\inf_{T\in\mathcal{F}(p_0,p_1)} \mathrm{KL}(T||W^{\epsilon}) = \inf_{T\in\mathcal{F}(p_0,p_1)} \mathrm{KL}(\pi^{T}||\pi^{W^{\epsilon}}).$$

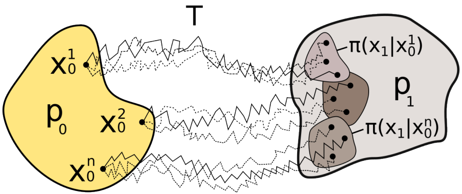
By expanding the outer part, we obtain:

$$\mathrm{KL}(\pi^{T}||\pi^{W^{\epsilon}}) =$$

$$\underbrace{\int_{\mathcal{X}\times\mathcal{Y}} \frac{||x-y||^2}{2\epsilon} d\pi^{T}(x,y) - H(\pi^{T})}_{\text{equivalent to EOT functional.}} + C.$$




Entropic OT.




Schrödinger Bridge.

# Characterization for EOT and SB solutions.

**Moreover, both solutions for EOT and SB problems are characterized by the starting distribution $p_0$ and one scalar-valued function $v^*$.**

**EOT solution.**
The EOT solution $\pi^*$ can be represented through the input density $p_0$ and a function $v^* : \mathbb{R}^D \to \mathbb{R}_+$:

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

where $c_{v^*}(x_0) = \int_{\mathbb{R}^D} \exp\Big(\langle x_0, x_1\rangle\epsilon\Big) v^*(x_1) dy$.

Here $v^*$ is the **adjusted Schrödinger potential.**

**SB solution.**
The solution $T^*$ for the Schrödinger Bridge is a Markovian process given by the following SDE:

$$dX_t = g^*(X_t, t)dt + \sqrt{\epsilon} dW_t^{\epsilon}, \quad X_0 \sim p_0$$

In turn, the optimal drift $g^*(x_t, t)$ is given by:

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

i.e., is a convolution with the adjusted potential $v^*$.

# Theoretical summary.

**Equivalence of EOT and SB problems.**

We can solve SB by solving the related EOT problem since:

$$\inf_{T \in \mathcal{F}(p_0,p_1)} \mathrm{KL}(T||W^\epsilon) = \inf_{T \in \mathcal{F}(p_0,p_1)} \mathrm{KL}(\pi^T||\pi^{W^\epsilon}) = \inf_{\pi \in \Pi(p_0,p_1)} \mathrm{KL}(\pi||\pi^{W^\epsilon}),$$

where $\Pi(p_0, p_1)$ is a set of joint distributions on $t = 0$ and $t = 1$ with marginals $p_0$ and $p_1$.

**Optimal form of the solution.**

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

Still not obvious how to solve. The EOT problem:

$$\inf_{\pi \in \Pi(p_0,p_1)} \mathrm{KL}(\pi||\pi^{W^\epsilon}) = \inf_{\pi \in \Pi(p_0,p_1)} \int_{\mathbb{R}^D \times \mathbb{R}^D} \frac{||x-y||^2}{2\epsilon} d\pi^T(x,y) - H(\pi^T) + C.$$

is a constrained optimization problem, and we do not know how to parametrize a set $\Pi(p_0, p_1)$.

# Direct optimization of KL with the solution.

**Our new objective:**

Instead of trying to solve the constrained optimization problem of EOT, let's just minimize KL with the solution $\pi^*$:

$$\underbrace{\arg\min_{\pi \in \Pi(p_0,p_1)} \mathrm{KL}(\pi || \pi^{W^\epsilon})}_{\text{constrained optimization}} \rightarrow \underbrace{\arg\min_{\pi} \mathrm{KL}(\pi^* || \pi)}_{\text{unconstrained optimization}}$$

The problem: we do not know $\pi^*$.

**Our proposed optimal form parametrization:**

It is possible with a proper parametrization of $\pi_\theta$.

$$\pi_\theta(x_0, x_1) = p_0(x_0)\pi_\theta(x_1|x_0) = p_0(x_0)\frac{\exp\left(\langle x_0, x_1\rangle \epsilon\right) v_\theta(x_1)}{c_\theta(x_0)}.$$

We parameterize $v^*$ as $v_\theta$. In turn, $c_\theta(x_0) = \int_{\mathbb{R}^D} \exp\left(\langle x_0, x_1\rangle / \epsilon\right) v_\theta(x_1) dx_1$ is the normalization.

# Deriving the Learning Objective

**Magic of KL-divergence.**

$$\mathrm{KL}(\pi^*||\pi_\theta) = \int_{\mathbb{R}^D\times\mathbb{R}^D} \pi^*(x_0,x_1)\log\frac{\pi^*(x_0,x_1)}{\pi_\theta(x_0,x_1)}dx_0dx_1 =$$

$$C - \int_{\mathbb{R}^D\times\mathbb{R}^D} \pi^*(x_0,x_1)\log \underbrace{\frac{\exp\big(\langle x_0,x_1\rangle/\epsilon\big)v_\theta(x_1)}{c_\theta(x_0)}}_{\pi_\theta(x_1|x_0)} dx_0dx_1 = C - \underbrace{\int_{\mathbb{R}^D\times\mathbb{R}^D} \pi^*(x_0,x_1)\big(\langle x_0,x_1\rangle/\epsilon\big)dx_0dx_1}_{\text{also constant}} +$$

$$\underbrace{\int_{\mathbb{R}^D\times\mathbb{R}^D} \pi^*(x_0,x_1)\log c_\theta(x_0)dx_0dx_1}_{\text{expectation of a function of } x_0} - \underbrace{\int_{\mathbb{R}^D\times\mathbb{R}^D} \pi^*(x_0,x_1)\log v_\theta(x_1)dx_0dx_1}_{\text{expectation of a function of } x_1} =$$

$$\widetilde{C} + \underbrace{\int_{\mathbb{R}^D} p_0(x_0)\log c_\theta(x_0)dx_0 - \int_{\mathbb{R}^D} p_1(x_1)\log v_\theta(x_1)dx_1}_{=\mathcal{L}(\theta)} = \mathrm{Const} + \mathcal{L}(\theta).$$

We can estimate $\mathrm{KL}(\pi^*||\pi_\theta)$ up to a constant, which depends only on $\pi^*$. Hence, we can directly optimize $\mathrm{KL}(\pi^*||\pi_\theta)$ knowing nothing about $\pi^*$ except its marginals $p_0$ and $p_1$.

# Gaussian parametrization

**The functional for optimization.**

$$\min_{\theta} \text{KL}(\pi^* || \pi_\theta) - C = \min_{\theta} \mathcal{L}(\theta) = \min_{\theta} \int_{\mathbb{R}^D} p_0(x_0) \log c_\theta(x_0) dx_0 - \int_{\mathbb{R}^D} p_1(x_1) \log v_\theta(x_1) dx_1.$$

The problem: it is hard to compute normalization constant $c_\theta(x_0)$ for arbitrary potential $v_\theta$.

**Gaussian parametrization of adjusted Schrödinger potential.**

We recall that:

$$\pi_\theta(x_1|x_0) = \frac{\exp\big(\langle x_0, x_1\rangle/\epsilon\big) v_\theta(x_1)}{c_\theta(x_0)},$$

For $x = 0$, we have $\pi_\theta(x_1|0) = \frac{v_\theta(x_1)}{c_\theta(x_0)}$, i.e. $v_\theta(x_1)$ is an unnormalized density.

$\implies$ Let us approximate $v_\theta$ by a Gaussian mixture:

$$v_\theta(x_1) \stackrel{\text{def}}{=} \sum_{k=1}^{K} \alpha_k \mathcal{N}(x_1|r_k, S_k),$$

where $\theta \stackrel{\text{def}}{=} \{\alpha_k, r_k, S_k\}_{k=1}^K$ are the parameters: $\alpha_k \geq 0$, $r_k \in \mathbb{R}^D$ and symmetric $0 \prec S_k \in \mathbb{R}^{D\times D}$.

# Gaussian parametrization

**Conditional distribution for the Gaussian mixture parametrization.**

For a Gaussian mixture approximation $v_\theta(x_1) \stackrel{\text{def}}{=} \sum_{k=1}^{K} \alpha_k \mathcal{N}(x_1 | r_k, S_k)$, it holds that

$$\pi_\theta(x_1|x_0) = \frac{1}{c_\theta(x_0)} \sum_{k=1}^{K} \widetilde{\alpha}_k(x_0) \mathcal{N}(x_1 | r_k(x_0), \epsilon S_k) \qquad \text{where} \qquad r_k(x_0) \stackrel{\text{def}}{=} r_k + S_k x_0,$$

$$\widetilde{\alpha}_k(x_0) \stackrel{\text{def}}{=} \alpha_k \exp\Big(\frac{x_0^T S_k x_0 + 2 r_k^T x_0}{2\epsilon}\Big), \qquad c_\theta(x_0) \stackrel{\text{def}}{=} \sum_{k=1}^{K} \widetilde{\alpha}_k(x_0).$$

**The functional for optimization.**

$$\min_\theta \mathrm{KL}(\pi^* || \pi_\theta) - \mathrm{C} = \min_\theta \mathcal{L}(\theta) = \min_\theta \int_{\mathbb{R}^D} p_0(x_0) \log c_\theta(x_0) dx_0 - \int_{\mathbb{R}^D} p_1(x_1) \log v_\theta(x_1) dx_1.$$

With such parametrization, we can easily estimate and optimize our objective.

# Training

**The functional for optimization:**

$$\min_{\theta} \mathcal{L}(\theta) = \min_{\theta} \int_{\mathbb{R}^D} p_0(x_0) \log c_\theta(x_0) dx_0 - \int_{\mathbb{R}^D} p_1(x_1) \log v_\theta(x_1) dx_1.$$

**The empirical functional for optimization.**

As the distributions $p_0, p_1$ are accessible only via samples $X^0 = \{x_0^1, \ldots, x_0^N\} \sim p_0$ and $X^1 = \{x_1^1, \ldots, x_1^M\} \sim p_1$, we optimize the empirical counterpart of $\mathcal{L}(\theta)$:

$$\widehat{\mathcal{L}}(\theta) \stackrel{\text{def}}{=} \frac{1}{N} \sum_{n=1}^{N} \log c_\theta(x_0^n) - \frac{1}{M} \sum_{m=1}^{M} \log v_\theta(x_1^m) \approx \mathcal{L}(\theta).$$

We use the (minibatch) gradient descent w.r.t. parameters $\theta$.

# EOT-based inference

**Sampling starting and ending points.**
The conditional distributions $\pi_\theta(x_1|x_0)$ are mixtures of Gaussians:

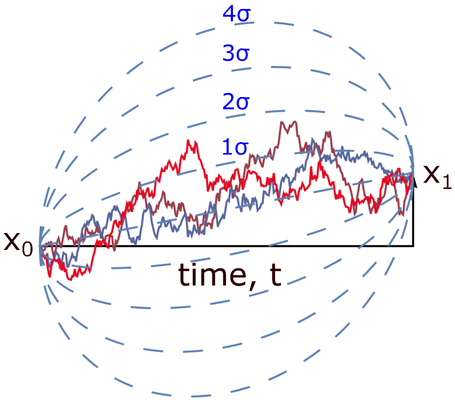
$$\pi_\theta(x_1|x_0) = \frac{1}{c_\theta(x_0)} \sum_{k=1}^{K} \widetilde{\alpha}_k(x_0)\mathcal{N}(x_1|r_k(x_0), \epsilon S_k)$$

Sampling of the pair $(x_0, x_1)$ is straightforward and lightspeed.

**Inner trajectory sampling.**

To sample trajectory $x_0, x_{t_1}, \dots, x_{t_L}, x_1$ with $0 < t_1 < \cdots < t_L < 1$ it is enough to sample from the Brownian Bridge $W^{\epsilon}_{|x_0,x_1}$.

**Brownian Bridge.**
The process $W^{\epsilon}_{|x_0,x_1}$ is a Brownian Bridge. It is a Gaussian process starting at $x_0$ and ending at $x_1$.



# SDE-based inference

**SDE form of the learned process.**

The process $T_\theta$ given by the potential $v_\theta$ is a diffusion process governed by the following SDE:

$$T_\theta \quad : \quad dX_t = g_\theta(X_t, t)dt + \sqrt{\epsilon}dW_t, \qquad X_0 \sim p_0,$$

$$g_\theta(x, t) \stackrel{\text{def}}{=} \epsilon \nabla_x \log \Big( \mathcal{N}(x|0, \epsilon(1-t)I_D) \sum_{k=1}^{K} \big\{ \alpha_k \mathcal{N}(r_k|0, \epsilon S_k) \mathcal{N}(h(x,t)|0, A_k^t) \big\} \Big),$$

with $A_k^t \stackrel{\text{def}}{=} \frac{t}{\epsilon(1-t)} I_D + \frac{S_k^{-1}}{\epsilon}$ and $h_k(x, t) \stackrel{\text{def}}{=} \frac{1}{\epsilon(1-t)} x + \frac{1}{\epsilon} S_k^{-1} r_k$.

- Any SDE solver can be applied to the sample from this SDE, e.g. Euler-Maruyama.
- EOT-based sampling is always better since it is the analytical solution of this SDE.

# Summary

**We developed a blazing-fast method for solving the Schrödinger Bridge problem.**

The method is based on:

1. New loss function for training the Schrödinger bridge:

$$\mathcal{L}(\theta) = \int_{\mathbb{R}^D} \log c_\theta(x_0) p_0(x_0) dx_0 - \int_{\mathbb{R}^D} \log v_\theta(x_1) p_1(x_1) dx_1, \quad c_\theta(x_0) = \int_{\mathbb{R}^D} \exp\big(\langle x_0, x_1 \rangle / \epsilon\big) v_\theta(x_1) dx_1,$$

where $v_\theta$ is an adjusted Schrödinger potential which completely defines the entire Schrödinger Bridge $T_\theta$.

2. Optimal parameterization of the Schrödinger bridge using mixtures of Gaussians:

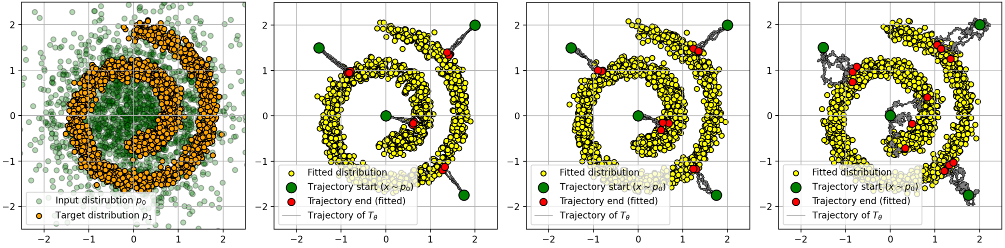
$$v_\theta(x_1) = \sum_{k=1}^{K} \alpha_k \mathcal{N}(x_1 | r_k, S_k), \quad c_\theta(x_0) = \sum_{k=1}^{K} \alpha_k \exp\Big(\frac{x_0^T S_k x_0 + 2 r_k^T x_0}{2\epsilon}\Big).$$

Our method's advantages:

- Fast training ($<1$ minute on 4 CPU cores, not hours of training on GPU, like others).
- Theoretical validity (in this work we prove the guarantees of the method's learning ability from the point of view of statistical learning theory and approximation theory).

# Experimental results

1. **Qualitative** results of our algorithm applied to 2D model distributions ("Gaussian" → "swiss-roll").




**(a)** $x \sim p_0$, $y \sim p_1$. **(b)** $\epsilon = 2 \cdot 10^{-3}$. **(c)** $\epsilon = 0.01$. **(d)** $\epsilon = 0.1$.

2. **Quantitative** results of our solver on the standard benchmark for the Schrödinger bridge problem.

Best from the existing methods →

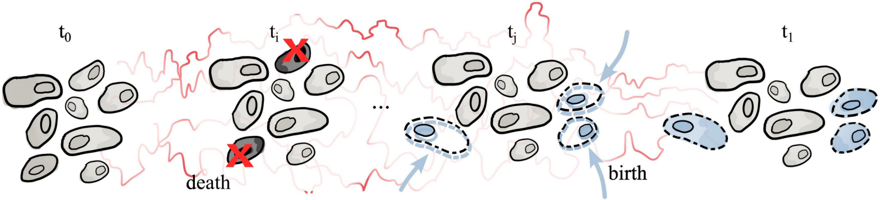
Our method →

| | $\epsilon=0.1$ | | | | $\epsilon=1$ | | | | $\epsilon=10$ | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $D=2$ | $D=16$ | $D=64$ | $D=128$ | $D=2$ | $D=16$ | $D=64$ | $D=128$ | $D=2$ | $D=16$ | $D=64$ | $D=128$ |
| Best solver | 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] | **0.03** | **0.08** | **0.28** | **0.60** | **0.05** | **0.09** | **0.24** | **0.62** | **0.07** | **0.11** | **0.21** | **0.37** |
| $\pm std$ | ±0.01 | ±0.04 | ±0.02 | ±0.02 | ±0.003 | ±0.006 | ±0.007 | ±0.007 | ±0.02 | ±0.01 | ±0.01 | ±0.01 |

*The metric cBW-UVP is used for comparising build schrödinger bridge with ground-truth bridge (lower=better).

# Experiments with Single-cell data[12]

3. **Quantitative** results in the problem of predicting single-cell trajectories in the feature space (single-cell trajectory inference).




Our method is superior to analogues in quality of work and speed. →

| Setup | Solver type | Solver \ DIM | 50 | 100 | 1000 |
|---|---|---|---|---|---|
| Discrete EOT | Sinkhorn | (Cuturi, 2013) [1 GPU V100] | 2.34 (90 s) | 2.24 (2.5 m) | 1.864 (9 m) |
| Continuous EOT | Langevin-based | (Mokrov et al., 2023) [1 GPU V100] | 2.39 ± 0.06 (19 m) | 2.32 ± 0.15 (19 m) | 1.46 ± 0.20 (15 m) |
| Continuous EOT | Minimax | (Gushchin et al., 2023) [1 GPU V100] | 2.44 ± 0.13 (43 m) | 2.24 ± 0.13 (45 m) | 1.32 ± 0.06 (71 m) |
| Continuous EOT | 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) |
| Continuous EOT | KL minimization | LightSB (**ours**) [4 CPU cores] | 2.31 ± 0.27 (65 s) | 2.16 ± 0.26 (66 s) | 1.27 ± 0.19 (146 s) |

**The Energy distance metric is used to compare the predicted cell position and the observed one (smaller=better).
The operating time of the method in question is indicated in parentheses. 50, 100, 1000 - dimension of the feature space.

[12] Alexander Y Tong et al. (2024). "Simulation-Free Schrödinger Bridges via Score and Flow Matching". In: *International Conference on Artificial Intelligence and Statistics*. PMLR, pp. 1279–1287.

# Unpaired Image translation in latent space

4. **Qualitative** results of the method for solving the **unpaired** domain translation problem (in the latent space of the ALAE autoencoder[13]).

The latent space size is 512. Images resolution is 1024x1024.

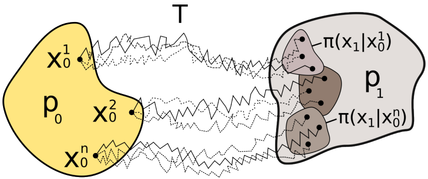
**(a)** Male → Female. **(b)** Female → Male. **(c)** Adult → Child. **(d)** Child → Adult.

[13] Stanislav Pidhorskyi, Donald A Adjeroh, and Gianfranco Doretto (2020). "Adversarial latent autoencoders". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14104–14113.

# Thank you

## Light Schrödinger Bridge (ICLR 2024)

The novel light and fast algorithm
to solve the Schrödinger Bridge problem.




GitHub

https://github.com/ngushchin/LightSB

# Part II. Light and Optimal Schrödinger Bridge Matching (ICML 2024)

# Reciprocal and Markovian processes

Let $\mathcal{F}$ denote the set of all stochastic processes in $\mathbb{R}^D$ for time interval $[0,1]$ with continuous trajectories $\{x_t\}_{t\in[0,1]}$. Recall that we already use $\mathcal{F}(p_0,p_1)\subset\mathcal{F}$ to denote its subset of processes whose marginals at times $t=0,1$ are $p_0$ and $p_1$, respectively.

**Reciprocal processes** Let $\mathcal{R}\subset\mathcal{F}$ denote the subset of **reciprocal** processes, i.e., those processes can be represented as mixtures of Brownian bridges:

$$T\in\mathcal{R} \qquad \Leftrightarrow \qquad \exists\pi=\pi^T\in\mathcal{P}(\mathbb{R}^D\times\mathbb{R}^D) \text{ s.t. } T=T_\pi \stackrel{\text{def}}{=} \int W^{\epsilon}_{|x_0,x_1}\pi(x_0,x_1)dx_0dx_1.$$

We use $\mathcal{R}(p_0,p_1)$ to denote its subset of processes which satisfy $\pi^T\in\Pi(p_0,p_1)$.

**Markov Processes** Let $\mathcal{M}\subset\mathcal{F}$ denote the subset of **Markovian** processes, i.e.,

$$T\in\mathcal{M} \qquad \Leftrightarrow \qquad \forall N>1,\ 0\le t_1<\dots<t_N\le 1:\ p^T(x_{t_N}|x_{t_{N-1}}\dots,x_1)=p^T(x_{t_N}|x_{t_{N-1}}).$$

In turn, let $\mathcal{M}(p_0,p_1)$ denote its subset of processes which satisfy $\pi^T\in\Pi(p_0,p_1)$.

# Schrödinger Bridge is both Markovian and Reciprocal Process

In fact, we already know that $T^* \in \mathcal{M}(p_0, p_1) \cap \mathcal{R}(p_0, p_1)$. Indeed,

- We already derived that SB $T^*$ is a **mixture of Brownian Bridges** $W^{\epsilon}_{|x_0,x_1}$:

$$T^* = \int W^{\epsilon}_{|x_0,x_1} \pi^*(x_0, x_1) dx_0 dx_1 \in \mathcal{M} \cap \mathcal{R},$$

  where $\pi^*(x_0, x_1)$ is the EOT plan. Therefore, $T^* \in \mathcal{R}(p_0, p_1) \subset \mathcal{R}$.
- We have already seen that the solution $T^*$ is a **diffusion** process:

$$dX_t = g^*(X_t, t)dt + \sqrt{\epsilon} dW^{\epsilon}_t, \quad X_0 \sim p_0$$

  for some drift $g^*$. Therefore, $T^*$ is Markovian, i.e., $T^* \in \mathcal{M}(p_0, p_1) \subset \mathcal{M}$.

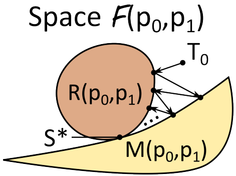
# Iterative Projections onto the Sets of Interest

The Schrödinger Bridge has an **awesome property**[14]: it is the *unique* process (starting at $p_0$ and ending at $p_1$) that satisfies both the *markovian* and *reciprocal* property, i.e.,

$$\{T^*\} = \mathcal{M}(p_0, p_1) \cap \mathcal{R}(p_0, p_1).$$

**Idea:** why not to try to find the process that is both markovian and reciprocal by using some sort of **projections** onto Reciprocal $\mathcal{R}(p_0, p_1)$ and Markovian $\mathcal{M}(p_0, p_1)$ sets of processes?

**Note:** The subset $R \subset \mathcal{F}$ is convex, while $\mathcal{M} \subset \mathcal{F}$ is, in general, not convex. The latter statement is not obvious and is a good excersize to think about.



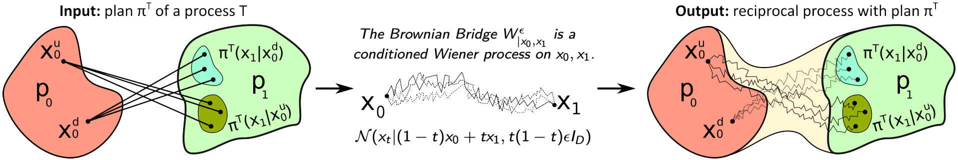

[14] Christian Léonard (2014). "A survey of the Schrödinger problem and some of its connections with optimal transport". In: *Discrete & Continuous Dynamical Systems-A* 34.4, pp. 1533–1574.

# Reciprocal Projection

The projection is defined for every $T \in \mathcal{F}$ as follows:

$$\text{proj}_{\mathcal{R}}(T) \overset{\text{def}}{=} \text{argmin}_{R \in \mathcal{R}} \text{KL}(T \| R).$$

One may easily prove that the reciprocal projection creates a **mixture** of Brownian Bridges $W^{\epsilon}_{|x_0,x_1}$ with the distribution $\pi^T$ of a stochastic process $T \in \mathcal{F}$ at times $t = 0, 1$, i.e.,




$$\text{proj}_{\mathcal{R}}(T) = \int W^{\epsilon}_{|x_0,x_1} \pi^T(x_0, x_1) dx_0 dx_1.$$

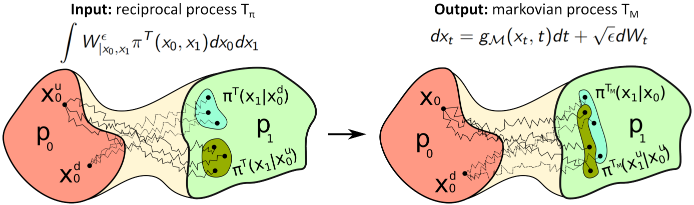
The projection depends only on $\pi^T$ (transport plan) rather than on the entire process $T$. Furthermore, $\pi^{\text{proj}_{\mathcal{R}}(T)} = \pi^T$, i.e., this *transport plan is preserved* during the projection.

# Markovian Projection

The projection is defined for *reciprical* processes $T = T_\pi \in \mathcal{R}$ as follows:

$$\text{proj}_{\mathcal{M}}(T) \stackrel{\text{def}}{=} \text{argmin}_{M \in \mathcal{M}} \text{KL}(T \| M).$$

It finds the **diffusion** process $T_{\mathcal{M}}$ which is the most similar to $T_\pi$:




The drift of the Markovian projection is: $g_{\mathcal{M}} \stackrel{\text{def}}{=} \arg\min_g \int_0^1 \mathbb{E}_{(x_t,x_1)\sim T_\pi} ||g(x_t,t) - \frac{x_1 - x_t}{1-t}||^2 dt.$

The markovian projections *preserves the marginals* of the process at every time $t$ (including $t = 0, 1$), but *alters the transport plan*, i.e., $\pi^T \neq \pi^{T_{\mathcal{M}}}$ (unless $T$ is the Schrodinger Bridge).

# Reciprocal and Markovian projections

## Reciprocal projection

- Defined for any process $T \in \mathcal{F}$:

$$\text{proj}_{\mathcal{R}}(T) \stackrel{\text{def}}{=} \text{argmin}_{R \in \mathcal{R}} \text{KL}(T \| R)$$

- Yields a mixture of Brownian Bridges:

$$\int W^{\epsilon}_{|x_0, x_1} \pi^T(x_0, x_1) dx_0 dx_1$$

## Markovian projection

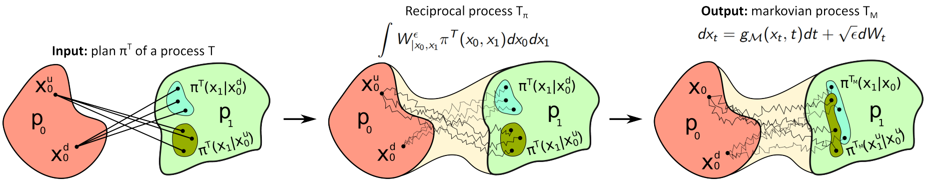
- Defined for a *reciprocal* process $T_\pi \in \mathcal{R}$:

$$\text{proj}_{\mathcal{M}}(T_\pi) \stackrel{\text{def}}{=} \text{argmin}_{M \in \mathcal{M}} \text{KL}(T_\pi \| M)$$

- Yields a **diffusion** with the SDE

$$dx_t = g_{\mathcal{M}}(x_t, t)dt + \sqrt{\epsilon} dW_t, \qquad x_0 \sim p_0.$$

**Bridge matching** = combination of Reciprocal and Markovian Projections



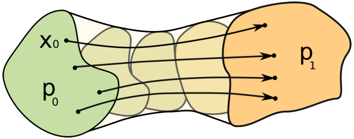

It is a popular way to learn diffusion processes between data distributions $p_0, p_1$.

# Flow Matching and Bridge Matching: a Reminder

Flow matching is a limiting case of the Bridge Matching when $\epsilon \to 0$.

**Flow Matching**

$$x_t = g(x_t, t)t$$



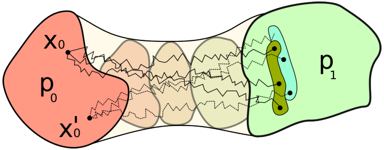

Define interpolation: $x_t \stackrel{\text{def}}{=} x_0 \cdot (1-t) + x_1 \cdot t$.

$$\min_g \mathop{\mathbb{E}}_{\substack{x_0 \sim p_0 \\ x_1 \sim p_1}} \mathop{\mathbb{E}}_{t \sim [0,1]} \left\| g(x_t, t) - (x_1 - x_0) \right\|^2.$$

Can be iterated to straighten the flows.
Related to the **Optimal Transport** (OT).

**Bridge Matching**

$$x_t = g(x_t, t)t + \sqrt{\epsilon} W_t \quad (\epsilon > 0).$$




Define a distribution: $p_t^\epsilon \stackrel{\text{def}}{=} \mathcal{N}(x_t, \epsilon t(1-t))$

$$\min_g \mathop{\mathbb{E}}_{\substack{x_0 \sim p_0 \\ x_1 \sim p_1}} \mathop{\mathbb{E}}_{t \sim [0,1]} \mathop{\mathbb{E}}_{\tilde{x}_t \sim p_t^\epsilon} \left\| g(\tilde{x}_t, t) - \frac{x_1 - \tilde{x}_t}{1-t} \right\|^2.$$

Can be iterated and converges to the **Schrödinger bridge**.

# Iterative Markovian Fitting (IMF)[15][16]

Alternating Markovian and Reciprocal projections is called the **Iterative Markovian Fitting** (IMF) procedure, or, alternatively, **Iterative Diffusion Bridge Matching** (IDBM).

Starting from a reciprocal process $T_0 = \int W^{\epsilon}_{|x_0,x_1} d\pi(x_0, x_1)$ induced by some initial plan $\pi(x_0, x_1)$, one performs iterative updates

$$T^{2n+1} = \text{proj}_{\mathcal{M}}(T^{2n}), \qquad T^{2n+2} = \text{proj}_{\mathcal{R}}(T^{2n+1})$$

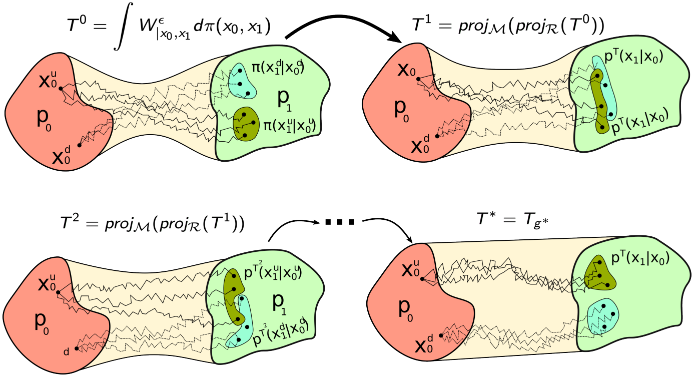
The sequence $\{T^n\}_{n=1}^{\infty}$ converges to the Schrodinger Bridge $T^*$:

$$\lim_{n\to+\infty} \text{KL}(T^n \| T^*) = 0.$$

[15] Stefano Peluchetti (2023). "Diffusion bridge mixture transports, Schrödinger bridge problems and generative modeling". In: *Journal of Machine Learning Research* 24.374, pp. 1–51.

[16] Yuyang Shi et al. (2023). "Diffusion Schrödinger Bridge Matching". In: *Thirty-seventh Conference on Neural Information Processing Systems*. URL: https://openreview.net/forum?id=qy07OHsJT5.

# Iterative Markovian Fitting: An Illustration



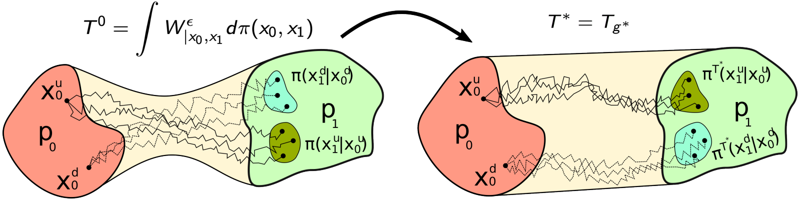

**Limitations**: The procedure is iterative, i.e., it requires many bridge matching steps.

- Each bridge matching step is a non-trivial drift learning (optimization) procedure.
- Errors in matching the target ($p_1$) may accumulate during IMF steps.[17]

[17] Rectified flow is a limiting case of the IMF when $\to 0$.

# Optimal Schrödinger Bridge Matching[18]

While IMF performs iterative Bridge Matching (reciprical and Markovian projections) to recover SB, we propose a novel concept of the **optimal projection**. It Recovers the Schrödinger Bridge $T^*$ is just one iteration of the Bridge Matching.




[18] Nikita Gushchin, Sergei Kholkin, et al. (n.d.). "Light and Optimal Schrödinger Bridge Matching". In: *Forty-first International Conference on Machine Learning*.

# Optimality of the "Optimal Projection"

Projection on the set $\mathcal{S}$ of SBs or *"Optimal Projection"* is the foundation of our method.

$$\mathcal{S} \stackrel{\text{def}}{=} \mathcal{R} \cap \mathcal{M}.$$

For **any** reciprocal process $T_\pi$ with marginals $p_0$ and $p_1$, we define the **optimal projection** by

$$\text{proj}_{\mathcal{S}}(T_\pi) = \text{argmin}_{S \in \mathcal{S}} \text{KL}(T_\pi \| S)$$

**Theorem (Optimal Projection)**

*Consider any reciprocal process $T_\pi$ that has marginals $p_0$ and $p_1$ at $t = 0$ and $t = 1$, respectively. Then the Optimal Projection yields the Schrödinger Bridge $T^*$, i.e.,*

$$T^* = proj_{\mathcal{S}}(T_\pi).$$

Looks nice, but how to implement this projection in practice? How to optimize over $S \in \mathcal{S}$?

# Characterization for EOT and SB solutions: a Reminder

The solutions for SB problems can be **characterized** by two things:

1. the starting distribution $p_0$;
2. the scalar-valued function $v$ (potential).

More precisely, the following process (which we denote by $S_v$)

$$S_v: \qquad dX_t = g_v(X_t, t)dt + \sqrt{\epsilon}dW_t^{\epsilon}, \quad X_0 \sim p_0,$$

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

belongs to $\mathcal{S}(p_0) \subset \mathcal{S}$ and is the Schrodinger bridge between $p_0$ and its marginal at time $t = 1$. Here $\mathcal{S}(p_0)$ denotes the subset of all Schrodinger Bridges which start at $p_0$.

**Idea:** optimize $\arg\min_{S \in \mathcal{S}(p_0)} \mathrm{KL}(T_\pi \| S_v)$ instead of $\arg\min_{S \in \mathcal{S}} \mathrm{KL}(T_\pi \| S)$.[19]

[19] These problems lead to the same solution $T^* \in \mathcal{S}(p_0) \subset \mathcal{S}$.

# Tractable Optimization Objective for the Optimal Projection

Optimal projection can be implemented using the *constrained* Bridge matching procedure.

**Theorem (Tractable Objective for Optimal Projecton)**

*Consider set of SBs that start at $p_0$, i.e., $S_\theta \in \mathcal{S}(p_0)$. Let reciprocal process $T_\pi$ be a reciprocal process. Then the optimal projection objective satisfies*

$$KL(T_\pi \| S_v) = C(\pi) + \underbrace{\frac{1}{2\epsilon}\int_0^1 \mathbb{E}_{(x_t, x_1)\sim T_\pi} \|g_v(x_t, t) - \frac{x_1 - x_t}{1-t}\|^2 dt}_{\text{Bridge Matching}},$$

*where*

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

*is the the drift of $S_v$. Here constant $C(\pi)$ does not depend on $S_v$.*

Nice, but how to compute drift $g_v$ and optimize this objective?

# Optimal Drift Computation Problem

For general parameterization of the potential $v$, e.g., with a neural network $v_\theta$, the computation of the drift $g_v = g_{v_\theta}$ is tricky, so is the computation of the loss. It requires tricky Monte Carlo Markov Chain techniques (MCMC), see the appendices of the paper.[20]

Fortunately, for a Gaussian mixture parameterization (as in **LightSB**) $v_\theta(x_1) \stackrel{\text{def}}{=} \sum_{k=1}^{K} \alpha_k \mathcal{N}(x_1 | r_k, S_k)$, the drift $g_{v_\theta}$ is available in the closed form

$$g_{v_\theta}(x_t, t) = \epsilon \nabla_x (\mathcal{N}(x|0, \epsilon(1-t)I_D) \sum_{k=1}^{K} \{\alpha_k \mathcal{N}(r_k|0, \epsilon S_k) \mathcal{N}(h_k(x,t)|o, A_k^t)\}.$$

Then we can implement the optimal Projection by optimizing

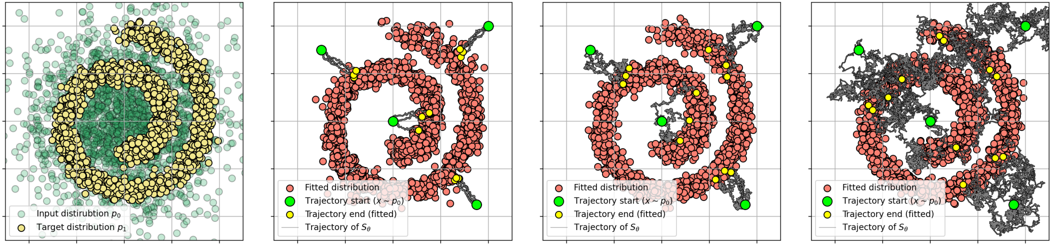
$$\theta^* = \arg\min_\theta \mathrm{KL}(T_\pi \| S_{v_\theta}) = \arg\min_\theta \frac{1}{2\epsilon} \int_0^1 \mathbb{E}_{(x_t, x_1) \sim T_\pi} \| g_\theta(x_t, t) - \frac{x_1 - x_t}{1-t} \|^2 dt.$$

We call the approach by **LightSB-M**. Here **M** stands for **m**atching.

[20] Nikita Gushchin, Sergei Kholkin, et al. (n.d.). "Light and Optimal Schrödinger Bridge Matching". In: *Forty-first International Conference on Machine Learning*.

# Qualitative Experiments. 2D Swiss Roll

The process $S_\theta = S_{v_\theta}$ learned with LightSB-M in *Gaussian* $\rightarrow$ *Swiss roll* example.




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

# Experiments. Quantitative SB Benchmark[21]

LightSB-M is the best Bridge Matching method on the SB benchmark. It has comparable performance to LightSB. Also, it yields the same solution for different starting plans $\pi(x_0, x_1)$: independent (**ID**), mini-batch OT (**MB**), ground truth (**GT**).

| | | $\epsilon = 0.1$ | | | | $\epsilon = 1$ | | | | $\epsilon = 10$ | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 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 SB bench† | 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 | Bridge matching | 5.2 | 16.8 | 37.3 | 35 | 0.3 | 1.1 | 9.7 | 31 | 3.7 | 105 | 3557 | 15000 |
| $SF^2M$-Sink | Bridge matching | 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**) | Bridge matching | **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**) | Bridge matching | **0.02** | **0.1** | **0.49** | **1.16** | **0.09** | **0.18** | 0.47 | **1.2** | 0.13 | **0.18** | **0.36** | **0.69** |

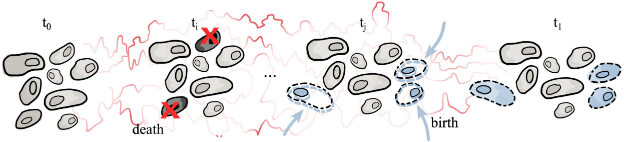
Comparisons of $cB\mathbb{W}_2^2$-UVP ↓ (%) between the optimal plan $\pi^*$ and the learned plan $\pi_\theta$ on the EOT/SB benchmark.

The best metric over *bridge matching* solvers is **bolded**. Results marked with † are taken from LightSB paper.

[21]Nikita Gushchin, Alexander Kolesov, Petr Mokrov, et al. (2023). "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*. URL: https://openreview.net/forum?id=OHimIaixXk.

# Experiments. Quantitative Evaluation on Biological Data

Predicting single-cell trajectories in the feature space.

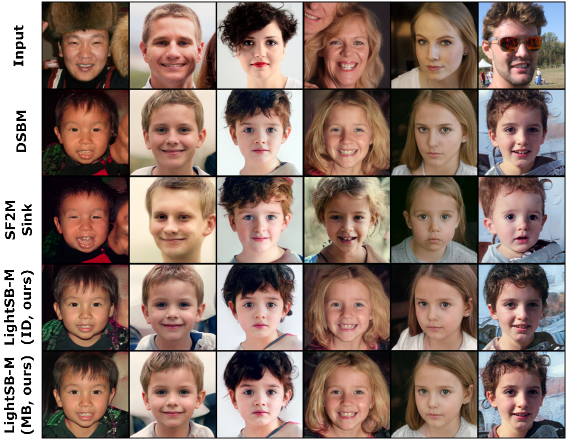



| Solver type | SolverDIM | 50 | 100 | 1000 |
|---|---|---|---|---|
| Langevin-based | EgNOT† [1 GPU V100] | 2.39 ± 0.06 (19 m) | 2.32 ± 0.15 (19 m) | 1.46 ± 0.20 (15 m) |
| Minimax | ENOT† [1 GPU V100] | 2.44 ± 0.13 (43 m) | 2.24 ± 0.13 (45 m) | 1.32 ± 0.06 (71 m) |
| IPF | DSB† [1 GPU V100] | 3.14 ± 0.27 (8 m) | 2.86 ± 0.26 (8 m) | 2.05 ± 0.19 (11 m) |
| KL minimization | LightSB† [4 CPU cores] | 2.31 ± 0.27 (65 s) | 2.16 ± 0.26 (66 s) | 1.27 ± 0.19 (146 s) |
| Bridge matching | DSBM [1 GPU V100] | 2.46 ± 0.1 (6.6 m) | 2.35 ± 0.1 (6.6 m) | 1.36 ± 0.04 (8.9 m) |
| | $SF^2$M-Sink [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] | **2.33** ± 0.09 (80 s) | **2.172** ± 0.08 (80 s) | **1.33** ± 0.05 (176 s) |

**Table 1:** Energy distance (averaged for two setups and 5 random seeds) on the MSCI dataset

LightSB-M is the best **Bridge Matching** method in thie experiment with Biological data. It provides comparable performance to LightSB that is based on the **KL** minimization principle.

# Experiments. Comparison on Unpaired Image-to-image Transfer




Adult to Child Unpaired Translation in the latent space of ALAE[22], 1024x1024 images.

[22]Stanislav Pidhorskyi, Donald A Adjeroh, and Gianfranco Doretto (2020). "Adversarial latent autoencoders". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14104–14113.

# Summary

**LightSB-M is a method to solve the SB problem in a single Bridge Matching step.**

The solver is based on:

- The "Optimal Projection" that translates **any** $\pi$ with marginals $p_0$ and $p_1$ to SB
- Novel Bridge Matching-like optimization objective

$$L_\theta(\pi) = \int_0^1 \mathbb{E}_{(x_t, x_1)\sim T_\pi} \|g_\theta(x_t, t) - \frac{x_1 - x_t}{1 - t}\|^2 dt$$

$$g_\theta(x_t, t) = \epsilon \nabla_{x_t} \log \int_{\mathbb{R}^D} \mathcal{N}(x'|x_t, (1-t)\epsilon I_D) \exp(\frac{\|x'\|^2}{2\epsilon}) v_\theta(x') dx'$$

- Parameterization of the SB using mixtures of Gaussians $v_\theta(x) = \sum_{k=1}^{K} \alpha_k \mathcal{N}(x|r_k, S_k)$. In this case, $g_\theta$ admits closed form expression.

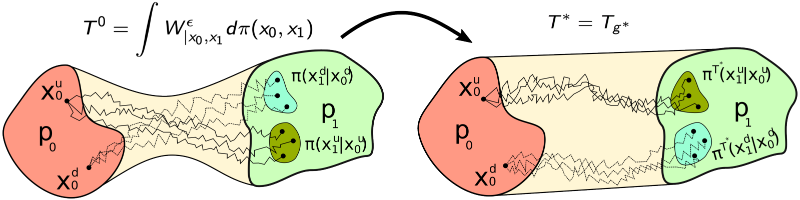
LightSB-M's advantages:

- Theoretical novelty (first method solving SB in one Bridge Matching iteration).
- Fast training ($< 1$ minute on 4 CPU cores, not hours of training on GPU, like others).

## Thank you

### Light and Optimal Schrödinger Bridge Matching (ICML 2024)

The novel light and fast algorithm based on the bridge matching
to solve the Schrödinger Bridge problem.



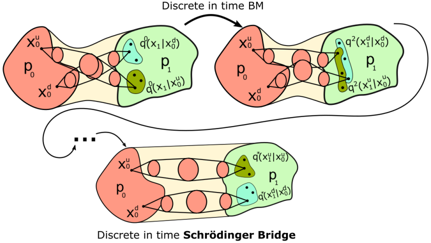

GitHub

`https://github.com/SKholkin/LightSB-Matching`

# Other works

# Adversarial Schrödinger Bridge Matching[23]

We present Discrete in time Bridge Matching and prove that Iterative Discrete in time Bridge Matching (**D-IMF**) converges to discrete in time **Schrödinger Bridge**.



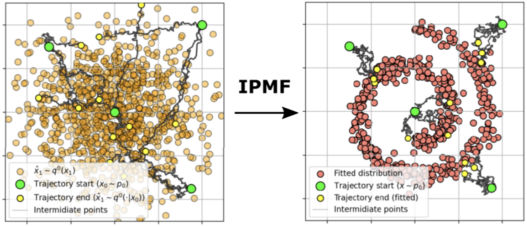


**Idea:** Substitute the Bridge Matching Diffusion by the **Denoising Diffusion GAN** (DD-GAN). That allows to **speed up** the generation **x25** times while having even better quality.

[23] Nikita Gushchin, Daniil Selikhanovych, et al. (2024). "Adversarial Schrödinger Bridge Matching". In: *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. URL: https://openreview.net/forum?id=L3Knnigicu.

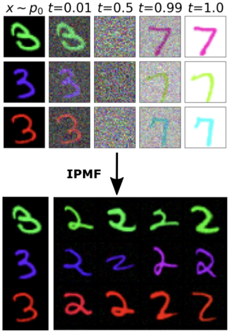
# Iterative Proportional Markovian Fitting[24]

Practical implementation of **IMF** algorithm secretly utilizes another popular algorithm **IPF**. We propose Iterative Proportional Markovian Fitting (**IPMF**) algorithm, argue that IMF used in practice and IPF algorithms are a particular cases of IPMF.




Independent coupling
Gaussian -> Gaussian

Schrödinger Bridge
Gaussian -> Swiss Roll




We show empirically and in some cases theoretically that IPMF converges to the Schrödinger Bridge.

[24]Sergei Kholkin et al. (2024). "Diffusion & Adversarial Schr\" odinger Bridges via Iterative Proportional Markovian Fitting". In: *arXiv preprint arXiv:2410.02601*.