Generative modeling via Schrödinger bridge (Diffusion Schrödinger Bridges)

Valentin De Bortoli

March 5, 2023

Summary of the previous lecture (1/4)

- In the previous lecture we introduced the **Schrödinger bridge** problem and the **Sinkhorn** algorithm:
  - Introduction of **Schrödinger bridges**.
  - Theoretical properties and link with **Optimal Transport**.
  - Introduction of the **Sinkhorn algorithm**.
  - **Exponential convergence** of the algorithm in compact spaces.
  - **Convergence results** in non-compact space.
- Recall that the **dynamical** formulation is given by

$$\boxed{\pi^\star = \arg\min\{\mathrm{KL}(\pi|\pi^0) \ : \ \pi \in \mathcal{P}((\mathbb{R}^d)^N), \pi_0 = \nu_0, \ \pi_N = \nu_1\} \ ,}$$

- Link with **generative modeling**:
  - $\pi^0 \in \mathcal{P}((\mathbb{R}^d)^N)$ is the discretization of the **Ornstein-Ulhenbeck** process.
  - ν_0 is the **data distribution**.
  - $\nu_1 = \mathrm{N}(0, \mathrm{Id})$ is the **easy-to-sample** distribution.

Summary of the previous lecture (2/4)

- **Advantage** of the Schrödinger bridge formulation:
  - The terminal distribution is **Gaussian** (no approximation).
  - The number of steps is **arbitrary**.
  - This is a **more flexible** framework.
  - Links with **Optimal Transport** and **Stochastic Control**.
- Some **drawbacks**:
  - **Longer training** times.
  - Paying the price of the **approximation**.

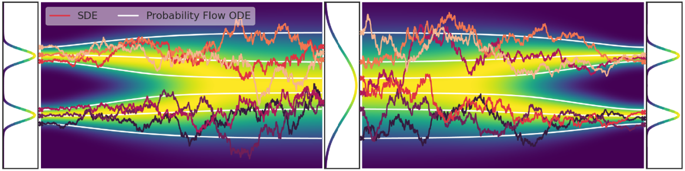


Figure 1: Noising and generative processes in SGM. Image extracted from **?**.

Summary of the previous lecture (3/4)

- Recall the **Sinkhorn algorithm**:

$$\pi^{2n+1} = \arg\min\{\mathrm{KL}(\pi|\pi^{2n}) \ : \ \pi \in \mathcal{P}((\mathbb{R}^d)^2),\ \pi_1 = \nu_1\} \ ,$$
$$\pi^{2n+2} = \arg\min\{\mathrm{KL}(\pi|\pi^{2n+1}) \ : \ \pi \in \mathcal{P}((\mathbb{R}^d)^2),\ \pi_0 = \nu_0\} \ .$$

- The **Sinkhorn algorithm** amounts to solving **half-bridges**.
- This is an **alternate projection** scheme w.r.t. the Kullback-Leibler divergence.

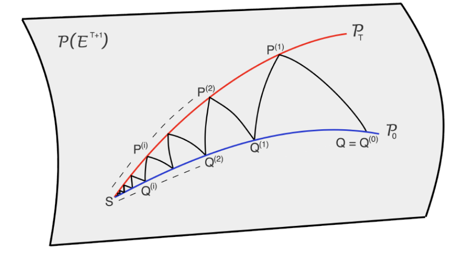


Figure 2: Solving half-bridges. Image extracted from Bernton et al. (2019).

Summary of the previous lecture (4/4)

- **Exponential convergence** in the **compact setting**.

Geometric convergence

- Let $(\pi^n)_{n\in\mathbb{N}}$ be the sequence obtained with the **Sinkhorn** algorithm and $\pi^\star$ the **Schrödinger bridge**. Under mild assumptions, we have

$$\boxed{\mathbf{W}_1(\pi^n, \pi^\star) \leq C\rho^n \ .}$$

- In fact the main result is a **geometric convergence** results on the potentials w.r.t. the **Hilbert-Birkhoff** metric.
- In the **non-compact setting** we still have convergence.

Convergence of the Sinkhorn algorithm Nutz (2021)

- Assume that $\int_{\mathbb{R}^d} \exp[r|\log \pi^0(x,y)|]\mathrm{d}(\nu_0 \otimes \nu_1)(x,y) < +\infty$ for some $r > 1$.
- Then $\lim_{n\to+\infty} \mathrm{KL}(\pi^n|\pi^\star) = 0$.

Outline of the course

- We introduce **Schrödinger bridges** for generative modeling.
- **Goal of the course**:
  - Introduce the **Diffusion Schrödinger Bridge (DSB)** algorithm.
  - Present a **conditional** extension of DSB.
- **Outline of the course**
  - Methodology of **Diffusion Schrödinger Bridges**.
  - **Conditional** generative modeling.

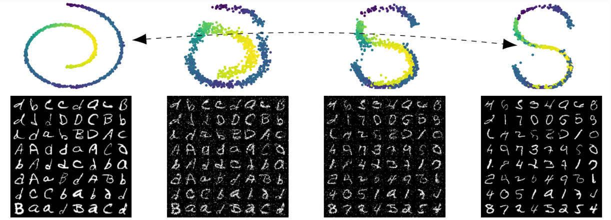


Figure 3: A Schrödinger Bridge between two data distributions. Image extracted from De Bortoli et al. (2021).

Diffusion Schrödinger Bridge

Outline of this section

- In this section:
  - We present **Diffusion Schrödinger Bridge**.
  - A **continuous time** formulation and a connection with **normalizing flows**.
  - Some **experimental results**.

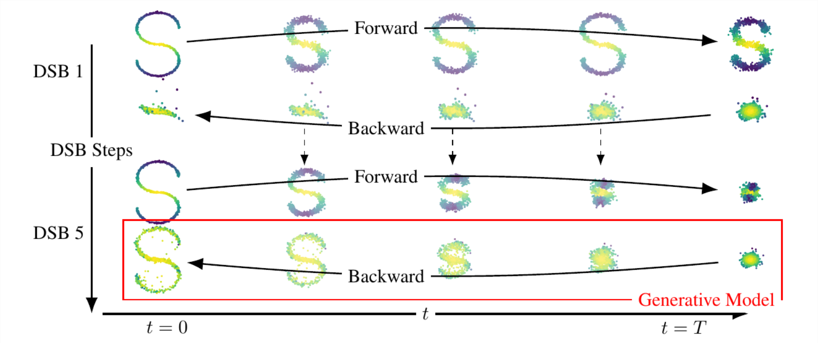


Figure 4: The Diffusion Schrödinger Bridge (DSB) algorithm. Image extracted from De Bortoli et al. (2021).

Methodology

Revisiting Generative Modeling using Schrödinger Bridges

- The **Schrödinger Bridge (SB) problem** is a classical problem appearing in applied mathematics, optimal control and probability.
- Recall that the **dynamical** formulation is given by

$$\boxed{\pi^\star = \arg\min\{\mathrm{KL}(\pi|\pi^0) \ : \ \pi \in \mathcal{P}((\mathbb{R}^d)^{N+1}), \pi_0 = \nu_0, \ \pi_N = \nu_1\} \ ,}$$

  - ν_0 is the **data distribution**.
  - ν_1 is the **easy-to-sample** distribution.
- If $\pi^\star$ is available: $X_N \sim \nu_1$, then $X_k \sim \pi^\star_{k|k+1}(\cdot|X_{k+1})$ for $k \in \{N-1, \dots, 0\}$.
- Recall the **Sinkhorn algorithm**:

$$\pi^{2n+1} = \arg\min\{\mathrm{KL}(\pi|\pi^{2n}) \ : \ \pi \in \mathcal{P}((\mathbb{R}^d)^{N+1}), \ \pi_N = \nu_1\} \ ,$$
$$\pi^{2n+2} = \arg\min\{\mathrm{KL}(\pi|\pi^{2n+1}) \ : \ \pi \in \mathcal{P}((\mathbb{R}^d)^{N+1}), \ \pi_0 = \nu_0\} \ .$$

- Updating the **potentials** is not efficient.
  - **Computing** the potentials is challenging (**dynamic programming**).
  - **Sampling** from **twisted kernels** is challenging.

Solving the Schrödinger Bridge Problem

- The SB problem can be solved using **Iterative Proportional Fitting (IPF)** (Fortet, 1940; Kullback, 1968), i.e. set $\pi^0 = p$ and for $n \geq 1$

$$\pi^{2n+1} = \arg\min\{\mathrm{KL}(\pi|\pi^{2n}), \ \ \pi_N = p_{\mathrm{prior}}\},$$
$$\pi^{2n+2} = \arg\min\{\mathrm{KL}(\pi|\pi^{2n+1}), \ \ \pi_0 = p_{\mathrm{data}}\}.$$

- This is akin to **alternative projection** in a Euclidean setting.
- $\lim_{n\to+\infty} \pi^n = \pi^{\mathrm{s},\star}$ under regularity conditions (Ruschendorf, 1995; Léger, 2021; De Bortoli et al., 2021).
- Explicit solution of the first IPF step

$$\mathrm{KL}(\pi||\pi^0) = \mathrm{KL}(\pi_N|p_N) + \mathbb{E}_{\pi_N}[\mathrm{KL}(\pi_{|N}||p_{|N})]$$

  Therefore,

$$\begin{aligned}\pi^1(x_{0:N}) &= p_{\mathrm{prior}}(x_N)p(x_{0:N-1}|x_N)\\ &= p_{\mathrm{prior}}(x_N)\textstyle\prod_{k=N-1}^{0} p_{k|k+1}(x_k|x_{k+1})\end{aligned}$$

- **Take-home message:** Approximation to first iteration of IPF corresponds to current **Score-Based Generative models**.

Solving the Schrödinger Bridge Problem

- The second iteration requires solving

$$\pi^2 = \arg\min\{\mathrm{KL}(\pi||\pi^1), \ \ \pi_0 = p_{\mathrm{data}}\}.$$

Therefore,

$$\begin{aligned}\pi^2(x_{0:N}) &= p_{\mathrm{data}}(x_0)\pi^1(x_{1:N}|x_0) \\ &= p_{\mathrm{data}}(x_0)\textstyle\prod_{k=1}^{N}\pi^1_{k+1|k}(x_{k+1}|x_k)\end{aligned}$$

- On an algorithmic level:
  - IPF1: the time-reversal of the **forward process** $\pi^0 = p$ is initialized by p_{prior} at time N to define the **backward process** π^1.
  - IPF2: the time-reversal of the **backward process** π^1 is initialized by p_{data} at time 0 to define the **forward process** π^2.
  - IPF3: the time-reversal of the **forward process** π^2 is initialized by p_{prior} at time N to define the **backward process** π^3.
  - ...

IPF and score networks

- Denote the forward processes $p^n := \pi^{2n}$ and backward processes $q^n := \pi^{2n+1}$.
- If $p^n_{k+1|k}(x_{k+1}|x_k) = \mathcal{N}(x_{k+1}; x_k + \gamma f^n_k(x_k), 2\gamma I_d)$ where $p^0 = p, f^0_k = f$, then

$$\boxed{q^n_{k|k+1}(x_k|x_{k+1}) \approx \mathcal{N}(x_k; x_{k+1} + \gamma b^n_{k+1}(x_{k+1}), 2\gamma I_d)\ ,}$$

  with $b^n_{k+1}(x_{k+1}) = -f^n_k(x_{k+1}) + 2\nabla \log p^n_{k+1}(x_{k+1})$.
- Similarly, we have

$$\boxed{p^{n+1}_{k+1|k}(x_{k+1}|x_k) \approx \mathcal{N}(x_{k+1}; x_k + \gamma f^{n+1}_k(x_k), 2\gamma I_d)\ ,}$$

  with $f^{n+1}_k(x_k) = -b^n_{k+1}(x_k) + 2\nabla \log q^n_k(x_k)$
- **Problem:** if we store the **score networks** they **accumulate**. Memory issue at step n we need to store $2n$ networks!

Approximating IPF via Mean Matching

- We change the **score-matching** to a **mean-matching** regression. This allows us to update only 2 networks.

Mean-matching (De Bortoli et al., 2021)

- Let $B_{k+1}^n(x) = x + \gamma_{k+1} b_{k+1}^n(x)$, $F_k^n(x) = x + \gamma_{k+1} f_k^n(x)$ and

$$q_{k|k+1}^n(x_k|x_{k+1}) = \mathcal{N}(x_k; B_{k+1}^n(x_{k+1}), 2\gamma I_d),$$
$$p_{k+1|k}^n(x_{k+1}|x_k) = \mathcal{N}(x_{k+1}; F_k^n(x_k), 2\gamma I_d).$$

- We have

$$B_{k+1}^n = \arg\min_{\mathrm{B}} \; \mathbb{E}_{p_{k,k+1}^n}[||\mathrm{B}(X_{k+1}) - (X_{k+1} + F_k^n(X_k) - F_k^n(X_{k+1}))||^2],$$
$$F_k^{n+1} = \arg\min_{\mathrm{B}} \; \mathbb{E}_{q_{k,k+1}^n}[||\mathrm{F}(X_k) - (X_k + B_{k+1}^n(X_{k+1}) - B_{k+1}^n(X_k))||^2].$$

- We use **neural networks** $B_{\beta^n}(k,x) \approx B_k^n(x)$ and $F_{\alpha^n}(k,x) \approx F_k^n(x)$, *i.e.* we have one forward and one backward neural net.

- PROOF OF THIS RESULT

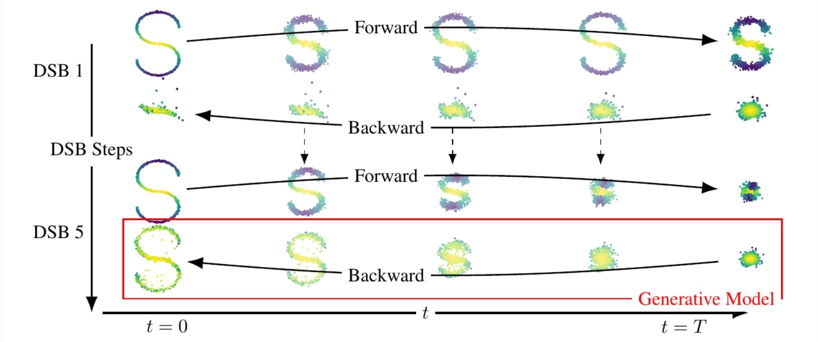
Diffusion Schrödinger Bridge

Algorithm 1 Diffusion Schrödinger Bridge

for $n \in \{0, \ldots, L\}$ **do**
    while not converged **do**
        Sample $\{X_k^j\}_{k,j=0}^{N,M}$, where $X_0^j \sim p_{\text{data}}$, and $X_{k+1}^j = F_{\alpha^n}(k, X_k^j) + \sqrt{2\gamma_{k+1}} Z_{k+1}^j$
        Compute $\hat{\ell}_n^b(\beta^n)$ approximating (12)
        $\beta^n \leftarrow$ Gradient Step$(\hat{\ell}_n^b(\beta^n))$
    end while
    while not converged **do**
        Sample $\{X_k^j\}_{k,j=0}^{N,M}$, where $X_N^j \sim p_{\text{prior}}$, and $X_{k-1}^j = B_{\beta^n}(k, X_k^j) + \sqrt{2\gamma_k} \tilde{Z}_k^j$
        Compute $\hat{\ell}_{n+1}^f(\alpha^{n+1})$ approximating (13)
        $\alpha^{n+1} \leftarrow$ Gradient Step$(\hat{\ell}_{n+1}^f(\alpha^{n+1}))$
    end while
end for
Output: (α^{L+1}, β^L)

Sample generation: $X_N \sim p_{\text{prior}}$ and $X_{k-1} = B_{\beta^L}(k, X_k) + \sqrt{2\gamma_k} Z_k$.

Diffusion Schrödinger Bridge: 2D example



- Diffusion Schrödinger Bridge (DSB) gives a solution to the **"small time problem"**.
- (Approximation of **Optimal Transport**).

Continuous time IPF and normalizing flows

Continuous-Time IPF

- IPF can be formulated in **continuous time**

$$\Pi^\star = \arg\min\{\mathrm{KL}(\Pi||\mathbb{P}) : \Pi \in \mathcal{P}(\mathcal{C}), \Pi_0 = p_{\text{data}}, \Pi_T = p_{\text{prior}}\} .$$

Similarly, we define the IPF (Π^n) recursively $\Pi^0 = \mathcal{P}$ using

$$\Pi^{2n+1} = \arg\min\{\mathrm{KL}(\Pi||\Pi^{2n}) : \Pi \in \mathcal{P}(\mathcal{C}), \Pi_T = p_{\text{prior}}\} ,$$
$$\Pi^{2n+2} = \arg\min\{\mathrm{KL}(\Pi||\Pi^{2n+1}) : \Pi \in \mathcal{P}(\mathcal{C}), \Pi_0 = p_{\text{data}}\} .$$

- Under regularity conditions, then

$$(\Pi^{2n+1})^R \to \mathrm{d}\mathbf{Y}_t^{2n+1} = b_{T-t}^n(\mathbf{Y}_t^{2n+1})\mathrm{d}t + \sqrt{2}\mathrm{d}\mathbf{B}_t, \mathbf{Y}_0^{2n+1} \sim p_{\text{prior}} ,$$
$$\Pi^{2n+2} \to \mathrm{d}\mathbf{X}_t^{2n+2} = f_t^{n+1}(\mathbf{X}_t^{2n+2})\mathrm{d}t + \sqrt{2}\mathrm{d}\mathbf{B}_t, \mathbf{X}_0^{2n+2} \sim p_{\text{data}} ,$$

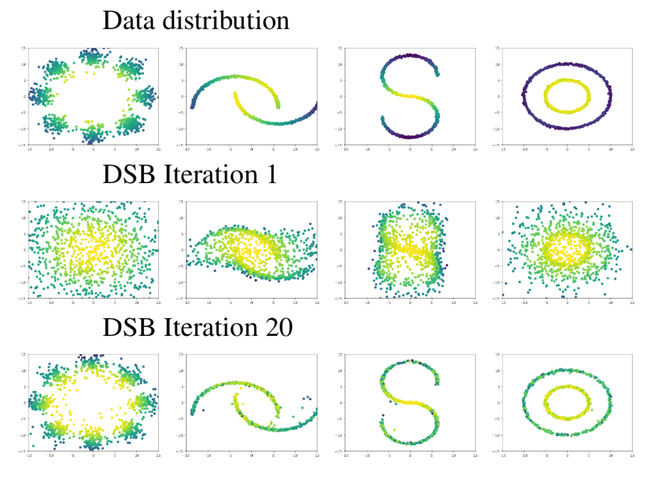
- where

$$b_t^n(x) = -f_t^n(x) + 2\nabla \log p_t^n(x) ,$$
$$f_t^{n+1}(x) = -b_t^n(x) + 2\nabla \log q_t^n(x) ,$$

with $f_t^0(x) = f(x)$, and p_t^n, q_t^n the densities of Π_t^{2n} and Π_t^{2n+1}.

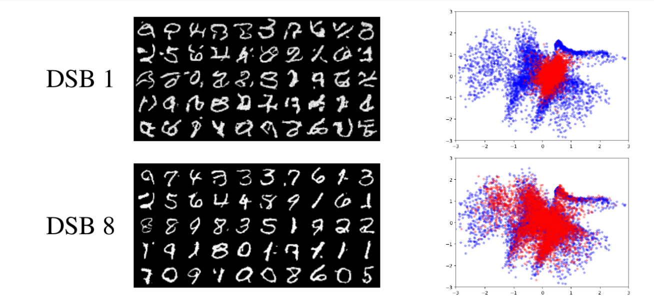
Some experiments

Applications: 2D distributions



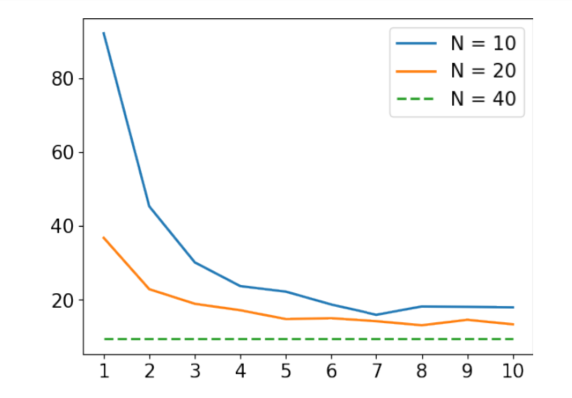
- Data distributions p_{data} VS distribution at $t = 0$ for $T = 0.2$ after 1 and 20 DSB steps

Applications: MNIST



- Generated samples ($N = 12$) and two-dimensional visualization of samples (red) compared to original MNIST data (blue) using pre-trained VAE ($d = 784$).

Applications: MNIST



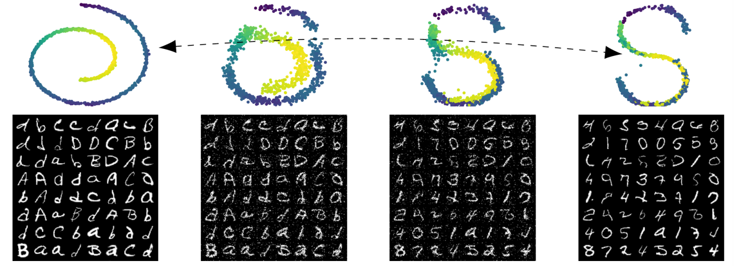
- Fréchet Inception Score vs DSB steps. Green line: FID obtained with 1 DSB step and $N = 40$

Applications: Downscaled CelebA

$t = 0$ $t = 0.31$ $t = 0.60$ $t = 0.63$

- Generative model for CelebA after 10 DSB steps with $N = 50$, $T = 0.63$ ($d = 32 \times 32 \times 3 = 3072$).

Applications: Datasets Interpolation



- First row: Swiss-roll to S-curve (2D). Step 9 of DSB with $T = 1$ ($N = 50$). From left to right: $t = 0, 0.4, 0.6, 1$. Second row: EMNIST to MNIST. Step 10 of DSB with $T = 1.5$ ($N = 30$). From left to right: $t = 0, 0.4, 1.25, 1.5$.

Conditional Schrödinger Bridge

Conclusion

Conclusion

- We have introduced the **Schrödinger Bridge** framework.
  - Introduction of **Diffusion Schrödinger Bridge**.
  - **Implementation** of DSB.
  - **Extensions** of DSB.

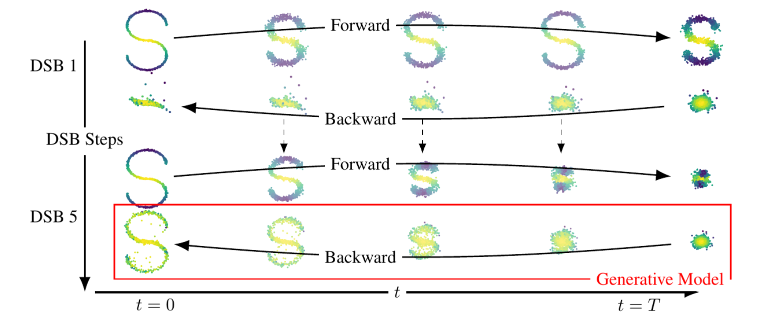


Figure 5: Diffusion Schrödinger Bridge. Image extracted from De Bortoli et al. (2021).

Thank you all!

References

Espen Bernton, Jeremy Heng, Arnaud Doucet, and Pierre E Jacob. Schrodinger bridge samplers. *arXiv preprint arXiv:1912.13170*, 2019.

Valentin De Bortoli, James Thornton, Jeremy Heng, and Arnaud Doucet. Diffusion schrödinger bridge with applications to score-based generative modeling. *NeurIPS*, 2021.

Marcel Nutz. Introduction to entropic optimal transport, 2021.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *ICLR*, 2021.