Score-based generative modeling (introduction & practice)

Valentin De Bortoli

February 28, 2022

Summary of the previous lecture

- The previous lecture was an **overview** of **generative modeling**:
  - Energy-based models (EBMs).
  - Variational Autoencoders (VAEs).
  - Normalizing flows.
  - Generative Adversarial Networks (GANs).

- **Strengths and limitations of these methods**.
  - **Link** between EBM and Maximum Entropy Models.
  - VAEs, ELBO and **latent representations**.
  - **Combining VAE with Normalizing Flows** yield competitive generative models.
  - ODEs, SDEs, Fokker-Planck equation and **Continuous Normalizing Flows**.
  - **State-of-the art results with GANs**.

Introduction of score-based generative models

- A new **contender**:
  - **Score-Based Generative Models** (SGM), also called **diffusion models**.
- Having a hard time keeping up with the literature?
  - List of references: https://scorebasedgenerativemodeling.github.io/
- Advantages of the method:
  - **State-of-the-art** results Dhariwal and Nichol (2021).
  - **Theoretical analysis** is possible De Bortoli et al. (2021b).
- Some drawbacks:
  - Naive implementation is **slow at sampling** times.
  - **Statistical understanding** is still limited.

Figure 1: SGM results. Image extracted from Dhariwal and Nichol (2021).

State-of-the-art results

- State-of-the art results on **CIFAR-10**.
- Recently, state-of-the-art results on **LSUN** (currently hold by Sauer et al. (2021)).
- In practice: good **fidelity** and good **diversity**.
- Have a look at https://paperswithcode.com/task/image-generation for latest advances.

Figure 2: Evolution of FID score (16/02/2022).

Outline of the course

- **Goal of the course**:
  - Introduce SGM in with time-reversal (without relying on stochastic calculus).
  - Present link with other models.

- **Outline of the course**:
  - Introduction of SGM with discrete-time reversal.
  - SGM and EBMs.
  - Introduction of SGM with variational approaches.

Figure 3: Noising process in SGM. Image extracted from Song et al. (2020b).

Discrete time-reversal and score-based generative modeling

Outline of the section

- In this section we introduce SGM in a **"direct" manner**.
- A bit of "history":
  - First paper (variational approach) Sohl-Dickstein et al. (2015).
  - First successful application Song and Ermon (2019).
  - Concurrently (variational approach) Ho et al. (2020).
- Our presentation is inspired from Song et al. (2020b) and De Bortoli et al. (2021b).
  - Everything is presented in **discrete-time**.
  - Next lecture we will look at a **continuous-time** version.
  - No **variational** interpretation here
- We present some **techniques** to train SGM.
- In what follows:
  - **Time-reversal in discrete-time**.
  - Links with **annealed Langevin**.
  - **Implementation details and tricks**.

Time-reversal in discrete-time

Principles of SGM

Figure 4: Noising and generative processes in SGM. Image extracted from Song et al. (2020b).

- **Interpolating** between two distributions:
  - The data distribution is denoted $\pi \in \mathcal{P}(\mathbb{R}^d)$.
  - The easy-to-sample distribution is denoted $\pi_0 \in \mathcal{P}(\mathbb{R}^d)$.
  - π_0 is usually the standard Gaussian.
- Going from the data to the easy-to-sample distribution: **noising process**.
- Going from the easy-to-sample to the data distribution: **generative process**.
- How to **invert** the forward noising process?

Ancestral sampling

- Let $N \in \mathbb{N}$ with $N > 0$ and consider p a density on $(\mathbb{R}^d)^{N+1}$ such that for any $x_{0:N} = \{x_k\}_{k=0}^N$ we have

$$\boxed{p(x_{0:N}) = p_0(x_0) \prod_{k=0}^{N-1} p_{k+1|k}(x_{k+1}|x_k) \ .}$$

- This the **forward** decomposition of p.
- For any $k \in \{0, \ldots, N-1\}$ we define the **marginal** p_{k+1} for any $x_{k+1} \in \mathbb{R}^d$

$$p_{k+1}(x_{k+1}) = \int_{\mathbb{R}^d} p_k(x_k) p_{k+1|k}(x_{k+1}|x_k) \mathrm{d}x_k \ .$$

- Assume that for any $k \in \{0, \ldots N\}$, $p_k > 0$ and define $p_{k|k+1}$ for any $x_k, x_{k+1} \in \mathbb{R}^d$

$$\boxed{p_{k|k+1}(x_k|x_{k+1}) = p_{k+1|k}(x_{k+1}|x_k) p_k(x_k) / p_{k+1}(x_{k+1}) \ .}$$

- We obtain the **backward** decomposition

$$\boxed{p(x_{0:N}) = p_N(x_N) \prod_{k=0}^{N-1} p_{k|k+1}(x_k|x_{k+1}) \ .}$$

The noising process (1/2)

- The **ancestral sampling** procedure allows to sample from p **backward**.
  - Access to the **backward transitions** $\{p_{k|k+1}\}_{k=0}^{N-1}$?
  - Tractability of the **forward transitions**?
- In practice we consider:
  - π (the data distribution) admits a density p_0 w.r.t. the Lebesgue measure.
  - The **forward decomposition** is a **noising process**

$$\boxed{p(x_{0:N}) = p_0(x_0) \prod_{k=0}^{N-1} p_{k+1|k}(x_{k+1}|x_k) \ .}$$

- How do we go from the data distribution to the easy-to-sample distribution?
  - **Ornstein-Ulhenbeck** process: $\mathrm{d}\mathbf{X}_t = -\mathbf{X}_t \mathrm{d}t + \sqrt{2}\mathrm{d}\mathbf{B}_t$.
  - **Euler-Maruyama** discretization: $X_{k+1} = (1-\gamma)X_k + \sqrt{2\gamma}Z_{k+1}$.
  - $\{Z_k\}_{k\in\mathbb{N}}$ i.i.d. $\mathrm{N}(0, \mathrm{Id})$ Gaussian.

The noising process (2/2)

- Let X_0, Y_0 with distribution $\pi_1, \pi_2 \in \mathcal{P}(\mathbb{R}^d)$ and consider the recursion

$$X_{k+1} = (1-\gamma)X_k + \sqrt{2\gamma}Z_{k+1} \ ,$$
$$Y_{k+1} = (1-\gamma)Y_k + \sqrt{2\gamma}Z_{k+1} \ .$$

- We study $u_k = \mathbb{E}[\|X_k - Y_k\|^2]$
  - This is called a **coupling** for the Markov chain.
  - Here we add the same noise. This is called a **synchronous** coupling.
- We have that $u_k \leq (1-\gamma)^{2k} u_0$. Hence, we get that

$$\boxed{\mathbf{W}_2(\pi_1 \mathrm{R}_k, \pi_2 \mathrm{R}_k) \leq (1-\gamma)^k u_0^{1/2} \ .}$$

  - R_k is the **Markov kernel** associated with X_k.
  - $\pi \mathrm{R}(\mathrm{A}) = \int_{\mathbb{R}^d} \mathrm{R}(x, \mathrm{A}) \mathrm{d}\pi(x)$.
  - $\pi_\infty \mathrm{R} = \pi_\infty$ if (and only if) $\pi_\infty = \mathrm{N}(0, \mathrm{Id}/(1-\gamma/2))$.
- We have that for any $\pi \in \mathcal{P}(\mathbb{R}^d)$ and $k \in \mathbb{N}$, $\mathbf{W}_2(\pi \mathrm{R}_k, \pi_\infty) \leq C_0 (1-\gamma)^k$
- C_0 is a constant depending on π and π_∞.

Inverting the noising process (1/3)

- **Euler-Maruyama** discretization of the **Ornstein-Ulhenbeck** process converges **exponentially** fast towards $\mathrm{N}(0, \mathrm{Id}/(1-\gamma/2))$.
- Now let us try to **invert** the forward noising process.

$$
\begin{aligned}
&p_{k|k+1}(x_k|x_{k+1}) = p_{k+1|k}(x_{k+1}|x_k)p_k(x_k)/p_{k+1}(x_{k+1}) \\
&= C_{0,k}\exp[-\|x_{k+1}-(1-\gamma)x_k\|^2/(4\gamma)]\exp[\log(p_k(x_k)) - \log(p_{k+1}(x_{k+1}))] \\
&= C_{1,k}\exp[-\|x_{k+1}-(1-\gamma)x_k\|^2/(4\gamma)]\exp[\log(p_k(x_k)) - \log(p_k(x_{k+1}))] \\
&= C_{1,k}\exp[-(\|x_{k+1}-(1-\gamma)x_k\|^2 + 4\gamma\{\log(p_k(x_k)) - \log(p_k(x_{k+1}))\})/(4\gamma)] \ .
\end{aligned}
$$

  - $C_{0,k}, C_{1,k} > 0$ constants which depend only on x_{k+1}.
  - $\|x_{k+1}-(1-\gamma)x_k\|^2 = \|x_k-(1+\gamma)x_{k+1}\|^2 - 2\gamma\|x_{k+1}-x_k\|^2 + \gamma^2\{\|x_k\|^2 - \|x_{k+1}\|^2\}$.
  - $\log(p_k(x_k)) = \log(p_k(x_{k+1})) + \langle \nabla \log p_k(x_{k+1}), x_k - x_{k+1}\rangle + \int_0^1 \nabla^2 \log p_k((1-t)x_{k+1} + tx_k)(x_k - x_{k+1})^{\otimes 2}\mathrm{d}t$.

Inverting the noising process (2/3)

- **Assumption:** $\|x_{k+1} - x_k\|^2 \leq C\gamma$ and $\max(\|x_k\|, \|x_{k+1}\|) \leq C$
  - $\left| \|x_{k+1} - (1-\gamma)x_k\|^2 - \|x_k - (1+\gamma)x_{k+1}\|^2 \right| \leq 4C\gamma^2$.
  - $|\log(p_k(x_k)) - \log(p_{k+1}(x_{k+1})) - \langle \nabla \log p_k(x_{k+1}), x_k - x_{k+1} \rangle| \leq D\gamma$.
- Hence, we get that

$$
\begin{aligned}
&|\|x_{k+1} - (1-\gamma)x_k\|^2 + 4\gamma\{\log(p_k(x_k)) - \log(p_{k+1}(x_{k+1}))\} \\
&\qquad - \|x_k - (1+\gamma)x_{k+1}\|^2 - 4\gamma\langle \nabla \log p_k(x_{k+1}), x_k - x_{k+1} \rangle| \leq 4(C+D)\gamma^2 \ .
\end{aligned}
$$

- We also have $\|x_k - (1+\gamma)x_{k+1}\|^2 = \|x_k - x_{k+1} - \gamma x_{k+1}\|^2 = \gamma^2\|x_{k+1}\|^2 + \|x_k - x_{k+1}\|^2 - 2\gamma\langle x_{k+1}, x_k - x_{k+1} \rangle$.
- Hence, we have

$$
\begin{aligned}
&\|x_k - (1+\gamma)x_{k+1}\|^2 + 4\gamma\langle \nabla \log p_k(x_{k+1}), x_k - x_{k+1} \rangle \\
&\qquad = \|x_k - x_{k+1}\|^2 + \gamma^2\|x_{k+1}\|^2 + 2\gamma\langle -x_{k+1} + 2\nabla \log p_k(x_{k+1}), x_k - x_{k+1} \rangle \\
&\qquad = \|x_k - x_{k+1} - \gamma x_{k+1} - 2\gamma\nabla \log p_k(x_{k+1})\|^2 \\
&\qquad\qquad + \gamma^2(\|x_{k+1}\|^2 + \|x_{k+1} + 2\nabla \log p_k(x_{k+1})\|^2) \ .
\end{aligned}
$$

Inverting the noising process (3/3)

- We have

$$\begin{aligned}\|x_k - (1+\gamma)x_{k+1}\|^2 + 4\gamma\langle \nabla \log p_k(x_{k+1}), x_k - x_{k+1}\rangle \\ = \|x_k - x_{k+1} - \gamma x_{k+1} - 2\gamma \nabla \log p_k(x_{k+1})\|^2 \\ + \gamma^2(\|x_{k+1}\|^2 + \|x_{k+1} + 2\nabla \log p_k(x_{k+1})\|^2)\ .\end{aligned}$$

- Hence, there exists $E \geq 0$ such that

$$\begin{aligned}|\|x_{k+1} - (1-\gamma)x_k\|^2 + 4\gamma\{\log(p_k(x_k)) - \log(p_{k+1}(x_{k+1}))\} \\ - \|x_k - x_{k+1} - \gamma x_{k+1} - 2\gamma \nabla \log p_k(x_{k+1})\|^2 | \leq E\gamma^2\ .\end{aligned}$$

- Hence, we get that

$$p_{k|k+1}(x_k|x_{k+1}) \approx C_\gamma^{-1} \exp[-\|x_k - x_{k+1} - \gamma\{x_{k+1} + 2\gamma \log p_k(x_{k+1})\}\|^2/(4\gamma)]$$

  - The **approximation** is up to a term of order γ in the exponential.
  - $C_\gamma = (4\pi\gamma)^{-d/2}$
- **Sampling** from the **backward chain**: $X_N \sim p_N$

$$\boxed{X_k = X_{k+1} + \gamma\{X_{k+1} + 2\nabla \log p_k(X_{k+1})\} + \sqrt{2\gamma} Z_{k+1}\ .}$$

- $\nabla \log p_k$ is **untractable**. We are going to approximate this term.

Score-matching (1/4)

- The term $\nabla \log p_k$ is the called the **(Stein) score**.
- Literature on **score matching** Hyvärinen and Dayan (2005); Vincent (2011)
- We have the following identity Efron (2011)

$$\begin{aligned}\nabla \log p_k(x_k) &= \nabla p_k(x_k)/p_k(x_k) \\ &= \int_{\mathbb{R}^d} \nabla \log p_{k|0}(x_k|x_0) p_{0,k}(x_0, x_k) \mathrm{d}x_0 / p_k(x_k) \\ &= \int_{\mathbb{R}^d} \nabla \log p_{k|0}(x_k|x_0) p_{0|k}(x_0|x_k) \mathrm{d}x_0 \ .\end{aligned}$$

- This can be rewritten

$$\boxed{\nabla \log p_k(x_k) = \mathbb{E}_{p_{0|k}(\cdot|x_k)}[\nabla \log p_{k|0}(x_k|X_0)] \ .}$$

- An **intermediate expression**:
  - $\nabla \log p_{k|0}(x_k|x_0)$ is tractable (forward transition).
  - The **conditional expectation** is not (backward conditional).
- We are going to use the property of the conditional expectation to obtain a **loss function**.

Score matching (2/4)

- We use the following properties of the **conditional expectation**:
  - $Y = \mathbb{E}[X \mid \mathcal{F}]$ if and only if $Y = \arg\min\{\mathbb{E}[\|X - Z\|^2] \ : \ Z \in \mathrm{L}^2(\mathcal{F})\}$.
  - $Y \in \sigma(X)$ if and only if there exists $f : \mathbb{R}^d \to \mathbb{R}^d$ (measurable) with $Y = f(X)$.
  - $Y = \mathbb{E}[X \mid U]$ if $Y = f(U)$, with $f = \arg\min\{\mathbb{E}[\|X - f(U)\|^2] \ : \ f \in \mathrm{L}^2(U)\}$.
- Recall that we have

$$\boxed{\nabla \log p_k(X_k) = \mathbb{E}[\nabla \log p_{k|0}(X_k|X_0)|X_k] \ .}$$

- Using the previous property we have

$$\boxed{\nabla \log p_k = \arg\min\{\mathbb{E}[\|f(X_k) - \nabla \log p_{k|0}(X_k|X_0)\|^2] \ : \ f \in \mathrm{L}^2(p_k)\} \ .}$$

- We obtain a **loss function**:
  - $\nabla \log p_{k|0}(x_k|x_0)$ is tractable (forward transition).
  - The expectation can be approximated with **Monte Carlo** (joint distribution).
- Note that this is valid for $k \in \{0, \ldots, N-1\}$.

Score matching (3/4)

- Recall that the loss function is given by

$$\boxed{\nabla \log p_k = \arg\min\{\mathbb{E}[\|f(X_k) - \nabla \log p_{k|0}(X_k|X_0)\|^2] \ : \ f \in \mathrm{L}^2(p_k)\} \ .}$$

- This **loss function** is called the **D**enoising **S**core **M**atching loss.
  - $\log p_{k|0}(x_k|x_0) = -\|x_k - m_k x_0\|^2/(2\sigma_k^2) + C_k$
  - $m_k = (1-\gamma)^k$, $\sigma_k^2 = \{1-(1-\gamma)^{2k}\}/(1-\gamma/2)$, C_k independent of x_k.
  - $\nabla \log p_{k|0}(x_k|x_0) = -(x_k - m_k x_0)/\sigma_k^2$.
  - $X_k = m_k X_0 + \sqrt{2\gamma}\sum_{j=1}^k (1-\gamma)^{k-j} Z_k = m_k X_0 + \sigma_k \hat{Z}_{j+1}$, $\hat{Z}_k \sim \mathrm{N}(0, \mathrm{Id})$.
  - $\nabla \log p_{k|0}(X_k|X_0) = -\hat{Z}_k/\sigma_k$.
  - f tries to **predict the residual noise**.
- Another formulation: **I**mplicit **S**core **M**atching (**divergence theorem**):

$$\begin{aligned}
\mathbb{E}[\,\langle f(X_k), \nabla \log p_{k|0}(X_k|X_0)\rangle \,|\, X_0] &= \int_{\mathbb{R}^d} \langle f(x_k), \nabla \log p_{k|0}(x_k|X_0)\rangle p_{k|0}(x_k|X_0)\mathrm{d}x_k \\
&= \int_{\mathbb{R}^d} \langle f(x_k), \nabla \log(x_k|X_0)\rangle \mathrm{d}x_k \\
&= -\int_{\mathbb{R}^d} \mathrm{div}(f(x_k)) p_{k|0}(x_k|X_0)\mathrm{d}x_k \\
&= -\mathbb{E}[\,\mathrm{div}(f(X_k)) \,|\, X_0] \ .
\end{aligned}$$

Score matching (4/4)

- Recall that the **DSM loss function** is given by

$$\boxed{\nabla \log p_k = \arg\min\{\mathbb{E}[\|f(X_k) - \nabla \log p_{k|0}(X_k|X_0)\|^2] \ : \ f \in \mathrm{L}^2(p_k)\} \ .}$$

- We have $\mathbb{E}[\langle f(X_k), \nabla \log p_{k|0}(X_k|X_0)\rangle] = -\mathbb{E}[\mathrm{div}(f(X_k))]$.
- Hence we have

$$\begin{aligned}\mathbb{E}[\|f(X_k) - \nabla \log p_{k|0}(X_k|X_0)\|^2] \\ = \mathbb{E}[\|f(X_k)\|^2 + 2\mathrm{div}(f(X_k))] + \mathbb{E}[\|\nabla \log p_{k|0}(X_k|X_0)\|^2] \ .\end{aligned}$$

- We obtain the **I**mplicit **S**core **M**atching loss function

$$\boxed{\nabla \log p_k = \arg\min\{\mathbb{E}[(1/2)\|f(X_k)\|^2 + \mathrm{div}(f(X_k))] \ : \ f \in \mathrm{L}^2(p_k)\} \ .}$$

- Comparison between ISM/DSM:
  - **DSM**: access to $\nabla \log p_{k|0}$.
  - **ISM**: no need of the **transition density** but computation of a **divergence**.
  - Approximation with the **Hutchinson estimator**.

Training algorithm

- We choose the **DSM** or **ISM** loss for all $k \in \{1, \ldots, N\}$
  - $\mathrm{DSM}_k(f) = \mathbb{E}[\|f(X_k) - \nabla \log p_{k|0}(X_k|X_0)\|^2]$.
  - $\mathrm{ISM}_k(f) = \mathbb{E}[(1/2)\|f(X_k)\|^2 + \mathrm{div}(f(X_k))]$.

- Defining the **integrated** loss:
  - $\ell^{\mathrm{DSM}}(f) = \sum_{k=1}^{N} \lambda_k \mathrm{DSM}_k(f(k, \cdot))$,
  - $\ell^{\mathrm{ISM}}(f) = \sum_{k=1}^{N} \lambda_k \mathrm{ISM}_k(f(k, \cdot))$.
  - We define a **weighting** function $\lambda_k \geq 0$.

- Let $\{\mathbf{s}_\theta\}_{\theta \in \Theta}$ a **a parametric family of functions** such that $\mathbf{s}_\theta : \mathbb{R}_+ \times \mathbb{R}^d \to \mathbb{R}^d$.
  - Usually $\{\mathbf{s}_\theta\}_{\theta \in \Theta}$ is a family of **neural networks**.
  - We optimize $\ell^{\mathrm{DSM}}(\theta) = \ell^{\mathrm{DSM}}(\mathbf{s}_\theta)$ or $\ell^{\mathrm{ISM}}(\theta) = \ell^{\mathrm{ISM}}(\mathbf{s}_\theta)$.

Backward sampling

- Recall the goal:
  - Sample from $p(x_{0:N}) = p_N(x_N) \prod_{k=0}^{N-1} p_{k|k+1}(x_k|x_{k+1})$ (**ancestral sampling**).
  - **Approximate backward**

$$\boxed{p_{k|k+1}(x_k|x_{k+1}) \approx C_\gamma^{-1} \exp[-\|x_k - x_{k+1} - \gamma\{x_{k+1} + 2\log p_k(x_{k+1})\}\|^2/(4\gamma)] \ .}$$

  - **Approximation of the score** (**DSM** or **ISM** losses).
- Once $\mathbf{s}_{\theta^\star}$ is learned via DSM or ISM losses, i.e. $\mathbf{s}_{\theta^\star}(k, \cdot) \approx \nabla \log p_k$.
- **Sampling scheme**:
  - $X_N \sim \mathrm{N}(0, \mathrm{Id})$ (approximate sampling from p_N).
  - **Approximate ancestral sampling**

$$\boxed{X_k = X_{k+1} + \gamma\{X_{k+1} + \mathbf{s}_{\theta^\star}(k\gamma, X_{k+1})\} + \sqrt{2\gamma} Z_{k+1} \ .}$$

  - X_1 is approximately distributed according to the **data-distribution**.
- Some remarks:
  - Time-reversal can be obtained in **continuous-time**.
  - Original approach relies on **annealed Langevin** Song and Ermon (2019).
  - Other approaches Ho et al. (2020); Gao et al. (2020).

Score-based generative models in action

Videos extracted from an online tutorial blogpost.

Results on CelebA

Video extracted from an online tutorial blogpost.

Links with annealed Langevin

Failure of the score-estimation

- We now present (one) **original approach** by Song and Ermon (2019).
- Goal: **sampling** from the **data distribution** p_0.
  - **Langevin algorithm**: $X_{k+1} = X_k + \gamma \nabla \log p_0(X_k) + \sqrt{2\gamma} Z_{k+1}$.
  - Estimation of the **Stein score** $\nabla \log p_0$ with ISM

$$\boxed{\nabla \log p_0 = \arg\min\{\mathbb{E}[(1/2)\|f(X)\|^2 + \mathrm{div}(f(X))] \ : \ f \in \mathrm{L}^2(p_0)\} \ .}$$

  - $X \sim p_0$.
- **Problems**:
  - **Slow mixing** with Langevin algorithm (non-convexity Eberle (2016)).
  - **Bad score** approximation.

Figure 5: Image extracted from an online tutorial blogpost.

The power of smoothing

- A solution: **smoothing** the density.
  - **Spreading** the observations lead to **better score estimations**.
  - Smoothing leads to **better landscapes** of the potential and **faster mixing** (removal of spurious minima).
- Problem: we do not target the right density.
  - $p_\sigma = p * \mathrm{N}(0, \sigma)$.
  - We have that $\mathrm{Var}(p_\sigma) = \mathrm{Var}(p) + \sigma^2$.
- **Trade-off**:
  - **Small value** of σ: close to p_0, hard to sample.
  - **Large value** of σ: far from p_0, easy to sample.

Figure 6: Image extracted from an online tutorial blogpost.

The best of both worlds

- The main of idea of Song and Ermon (2019): **annealed Langevin** dynamics.
  - Starting from a large value of σ_T, sample easily using the **Langevin dynamics**.
  - Reduce the value of $\sigma_T > \sigma_{T-1}$ and **warm-start** the new Langevin dynamics with the previous samples.
  - Repeat the procedure with σ_0 very small (close to the target density).
  - This is an **annealed** procedure.

Figure 7: Image extracted from an online tutorial blogpost.

Annealing algorithm

Algorithm 1 Sampling of annealing Langevin dynamics

Input: $\{\sigma_t\}_{t=1}^T$, $\{\gamma\}_{t=1}^T$, K
Initialize $X_T^0 \sim \mathcal{N}(0, \sigma_T \operatorname{Id})$.
for $t = T$ to 1 **do**
  for $k = 0$ to $K - 1$ **do**
    Sample $X_t^{k+1} = X_t^k + \gamma_t \mathbf{s}_\theta(\sigma_t, X_t^k) + \sqrt{2\gamma_t} Z_t^{k+1}$
  end for
  $X_{t-1}^0 = X_t^K$
end for
Return X_0^0.

- If $K = 1$ then it is **equivalent to the time-reversal** except that:
  - $\{\gamma_t\}_{t=1}^T$ is *a priori* unrelated to $\{\sigma_t\}_{t=1}^T$ contrary to the time-reversal approach where we would have $\gamma_t = \gamma$ and $\sigma_t^2 = t\gamma$.
  - Main difference is that the **forward** process is the discretization of a **Brownian motion** and not a **Ornstein-Ulhenbeck** process.
  - $X_{k+1} = X_k - \gamma X_k + \sqrt{2\gamma} Z_{k+1}$ in the Ornstein-Ulhenbeck setting and $X_{k+1} = X_k + \sqrt{2\gamma} Z_{k+1}$ in the Brownian case.

Implementation details and tricks

Careful implementation is necessary

- Originally these models were **hard** to train Song and Ermon (2019), see also this blogpost.
- In what follows we describe a **series of tricks** which greatly facilitate the training of these models. These tricks can be found in Song et al. (2020b); Song and Ermon (2020); Nichol and Dhariwal (2021); Ho and Salimans (2021); De Bortoli et al. (2021a).
- We *do not* discuss the **architecture** here.
- In what follows we discuss the following tricks:
  - Ornstein-Ulhenbeck and discretization
  - Loss function weighting.
  - Exponential Moving Average.
  - Adapted variance and predictor-corrector.
  - Conditional sampling and classifier-free guidance.

Ornstein-Ulhenbeck and discretization

- We have introduced **score-based generative models** as the discretization of a **Ornstein-Ulhenbeck** process:
  - **Target measure** is $\mathrm{N}(0, \mathrm{Id})$ (approximately), the data should be centered and reduced.
  - **Constant** stepsize discretization is *not* what is done in practice.
- In practice we consider a **schedule** on the stepsize:

$$\boxed{X_k = X_{k+1} + \gamma_k \{X_{k+1} + \mathbf{s}_\theta(\textstyle\sum_{j=0}^{k} \gamma_j, X_{k+1})\} + \sqrt{2\gamma_k} Z_{k+1} \ .}$$

  - Linear schedule $\gamma_k = \gamma_{\min} + (\gamma_{\max} - \gamma_{\min})(N-k)/N$ Song et al. (2020b).
  - **Intuition**: we need more stepsizes near the data distribution.
  - Different schedules Song and Ermon (2019); Ho et al. (2020); Nichol and Dhariwal (2021).

Figure 8: Budget of stepsizes. Image extracted from Watson et al. (2021).

Loss function weighting

- In practice a **weighted** version of the **DSM** loss is used.
  - Recall that the **DSM** loss is given by $\mathrm{DSM}_k(f) = \mathbb{E}[\|f(X_k) - \nabla \log p_{k|0}(X_k|X_0)\|^2]$.
  - $\ell^{\mathrm{DSM}}(f) = \sum_{k=1}^{N} \lambda_k \mathrm{DSM}_k(f(k, \cdot))$.
  - $\nabla \log p_{k|0}(X_k|X_0) = -\hat{Z}_k/\sigma_k$
- **Intuition**: $\lambda_k = \sigma_k$ to **stabilize** the loss Song et al. (2020b).
- Additional remarks:
  - In Ho et al. (2020); Nichol and Dhariwal (2021) this loss is given by L_{simple}.
  - Justification with **Girsanov** theory in Song et al. (2021); Huang et al. (2021).
  - Changing the **discretization schedule** is equivalent to do a **time-change** in the original Ornstein-Ulhenbeck process then a **fixed discretization**.
- Another improvement: **unconditional model**
  - $\mathbf{s}_\theta(t, x) = \mathbf{s}_\theta(x)/t$.
  - **Intuition:** in practice $\mathbf{s}_\theta(t, x) \sim_{t \to 0} C(t, x)/t$.
  - Conditional models perform better than unconditional ones Ho et al. (2020).

Exponential Moving Average

- The training of the network is **unstable**.
- To regularize this we consider an **E**xponential **M**oving **A**verage of weights.

$$\boxed{\bar{\theta}_{n+1} = (1 - m)\bar{\theta}_n + m\theta_n \ .}$$

  - The parameter m corresponds to the **forgetting** of the initial conditions.
  - The parameters $\bar{\theta}_K$ are used at **sampling** times (K is the number of training steps).

Figure 9: Training instabilities. Image extracted from Song and Ermon (2020).

Adapted variance and predictor-corrector

- Recall that we consider the following **Euler-Maruyama discretization**

$$\boxed{X_k = X_{k+1} + \gamma_k \{X_{k+1} + \mathbf{s}_\theta(\textstyle\sum_{j=0}^k \gamma_j, X_{k+1})\} + \sqrt{2\gamma_k} Z_{k+1} \ .}$$

- Instead of a classical Euler-Maruyama discretization we can consider a **Modified Euler-Maruyama** scheme Durham and Gallant (2002).
  - Replace the term $2\gamma_k$ by $2\gamma_k \{\sum_{j=1}^{k-1} \gamma_j / \sum_{j=1}^k \gamma_j\}$.
  - This discretization scheme can be found in the literature on **stochastic bridges** De Bortoli et al. (2021a).
  - **Intuition:** smaller variance near the **data distribution**.
- We can also correct the Euler-Maruyama scheme using the **time-reversal** property.
  - We must have $\mathcal{L}(X_k) \approx p_k$.
  - Hence we go from X_{k+1} to $\hat{X}_k$ with the Euler-Maruyama scheme (**predictor**).
  - We refine $\hat{X}_k$ by running a Langevin chain targeting p_k (**corrector**).

$$\boxed{X_k^0 = \hat{X}_k \ , \qquad X_k^{\ell+1} = X_k^\ell + \delta_k \gamma \mathbf{s}_\theta(\textstyle\sum_{j=0}^k \gamma_j, X_k) + \sqrt{2\delta_k} Z_k^{\ell+1} \ .}$$

  - $\{\delta_k\}_{k=0}^N$ is a sequence of stepsizes and we set $X_k = X_k^L$ ($L \in \mathbb{N}$).

Conditional sampling and classifier-free guidance

- If the data distribution contains **classes** (like MNIST, CIFAR-10, LSUN, ImageNet or CelebA when classifying by attributes) then we can exploit this extra **structure**.
- Define a **conditional score** $\mathrm{DSM}_k(f) = \mathbb{E}[\|f(X_k^c, c) - \nabla \log p_{k|0}(X_k^c | X_0^c)\|^2]$.
  - $c \in \{1, \dots, C\}$ is the class of the image.
  - Then, we can (approximately) sample from the class c by considering $X_N^c \sim \mathrm{N}(0, \mathrm{Id})$

$$\boxed{X_k^c = X_{k+1}^c + \gamma_k \{X_{k+1}^c + \mathbf{s}_\theta(\textstyle\sum_{j=0}^k \gamma_j, X_{k+1}^c, c)\} + \sqrt{2\gamma_k} Z_{k+1} \ .}$$

Figure 10: Class conditional generation. Image extracted from Song et al. (2020b).

- Other improvements with **unconditional guidance** Ho and Salimans (2021) or **classifier guidance** Dhariwal and Nichol (2021).

Other approaches

Links with other models

- Until now we have presented two approaches to derive **score-based generative models** (SGMs) :
  - A discrete-time **time-reversal** approach.
  - An **annealed Langevin** approach.
- The time-reversal approach is now widely used Song et al. (2020b).

- We now present links with other **generative models**:
  - SGMs as **conditional EBMs** Gao et al. (2020).
  - SGMs as **variational autoencoders** Ho et al. (2020).
- The connection with variational autoencoders allows for:
  - **Extension** to discrete settings
  - **Acceleration** of the sampling dynamics Watson et al. (2021)

- In the next sessions we will see links with **normalizing flows** and **GANs**.

Connections with Energy-Based Models

Pitfalls of Energy-Based Models

- We follow the approach of Gao et al. (2020). We recall the basics on **EBM**.
  - Assume that the data distribution is modelled by a **parametric distribution** p_θ such that for any $x \in \mathbb{R}^d$

$$\boxed{p_\theta(x) = \exp[-f_\theta(x) + L(\theta)] \ , \qquad L(\theta) = -\log(\int_{\mathbb{R}^d} \exp[-f_\theta(\tilde{x})]\mathrm{d}\tilde{x}) \ .}$$

  - f_θ is a **neural network** ($\theta \in \Theta$ is a set of parameters).
  - Maximizing the **likelihood**

$$\boxed{\hat{\ell}(\theta) = \hat{\pi}[\log(p_\theta)] = -(1/N)\sum_{k=1}^{N} f_\theta(x^k) + L(\theta) \ .}$$

  - Sampling with a **Langevin dynamics**.
- The distribution is **multimodal** and/or **supported on a manifold**, i.e. it is complicated to sample from it using a Langevin dynamics.

Figure 11: Poor energy estimation. Image extracted from Gao et al. (2020).

Conditional Energy-Based Models

- To alleviate this issue **conditional models**:
  - This idea is also present in **multiscale synthesis** Saharia et al. (2021).
  - **Intuition**: easier to sample progressively from simple models.

Figure 12: Conditional EBM energy. Image extracted from Gao et al. (2020).

- Going from **marginal** to **conditional**.
  - We define $p_N = \mathrm{N}(0, \sigma_N \mathrm{Id})$ (σ_N large).
  - We define $p_{k|k+1,\theta}(x_k|x_{k+1}) \propto \exp[-f_{k,\theta}(x_k) - \|x_k - x_{k+1}\|^2/(4\gamma)]$.
- $p_{k|k+1,\theta}$ models the backward $p_{k|k+1}$ given for any $x_k, x_{k+1} \in \mathbb{R}^d$

$$p_{k|k+1}(x_k|x_{k+1}) = p_{k+1|k}(x_{k+1}|x_k)p_k(x_k)/p_{k+1}(x_{k+1}) \ ,$$
$$p_{k+1|k}(x_{k+1}|x_k) = (4\pi\gamma)^{-d/2} \exp[-\|x_{k+1} - x_k\|^2/(4\gamma)] \ .$$

- Hence $f_{k,\theta}$ approximates p_k (**noisy marginal**).

Recovery likelihood and first order expansion

- We need to optimize $\{f_{k,\theta}\}_{k=0}^{K-1}$. We consider the **recovery likelihood**

$$\boxed{\ell(\theta) = (1/N) \sum_{k=0}^{K-1} \sum_{i=1}^{N} \log p_{k|k+1,\theta}(x_k^i | x_{k+1}^i) \ ,}$$

  - where $\{x_0^i\}_{i=1}^N$ is a i.i.d. sample from the data distribution,
  - where $x_{k+1}^i = x_k^i + \sqrt{2\gamma} Z_k^i$ (**noising process**).
- **First order expansion** of the recovery-likelihood (under the assumption $\|x_{k+1} - x_k\|^2 \leq C\gamma$)

$$\boxed{|f_{k,\theta}(x_k) - f_{k,\theta}(x_{k+1}) - \langle \nabla f_{k,\theta}(x_{k+1}), x_k - x_{k+1} \rangle| \leq D\gamma \ ,}$$

- with $D \geq 0$. Therefore, up to a term of order γ in the exponential we have

$$\boxed{p_{k|k+1,\theta}(x_k | x_{k+1}) \approx D_k \exp[-\|x_k - x_{k+1} + 2\gamma \nabla f_{k,\theta}(x_{k+1})\|^2 / (4\gamma)] \ .}$$

- Link with **score-based generative models**:
  - We obtain that the **recovery likelihood** is the **DSM** loss.
  - The algorithm to sample from p_0 is similar to **ancestral sampling**.
  - Main difference: discretization of a **Brownian motion** and not the **Ornstein-Ulhenbeck** (also possible with the diffusion recovery likelihood model Gao et al. (2020)).

Connections with Variational AutoEncoders

A variational perspective

- We follow the approach of Ho et al. (2020).
- **Variational approach** offers great flexibility:
  - Optimization of the stepsize Watson et al. (2021).
  - Learning of the covariance matrix Nichol and Dhariwal (2021).
  - Non-Markov dynamics Song et al. (2020a).
- Ho et al. (2020) was the first to propose a **discretized Ornstein-Ulhenbeck** Markov chain as a forward process.

Figure 13: CelebA and CIFAR10 results. Image extracted from Ho et al. (2020).

An Evidence Lower BOund (1/2)

- We start by deriving an **ELBO** for the **score-based generative models**. Note that such a derivation was already obtained by Sohl-Dickstein et al. (2015).
- Similar to VAE we maximize the **log-likelihood**

$$\begin{aligned}\log(p_{\theta,0}(x_0)) &= \log(\int_{(\mathbb{R}^d)^N} \prod_{k=0}^{N-1} p_{\theta,k|k+1}(x_k|x_{k+1}) p_N(x_N) \mathrm{d}x_{1:N}) \\ &= \log(\int_{(\mathbb{R}^d)^N} \prod_{k=0}^{N-1} p_{\theta,k|k+1}(x_k|x_{k+1}) p_N(x_N) / q(x_{1:N}|x_0) q(x_{1:N}|x_0) \mathrm{d}x_{1:N}) \\ &\geq \int_{(\mathbb{R}^d)^N} \log(\prod_{k=0}^{N-1} p_{\theta,k|k+1}(x_k|x_{k+1}) p_N(x_N) / q(x_{1:N}|x_0)) q(x_{1:N}|x_0) \mathrm{d}x_{1:N} \ .\end{aligned}$$

- The last inequality is obtained the **concavity** of the logarithm.
- We now choose the **variational distribution** $q(x_{1:N}|x_0)$:
  - **Factorization** $q(x_{1:N}|x_0) = q_{N|0}(x_N|x_0) \prod_{k=1}^{N-1} q_{k|k+1,0}(x_k|x_{k+1}, x_0)$.
  - **Tractability** of $q_{k|k+1,0}$.
  - We choose a tractable (Gaussian) decomposition

  $q(x_{1:N}|x_0) = \prod_{k=0}^{N-1} q_{k+1|k}(x_{k+1}|x_k)$.
- Here, we consider

$$\boxed{q_{k+1|k}(x_{k+1}|x_k) = (4\pi\gamma)^{-d/2} \exp[-\|x_{k+1} - x_k + \gamma x_k\|^2/(4\gamma)] \ .}$$

- This is a slightly different discretization from the one of Ho et al. (2020).

An Evidence Lower BOund (2/2)

- Recall that we have $\log(p_{\theta,0}(x_0)) \geq \mathcal{L}$ with

$$\boxed{\mathcal{L} = \int_{(\mathbb{R}^d)^N} \log\left(\prod_{k=0}^{N-1} p_{\theta,k|k+1}(x_k|x_{k+1}) p_N(x_N) / q(x_{1:N}|x_0)\right) q(x_{1:N}|x_0) \mathrm{d}x_{1:N} \ .}$$

- We use the **backward decomposition** of $q(x_{1:N}|x_0)$ and we get

$$\boxed{\mathcal{L} = \mathcal{L}_N + \sum_{k=1}^{N-1} \mathcal{L}_k + \mathcal{L}_0 \ ,}$$

- with:
  - $\mathcal{L}_N = \int_{\mathbb{R}^d} \log(p_N(x_N)/q(x_N|x_0)) q_{N|0}(x_N|x_0) \mathrm{d}x_N$.
  - $\mathcal{L}_k = \int_{\mathbb{R}^d} \log(p_{\theta,k|k+1}(x_k|x_{k+1})/q_{k|k+1,0}(x_k|x_{k+1}, x_0)) q_{k,k+1|0}(x_k, x_{k+1}|x_0) \mathrm{d}x_k$.
  - $\mathcal{L}_0 = \int_{\mathbb{R}^d} \log(p_{\theta,0|1}(x_0|x_1)) q_{1|0}(x_1|x_0) \mathrm{d}x_1$.
- **The different terms**:
  - $\mathcal{L}_N$ does not depend on θ.
  - $\mathcal{L}_k$ is related to **score-matching**.
  - $\mathcal{L}_0$ is more complicated and will be dealt with later.

The backward $q_{k|k+1,0}$ (1/2)

- To compute $\mathcal{L}_k$ we need to compute $q_{k|k+1,0}$.
- We know that $q_{k|k+1,0}$ is **Gaussian** with **diagonal covariance** and just need to compute its parameter.
- $q_{k|0} = \mathrm{N}(\alpha_k x_0, \sigma_k \mathrm{Id})$ and $q_{k+1|k} = \mathrm{N}(\alpha_{k+1|k}, \sigma_{k+1|k} \mathrm{Id})$.
- **Computing the parameters**:
  - $\alpha_{k+1|k} = 1 - \gamma$, $\sigma^2_{k+1|k} = 2\gamma$.
  - $X_{k+1} = (1-\gamma)^k X_0 + \sqrt{2\gamma} \sum_{j=1}^k (1-\gamma)^{k-j} Z_{j+1} = (1-\gamma)^k X_0 + \sigma_{k+1} \hat{Z}_{k+1}$.
  - $\hat{Z}_{k+1} \sim \mathrm{N}(0, \mathrm{Id})$ and $\sigma^2_{k+1} = (1 - (1-\gamma)^{2k})/(1-\gamma/2)$.
  - $\alpha_{k+1} = (1-\gamma)^k$, $\sigma^2_{k+1} = (1 - (1-\gamma)^{2k})/(1-\gamma/2)$.
- We have that

$$\boxed{q_{k|k+1,0}(x_k|x_{k+1}, x_0) = q_{k+1|k}(x_{k+1}|x_k) q_{k|0}(x_k|x_0) / q_{k+1|0}(x_{k+1}|x_0) \ .}$$

  - We can **discard the denominator** (normalizing constant).
  - We can focus on $\log(q_{k+1|k}(x_{k+1}|x_k) q_{k|0}(x_k|x_0))$.

The backward $q_{k|k+1,0}$ (2/2)

- We have that

$$
\begin{aligned}
&-2\log(q_{k+1|k}(X_{k+1}|X_k)q_{k|0}(X_k|X_0)) \\
&= \|X_{k+1} - \alpha_{k+1|k}X_k\|^2/\sigma^2_{k+1|k} + \|X_k - \alpha_k X_0\|^2/\sigma^2_k \\
&= \{\alpha^2_{k+1|k}/\sigma^2_{k+1|k} + (1/\sigma^2_k)\}\|X_k\|^2 - 2\langle \alpha_{k+1|k}/\sigma^2_{k+1|k}X_{k+1} + \alpha_k/\sigma^2_k X_0, X_k\rangle + C \\
&= (1/\sigma^2_{k|k+1})\|X_k\|^2 \\
&\qquad - 2\langle \{\alpha_{k+1|k}/\sigma^2_{k+1|k} + \alpha_k/(\alpha_{k+1}\sigma^2_k)\}X_{k+1} + \alpha_k\sigma_{k+1}/(\alpha_{k+1}\sigma^2_k)\hat{Z}_{k+1}, X_k\rangle + C \\
&= (1/\sigma^2_{k|k+1})\|X_k\|^2 - (2/\sigma^2_{k|k+1})\langle A_{k|k+1}X_{k+1} - B_{k|k+1}\hat{Z}_{k+1}, X_k\rangle + C \\
&= \|X_k - A_{k|k+1}X_{k+1} + B_{k|k+1}\hat{Z}_{k+1}\|^2/(2\sigma^2_{k|k+1}) + D\ .
\end{aligned}
$$

  - C, D constants **independent** from x_k.
  - $\sigma^2_{k|k+1} = (\alpha^2_{k+1|k}/\sigma^2_{k+1|k} + (1/\sigma^2_k))^{-1}$.
  - $A_{k|k+1} = \alpha_{k+1|k}(\sigma_{k|k+1}/\sigma_{k+1|k})^2 + \alpha_k\sigma^2_{k|k+1}/(\alpha_{k+1}\sigma^2_k)$.
  - $B_{k|k+1} = (\alpha_k\sigma_{k+1}\sigma^2_{k+1|k})/(\alpha_{k+1}\sigma^2_k)$.
- Therefore, we **choose the family**

$$
\boxed{\log(p_{\theta,k|k+1}(x_k|x_{k+1})) = \|x_k - A_{k|k+1}x_{k+1} + B_{k|k+1}\hat{z}_{\theta,k+1}(x_{k+1})\|^2/(2\sigma^2_{k|k+1}) + E\ .}
$$

- E is a constant, $\hat{z}_{\theta,k+1}(x_{k+1})$ is a function of x_{k+1} (**estimator of the noise**).

Sampling from the model

- How to **train** and **sample** the model?
- Recall that we have set

$$\boxed{\log(p_{\theta,k|k+1}(x_k|x_{k+1})) = \|x_k - A_{k|k+1}x_{k+1} + B_{k|k+1}\hat{z}_{\theta,k+1}(x_{k+1})\|^2/(2\sigma^2_{k|k+1}) + E\ .}$$

- Recall that $p_N = \mathrm{N}(0, \mathrm{Id})$. To **sample from the model**:
  - We sample $X_N \sim \mathrm{N}(0, \mathrm{Id})$
  - We consider the **backward update**

$$\boxed{X_k = A_{k|k+1}X_{k+1} - B_{k|k+1}\hat{z}_{\theta,k+1}(X_{k+1}) + \sigma_{k|k+1}Z_{k+1}\ .}$$

- To **train the model** (without the therm $\mathcal{L}_{1|0}$):
  - Minimize $\sum_{k=1}^{N} \ell_k(\theta)$, with

$$\boxed{\ell_k(\theta) = \mathbb{E}[\|X_k - A_{k|k+1}X_{k+1} + B_{k|k+1}\hat{z}_{\theta,k+1}(X_{k+1}) + \sigma_{k|k+1}Z_{k+1}\|^2]\ .}$$

Taylor expansion and comparison with SGM (2/2)

- The model is already **similar to SGM**:
  - We sample from $\mathrm{N}(0, \mathrm{Id})$ and use **ancestral sampling**.
  - We train part of the **drift term**.
- The analogy becomes even stronger when considering **Taylor expansion** of $A_{k|k+1}$, $B_{k|k+1}$ and $\sigma_{k|k+1}$:
  - $A_{k|k+1} = 1 + \gamma + o(\gamma)$.
  - $B_{k|k+1} = 2\gamma + o(\gamma)$.
  - $\sigma^2_{k|k+1} = 2\gamma + o(\gamma)$.
- Hence

$$X_k = A_{k|k+1} X_{k+1} - B_{k|k+1} \hat{z}_{\theta,k+1}(X_{k+1}) + \sigma_{k|k+1} Z_{k+1} \ ,$$

- becomes (up to the first order)

$$\boxed{X_k = (1+\gamma) X_{k+1} - 2\gamma \hat{z}_{\theta,k+1}(X_{k+1}) + \sqrt{2\gamma} Z_{k+1} \ .}$$

- We can identify this recursion with the one of SGM if $\hat{z}_{\theta,k+1} \approx -\nabla \log p_{k+1}$, i.e. the neural network approximates the **score**.

Taylor expansion and comparison with SGM (2/2)

- We want to show that $\hat{z}_{\theta,k+1} \approx -\nabla \log p_{k+1}$, i.e. the neural network approximates the **score**.
- Recall that we minimize the following **loss function**

$$\ell_k(\theta) = \mathbb{E}[\|X_k - A_{k|k+1}X_{k+1} + B_{k|k+1}\hat{z}_{\theta,k+1}(X_{k+1}) + \sigma_{k|k+1}Z_{k+1}\|^2/(2\sigma^2_{k|k+1})] \ .$$

- Up to the first order we get that

$$\boxed{\ell_k(\theta) = \mathbb{E}[\|X_k - (1+\gamma)X_{k+1} + 2\gamma\hat{z}_{\theta,k+1}(X_{k+1})\|^2]/(2\gamma) \ .}$$

- But we have $(1+\gamma)X_{k+1} = (1-\gamma^2)X_k + \sqrt{2\gamma}(1+\gamma)Z_{k+1}$.
- Hence, up to the first order we get that

$$\boxed{\ell_k(\theta) = \mathbb{E}[\|\sqrt{2\gamma}Z_{k+1} + 2\gamma\hat{z}_{\theta,k+1}(X_{k+1})\|^2]/(2\gamma) \ ,}$$

- This is exactly the **Denoising Score Matching** loss (up to a minus term) times λ_k (the **weighting function** appearing score-based models and fixed to $\lambda_k = 2\gamma$).

The term $\mathcal{L}_0$

- The previous recursion is valid up to $k = 1$.
- p_θ is an **independent decoder** on the pixel of the image.
- We assume that $x_0 \in [-1, 1]^d$

$$\boxed{p_\theta(x_0|x_1) = \prod_{i=1}^d \int_{a(x_0^i)}^{b(x_0^i)} \exp[-\|x - \mu_\theta(x_1)\|^2 / \sigma_1^2] / (2\pi\sigma_1^2)^d \mathrm{d}x \ .}$$

- $a(t) = t + 1/255$ if $t < 1$ and $+\infty$ otherwise.
- $b(t) = t - 1/255$ if $t > -1$ and $-\infty$ otherwise.
- We could also have chosen the classical (non-discrete) **decoding Gaussian of the VAE**.

Figure 14: CelebA results. Image extracted from Ho et al. (2020).

Conclusion

Conclusion

- We have introduced **score-based generative models**.
  - Motivation with **state-of-the-art** results.
  - **Ancestral sampling** and time-reversal (discrete-time).
  - **Tricks** and **implementation**.
  - Connection with **EBM** and **VAE**.

- Next time:
  - Score-based generative models in **continuous time** and results.
  - **Normalizing flows** and **Likelihood computation**.
  - **Acceleration** of SGMs.
  - A continuous-time **ELBO** with Girsanov and Feynman-Kac theory.

See you all on the 07/03!

References

Valentin De Bortoli, Arnaud Doucet, Jeremy Heng, and James Thornton. Simulating diffusion bridges with score matching. *arXiv preprint arXiv:2111.07243*, 2021a.

Valentin De Bortoli, James Thornton, Jeremy Heng, and Arnaud Doucet. Diffusion schrödinger bridge with applications to score-based generative modeling. *Advances in Neural Information Processing Systems*, 34, 2021b.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34, 2021.

Garland B Durham and A Ronald Gallant. Numerical techniques for maximum likelihood estimation of continuous-time diffusion processes. *Journal of Business & Economic Statistics*, 20(3):297–338, 2002.

Andreas Eberle. Reflection couplings and contraction rates for diffusions. *Probability theory and related fields*, 166(3):851–886, 2016.

References ii

Bradley Efron. Tweedie's formula and selection bias. *Journal of the American Statistical Association*, 106(496):1602–1614, 2011.

Ruiqi Gao, Yang Song, Ben Poole, Ying Nian Wu, and Diederik P Kingma. Learning energy-based models by diffusion recovery likelihood. *arXiv preprint arXiv:2012.08125*, 2020.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

Chin-Wei Huang, Jae Hyun Lim, and Aaron C Courville. A variational perspective on diffusion-based generative models and score matching. *Advances in Neural Information Processing Systems*, 34, 2021.

Aapo Hyvärinen and Peter Dayan. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(4), 2005.

References iii

Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, pages 8162–8171. PMLR, 2021.

Chitwan Saharia, Jonathan Ho, William Chan, Tim Salimans, David J Fleet, and Mohammad Norouzi. Image super-resolution via iterative refinement. *arXiv preprint arXiv:2104.07636*, 2021.

Axel Sauer, Kashyap Chitta, Jens Müller, and Andreas Geiger. Projected gans converge faster. *Advances in Neural Information Processing Systems*, 34, 2021.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR, 2015.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020a.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in Neural Information Processing Systems*, 32, 2019.

References iv

Yang Song and Stefano Ermon. Improved techniques for training score-based generative models. *Advances in neural information processing systems*, 33: 12438–12448, 2020.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020b.

Yang Song, Conor Durkan, Iain Murray, and Stefano Ermon. Maximum likelihood training of score-based diffusion models. *Advances in Neural Information Processing Systems*, 34, 2021.

Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural Computation*, 23(7):1661–1674, 2011.

Daniel Watson, Jonathan Ho, Mohammad Norouzi, and William Chan. Learning to efficiently sample from diffusion probabilistic models. *arXiv preprint arXiv:2106.03802*, 2021.