Diffusion models for Gaussian distributions: Exact solutions and Wasserstein errors

Émile Pierret[a], supervised by Bruno Galerne[a,b]

Séminaire Imaging in Paris

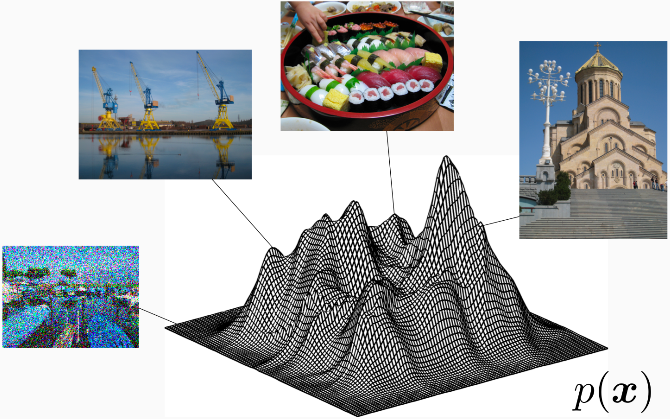
January 6^{th} 2025

[a] Institut Denis Poisson – Université d'Orléans, Université de Tours, CNRS
[b] Institut universitaire de France (IUF)

Introduction

What is a generative model ?

Goal: Sample from a data distribution of images.



$p(\boldsymbol{x})$

CelebA dataset

Dataset samples

50K samples

CelebA dataset

Dataset samples

50K samples

Generated (Fake) samples

Style GAN, (Karras et al., 2018) (NVIDIA)

Challenge: Given a model $G(\cdot;\Theta)$, find $\Theta^\star$ such that $G(\mathcal{N}(\mathbf{0},\boldsymbol{I}_N),\Theta^\star) \approx p$

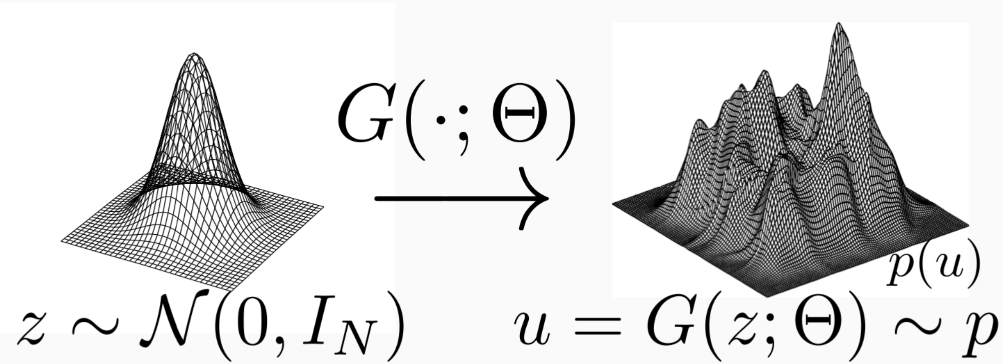


Image extracted from Bruno Galerne's slides

Diffusion models through SDE

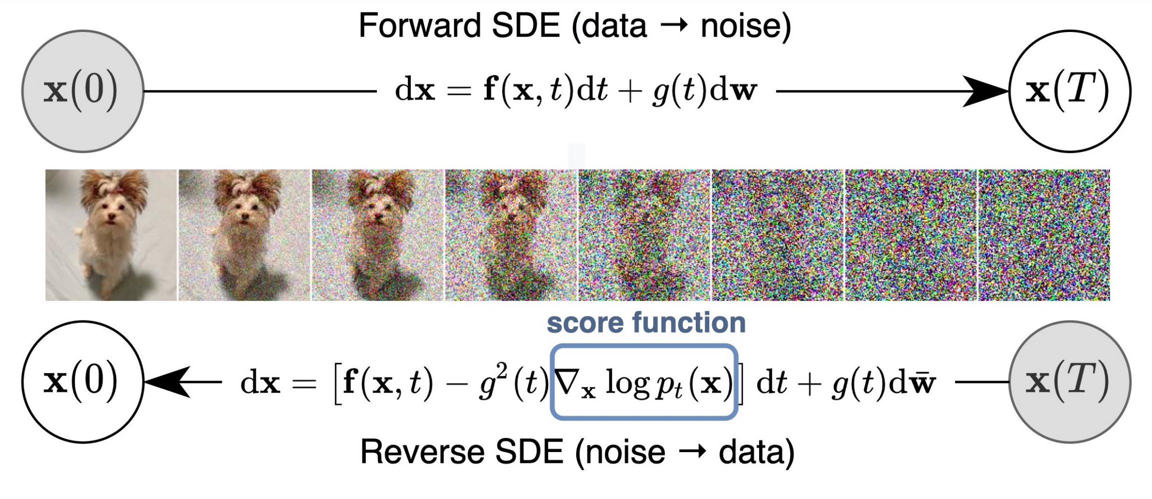


Image extracted from [Song et al. 2021]

Focus on the VP-SDE: the forward process

$$dx_t = -\beta_t x_t dt + \sqrt{2\beta_t} dw_t, \quad 0 \leqslant t \leqslant T, \quad x_0 \sim p_{\text{data}} \tag{1}$$

where β_t is an affine non-decreasing function. We denote $(p_t)_{0<\leqslant t \leqslant T}$ the density of x_t.

Focus on the VP-SDE: the forward process

$$dx_t = -\beta_t x_t dt + \sqrt{2\beta_t} dw_t, \quad 0 \leqslant t \leqslant T, \quad x_0 \sim p_{\text{data}} \tag{1}$$

where β_t is an affine non-decreasing function. We denote $(p_t)_{0<\leqslant t \leqslant T}$ the density of x_t.

The strong solution of Equation (1) is:

$$x_t = e^{-B_t} x_0 + \boldsymbol{\eta}_t, \quad 0 \leqslant t \leqslant T. \tag{2}$$

with $\eta_t \sim \mathcal{N}(\mathbf{0}, \left(1 - e^{-2B_t}\right) \boldsymbol{I}), B_t = \int_0^t \beta_u du$.

Focus on the VP-SDE: the forward process

$$dx_t = -\beta_t x_t dt + \sqrt{2\beta_t} dw_t, \quad 0 \leqslant t \leqslant T, \quad x_0 \sim p_{\text{data}} \tag{1}$$

where β_t is an affine non-decreasing function. We denote $(p_t)_{0<\leqslant t \leqslant T}$ the density of x_t.

The strong solution of Equation (1) is:

$$x_t = e^{-B_t} x_0 + \boldsymbol{\eta}_t, \quad 0 \leqslant t \leqslant T. \tag{2}$$

with $\eta_t \sim \mathcal{N}(\mathbf{0}, \left(1 - e^{-2B_t}\right) \boldsymbol{I}), B_t = \int_0^t \beta_u du$.

Consequently, if $t \to +\infty$, $x_\infty \sim \mathcal{N}_0$.

Focus on the VP-SDE: the forward process

$$dx_t = -\beta_t x_t dt + \sqrt{2\beta_t} dw_t, \quad 0 \leqslant t \leqslant T, \quad x_0 \sim p_{\text{data}} \tag{1}$$

where β_t is an affine non-decreasing function. We denote $(p_t)_{0<\leqslant t \leqslant T}$ the density of x_t.

The strong solution of Equation (1) is:

$$x_t = e^{-B_t} x_0 + \boldsymbol{\eta}_t, \quad 0 \leqslant t \leqslant T. \tag{2}$$

with $\eta_t \sim \mathcal{N}(\mathbf{0}, \left(1 - e^{-2B_t}\right) \boldsymbol{I}), B_t = \int_0^t \beta_u du.$

Consequently, if $t \to +\infty$, $x_\infty \sim \mathcal{N}_0$.

Furthermore, denoting $\boldsymbol{\Sigma}_t = \operatorname{Cov}(x_t)$,

$$d\boldsymbol{\Sigma}_t = 2\beta_t(\boldsymbol{I} - \boldsymbol{\Sigma}_t)dt, \quad 0 < t \leqslant T. \tag{3}$$

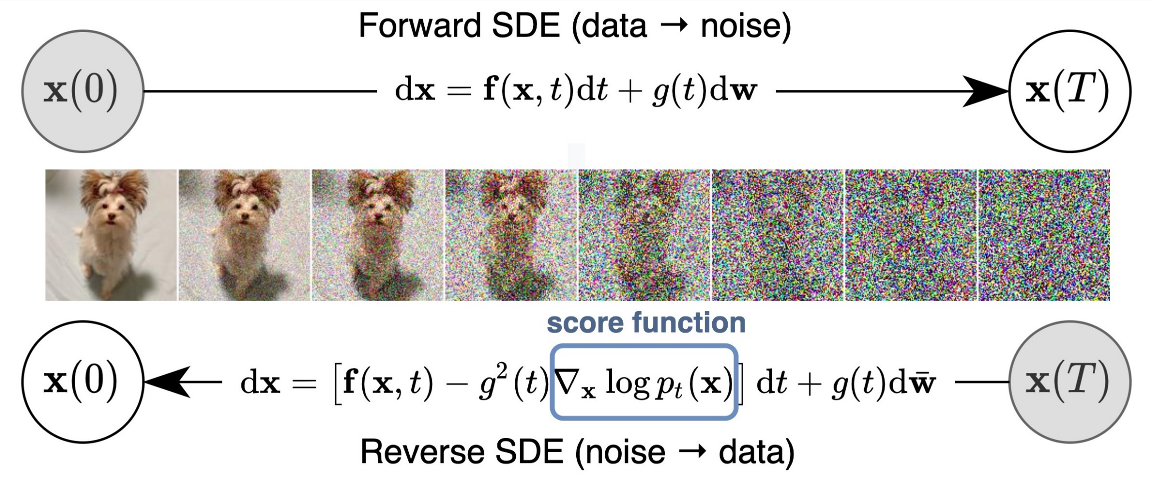


Image extracted from [Song et al. 2021]

[Pardoux 1986][1]

If X_t solution of

$$d\boldsymbol{X}_t = b(t, \boldsymbol{X}_t)dt + \sigma(t, \boldsymbol{X}_t)d\boldsymbol{W}_t \tag{4}$$

Under assumptions

1. **(H1)** $\exists K > 0$ s.t. $\forall (t, x, y) \in [0, 1] \times \mathbb{R}^d \times \mathbb{R}^d$,

$$|b(t, x) - b(t, y)| + |\sigma(t, x) - \sigma(t, y)| \leqslant K|x - y|$$
$$|b(t, x)| + |\sigma(t, x)| \leqslant K(1 + |x|).$$

2. **(H2)** p_{data} has a density distribution in $L^2\left(\mathbb{R}^d, \frac{dx}{1+|x|^k}\right)$ for a certain $k \in \mathbb{N}$.
3. **(H3)** $\frac{\partial^2 \sigma^2}{\partial x_i \partial x_j}(t, x) \in L^\infty(]0, 1[\times \mathbb{R}^d)$ for $1 \leqslant i, j \leqslant d$.

then $\overline{\boldsymbol{X}}_t = \boldsymbol{X}_{1-t}$ is solution of

$$d\overline{\boldsymbol{X}}_t = \bar{b}(t, \overline{\boldsymbol{X}}_t)dt + \overline{\sigma}(t, \overline{\boldsymbol{X}}_t)d\overline{\boldsymbol{W}}_t \tag{5}$$

In our case, $b(t, x) = -\beta_t x, \sigma(t, x) = \sqrt{2\beta_t}$

[1] E. Pardoux (1986). "Grossissement d'une filtration et retournement du temps d'une diffusion". In: *Séminaire de Probabilités XX 1984/85*. Ed. by Jacques Azéma and Marc Yor. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 48–55. ISBN: 978-3-540-39860-8

Study of the backward process [Pardoux 1986][2]

Under some assumptions on the distribution p_{data} [Pardoux 1986], the backward process $(x_{T-t})_{0\leqslant t\leqslant T}$ verifies the backward SDE

$$dy_t = \beta_{T-t}(y_t + 2\nabla \log p_{T-t}(y_t))dt + \sqrt{2\beta_{T-t}}d\overline{w}_t, \quad 0 \leqslant t < T, \quad y_0 \sim p_T. \tag{6}$$

[2] E. Pardoux (1986). "Grossissement d'une filtration et retournement du temps d'une diffusion". In: *Séminaire de Probabilités XX 1984/85*. Ed. by Jacques Azéma and Marc Yor. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 48–55. ISBN: 978-3-540-39860-8

Study of the backward process [Pardoux 1986][2]

Under some assumptions on the distribution p_{data} [Pardoux 1986], the backward process $(x_{T-t})_{0\leqslant t\leqslant T}$ verifies the backward SDE

$$dy_t = \beta_{T-t}(y_t + 2\nabla \log p_{T-t}(y_t))dt + \sqrt{2\beta_{T-t}}d\overline{w}_t, \quad 0 \leqslant t < T, \quad y_0 \sim p_T. \tag{6}$$

- The backward Brownian motion $\overline{w}$ is not defined on the same filtration than the forward w:

$$\overline{w}_t = w_t - w_T + \int_t^T \frac{1}{p(s, x_s)} \operatorname{div}(\sigma p)(s, x_s) ds. \tag{7}$$

[2] E. Pardoux (1986). "Grossissement d'une filtration et retournement du temps d'une diffusion". In: *Séminaire de Probabilités XX 1984/85*. Ed. by Jacques Azéma and Marc Yor. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 48–55. ISBN: 978-3-540-39860-8

Study of the backward process [Pardoux 1986][2]

Under some assumptions on the distribution p_{data} [Pardoux 1986], the backward process $(x_{T-t})_{0\leqslant t\leqslant T}$ verifies the backward SDE

$$dy_t = \beta_{T-t}(y_t + 2\nabla \log p_{T-t}(y_t))dt + \sqrt{2\beta_{T-t}}d\overline{w}_t, \quad 0 \leqslant t < T, \quad y_0 \sim p_T. \tag{6}$$

- The backward Brownian motion $\overline{w}$ is not defined on the same filtration than the forward w:

$$\overline{w}_t = w_t - w_T + \int_t^T \frac{1}{p(s, x_s)} \operatorname{div}(\sigma p)(s, x_s) ds. \tag{7}$$

- $\nabla \log p_{T-t}$ is called the score function.

[2] E. Pardoux (1986). "Grossissement d'une filtration et retournement du temps d'une diffusion". In: *Séminaire de Probabilités XX 1984/85*. Ed. by Jacques Azéma and Marc Yor. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 48–55. ISBN: 978-3-540-39860-8

Study of the backward process [Pardoux 1986][2]

Under some assumptions on the distribution p_{data} [Pardoux 1986], the backward process $(x_{T-t})_{0\leqslant t\leqslant T}$ verifies the backward SDE

$$dy_t = \beta_{T-t}(y_t + 2\nabla \log p_{T-t}(y_t))dt + \sqrt{2\beta_{T-t}}d\overline{w}_t, \quad 0 \leqslant t < T, \quad y_0 \sim p_T. \tag{6}$$

- The backward Brownian motion $\overline{w}$ is not defined on the same filtration than the forward w:

$$\overline{w}_t = w_t - w_T + \int_t^T \frac{1}{p(s, x_s)} \operatorname{div}(\sigma p)(s, x_s)ds. \tag{7}$$

- $\nabla \log p_{T-t}$ is called the score function.
- We are unable to derive the score function: **No closed-form solution !**

[2] E. Pardoux (1986). "Grossissement d'une filtration et retournement du temps d'une diffusion". In: *Séminaire de Probabilités XX 1984/85*. Ed. by Jacques Azéma and Marc Yor. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 48–55. ISBN: 978-3-540-39860-8

Study of the backward process [Pardoux 1986][2]

Under some assumptions on the distribution p_{data} [Pardoux 1986], the backward process $(x_{T-t})_{0\leqslant t\leqslant T}$ verifies the backward SDE

$$dy_t = \beta_{T-t}(y_t + 2\nabla \log p_{T-t}(y_t))dt + \sqrt{2\beta_{T-t}}d\overline{w}_t, \quad 0 \leqslant t < T, \quad y_0 \sim p_T. \tag{6}$$

- The backward Brownian motion $\overline{w}$ is not defined on the same filtration than the forward w:

$$\overline{w}_t = w_t - w_T + \int_t^T \frac{1}{p(s, x_s)} \operatorname{div}(\sigma p)(s, x_s)ds. \tag{7}$$

- $\nabla \log p_{T-t}$ is called the score function.
- We are unable to derive the score function: **No closed-form solution !**
- Learn $s_\theta(x, t) \approx \nabla \log p_t(x)$

[2] E. Pardoux (1986). "Grossissement d'une filtration et retournement du temps d'une diffusion". In: *Séminaire de Probabilités XX 1984/85*. Ed. by Jacques Azéma and Marc Yor. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 48–55. ISBN: 978-3-540-39860-8

Study of the backward process [Pardoux 1986][2]

Under some assumptions on the distribution p_{data} [Pardoux 1986], the backward process $(x_{T-t})_{0\leqslant t\leqslant T}$ verifies the backward SDE

$$dy_t = \beta_{T-t}(y_t + 2\nabla \log p_{T-t}(y_t))dt + \sqrt{2\beta_{T-t}}d\overline{w}_t, \quad 0 \leqslant t < T, \quad y_0 \sim p_T. \tag{6}$$

- The backward Brownian motion $\overline{w}$ is not defined on the same filtration than the forward w:

$$\overline{w}_t = w_t - w_T + \int_t^T \frac{1}{p(s, x_s)} \operatorname{div}(\sigma p)(s, x_s)ds. \tag{7}$$

- $\nabla \log p_{T-t}$ is called the score function.
- We are unable to derive the score function: **No closed-form solution !**
- Learn $s_\theta(x,t) \approx \nabla \log p_t(x)$

$$dy_t = \beta_{T-t}(y_t + 2s_\theta(y_t, T-t))dt + \sqrt{2\beta_{T-t}}d\overline{w}_t, \quad 0 \leqslant t < T, \quad y_0 \sim p_T. \tag{8}$$

[2] E. Pardoux (1986). "Grossissement d'une filtration et retournement du temps d'une diffusion". In: *Séminaire de Probabilités XX 1984/85*. Ed. by Jacques Azéma and Marc Yor. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 48–55. ISBN: 978-3-540-39860-8

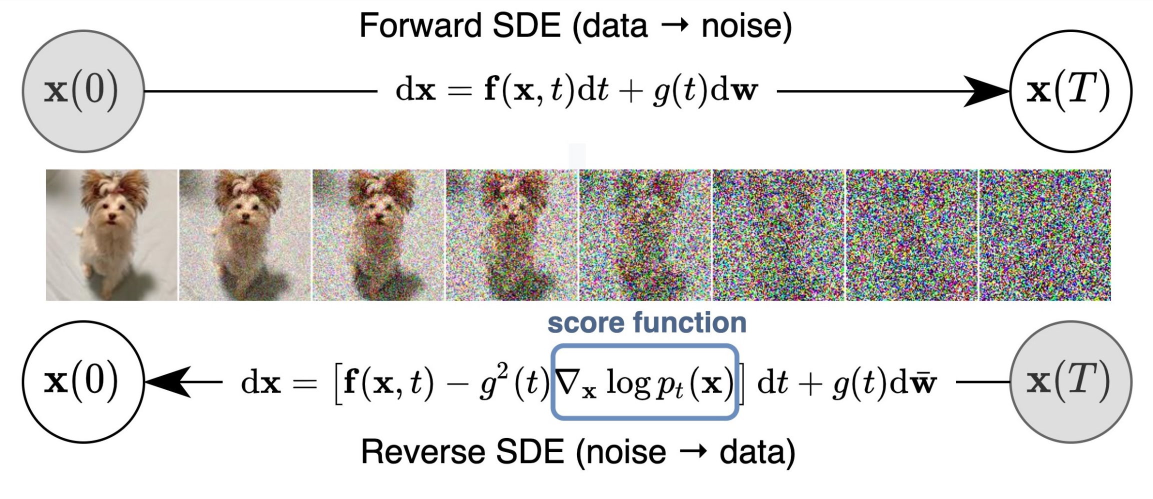
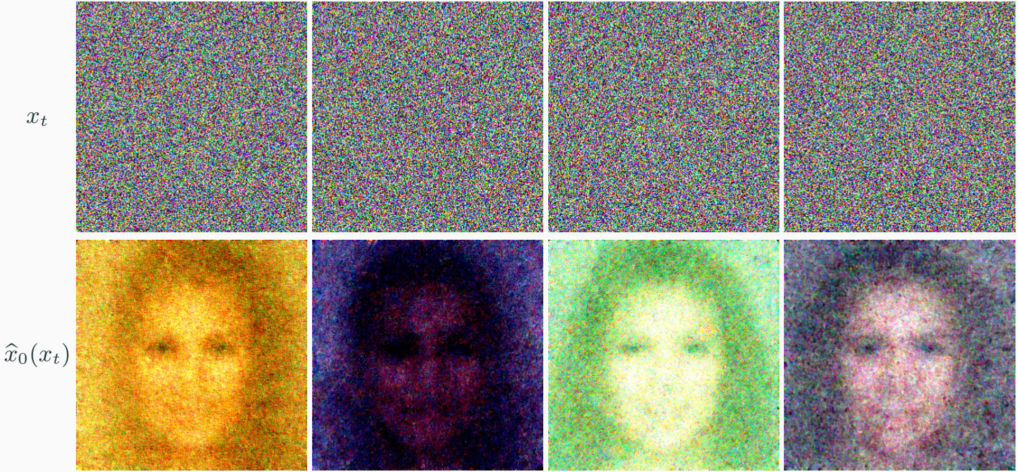


Image extracted from [Song et al. 2021]

Examples (generated with [Lugmayr et al. 2022][3])



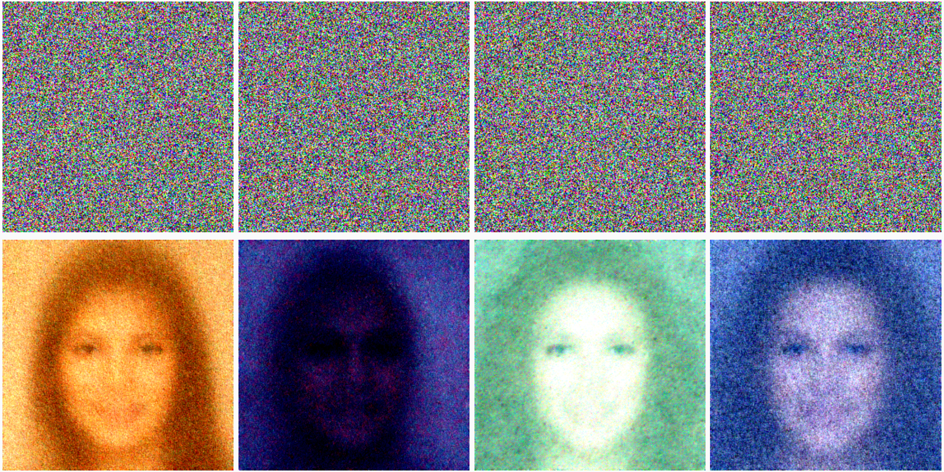
t = 249

[3] Andreas Lugmayr et al. (2022). "RePaint: Inpainting using Denoising Diffusion Probabilistic Models". In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11451–11461. URL: `https://api.semanticscholar.org/CorpusID:246240274`

Examples (generated with [Lugmayr et al. 2022][3])

x_t

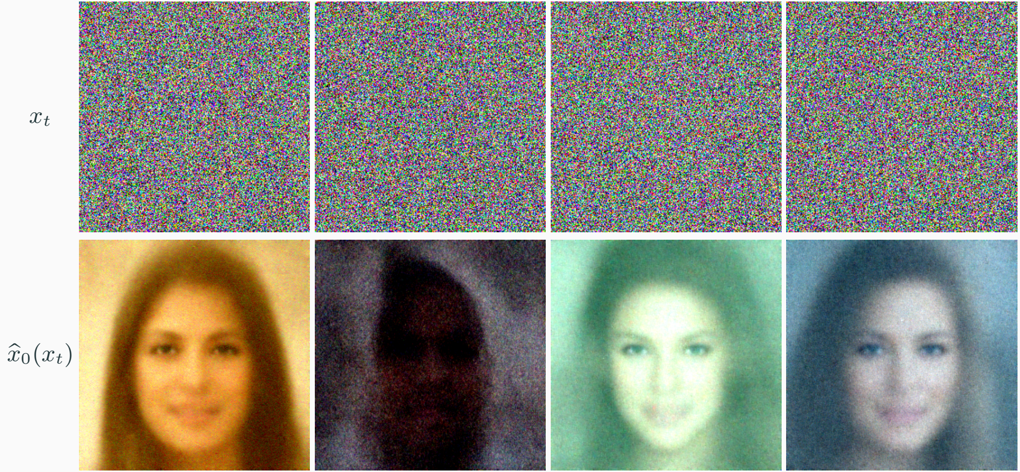
$\widehat{x}_0(x_t)$



t = 230

[3] Andreas Lugmayr et al. (2022). "RePaint: Inpainting using Denoising Diffusion Probabilistic Models". In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11451–11461. URL: `https://api.semanticscholar.org/CorpusID:246240274`

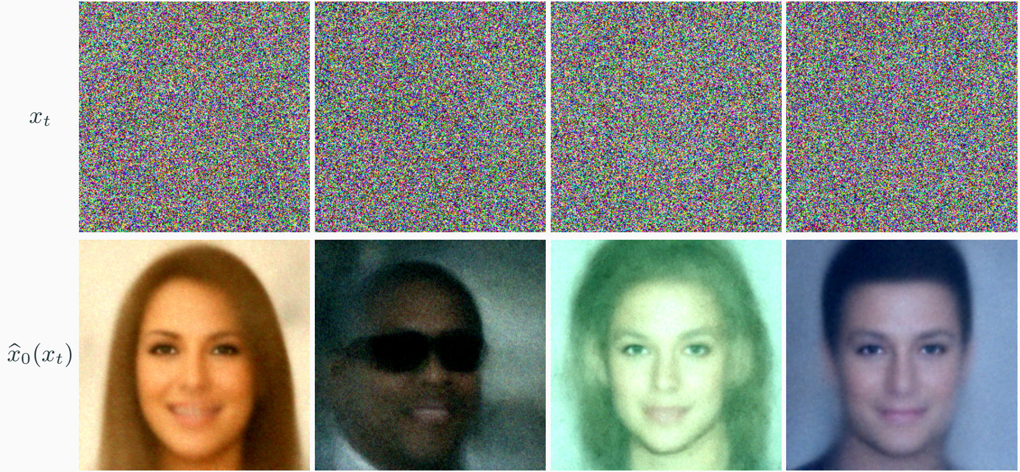
Examples (generated with [Lugmayr et al. 2022][3])



t = 210

[3] Andreas Lugmayr et al. (2022). "RePaint: Inpainting using Denoising Diffusion Probabilistic Models". In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11451–11461. URL: https://api.semanticscholar.org/CorpusID:246240274

Examples (generated with [Lugmayr et al. 2022][3])



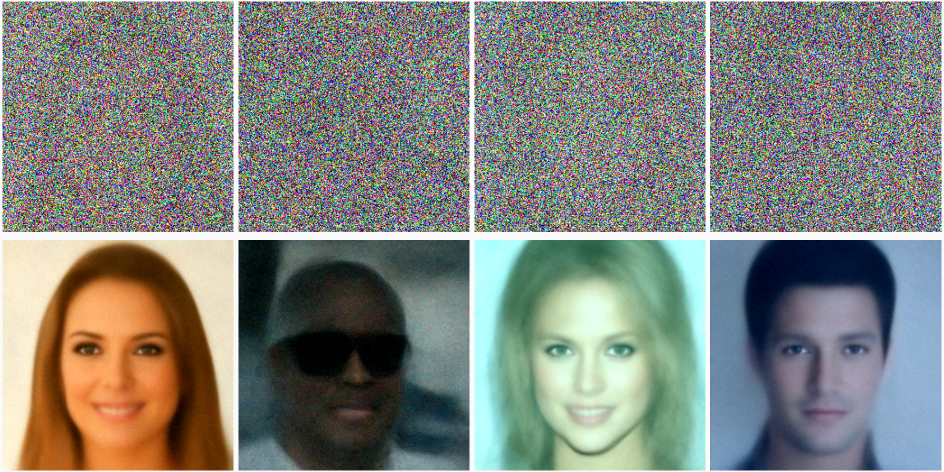
$t = 190$

[3] Andreas Lugmayr et al. (2022). "RePaint: Inpainting using Denoising Diffusion Probabilistic Models". In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11451–11461. URL: https://api.semanticscholar.org/CorpusID:246240274

Examples (generated with [Lugmayr et al. 2022][3])

x_t

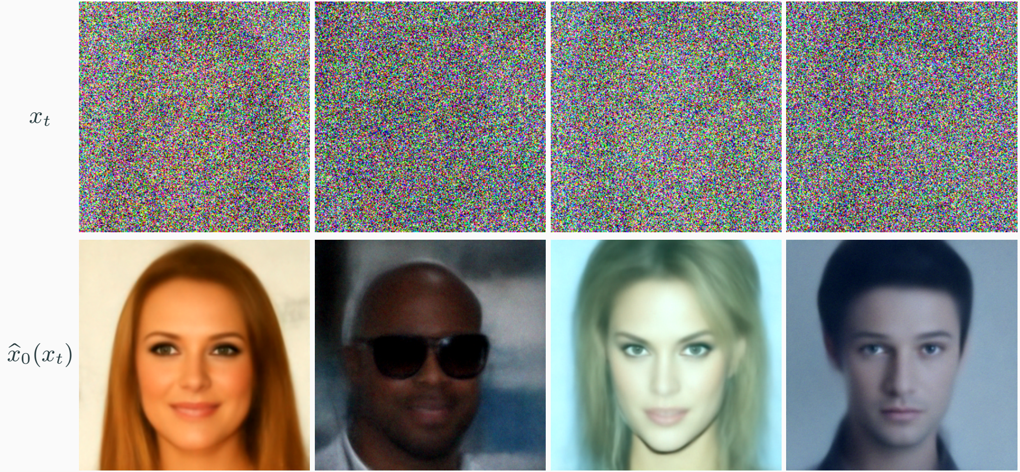
$\widehat{x}_0(x_t)$



$\mathrm{t} = 170$

[3] Andreas Lugmayr et al. (2022). “RePaint: Inpainting using Denoising Diffusion Probabilistic Models”. In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11451–11461. URL: `https://api.semanticscholar.org/CorpusID:246240274`

Examples (generated with [Lugmayr et al. 2022][3])

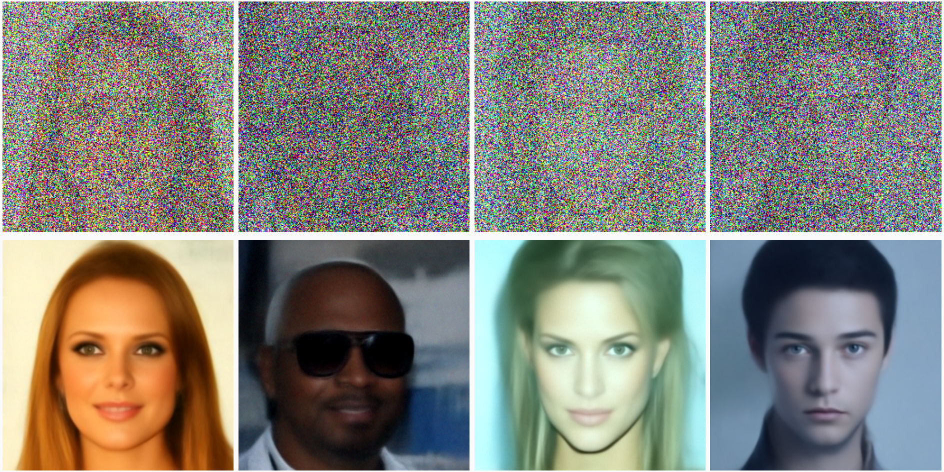


t = 150

[3]Andreas Lugmayr et al. (2022). "RePaint: Inpainting using Denoising Diffusion Probabilistic Models". In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11451–11461. URL: https://api.semanticscholar.org/CorpusID:246240274

x_t

$\widehat{x}_0(x_t)$

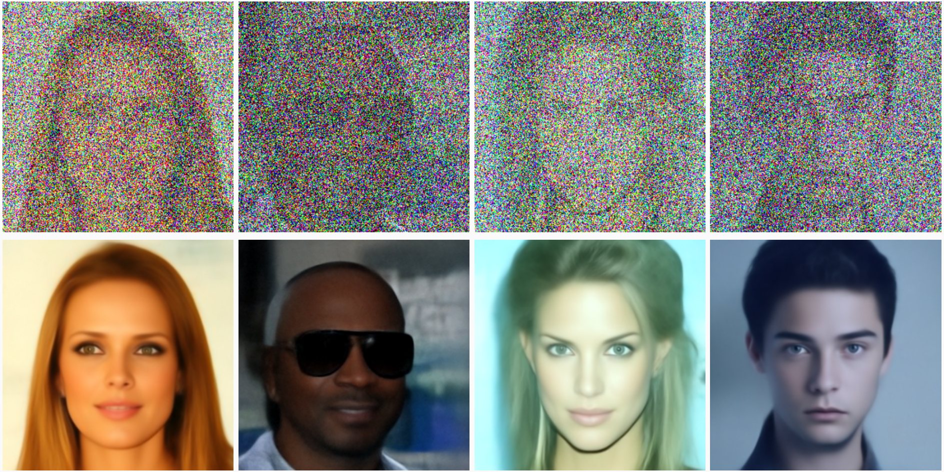


$t = 130$

[3] Andreas Lugmayr et al. (2022). "RePaint: Inpainting using Denoising Diffusion Probabilistic Models". In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11451–11461. URL: `https://api.semanticscholar.org/CorpusID:246240274`

x_t

$\widehat{x}_0(x_t)$



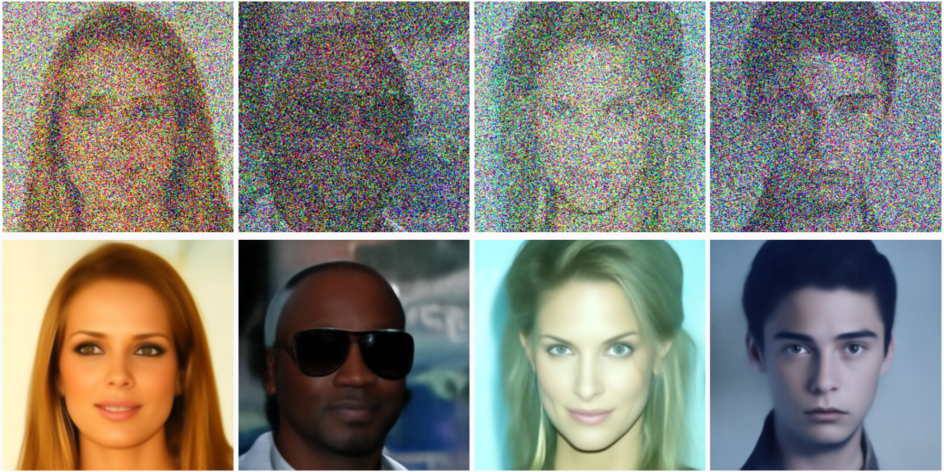
$t = 110$

[3] Andreas Lugmayr et al. (2022). "RePaint: Inpainting using Denoising Diffusion Probabilistic Models". In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11451–11461. URL: https://api.semanticscholar.org/CorpusID:246240274

Examples (generated with [Lugmayr et al. 2022][3])

x_t

$\widehat{x}_0(x_t)$



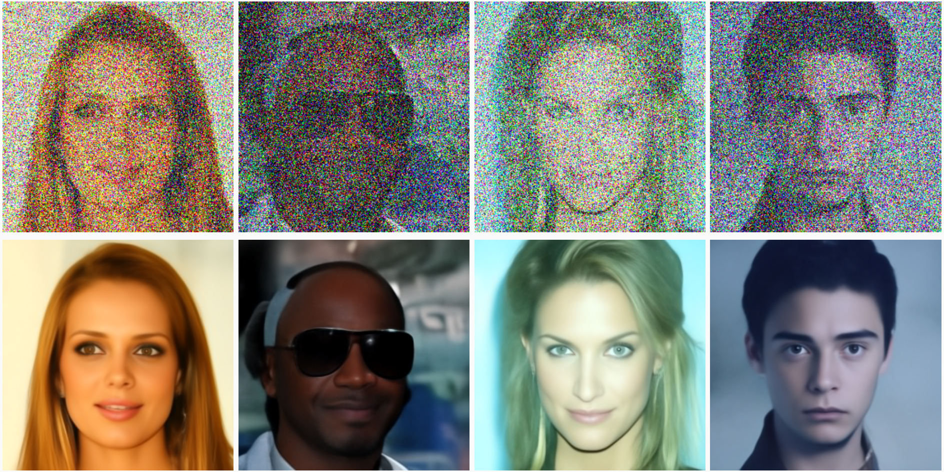
t = 90

[3] Andreas Lugmayr et al. (2022). "RePaint: Inpainting using Denoising Diffusion Probabilistic Models". In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11451–11461. URL: https://api.semanticscholar.org/CorpusID:246240274

Examples (generated with [Lugmayr et al. 2022][3])

x_t

$\widehat{x}_0(x_t)$



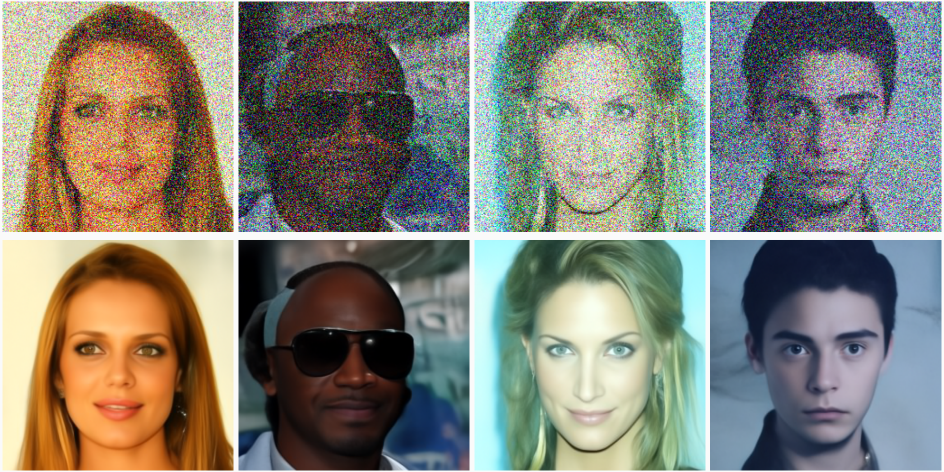
t = 70

[3] Andreas Lugmayr et al. (2022). "RePaint: Inpainting using Denoising Diffusion Probabilistic Models". In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11451–11461. URL: `https://api.semanticscholar.org/CorpusID:246240274`

Examples (generated with [Lugmayr et al. 2022][3])

x_t

$\widehat{x}_0(x_t)$



t = 50

[3] Andreas Lugmayr et al. (2022). "RePaint: Inpainting using Denoising Diffusion Probabilistic Models". In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11451–11461. URL: https://api.semanticscholar.org/CorpusID:246240274

Examples (generated with [Lugmayr et al. 2022][3])

x_t

$\widehat{x}_0(x_t)$

t = 30

[3] Andreas Lugmayr et al. (2022). "RePaint: Inpainting using Denoising Diffusion Probabilistic Models". In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11451–11461. URL: `https://api.semanticscholar.org/CorpusID:246240274`

Examples (generated with [Lugmayr et al. 2022][3])

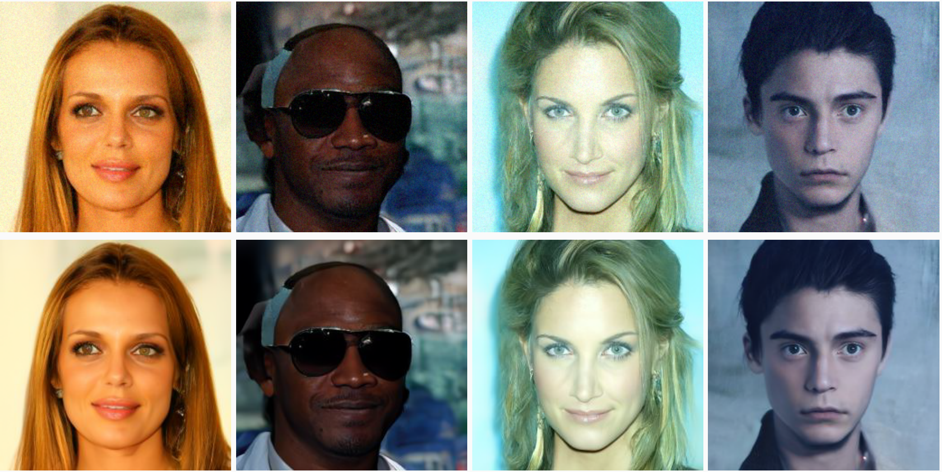
$$t = 10$$

[3] Andreas Lugmayr et al. (2022). "RePaint: Inpainting using Denoising Diffusion Probabilistic Models". In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11451–11461. URL: https://api.semanticscholar.org/CorpusID:246240274

Examples (generated with [Lugmayr et al. 2022][3])

x_t

$\widehat{x}_0(x_t)$



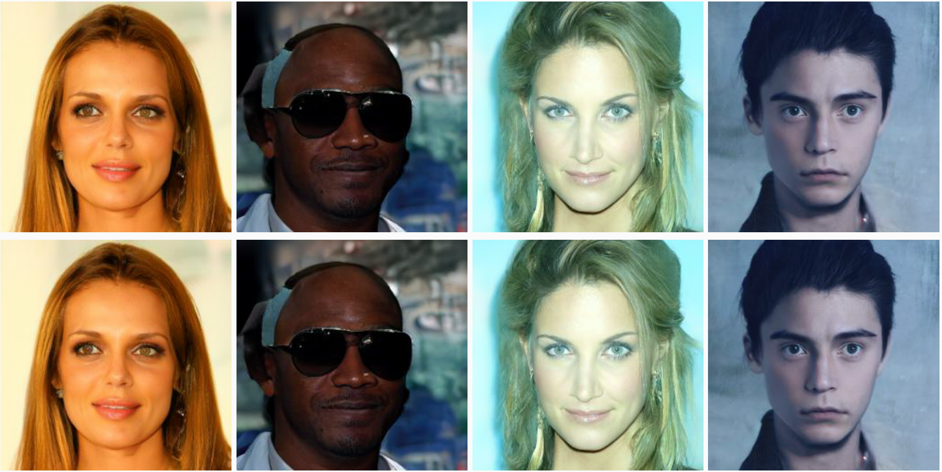
$\mathrm{t} = 5$

[3] Andreas Lugmayr et al. (2022). "RePaint: Inpainting using Denoising Diffusion Probabilistic Models". In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11451–11461. URL: `https://api.semanticscholar.org/CorpusID:246240274`

Examples (generated with [Lugmayr et al. 2022][3])

x_t

$\widehat{x}_0(x_t)$



$t = 0$

[3] Andreas Lugmayr et al. (2022). "RePaint: Inpainting using Denoising Diffusion Probabilistic Models". In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11451–11461. URL: `https://api.semanticscholar.org/CorpusID:246240274`

To the probability flow-ODE: the Fokker-Planck equation

$$dx_t = f(x_t, t)dt + g(t)dw_t \tag{9}$$

The marginals $(p_t)_{0 \leqslant t \leqslant T}$ follow the Fokker-Planck Equation:

$$\partial_t p_t(x) = -\operatorname{div}_x \left[f(x,t) p_t(x) \right] + \frac{1}{2} g(t)^2 \Delta_x p_t(x). \tag{10}$$

To the probability flow-ODE: the Fokker-Planck equation

$$dx_t = f(x_t, t)dt + g(t)dw_t \tag{9}$$

The marginals $(p_t)_{0 \leqslant t \leqslant T}$ follow the Fokker-Planck Equation:

$$\partial_t p_t(x) = -\operatorname{div}_x \left[f(x,t) p_t(x) \right] + \frac{1}{2} g(t)^2 \Delta_x p_t(x). \tag{10}$$

By denoting $q_t = p_{T-t}$, we search for a Fokker-Planck equation associated with q_t.

To the probability flow-ODE: the Fokker-Planck equation

$$dx_t = f(x_t, t)dt + g(t)dw_t \tag{9}$$

The marginals $(p_t)_{0 \leqslant t \leqslant T}$ follow the Fokker-Planck Equation:

$$\partial_t p_t(x) = -\operatorname{div}_x \left[f(x,t)p_t(x)\right] + \frac{1}{2}g(t)^2 \Delta_x p_t(x). \tag{10}$$

By denoting $q_t = p_{T-t}$, we search for a Fokker-Planck equation associated with q_t.

$$\begin{aligned}
\partial_t q_t(x) &= -\partial_t p_{T-t}(x) \\
&= \operatorname{div}_x \left[f(x, T-t)q_t(x)\right] + \left(-1 + \frac{1}{2}\right) g(T-t)^2 \Delta_x q_t(x) \\
&= -\operatorname{div}_x \left[\left(-f(x, T-t) + g(T-t)^2 \nabla_x \log q_t(x)\right) q_t(x)\right] + \frac{1}{2} g(T-t)^2 \Delta_x q_t(x).
\end{aligned}$$

To the probability flow-ODE: the Fokker-Planck equation

$$dx_t = f(x_t, t)dt + g(t)dw_t \tag{9}$$

The marginals $(p_t)_{0\leqslant t\leqslant T}$ follow the Fokker-Planck Equation:

$$\partial_t p_t(x) = -\operatorname{div}_x \left[f(x,t)p_t(x)\right] + \frac{1}{2}g(t)^2\Delta_x p_t(x). \tag{10}$$

By denoting $q_t = p_{T-t}$, we search for a Fokker-Planck equation associated with q_t.

$$\begin{aligned}
\partial_t q_t(x) &= -\partial_t p_{T-t}(x) \\
&= \operatorname{div}_x \left[f(x,T-t)q_t(x)\right] + \left(-1+\frac{1}{2}\right)g(T-t)^2\Delta_x q_t(x) \\
&= -\operatorname{div}_x \left[\left(-f(x,T-t) + g(T-t)^2\nabla_x \log q_t(x)\right) q_t(x)\right] + \frac{1}{2}g(T-t)^2\Delta_x q_t(x).
\end{aligned}$$

$$\begin{aligned}
\partial_t q_t(x) &= -\partial_t p_{T-t}(x) \\
&= \operatorname{div}_x \left[f(x,T-t)q_t(x)\right] + \left(0-\frac{1}{2}\right)g(T-t)^2\Delta_x q_t(x) \\
&= -\operatorname{div}_x \left[\left(-f(x,T-t) + \frac{1}{2}g(T-t)^2\nabla_x \log q_t(x)\right) q_t(x)\right].
\end{aligned}$$

Probability-flow ODE

The marginals $(p_t)_{0 \leqslant t \leqslant T}$ associated with the backward SDE

$$dy_t = -\beta_t \left[y_t + 2\nabla_y \log p_t(y_t) \right] dt + \sqrt{2\beta_t} d\overline{w}_t, \quad 0 \leqslant t \leqslant T, \quad y_T \sim p_T \tag{11}$$

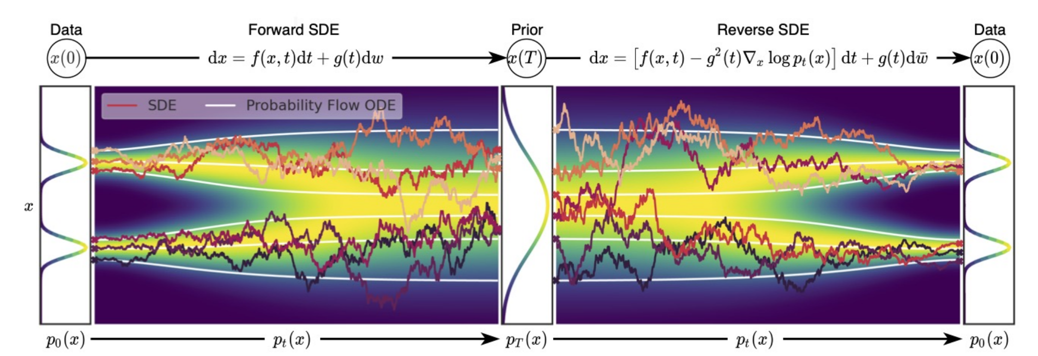
Probability-flow ODE

The marginals $(p_t)_{0\leqslant t\leqslant T}$ associated with the backward SDE

$$dy_t = -\beta_t \left[y_t + 2\nabla_y \log p_t(y_t)\right] dt + \sqrt{2\beta_t} d\overline{w}_t, \quad 0 \leqslant t \leqslant T, \quad y_T \sim p_T \tag{11}$$

are the same as those of this ODE

$$dy_t = -\beta_t \left[y_t + \nabla_y \log p_t(y_t)\right] dt, \qquad 0 \leqslant t \leqslant T, \quad y_T \sim p_T. \tag{12}$$



Study of the convergence

Sampling through diffusion models

$$\left.\begin{array}{l} dy_t = -\beta_t \left[y_t + 2\nabla_y \log p_t(y_t)\right] dt + \sqrt{2\beta_t} d\overline{w}_t, \\ \text{or} \\ dy_t = -\beta_t \left[y_t + \nabla_y \log p_t(y_t)\right] dt, \end{array}\right. \quad \text{where} \quad 0 \leqslant t \leqslant T, \quad y_T \sim p_T. \tag{13}$$

Sampling a distribution using diffusion models implies different choices and error types:

Sampling through diffusion models

$$\begin{array}{l} dy_t = -\beta_t \left[y_t + 2\nabla_y \log p_t(y_t) \right] dt + \sqrt{2\beta_t} d\overline{w}_t, \\ \text{or} \\ dy_t = -\beta_t \left[y_t + \nabla_y \log p_t(y_t) \right] dt, \end{array} \quad \text{where} \quad 0 \leqslant t \leqslant T, \quad \underset{y_T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})}{\cancel{y_T \sim p_T}} . \tag{13}$$

Sampling a distribution using diffusion models implies different choices and error types:

- p_T, which is unknown, is replaced by $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$ $\rightarrow$ **initialization error**

Sampling through diffusion models

$$
\begin{aligned}
&dy_t = -\beta_t \left[y_t + 2\nabla_y \log p_t(y_t)\right] dt + \sqrt{2\beta_t} d\overline{w}_t, \\
&\text{or} \qquad\qquad\qquad\qquad\qquad\qquad \text{where} \quad \cancel{0}_{\varepsilon} \leqslant t \leqslant T, \quad y_T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}). \\
&dy_t = -\beta_t \left[y_t + \nabla_y \log p_t(y_t)\right] dt,
\end{aligned} \tag{13}
$$

Sampling a distribution using diffusion models implies different choices and error types:

- p_T, which is unknown, is replaced by $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$ → **initialization error**
- In fact, another time ε to consider them on $[\varepsilon, T]$ → **truncation error**

Sampling through diffusion models

$$\begin{cases} dy_t = -\beta_t \left[y_t + 2\nabla_y \log p_t(y_t) \right] dt + \sqrt{2\beta_t} d\overline{w}_t, \\ \text{or} \\ dy_t = -\beta_t \left[y_t + \nabla_y \log p_t(y_t) \right] dt, \end{cases} \quad \text{where} \quad \varepsilon \leqslant t \leqslant T, \quad y_T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}). \tag{13}$$

Sampling a distribution using diffusion models implies different choices and error types:

- p_T, which is unknown, is replaced by $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$ → **initialization error**
- In fact, another time ε to consider them on $[\varepsilon, T]$ → **truncation error**
- A scheme to discretize the equations → **discretization error**

Sampling through diffusion models

$$dy_t = -\beta_t[y_t + 2\underbrace{\nabla_y \log p_t(y_t)}_{s_\theta(t,y_t)}]dt + \sqrt{2\beta_t}d\overline{w}_t,$$

or where $\varepsilon \leqslant t \leqslant T, \quad y_T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}).$ (13)

$$dy_t = -\beta_t[y_t + \underbrace{\nabla_y \log p_t(y_t)}_{s_\theta(t,y_t)}]dt,$$

Sampling a distribution using diffusion models implies different choices and error types:

- p_T, which is unknown, is replaced by $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$ → **initialization error**
- In fact, another time ε to consider them on $[\varepsilon, T]$ → **truncation error**
- A scheme to discretize the equations → **discretization error**
- A model/neural network s_θ to learn the score → **score approximation error**

Sampling through diffusion models

$$\begin{cases} dy_t = -\beta_t[y_t + 2s_\theta(t, y_t)]dt + \sqrt{2\beta_t}d\overline{w}_t, \\ \text{or} \\ dy_t = -\beta_t[y_t + s_\theta(t, y_t)]dt, \end{cases} \quad \text{where} \quad \varepsilon \leqslant t \leqslant T, \quad y_T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}). \tag{13}$$

Sampling a distribution using diffusion models implies different choices and error types:

- p_T, which is unknown, is replaced by $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$ $\longrightarrow$ **initialization error**
- In fact, another time ε to consider them on $[\varepsilon, T]$ $\longrightarrow$ **truncation error**
- A scheme to discretize the equations $\longrightarrow$ **discretization error**
- A model/neural network s_θ to learn the score $\longrightarrow$ **score approximation error**

Theorem 1. *Assume* **A**1, **A**2, **A**3, **A**4 *that* $T \geq 2\bar{\beta}(1+\log(1+\mathrm{diam}(\mathcal{M})))$, $\gamma_K = \varepsilon$ *and* $\varepsilon, \mathtt{M}, \delta \leq 1/32$. *Then, there exists* $\mathtt{D}_0 \geq 0$ *such that*

$$\mathbf{W}_1(\mathcal{L}(Y_K), \pi) \leq \mathtt{D}_0(\exp[\kappa/\varepsilon](\mathtt{M} + \delta^{1/2})/\varepsilon^2 + \exp[\kappa/\varepsilon]\exp[-T/\bar{\beta}] + \varepsilon^{1/2}),$$

with $\kappa = \mathrm{diam}(\mathcal{M})^2(1+\bar{\beta})/2$ *and*

$$\mathtt{D}_0 = D(1+\bar{\beta})^7(1+d+\mathrm{diam}(\mathcal{M})^4)(1+\log(1+\mathrm{diam}(\mathcal{M}))), \tag{7}$$

and D *is a numerical constant.*

Theorem 3. *Assume* **A**1, **A**2, **A**3, **A**4 *that* $T \geq 2\beta(1+\log(1+\mathrm{diam}(\mathcal{M})))$, $\gamma_K = \varepsilon$ *and* $\varepsilon, \mathtt{M}, \delta \leq 1/32$. *In addition, assume that there exists* $\Gamma \geq 0$ *such that for any* $t \in (0, T]$ *and* $x_t \in \mathbb{R}^d$

$$\|\nabla^2 \log p_t(x_t)\| \leq \Gamma/\sigma_t^2. \tag{9}$$

Then, there exists $\mathtt{D}_0 \geq 0$ *such that*

$$\mathbf{W}_1(\mathcal{L}(Y_K), \pi) \leq \mathtt{D}_0((\mathtt{M} + \delta^{1/2})/\varepsilon^{\Gamma+2} + \exp[-T/\bar{\beta}]/\varepsilon^{\Gamma} + \varepsilon^{1/2}),$$

From Valentin De Bortoli (2022). "Convergence of denoising diffusion models under the manifold hypothesis". In: *Transactions on Machine Learning Research*. ISSN: 2835-8856. URL: https://openreview.net/forum?id=MhK5aXo3gB

Restriction to the Gaussian case

Gaussian assumption

Gaussian assumption: p_{data} is a centered Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$. ($\mathbf{\Sigma}$ is not necessarily invertible)

Gaussian assumption

Gaussian assumption: p_{data} is a centered Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$. ($\mathbf{\Sigma}$ is not necessarily invertible) In this case,

$$\nabla \log p_t(x) = -\mathbf{\Sigma}_t^{-1} x, \quad 0 < t \leqslant T \tag{14}$$

with $\mathbf{\Sigma}_t = e^{-2B_t}\mathbf{\Sigma} + (1 - e^{-2B_t})\boldsymbol{I}$.

Gaussian assumption: p_{data} is a centered Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$. ($\mathbf{\Sigma}$ is not necessarily invertible) In this case,

$$\nabla \log p_t(x) = -\mathbf{\Sigma}_t^{-1} x, \quad 0 < t \leqslant T \tag{14}$$

with $\mathbf{\Sigma}_t = e^{-2B_t}\mathbf{\Sigma} + (1 - e^{-2B_t})\boldsymbol{I}$.

Proposition 3: Lineartiy of the score

he three following propositions are equivalent:

(i) $x_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$ for some covariance $\mathbf{\Sigma}$.

(ii) $\forall t > 0, \nabla_x \log p_t(x)$ is linear w.r.t x.

(iii) $\exists t > 0, \nabla_x \log p_t(x)$ is linear w.r.t x.

Initialization error

Explicit solution of the backward equations

Gaussian assumption: p_{data} is a centered Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$. ($\mathbf{\Sigma}$ is not necessarily invertible)

Gaussian assumption: p_{data} is a centered Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$. ($\mathbf{\Sigma}$ is not necessarily invertible)

Proposition 4: Solution to the equations under Gaussian assumption

Under Gaussian assumption, the strong solution to SDE (6) can be written as:

$$y_t^{\mathsf{SDE}} = e^{-(B_T - B_t)} \mathbf{\Sigma}_t \mathbf{\Sigma}_T^{-1} y_T + \boldsymbol{\xi}_t, \quad 0 \leqslant t \leqslant T \tag{15}$$

Under Gaussian assumption, the solution to ODE (12) can be written as:

$$y_t^{\mathsf{ODE}} = \mathbf{\Sigma}_T^{-1/2} \mathbf{\Sigma}_t^{1/2} y_T, \quad 0 \leqslant t \leqslant T, \tag{16}$$

with $\mathbf{\Sigma}_t = e^{-2B_t} \mathbf{\Sigma} + (1 - e^{-2B_t}) \boldsymbol{I}$.

Explicit solution of the backward equations

Gaussian assumption: p_{data} is a centered Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$. ($\mathbf{\Sigma}$ is not necessarily invertible)

Proposition 5: Solution to the equations under Gaussian assumption

If $\mathrm{Cov}(y_T)\mathbf{\Sigma} = \mathbf{\Sigma}\,\mathrm{Cov}(y_T)$, for $0 \leqslant t \leqslant T$,

$$\mathbf{\Sigma}_t^{\mathsf{SDE}} = \mathrm{Cov}(y_t^{\mathsf{SDE}}) = \mathbf{\Sigma}_t + e^{-2(B_T - B_t)}\mathbf{\Sigma}_t^2\mathbf{\Sigma}_T^{-2}\left[\mathrm{Cov}(y_T) - \mathbf{\Sigma}_T\right], \quad (15)$$

$$\mathbf{\Sigma}_t^{\mathsf{ODE}} = \mathrm{Cov}(y_t^{\mathsf{ODE}}) = \mathbf{\Sigma}_t\mathbf{\Sigma}_T^{-1}\,\mathrm{Cov}(y_T), \quad (16)$$

Explicit solution of the backward equations

Gaussian assumption: p_{data} is a centered Gaussian distribution $\mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$. ($\boldsymbol{\Sigma}$ is not necessarily invertible)

Proposition 6: Solution to the equations under Gaussian assumption

If $\operatorname{Cov}(y_T)\boldsymbol{\Sigma} = \boldsymbol{\Sigma}\operatorname{Cov}(y_T)$, for $0 \leqslant t \leqslant T$,

$$\boldsymbol{\Sigma}_t^{\mathsf{SDE}} = \operatorname{Cov}(y_t^{\mathsf{SDE}}) = \boldsymbol{\Sigma}_t + e^{-2(B_T - B_t)}\boldsymbol{\Sigma}_t^2\boldsymbol{\Sigma}_T^{-2}\left[\operatorname{Cov}(y_T) - \boldsymbol{\Sigma}_T\right], \tag{15}$$

$$\boldsymbol{\Sigma}_t^{\mathsf{ODE}} = \operatorname{Cov}(y_t^{\mathsf{ODE}}) = \boldsymbol{\Sigma}_t\boldsymbol{\Sigma}_T^{-1}\operatorname{Cov}(y_T), \tag{16}$$

If $y_T \sim p_T = \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_T)$,

$$\boldsymbol{\Sigma}_t^{\mathsf{SDE}} = \boldsymbol{\Sigma}_t^{\mathsf{ODE}} = \boldsymbol{\Sigma}_t, \quad 0 \leqslant t \leqslant T.$$

Explicit solution of the backward equations

Gaussian assumption: p_{data} is a centered Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$. ($\mathbf{\Sigma}$ is not necessarily invertible)

Proposition 7: Solution to the equations under Gaussian assumption

If $\text{Cov}(y_T)\mathbf{\Sigma} = \mathbf{\Sigma}\,\text{Cov}(y_T)$, for $0 \leqslant t \leqslant T$,

$$\mathbf{\Sigma}_t^{\text{SDE}} = \text{Cov}(y_t^{\text{SDE}}) = \mathbf{\Sigma}_t + e^{-2(B_T - B_t)}\mathbf{\Sigma}_t^2\mathbf{\Sigma}_T^{-2}\left[\text{Cov}(y_T) - \mathbf{\Sigma}_T\right], \quad (15)$$

$$\mathbf{\Sigma}_t^{\text{ODE}} = \text{Cov}(y_t^{\text{ODE}}) = \mathbf{\Sigma}_t\mathbf{\Sigma}_T^{-1}\,\text{Cov}(y_T), \quad (16)$$

If $y_T \sim p_T = \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_T)$,

$$\mathbf{\Sigma}_t^{\text{SDE}} = \mathbf{\Sigma}_t^{\text{ODE}} = \mathbf{\Sigma}_t, \quad 0 \leqslant t \leqslant T.$$

If $y_T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$,

$$\mathbf{\Sigma}_t^{\text{SDE}} = \mathbf{\Sigma}_t + e^{-2(B_T - B_t)}\mathbf{\Sigma}_t^2\mathbf{\Sigma}_T^{-2}(\boldsymbol{I} - \mathbf{\Sigma}_T), \quad 0 \leqslant t \leqslant T.$$

$$\mathbf{\Sigma}_t^{\text{ODE}} = \mathbf{\Sigma}_t\mathbf{\Sigma}_T^{-1}$$

Link between ODE and OT

Under Gaussian assumption, the solution to ODE (12) can be written as:

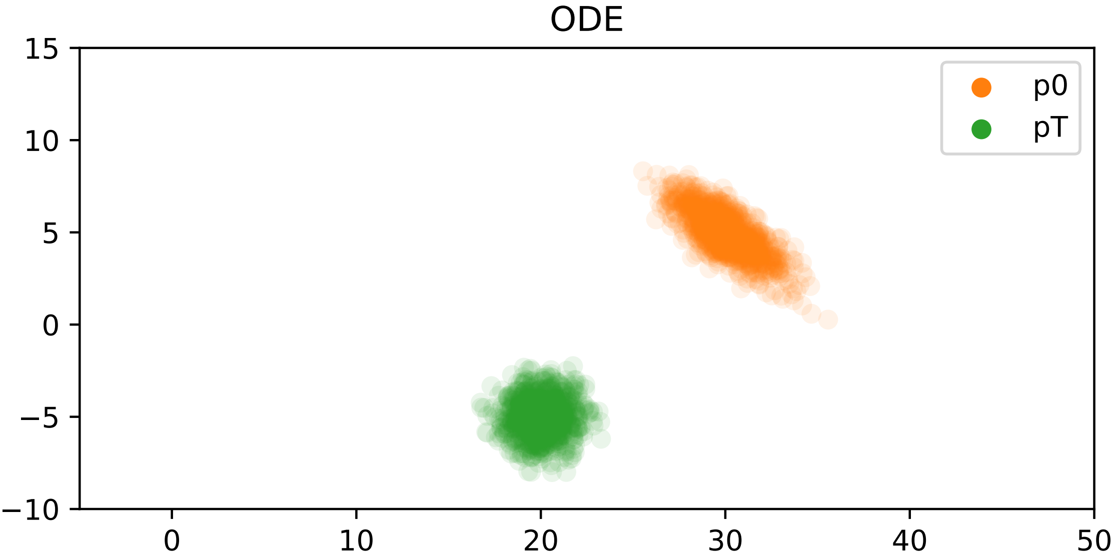
$$y_t^{\text{ODE}} = \mathbf{\Sigma}_T^{-1/2}\mathbf{\Sigma}_t^{1/2} y_T, \quad 0 \leqslant t \leqslant T, \tag{17}$$

- $y \mapsto \mathbf{\Sigma}_T^{-1/2}\mathbf{\Sigma}_t^{1/2} y$ is the transport map between p_T and p_t.
- False in general:, see [Lavenant and Santambrogio 2022][4]
- However, used in [Khrulkov et al. 2023][5]

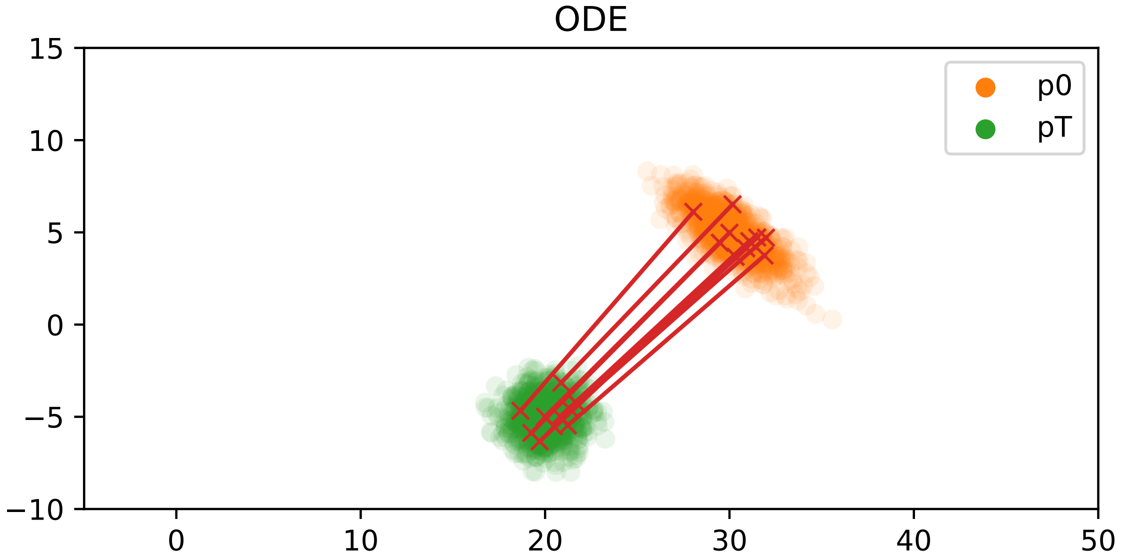
[4] Hugo Lavenant and Filippo Santambrogio (2022). "The flow map of the Fokker–Planck equation does not provide optimal transport". In: *Applied Mathematics Letters* 133, p. 108225. ISSN: 0893-9659. DOI: https://doi.org/10.1016/j.aml.2022.108225. URL: https://www.sciencedirect.com/science/article/pii/S089396592200180X

[5] Valentin Khrulkov et al. (2023). "Understanding DDPM Latent Codes Through Optimal Transport". In: *The Eleventh International Conference on Learning Representations*. URL: https://openreview.net/forum?id=6PIrhAx1j4i

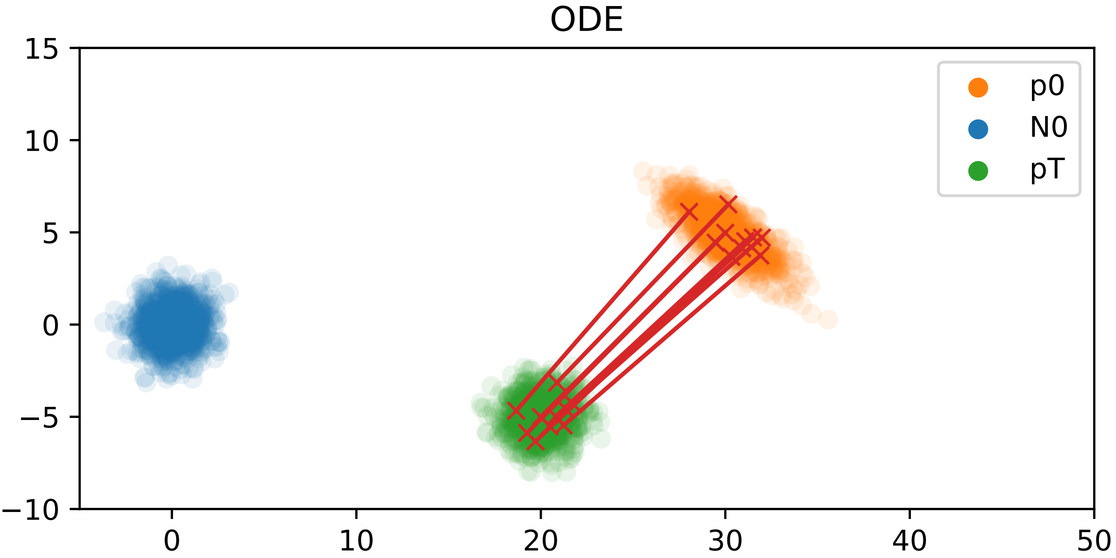
Initialization error: Focus on the ODE

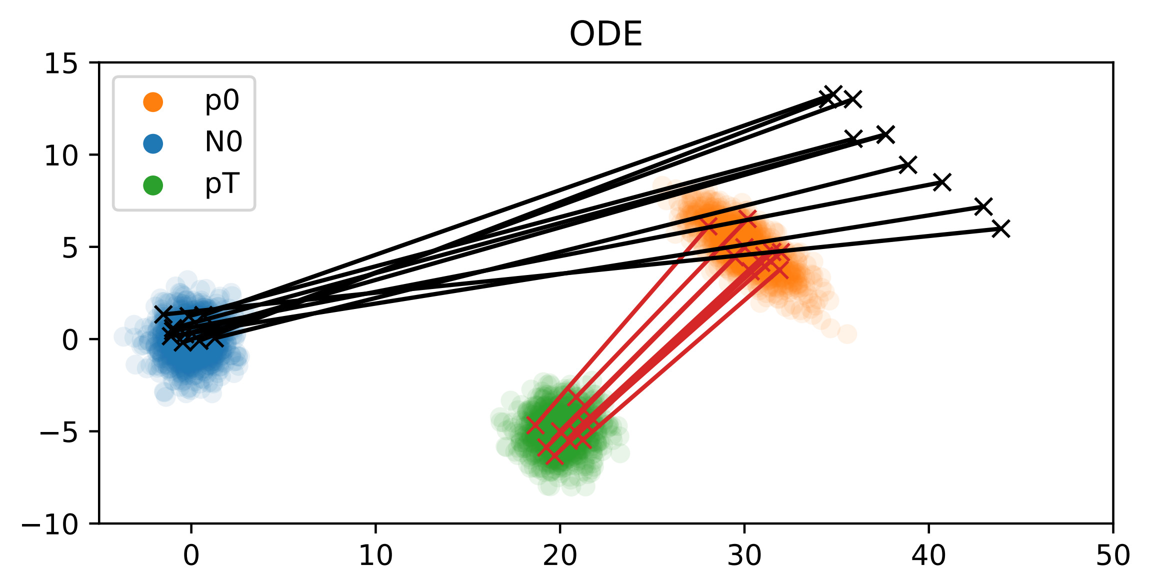


Initialization error: Focus on the ODE

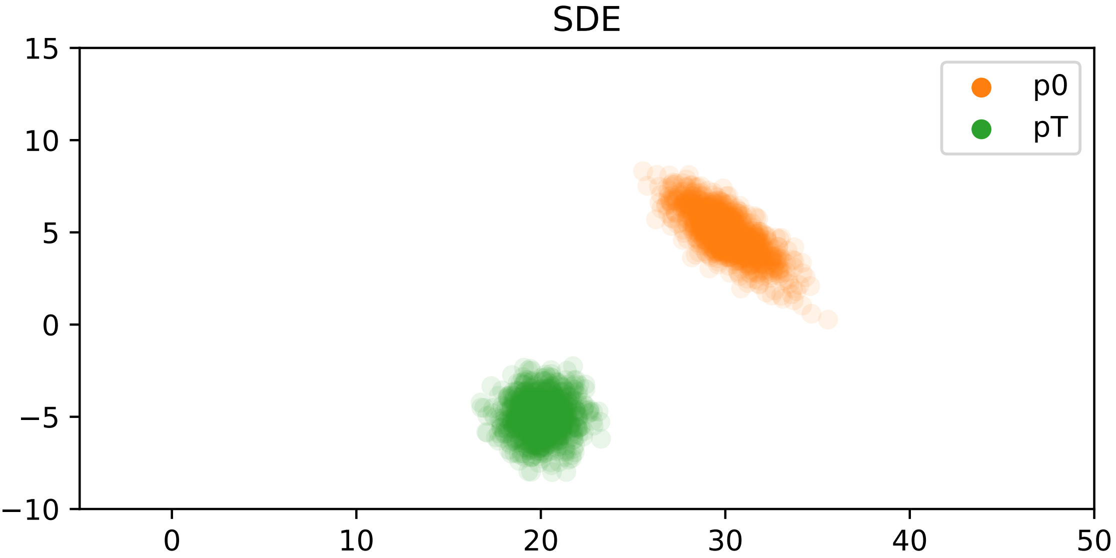


Initialization error: Focus on the ODE

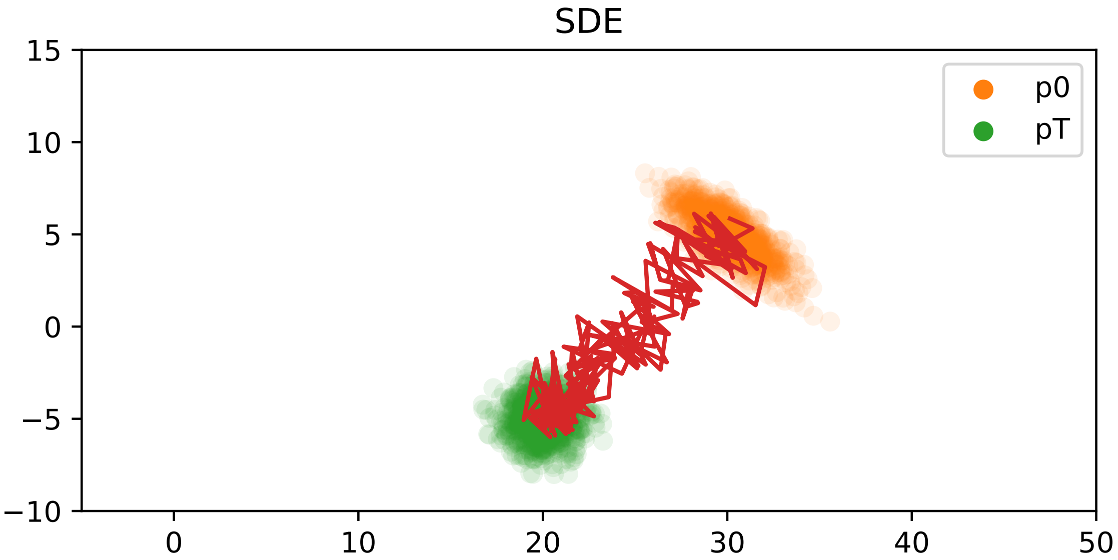



ODE
p0
N0
pT
15
10
5
0
−5
−10
0
10
20
30
40
50

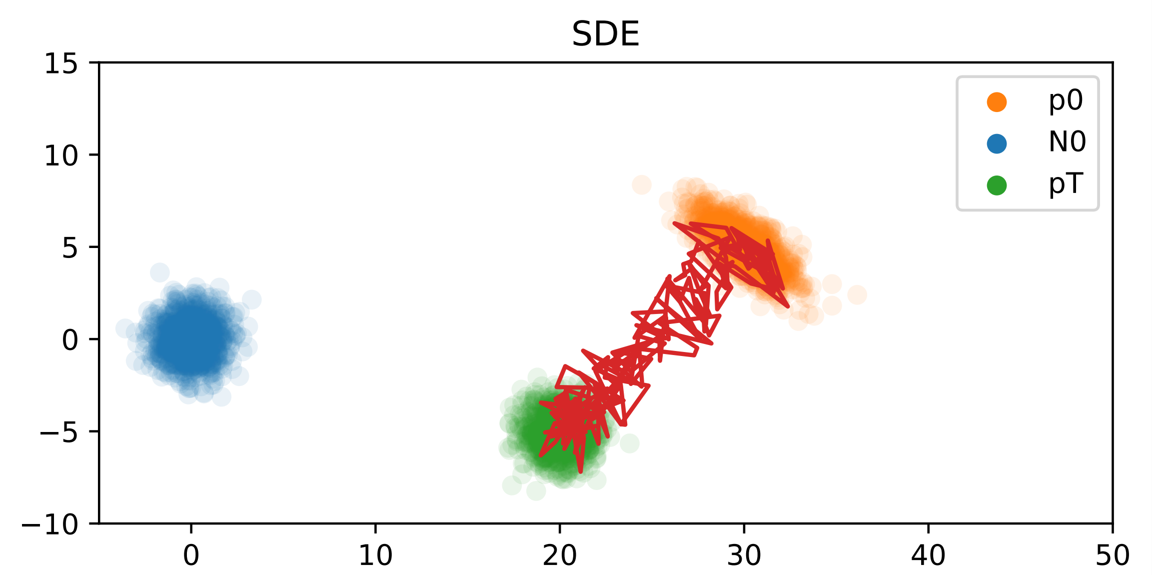
Initialization error: Focus on the SDE



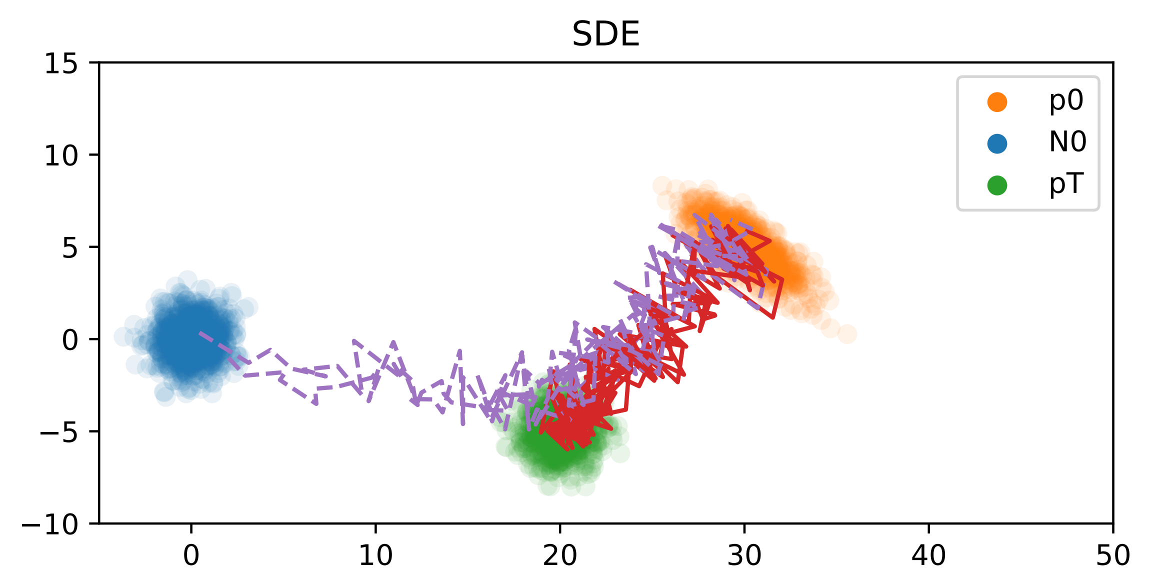
Initialization error: Focus on the SDE



Initialization error: Focus on the SDE



Initialization error: Focus on the SDE



Initialization error

Under Gaussian assumption, the strong solution to SDE (6) can be written as:

$$y_t^{\mathsf{SDE}} = e^{-(B_T - B_t)} \boldsymbol{\Sigma}_t \boldsymbol{\Sigma}_T^{-1} y_T + \boldsymbol{\xi}_t, \quad 0 \leqslant t \leqslant T \tag{18}$$

with $\mathrm{Cov}(\boldsymbol{\xi}_t) = \boldsymbol{\Sigma}_t - e^{-2(B_T - B_t)} \boldsymbol{\Sigma}_t^2 \boldsymbol{\Sigma}_T^{-1}$.

Finally,

$$\mathrm{Cov}(y_t) = \boldsymbol{\Sigma}_t + e^{-2(B_T - B_T)} \boldsymbol{\Sigma}_{T-t}^2 \boldsymbol{\Sigma}_T^{-2} \left[\mathrm{Cov}(y_0) - \boldsymbol{\Sigma}_T \right]. \tag{19}$$

Initialization error

Under Gaussian assumption, the strong solution to SDE (6) can be written as:

$$y_t^{\mathsf{SDE}} = e^{-(B_T - B_t)} \mathbf{\Sigma}_t \mathbf{\Sigma}_T^{-1} y_T + \boldsymbol{\xi}_t, \quad 0 \leqslant t \leqslant T \tag{18}$$

with $\mathrm{Cov}(\boldsymbol{\xi}_t) = \mathbf{\Sigma}_t - e^{-2(B_T - B_t)} \mathbf{\Sigma}_t^2 \mathbf{\Sigma}_T^{-1}$.

Finally,

$$\mathrm{Cov}(y_t) = \mathbf{\Sigma}_t + e^{-2(B_T - B_t)} \mathbf{\Sigma}_t^2 \mathbf{\Sigma}_T^{-2} \left[\mathrm{Cov}(y_T) - \mathbf{\Sigma}_T \right]. \tag{19}$$

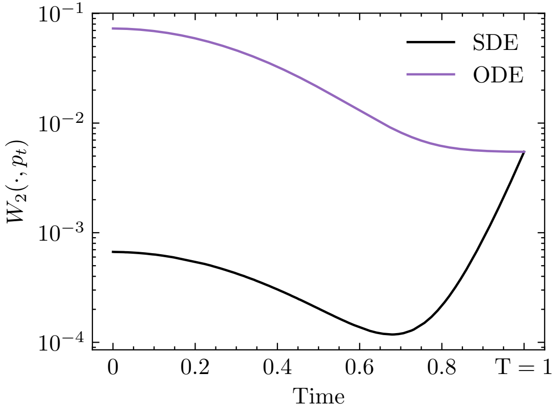
SDE vs ODE

Proposition 8: Marginals of the generative processes under Gaussian assumption

Under Gaussian assumption,

$$\mathbf{W}_2(p_t^{\mathsf{SDE}}, p_t) \leqslant \mathbf{W}_2(p_t^{\mathsf{ODE}}, p_t) \qquad (20)$$

which shows that at each time $0 \leqslant t \leqslant T$ and in particular for $t = 0$ which corresponds to the desired outputs of the sampler, the SDE sampler is a better sampler than the ODE sampler when the exact score is known.



Truncation error

Truncation error

$$dy_t = -\beta_t[y_t + 2s_\theta(t, y_t)]dt + \sqrt{2\beta_t}d\overline{w}_t, \quad \text{or} \quad dy_t = -\beta_t[y_t + s_\theta(t, y_t)]dt, \qquad \text{where} \quad \varepsilon \leqslant t \leqslant T, \quad y_T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}). \tag{21}$$

Sampling a distribution using diffusion models implies different choices and error types:

- p_T, which is unknown, is replaced by $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$ → **initialization error**
- In fact, another time ε to consider them on $[\varepsilon, T]$ → **truncation error**
- A scheme to discretize the equations → **discretization error**
- A model/neural network s_θ to learn the score → **score approximation error**

Truncation error under Gaussian assumption

Gaussian assumption: p_{data} is a centered Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$. ($\mathbf{\Sigma}$ is not necessarily invertible) In this case,

$$\nabla \log p_t(x) = -\mathbf{\Sigma}_t^{-1} x, \quad 0 < t \leqslant T \tag{22}$$

with $\mathbf{\Sigma}_t = e^{-2B_t}\mathbf{\Sigma} + (1 - e^{-2B_t})\boldsymbol{I}$.

Truncation error under Gaussian assumption

Gaussian assumption: p_{data} is a centered Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$. ($\mathbf{\Sigma}$ is not necessarily invertible) In this case,

$$\nabla \log p_t(x) = -\mathbf{\Sigma}_t^{-1} x, \quad 0 < t \leqslant T \tag{22}$$

with $\mathbf{\Sigma}_t = e^{-2B_t}\mathbf{\Sigma} + (1 - e^{-2B_t})\boldsymbol{I}$. At time $t = 0$,

$$\mathbf{\Sigma}_0 = \mathbf{\Sigma}$$

Truncation error under Gaussian assumption

Gaussian assumption: p_{data} is a centered Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$. ($\mathbf{\Sigma}$ is not necessarily invertible) In this case,

$$\nabla \log p_t(x) = -\mathbf{\Sigma}_t^{-1} x, \quad 0 < t \leqslant T \tag{22}$$

with $\mathbf{\Sigma}_t = e^{-2B_t}\mathbf{\Sigma} + (1 - e^{-2B_t})\boldsymbol{I}$. At time $t = 0$,

$$\mathbf{\Sigma}_0 = \mathbf{\Sigma}$$

Consequently, $\nabla \log p_0(x)$ is not defined in general.

Truncation error under Gaussian assumption

Gaussian assumption: p_{data} is a centered Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$. ($\mathbf{\Sigma}$ is not necessarily invertible) In this case,

$$\nabla \log p_t(x) = -\mathbf{\Sigma}_t^{-1} x, \quad 0 < t \leqslant T \tag{22}$$

with $\mathbf{\Sigma}_t = e^{-2B_t}\mathbf{\Sigma} + (1 - e^{-2B_t})\boldsymbol{I}$. At time $t = 0$,

$$\mathbf{\Sigma}_0 = \mathbf{\Sigma}$$

Consequently, $\nabla \log p_0(x)$ is not defined in general.

$\Longrightarrow$ The truncation error is vital.

Discretization error

Discretization error

$$\begin{aligned} dy_t &= -\beta_t[y_t + 2s_\theta(t, y_t)]dt + \sqrt{2\beta_t}d\overline{w}_t, \\ &\text{or} \\ dy_t &= -\beta_t[y_t + s_\theta(t, y_t)]dt, \end{aligned} \quad \text{where} \quad \varepsilon \leqslant t \leqslant T, \quad y_T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}). \tag{23}$$

Sampling a distribution using diffusion models implies different choices and error types:

- p_T, which is unknown, is replaced by $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$ → **initialization error**
- In fact, another time ε to consider them on $[\varepsilon, T]$ → **truncation error**
- A scheme to discretize the equations → **discretization error**
- A model/neural network s_θ to learn the score → **score approximation error**

Discretization schemes

SDE schemes	Euler-Maruyama (EM)	$\begin{cases} \tilde{\boldsymbol{y}}_0^{\Delta,\text{EM}} & \sim \mathcal{N}_0 \\ \tilde{\boldsymbol{y}}_{k+1}^{\Delta,\text{EM}} & = \tilde{\boldsymbol{y}}_k^{\Delta,\text{EM}} + \Delta_t \beta_{T-t_k} \left(\tilde{\boldsymbol{y}}_k^{\Delta,\text{EM}} - 2\boldsymbol{\Sigma}_{T-t_k}^{-1} \tilde{\boldsymbol{y}}_k^{\Delta,\text{EM}} \right) + \sqrt{2\Delta_t \beta_{T-t_k}} z_k, \; z_k \sim \mathcal{N}_0 \end{cases}$	(24)
	Exponential integrator (EI)	$\begin{cases} \tilde{\boldsymbol{y}}_0^{\Delta,\text{EI}} & \sim \mathcal{N}_0 \\ \tilde{\boldsymbol{y}}_{k+1}^{\Delta,\text{EI}} & = \tilde{\boldsymbol{y}}_k^{\Delta,\text{EI}} + \gamma_{1,k} \left(\tilde{\boldsymbol{y}}_k^{\Delta,\text{EI}} - 2\boldsymbol{\Sigma}_{T-t_k}^{-1} \tilde{\boldsymbol{y}}_k^{\Delta,\text{EI}} \right) + \sqrt{2\gamma_{2,k}} z_k, \; z_k \sim \mathcal{N}_0 \end{cases}$ where $\gamma_{1,k} = \exp(B_{T-t_k} - B_{T-t_{k+1}}) - 1$ and $\gamma_{2,k} = \frac{1}{2}(\exp(2B_{T-t_k} - 2B_{T-t_{k+1}}) - 1)$	(25)
ODE schemes	Explicit Euler	$\begin{cases} \hat{\boldsymbol{y}}_0^{\Delta,\text{Euler}} & \sim \mathcal{N}_0 \\ \hat{\boldsymbol{y}}_{k+1}^{\Delta,\text{Euler}} & = \hat{\boldsymbol{y}}_k^{\Delta,\text{Euler}} + \Delta_t f(t_k, \hat{\boldsymbol{y}}_k^{\Delta,\text{Euler}}) \quad \text{with } f(t,y) = \beta_{T-t} y - \beta_{T-t} \boldsymbol{\Sigma}_{T-t}^{-1} y \end{cases}$	(26)
	Heun's method	$\begin{cases} \hat{\boldsymbol{y}}_0^{\Delta,\text{Heun}} & \sim \mathcal{N}_0 \\ \hat{\boldsymbol{y}}_{k+1/2}^{\Delta,\text{Heun}} & = \hat{\boldsymbol{y}}_k^{\Delta,\text{Heun}} + \Delta_t f(t_k, \hat{\boldsymbol{y}}_k^{\Delta,\text{Heun}}) \quad \text{with } f(t,y) = \beta_{T-t} y - \beta_{T-t} \boldsymbol{\Sigma}_{T-t}^{-1} y \\ \hat{\boldsymbol{y}}_{k+1}^{\Delta,\text{Heun}} & = \hat{\boldsymbol{y}}_k^{\Delta,\text{Heun}} + \frac{\Delta_t}{2} \left(f(t_k, \hat{\boldsymbol{y}}_k^{\Delta,\text{Heun}}) + f(t_{k+1}, \hat{\boldsymbol{y}}_{k+1/2}^{\Delta,\text{Heun}}) \right) \end{cases}$	(27)

Discretization processes

The EM discretized process is a Gaussian processe:

$$\begin{cases} \tilde{\boldsymbol{y}}_0^{\Delta,\mathsf{EM}} & \sim \mathcal{N}_0 \\ \tilde{\boldsymbol{y}}_1^{\Delta,\mathsf{EM}} & = \tilde{\boldsymbol{y}}_0^{\Delta,\mathsf{EM}} + \Delta_t \beta_{T-t_0} \left(\tilde{\boldsymbol{y}}_0^{\Delta,\mathsf{EM}} - 2\boldsymbol{\Sigma}_{T-t_0}^{-1} \tilde{\boldsymbol{y}}_0^{\Delta,\mathsf{EM}} \right) + \sqrt{2\Delta_t \beta_{T-t_0}} z_0, \;\; z_0 \sim \mathcal{N}_0 \end{cases} \tag{14}$$

Discretization processes

The EM discretized process is a Gaussian processe:

$$\begin{cases} \tilde{\boldsymbol{y}}_0^{\Delta,\text{EM}} & \sim \mathcal{N}_0 \\ \tilde{\boldsymbol{y}}_1^{\Delta,\text{EM}} & = \tilde{\boldsymbol{y}}_0^{\Delta,\text{EM}} + \Delta_t \beta_{T-t_0} \left(\tilde{\boldsymbol{y}}_0^{\Delta,\text{EM}} - 2\boldsymbol{\Sigma}_{T-t_0}^{-1} \tilde{\boldsymbol{y}}_0^{\Delta,\text{EM}} \right) + \sqrt{2\Delta_t \beta_{T-t_0}} z_0, \ z_0 \sim \mathcal{N}_0 \end{cases} \tag{14}$$

$$\boldsymbol{\Sigma}_0^{\Delta,\text{EM}} = \boldsymbol{I} \tag{28}$$

$$\boldsymbol{\Sigma}_1^{\Delta,\text{EM}} = \left(\boldsymbol{I} + \Delta_t \beta_{T-t_0} \left(\boldsymbol{I} - 2\boldsymbol{\Sigma}_{T-t_0}^{-1} \right) \right)^2 \boldsymbol{\Sigma}_0^{\Delta,\text{EM}} + 2\Delta_t \beta_{T-t_0} \boldsymbol{I} \tag{29}$$

Discretization processes

The EM discretized process is a Gaussian processe:

$$\begin{cases} \tilde{\boldsymbol{y}}_0^{\Delta,\text{EM}} & \sim \mathcal{N}_0 \\ \tilde{\boldsymbol{y}}_1^{\Delta,\text{EM}} & = \tilde{\boldsymbol{y}}_0^{\Delta,\text{EM}} + \Delta_t \beta_{T-t_0} \left(\tilde{\boldsymbol{y}}_0^{\Delta,\text{EM}} - 2\boldsymbol{\Sigma}_{T-t_0}^{-1} \tilde{\boldsymbol{y}}_0^{\Delta,\text{EM}} \right) + \sqrt{2\Delta_t \beta_{T-t_0}} z_0, \; z_0 \sim \mathcal{N}_0 \end{cases} \tag{14}$$

$$\boldsymbol{\Sigma}_0^{\Delta,\text{EM}} = \boldsymbol{I} \tag{28}$$

$$\boldsymbol{\Sigma}_1^{\Delta,\text{EM}} = \left(\boldsymbol{I} + \Delta_t \beta_{T-t_0} \left(\boldsymbol{I} - 2\boldsymbol{\Sigma}_{T-t_0}^{-1} \right) \right)^2 \boldsymbol{\Sigma}_0^{\Delta,\text{EM}} + 2\Delta_t \beta_{T-t_0} \boldsymbol{I} \tag{29}$$

Denoting $(\boldsymbol{v}_i)_i$ the eigenvectors of $\boldsymbol{\Sigma}$, let $1 \leqslant i \leqslant d$,

$$\boldsymbol{\Sigma}_1^{\Delta,\text{EM}} \boldsymbol{v}_i = \left[\left(1 + \Delta_t \beta_{T-t_0} \left(\boldsymbol{I} - \frac{2}{\lambda_{i,T-t_0}} \right) \right)^2 \lambda_{i,0}^{\Delta,\text{EM}} + 2\Delta_t \beta_{T-t_0} \right] \boldsymbol{v}_i \tag{30}$$

Discretization processes

The EM discretized process is a Gaussian processe:

$$\begin{cases} \tilde{\boldsymbol{y}}_0^{\Delta,\text{EM}} & \sim \mathcal{N}_0 \\ \tilde{\boldsymbol{y}}_1^{\Delta,\text{EM}} & = \tilde{\boldsymbol{y}}_0^{\Delta,\text{EM}} + \Delta_t \beta_{T-t_0} \left(\tilde{\boldsymbol{y}}_0^{\Delta,\text{EM}} - 2\boldsymbol{\Sigma}_{T-t_0}^{-1} \tilde{\boldsymbol{y}}_0^{\Delta,\text{EM}} \right) + \sqrt{2\Delta_t \beta_{T-t_0}} z_0, \ z_0 \sim \mathcal{N}_0 \end{cases} \tag{14}$$

$$\boldsymbol{\Sigma}_0^{\Delta,\text{EM}} = \boldsymbol{I} \tag{28}$$

$$\boldsymbol{\Sigma}_1^{\Delta,\text{EM}} = \left(\boldsymbol{I} + \Delta_t \beta_{T-t_0} \left(\boldsymbol{I} - 2\boldsymbol{\Sigma}_{T-t_0}^{-1} \right) \right)^2 \boldsymbol{\Sigma}_0^{\Delta,\text{EM}} + 2\Delta_t \beta_{T-t_0} \boldsymbol{I} \tag{29}$$

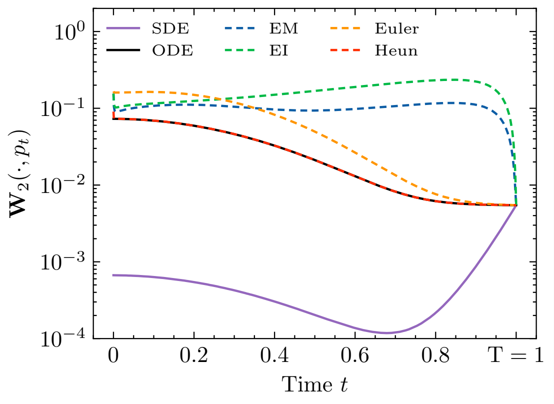
Denoting $(\boldsymbol{v}_i)_i$ the eigenvectors of $\boldsymbol{\Sigma}$, let $1 \leqslant i \leqslant d$,

$$\boldsymbol{\Sigma}_1^{\Delta,\text{EM}} \boldsymbol{v}_i = \left[\left(1 + \Delta_t \beta_{T-t_0} \left(\boldsymbol{I} - \frac{2}{\lambda_{i,T-t_0}} \right) \right)^2 \lambda_{i,0}^{\Delta,\text{EM}} + 2\Delta_t \beta_{T-t_0} \right] \boldsymbol{v}_i \tag{30}$$

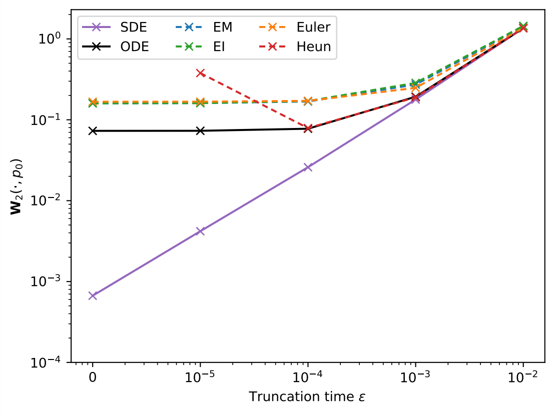
$$\lambda_{i,1}^{\Delta,\text{EM}} = \left(1 + \Delta_t \beta_{T-t_0} \left(\boldsymbol{I} - \frac{2}{\lambda_{i,T-t_0}} \right) \right)^2 \lambda_{i,0}^{\Delta,\text{EM}} + 2\Delta_t \beta_{T-t_0}, \quad 1 \leqslant i \leqslant d. \tag{31}$$

Errors study

Gaussian assumption: p_{data} is a centered Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$. ($\mathbf{\Sigma}$ is not necessarily invertible)



Initialization + discretization



Truncature + Initialization

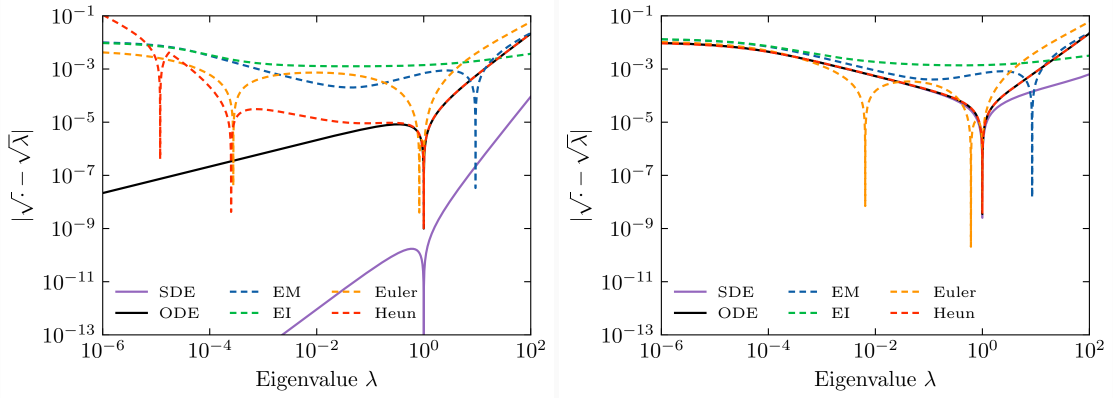
Discretization schemes

SDE schemes	Euler-Maruyama (EM)	$\begin{cases} \tilde{\boldsymbol{y}}_0^{\Delta,\text{EM}} & \sim \mathcal{N}_0 \\ \tilde{\boldsymbol{y}}_{k+1}^{\Delta,\text{EM}} & = \tilde{\boldsymbol{y}}_k^{\Delta,\text{EM}} + \Delta_t \beta_{T-t_k} \left(\tilde{\boldsymbol{y}}_k^{\Delta,\text{EM}} - 2\boldsymbol{\Sigma}_{T-t_k}^{-1} \tilde{\boldsymbol{y}}_k^{\Delta,\text{EM}} \right) + \sqrt{2\Delta_t \beta_{T-t_k}} z_k, \ z_k \sim \mathcal{N}_0 \end{cases}$	(32)
	Exponential integrator (EI)	$\begin{cases} \tilde{\boldsymbol{y}}_0^{\Delta,\text{EI}} & \sim \mathcal{N}_0 \\ \tilde{\boldsymbol{y}}_{k+1}^{\Delta,\text{EI}} & = \tilde{\boldsymbol{y}}_k^{\Delta,\text{EI}} + \gamma_{1,k} \left(\tilde{\boldsymbol{y}}_k^{\Delta,\text{EI}} - 2\boldsymbol{\Sigma}_{T-t_k}^{-1} \tilde{\boldsymbol{y}}_k^{\Delta,\text{EI}} \right) + \sqrt{2\gamma_{2,k}} z_k, \ z_k \sim \mathcal{N}_0 \end{cases}$ where $\gamma_{1,k} = \exp(B_{T-t_k} - B_{T-t_{k+1}}) - 1$ and $\gamma_{2,k} = \frac{1}{2}(\exp(2B_{T-t_k} - 2B_{T-t_{k+1}}) - 1)$	(33)
ODE schemes	Explicit Euler	$\begin{cases} \widehat{\boldsymbol{y}}_0^{\Delta,\text{Euler}} & \sim \mathcal{N}_0 \\ \widehat{\boldsymbol{y}}_{k+1}^{\Delta,\text{Euler}} & = \widehat{\boldsymbol{y}}_k^{\Delta,\text{Euler}} + \Delta_t f(t_k, \widehat{\boldsymbol{y}}_k^{\Delta,\text{Euler}}) \quad \text{with } f(t,y) = \beta_{T-t} y - \beta_{T-t} \boldsymbol{\Sigma}_{T-t}^{-1} y \end{cases}$	(34)
	Heun's method	$\begin{cases} \widehat{\boldsymbol{y}}_0^{\Delta,\text{Heun}} & \sim \mathcal{N}_0 \\ \widehat{\boldsymbol{y}}_{k+1/2}^{\Delta,\text{Heun}} & = \widehat{\boldsymbol{y}}_k^{\Delta,\text{Heun}} + \Delta_t f(t_k, \widehat{\boldsymbol{y}}_k^{\Delta,\text{Heun}}) \quad \text{with } f(t,y) = \beta_{T-t} y - \beta_{T-t} \boldsymbol{\Sigma}_{T-t}^{-1} y \\ \widehat{\boldsymbol{y}}_{k+1}^{\Delta,\text{Heun}} & = \widehat{\boldsymbol{y}}_k^{\Delta,\text{Heun}} + \frac{\Delta_t}{2} \left(f(t_k, \widehat{\boldsymbol{y}}_k^{\Delta,\text{Heun}}) + f(t_{k+1}, \widehat{\boldsymbol{y}}_{k+1/2}^{\Delta,\text{Heun}}) \right) \end{cases}$	(35)

Separable 2-Wasserstein distance

For two centered Gaussians $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_1)$ and $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_2)$ such that $\mathbf{\Sigma}_1, \mathbf{\Sigma}_2$ are simultaneously diagonalizable with respective eigenvalues $(\lambda_{i,1})_{1\leqslant i\leqslant d}$, $(\lambda_{i,2})_{1\leqslant i\leqslant d}$,

$$\mathbf{W}_2(\mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_1), \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_2))^2 = \sum_{1\leqslant i\leqslant d} (\sqrt{\lambda_{i,1}} - \sqrt{\lambda_{i,2}})^2 \quad (36)$$



Ablation study

		Continuous		$N=50$		$N=250$		$N=500$		$N=1000$	
		p_T	$\mathcal{N}_0$	p_T	$\mathcal{N}_0$	p_T	$\mathcal{N}_0$	p_T	$\mathcal{N}_0$	p_T	$\mathcal{N}_0$
EM	$\varepsilon=0$	0	6.7E-04	4.78	4.78	0.65	0.66	0.32	0.32	0.16	0.16
	$\varepsilon=10^{-5}$	4.1E-03	4.2E-03	4.77	4.77	0.66	0.66	0.32	0.32	0.16	0.16
	$\varepsilon=10^{-4}$	0.03	0.03	4.76	4.76	0.66	0.66	0.32	0.32	0.17	0.17
	$\varepsilon=10^{-3}$	0.18	0.18	4.68	4.68	0.70	0.70	0.40	0.40	0.27	0.27
EI	$\varepsilon=0$	0	6.7E-04	2.81	2.81	0.57	0.57	0.30	0.30	0.16	0.16
	$\varepsilon=10^{-5}$	4.1E-03	4.2E-03	2.81	2.81	0.57	0.57	0.30	0.30	0.16	0.16
	$\varepsilon=10^{-4}$	0.03	0.03	2.82	2.82	0.58	0.58	0.31	0.31	0.17	0.17
	$\varepsilon=10^{-3}$	0.18	0.18	2.91	2.91	0.67	0.67	0.41	0.41	0.29	0.29
Euler	$\varepsilon=0$	0	0.07	1.72	1.78	0.38	0.44	0.20	0.26	0.10	0.17
	$\varepsilon=10^{-5}$	4.1E-03	0.07	1.72	1.78	0.38	0.44	0.20	0.26	0.10	0.17
	$\varepsilon=10^{-4}$	0.03	0.08	1.72	1.78	0.38	0.44	0.20	0.26	0.11	0.17
	$\varepsilon=10^{-3}$	0.18	0.19	1.73	1.79	0.42	0.48	0.27	0.32	0.21	0.25
Heun	$\varepsilon=0$	0	0.07	-	-	-	-	-	-	-	-
	$\varepsilon=10^{-5}$	4.1E-03	0.07	23.42	23.42	2.86	2.87	1.05	1.06	0.37	0.38
	$\varepsilon=10^{-4}$	0.03	0.08	4.68	4.68	0.43	0.44	0.12	0.14	0.03	0.08
	$\varepsilon=10^{-3}$	0.18	0.19	0.58	0.59	0.13	0.15	0.16	0.18	0.17	0.19

Score approximation error

Score approximation error

$$dy_t = -\beta_t[y_t + 2s_\theta(t, y_t)]dt + \sqrt{2\beta_t}d\overline{w}_t,$$

or where $\varepsilon \leqslant t \leqslant T, \quad y_T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}).$ (37)

$$dy_t = -\beta_t[y_t + s_\theta(t, y_t)]dt,$$

Sampling a distribution using diffusion models implies different choices and error types:

- p_T, which is unknown, is replaced by $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$ → **initialization error**
- In fact, another time ε to consider them on $[\varepsilon, T]$ → **truncation error**
- A scheme to discretize the equations → **discretization error**
- A model/neural network alerts_θ to learn the score → **score approximation error**

The Asymptotic Discrete Spot Noise (ADSN) model [Galerne, Gousseau, and Morel 2011] [6]

A Gaussian distribution, named $\mathrm{ADSN}(\boldsymbol{u})$, can be associated with a texton $\boldsymbol{u}$

$\boldsymbol{u}$

Samples of $\mathrm{ADSN}(\boldsymbol{u})$

[6] Bruno Galerne, Yann Gousseau, and Jean-Michel Morel (2011). "Random Phase Textures: Theory and Synthesis". In: *IEEE Transactions on Image Processing* 20.1, pp. 257–267. DOI: 10.1109/TIP.2010.2052822

The Asymptotic Discrete Spot Noise (ADSN) model [Galerne, Gousseau, and Morel 2011] [6]

A Gaussian distribution, named $\mathrm{ADSN}(\boldsymbol{u})$, can be associated with a texton $\boldsymbol{u}$

$\boldsymbol{u}$

Samples of $\mathrm{ADSN}(\boldsymbol{u})$

[6] Bruno Galerne, Yann Gousseau, and Jean-Michel Morel (2011). "Random Phase Textures: Theory and Synthesis". In: *IEEE Transactions on Image Processing* 20.1, pp. 257–267. DOI: 10.1109/TIP.2010.2052822

A Gaussian distribution, named $\mathrm{ADSN}(\boldsymbol{u})$, can be associated with a texton $\boldsymbol{u}$

$\boldsymbol{u}$

Samples of $\mathrm{ADSN}(\boldsymbol{u})$

[6] Bruno Galerne, Yann Gousseau, and Jean-Michel Morel (2011). "Random Phase Textures: Theory and Synthesis". In: *IEEE Transactions on Image Processing* 20.1, pp. 257–267. DOI: 10.1109/TIP.2010.2052822

The Asymptotic Discrete Spot Noise (ADSN) model [Galerne, Gousseau, and Morel 2011] [6]

A Gaussian distribution, named $\mathrm{ADSN}(\boldsymbol{u})$, can be associated with a texton $\boldsymbol{u}$

$\boldsymbol{u}$

Samples of $\mathrm{ADSN}(\boldsymbol{u})$

[6] Bruno Galerne, Yann Gousseau, and Jean-Michel Morel (2011). "Random Phase Textures: Theory and Synthesis". In: *IEEE Transactions on Image Processing* 20.1, pp. 257–267. DOI: 10.1109/TIP.2010.2052822

The Asymptotic Discrete Spot Noise (ADSN) model [Galerne, Gousseau, and Morel 2011] [6]

A Gaussian distribution, named $\mathrm{ADSN}(\boldsymbol{u})$, can be associated with a texton $\boldsymbol{u}$

$\boldsymbol{u}$

Samples of $\mathrm{ADSN}(\boldsymbol{u})$

[6] Bruno Galerne, Yann Gousseau, and Jean-Michel Morel (2011). "Random Phase Textures: Theory and Synthesis". In: *IEEE Transactions on Image Processing* 20.1, pp. 257–267. DOI: 10.1109/TIP.2010.2052822

We learn $s_\theta \approx \nabla_x \log p_t(x)$ for the previous ADSN model by score matching, using *DDPM continuous* [Song et al. 2021][7].

$$\theta^\star = \operatorname{argmin}_\theta \mathbb{E}_t \left\{ \lambda(t) \mathbb{E}_{x_0} \mathbb{E}_{x_t|x_0} \left[s_\theta(x_t, t) - \nabla_{x_t} \log p_t(x_0 \mid x_t) \right] \right\} \quad (38)$$

x_0 is sampled during the training.

[7] Yang Song et al. (2021). "Score-Based Generative Modeling through Stochastic Differential Equations". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=PxTIG12RRHS

We learn $s_\theta \approx \nabla_x \log p_t(x)$ for the previous ADSN model by score matching, using *DDPM continuous* [Song et al. 2021][7].

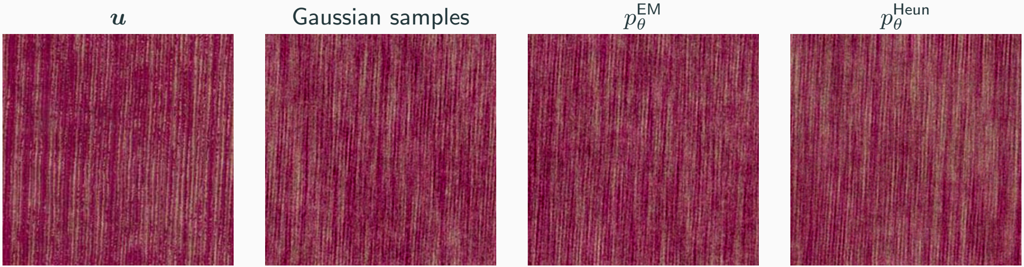
$$\theta^\star = \operatorname{argmin}_\theta \mathbb{E}_t \left\{ \lambda(t) \mathbb{E}_{x_0} \mathbb{E}_{x_t | x_0} \left[s_\theta(x_t, t) - \nabla_{x_t} \log p_t(x_0 \mid x_t) \right] \right\} \tag{38}$$

x_0 is sampled during the training.

Then, we apply the EM scheme and the Heun's scheme with this learned network.

[7] Yang Song et al. (2021). "Score-Based Generative Modeling through Stochastic Differential Equations". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=PxTIG12RRHS

Score approximation



Score approximation

$\boldsymbol{u}$

Gaussian samples

p_θ^{EM}

p_θ^{Heun}

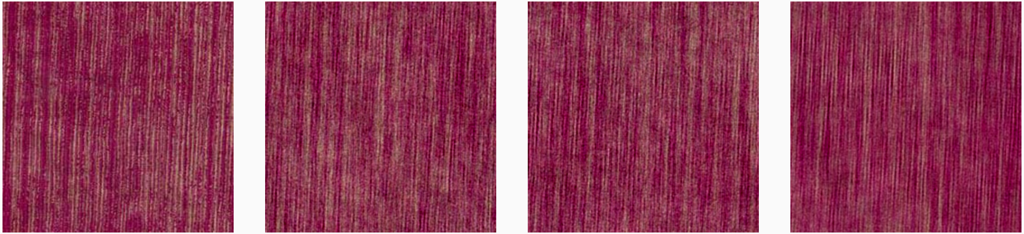
$\boldsymbol{u}$

Gaussian samples

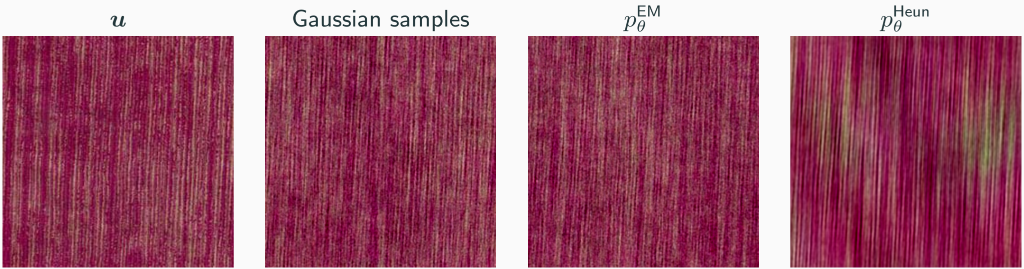
p_θ^{EM}

p_θ^{Heun}

Score approximation



Score approximation



$\boldsymbol{u}$

Gaussian samples

p_θ^{EM}

p_θ^{Heun}

	Exact score distribution			Learned score distribution	
p	$\mathbf{W}_2(p, p_{\text{data}}) \downarrow$	$\mathbf{W}_2^{\text{emp.}}(p^{\text{emp.}}, p_{\text{data}}) \downarrow$	$\text{FID}(p^{\text{emp.}}, p_{\text{data}}^{\text{emp.}}) \downarrow$	$\mathbf{W}_2^{\text{emp.}}(p_\theta^{\text{emp.}}, p_{\text{data}}^{\text{emp.}}) \downarrow$	$\text{FID}(p_\theta^{\text{emp.}}, p_{\text{data}}^{\text{emp.}}) \downarrow$
EM	5.16	5.1630	0.0891	15.6	1.02
Heun	3.73	3.7323	0.0447	56.7	19.4

Conclusion

Conclusion

- Extension to GMM ?

Conclusion

- Extension to GMM ?
  - Exact solutions are not known.

Conclusion

- Extension to GMM ?
  - Exact solutions are not known.
  - A closed form for the $\mathbf{W}_2$ distance is not known.

- Extension to GMM ?
  - Exact solutions are not known.
  - A closed form for the $\mathbf{W}_2$ distance is not known.
  - The discretized processes are no more GMM.

Conclusion

- Extension to GMM ?
  - Exact solutions are not known.
  - A closed form for the $\mathbf{W}_2$ distance is not known.
  - The discretized processes are no more GMM.
- To be compared with the bounds in the literature.
  (in progress) → Stanislas Strasman et al. (2024). *An analysis of the noise schedule for score-based generative models*. arXiv: 2402.04650 [math.ST]. URL: https://arxiv.org/abs/2402.04650

Conclusion

- Extension to GMM ?
  - Exact solutions are not known.
  - A closed form for the $\mathbf{W}_2$ distance is not known.
  - The discretized processes are no more GMM.
- To be compared with the bounds in the literature.
  (in progress) → Stanislas Strasman et al. (2024). *An analysis of the noise schedule for score-based generative models*. arXiv: 2402.04650 [math.ST]. URL: https://arxiv.org/abs/2402.04650
- Extension to conditional diffusion models ?
  (in progress) → Joël Greffier et al. (2024). "Photon-counting CT systems: A technical review of current clinical possibilities". In: *Diagnostic and Interventional Imaging*. ISSN: 2211-5684. DOI: https://doi.org/10.1016/j.diii.2024.09.002. URL: https://www.sciencedirect.com/science/article/pii/S2211568424001955

Thank you for your attention !

References

Choi, Jooyoung et al. (2021). "ILVR: Conditioning Method for Denoising Diffusion Probabilistic Models". In: *ILVR*. Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 14367–14376. URL: https://openaccess.thecvf.com/content/ICCV2021/html/Choi_ILVR_Conditioning_Method_for_Denoising_Diffusion_Probabilistic_Models_ICCV_2021_paper.html (visited on 2022-11-28).

Chung, Hyungjin et al. (2022). "Improving Diffusion Models for Inverse Problems using Manifold Constraints". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

De Bortoli, Valentin (2022). "Convergence of denoising diffusion models under the manifold hypothesis". In: *Transactions on Machine Learning Research*. ISSN: 2835-8856. URL: https://openreview.net/forum?id=MhK5aXo3gB.

Galerne, Bruno, Yann Gousseau, and Jean-Michel Morel (2011). "Random Phase Textures: Theory and Synthesis". In: *IEEE Transactions on Image Processing* 20.1, pp. 257–267. DOI: 10.1109/TIP.2010.2052822.

Greffier, Joël et al. (2024). "Photon-counting CT systems: A technical review of current clinical possibilities". In: *Diagnostic and Interventional Imaging*. ISSN: 2211-5684. DOI: https://doi.org/10.1016/j.diii.2024.09.002. URL: https://www.sciencedirect.com/science/article/pii/S2211568424001955.

Khrulkov, Valentin et al. (2023). "Understanding DDPM Latent Codes Through Optimal Transport". In: *The Eleventh International Conference on Learning Representations*. URL: https://openreview.net/forum?id=6PIrhAx1j4i.

Lavenant, Hugo and Filippo Santambrogio (2022). "The flow map of the Fokker–Planck equation does not provide optimal transport". In: *Applied Mathematics Letters* 133, p. 108225. ISSN: 0893-9659. DOI: https://doi.org/10.1016/j.aml.2022.108225. URL: https://www.sciencedirect.com/science/article/pii/S089396592200180X.

Lugmayr, Andreas et al. (2022). "RePaint: Inpainting using Denoising Diffusion Probabilistic Models". In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11451–11461. URL: https://api.semanticscholar.org/CorpusID:246240274.

Pardoux, E. (1986). "Grossissement d'une filtration et retournement du temps d'une diffusion". In: *Séminaire de Probabilités XX 1984/85*. Ed. by Jacques Azéma and Marc Yor. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 48–55. ISBN: 978-3-540-39860-8.

Song, Yang et al. (2021). "Score-Based Generative Modeling through Stochastic Differential Equations". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=PxTIG12RRHS.

Strasman, Stanislas et al. (2024). *An analysis of the noise schedule for score-based generative models*. arXiv: 2402.04650 [math.ST]. URL: https://arxiv.org/abs/2402.04650.

To the restoration problems ?

To the restoration problems ?

My thesis title: Stochastic super resolution using deep generative models

To the restoration problems ?

My thesis title: Stochastic super resolution using deep generative models

→ We need to use **conditional** diffusion model !

How to perform conditional simulation ?

What is the link with solving inverse problems $\boldsymbol{v} = \boldsymbol{A}x + \sigma\boldsymbol{\varepsilon}$?

How to perform conditional simulation ?

What is the link with solving inverse problems $\boldsymbol{v} = \boldsymbol{A}x + \sigma\varepsilon$?

A large literature [Song et al. 2021[8],Lugmayr et al. 2022[9],Chung et al. 2022[10],Choi et al. 2021[11]] uses the Bayes formula

$$\nabla_x \log p_t(x_t \mid \boldsymbol{v}) = \nabla_x \log p_t(\boldsymbol{v} \mid x_t) + \nabla_x \log p_t(x_t). \tag{39}$$

where $\nabla_x \log p_t(x_t)$ is the unconditional score. Consequently, studying the unconditional case provides information for the conditional one.

[8] Yang Song et al. (2021). "Score-Based Generative Modeling through Stochastic Differential Equations". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=PxTIG12RRHS

[9] Andreas Lugmayr et al. (2022). "RePaint: Inpainting using Denoising Diffusion Probabilistic Models". In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11451–11461. URL: https://api.semanticscholar.org/CorpusID:246240274

[10] Hyungjin Chung et al. (2022). "Improving Diffusion Models for Inverse Problems using Manifold Constraints". In: *Advances in Neural Information Processing Systems (NeurIPS)*

[11] Jooyoung Choi et al. (2021). "ILVR: Conditioning Method for Denoising Diffusion Probabilistic Models". In: *ILVR*. Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 14367–14376. URL: https://openaccess.thecvf.com/content/ICCV2021/html/Choi_ILVR_Conditioning_Method_for_Denoising_Diffusion_Probabilistic_Models_ICCV_2021_paper.html (visited on 2022-11-28)

Ablation study

		Continuous		$N = 50$		$N = 250$		$N = 500$		$N = 1000$	
		p_T	$\mathcal{N}_0$	p_T	$\mathcal{N}_0$	p_T	$\mathcal{N}_0$	p_T	$\mathcal{N}_0$	p_T	$\mathcal{N}_0$
EM	$\varepsilon = 0$	0	6.7E-4	4.77	4.77	0.65	0.65	0.31	0.31	0.15	0.16
	$\varepsilon = 10^{-5}$	2.5E-3	2.6E-3	4.77	4.77	0.65	0.65	0.31	0.31	0.16	0.16
	$\varepsilon = 10^{-3}$	0.17	0.17	4.67	4.67	0.69	0.69	0.39	0.39	0.27	0.27
	$\varepsilon = 10^{-2}$	1.35	1.35	4.56	4.56	1.69	1.69	1.50	1.50	1.42	1.42
EI	$\varepsilon = 0$	0	6.7E-4	2.81	2.81	0.57	0.57	0.30	0.30	0.16	0.16
	$\varepsilon = 10^{-5}$	2.5E-3	2.6E-3	2.81	2.81	0.57	0.57	0.30	0.30	0.16	0.16
	$\varepsilon = 10^{-3}$	0.17	0.17	2.91	2.91	0.66	0.66	0.41	0.41	0.28	0.28
	$\varepsilon = 10^{-2}$	1.35	1.35	3.93	3.93	1.76	1.76	1.55	1.55	1.45	1.45
Euler	$\varepsilon = 0$	0	0.07	1.72	1.78	0.38	0.44	0.19	0.26	0.10	0.17
	$\varepsilon = 10^{-5}$	2.5E-3	0.07	1.72	1.78	0.38	0.44	0.20	0.26	0.10	0.17
	$\varepsilon = 10^{-3}$	0.17	0.19	1.72	1.78	0.42	0.48	0.27	0.32	0.21	0.25
	$\varepsilon = 10^{-2}$	1.35	1.36	2.21	2.25	1.41	1.43	1.37	1.38	1.36	1.37
Heun	$\varepsilon = 0$	0	0.07	7.09	7.09	0.72	0.73	0.21	0.22	0.05	0.09
	$\varepsilon = 10^{-5}$	2.5E-3	0.07	6.48	6.48	0.64	0.65	0.18	0.20	0.05	0.09
	$\varepsilon = 10^{-3}$	0.17	0.19	0.56	0.57	0.13	0.15	0.16	0.18	0.17	0.19
	$\varepsilon = 10^{-2}$	1.35	1.36	1.37	1.38	1.35	1.36	1.35	1.36	1.35	1.36