A Precise Examination of Diffusion Models via Their Application to Gaussian Distributions

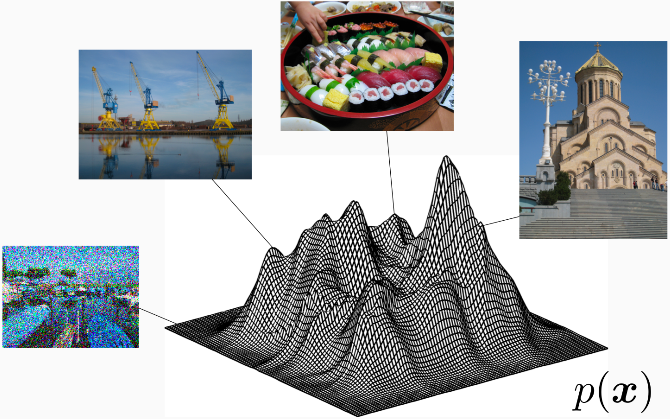
Émile Pierret[a], work supervised by Bruno Galerne[b,c]
CSD Seminar
October 9^{th} 2025

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

Introduction

What is a generative model ?

Goal: Sample from a data distribution of images.



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 $G(\mathcal{N}(\mathbf{0}, \boldsymbol{I}_N), \Theta^\star) \approx p$

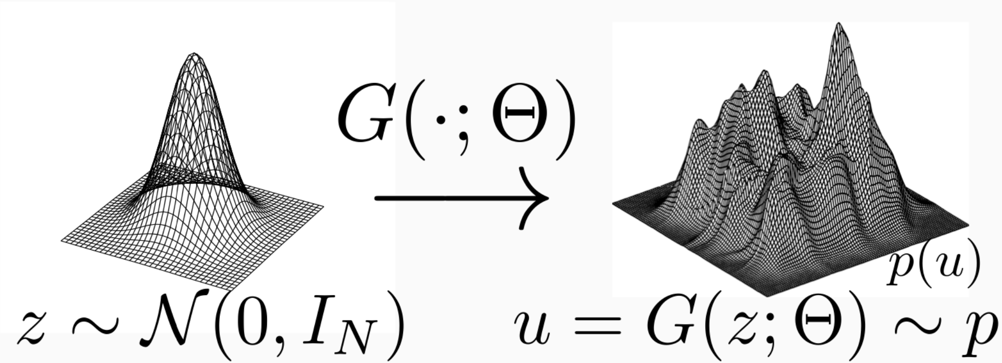


Image extracted from Bruno Galerne's slides

Diffusion models through SDE

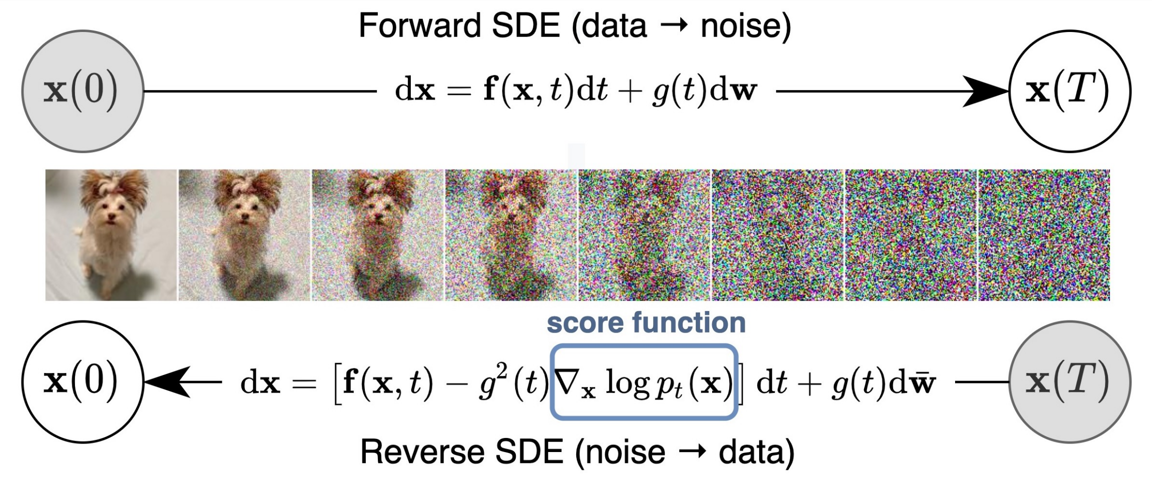
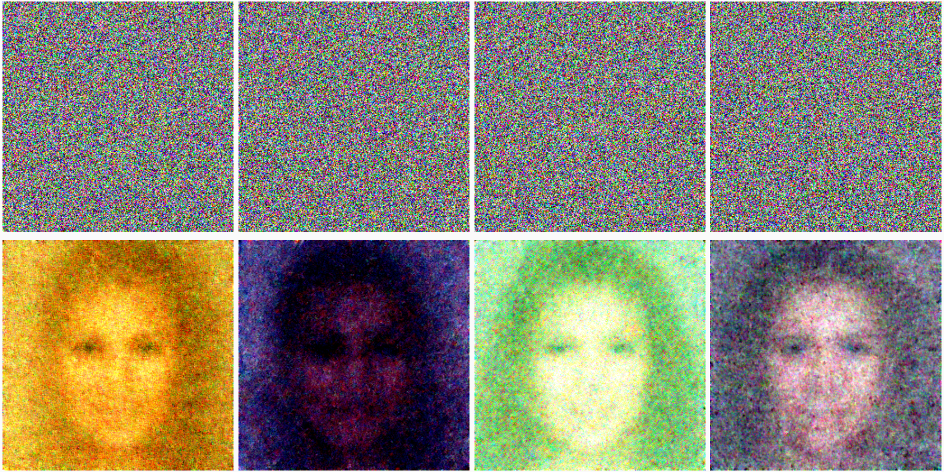


Image extracted from [Song et al. 2021]

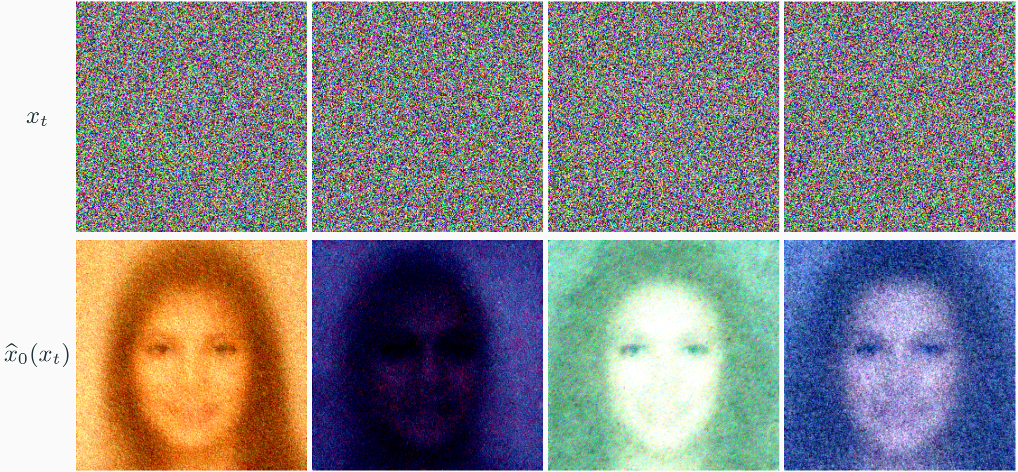
Examples (generated with [Lugmayr et al. 2022][1])



t = 249

[1] 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`

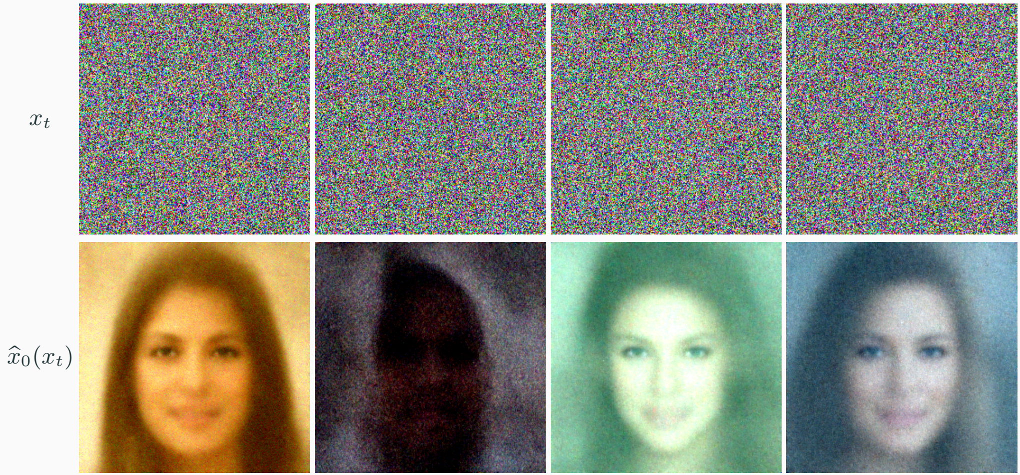
Examples (generated with [Lugmayr et al. 2022][1])



t = 230

[1] 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

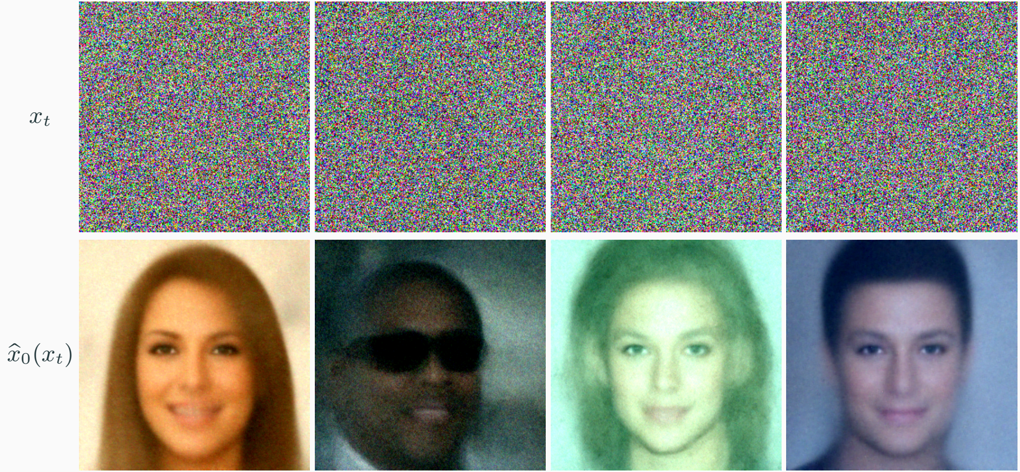
Examples (generated with [Lugmayr et al. 2022][1])



t = 210

[1] 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][1])



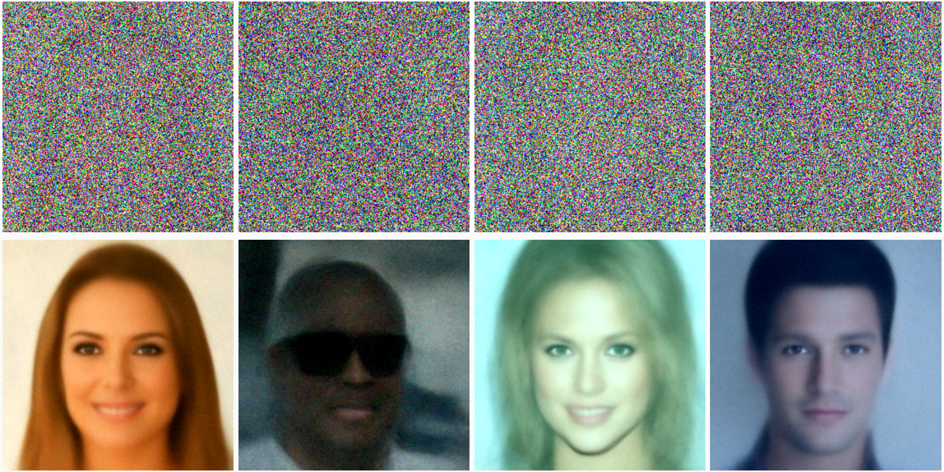
$t = 190$

[1] 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][1])

x_t

$\widehat{x}_0(x_t)$



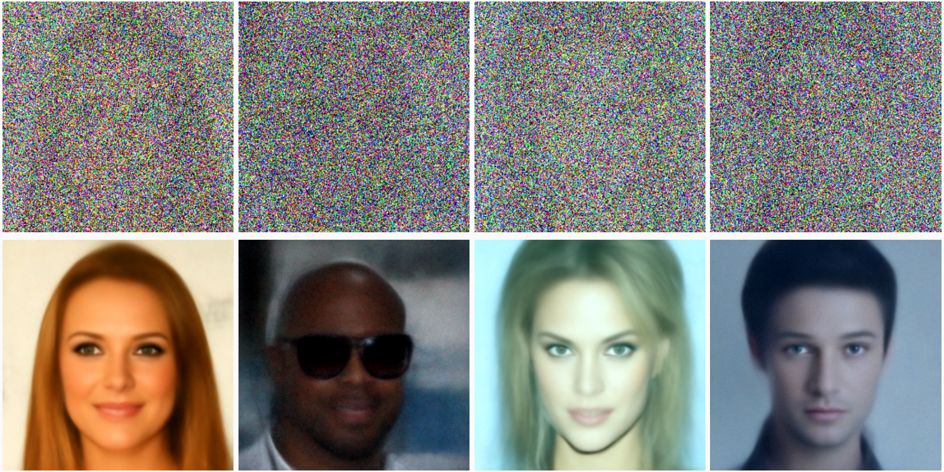
t = 170

[1] 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][1])

x_t

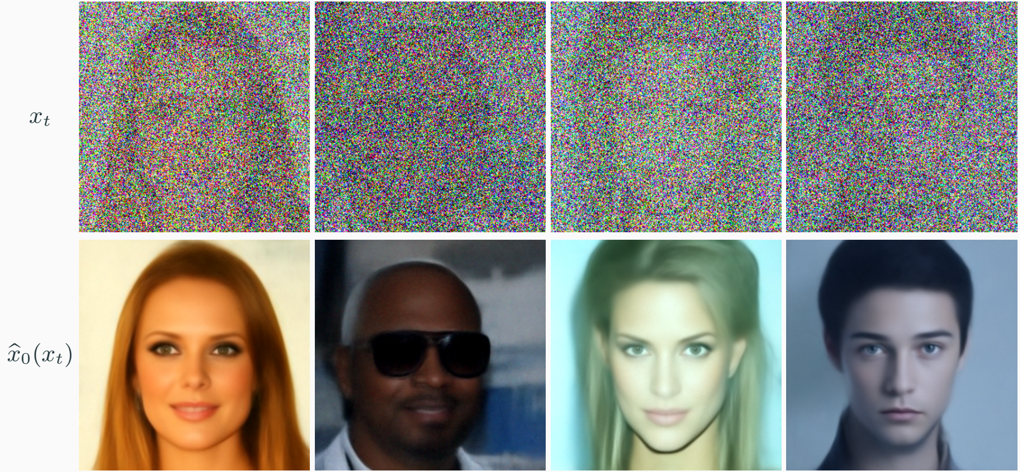
$\widehat{x}_0(x_t)$



t = 150

[1] 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][1])



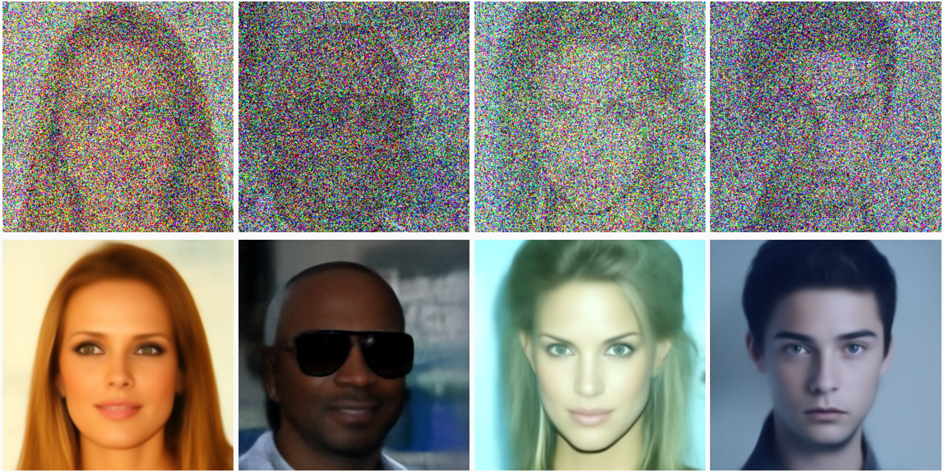
t = 130

[1] 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][1])

x_t

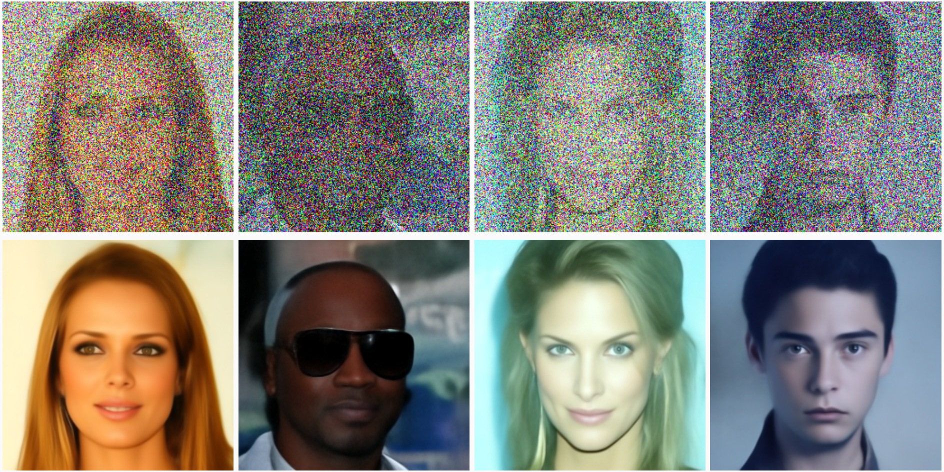
$\widehat{x}_0(x_t)$



$t = 110$

[1] 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][1])



x_t

$\widehat{x}_0(x_t)$

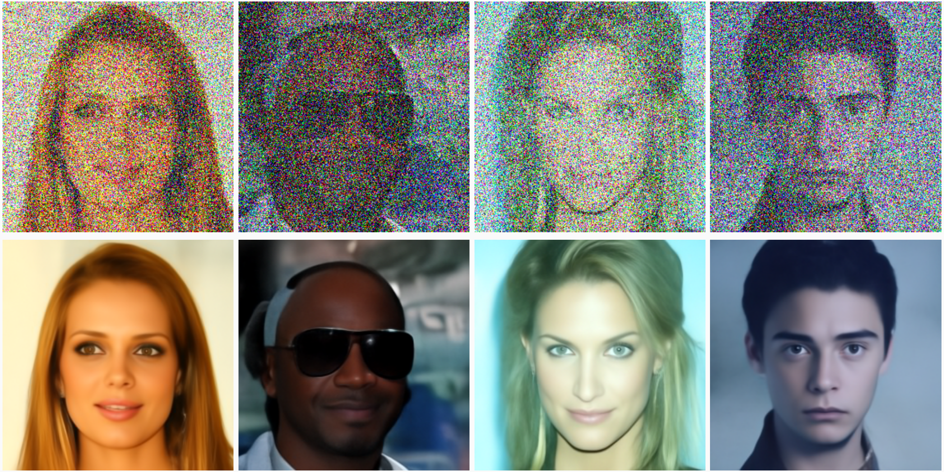
t = 90

[1] 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][1])

x_t

$\widehat{x}_0(x_t)$

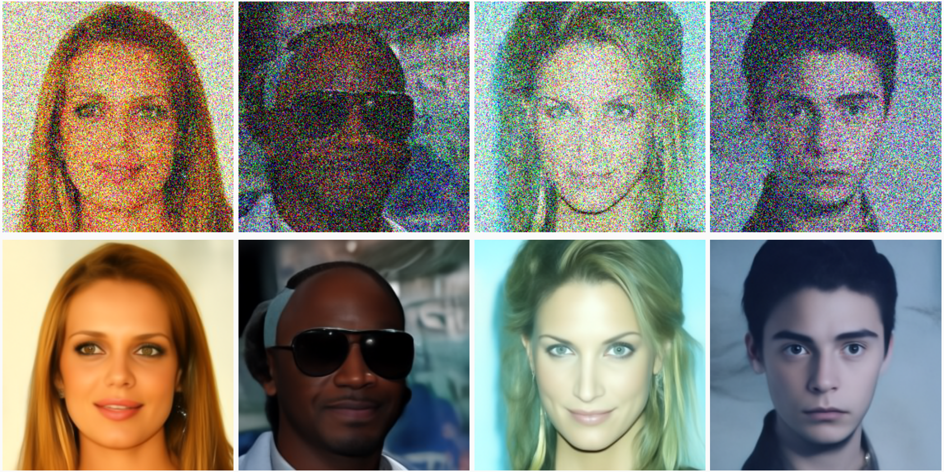


t = 70

[1] 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 = 50

[1] 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][1])

x_t

$\widehat{x}_0(x_t)$

t = 30

[1] 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][1])

x_t

$\widehat{x}_0(x_t)$

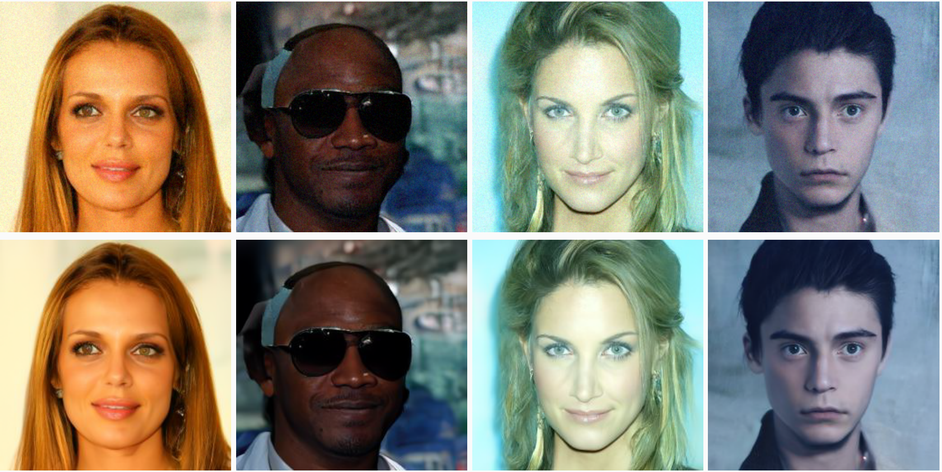
$t = 10$

[1] 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][1])

x_t

$\widehat{x}_0(x_t)$



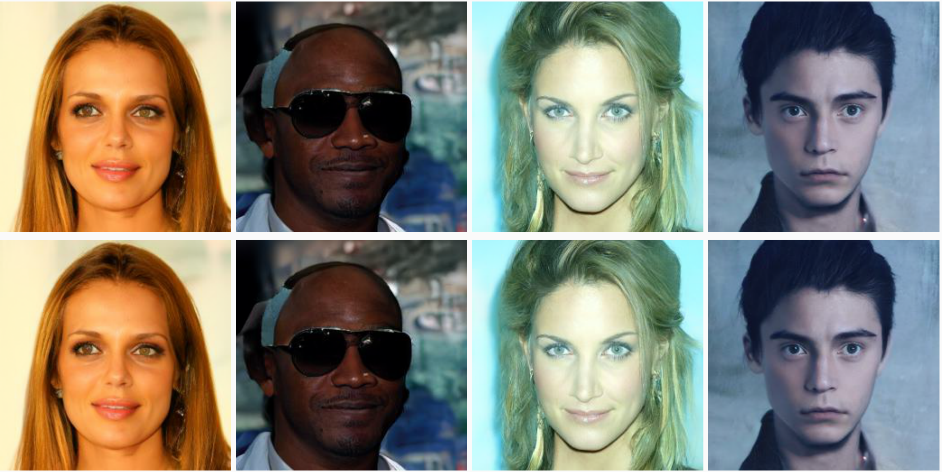
$t = 5$

[1] 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][1])

x_t

$\widehat{x}_0(x_t)$



t = 0

[1] 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`

Introduction to diffusion models through SDEs

Focus on the VP-SDE: the forward process

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

The strong solution of this equation is:

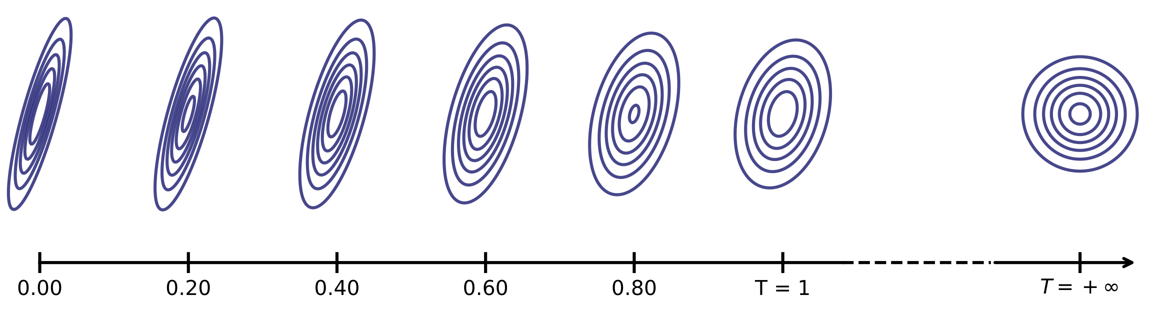
$$x_t = e^{-B_t} x_0 + \boldsymbol{\eta}_t, \quad 0 \leqslant t \leqslant T, \text{ with } \eta_t \sim \mathcal{N}(\mathbf{0}, \left(1 - e^{-2B_t}\right) \boldsymbol{I}), B_t = \int_0^t \beta_u du.$$

Focus on the VP-SDE: the forward process

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

The strong solution of this equation is:

$$x_t = e^{-B_t} x_0 + \boldsymbol{\eta}_t, \quad 0 \leqslant t \leqslant T, \text{ with } \eta_t \sim \mathcal{N}(\mathbf{0}, \left(1 - e^{-2B_t}\right) \boldsymbol{I}), B_t = \int_0^t \beta_u du.$$



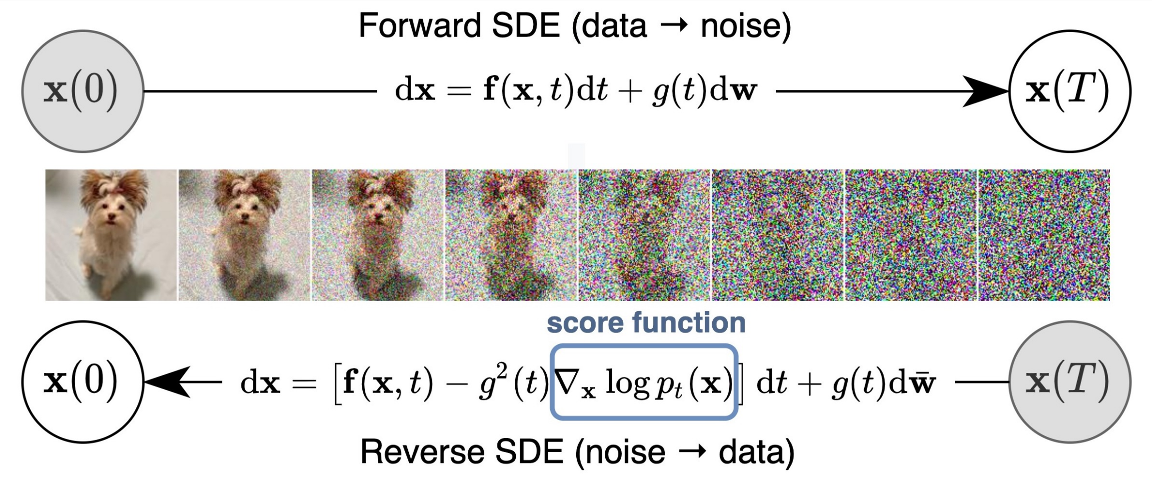


Image extracted from [Song et al. 2021]

Forward equation

$$dx_t = -\beta_t x_t + \sqrt{2\beta_t} dw_t, \quad 1 \leqslant t \leqslant T$$

↓ via p_t density distribution of x_t

Fokker-Planck equation

$$\partial_t p_t(x) = -\operatorname{div}(f(x,t)p_t(x)) + \frac{1}{2}\Delta_x p_t(x), \quad 1 \leqslant t \leqslant T$$

↓ Reversed in time

"Reversed" Fokker-Planck equation

$$\partial_t q_t(x) = -\operatorname{div}_x\left[\left(-f(x,T-t) + g(T-t)^2\nabla_x \log q_t(x)\right) q_t(x)\right] + \frac{(g(T-t))^2}{2} q_t(x), \quad 1 \leqslant t \leqslant T$$

↓

A family of backward equations

$$dy_t = \left[-\beta_{T-t} y_t + (1+\alpha)\beta_t \nabla_y \log p_{T-t}(y_t)\right] dt + \sqrt{2\alpha\beta_{T-t}} dw_t, \quad 1 \leqslant t \leqslant T.$$

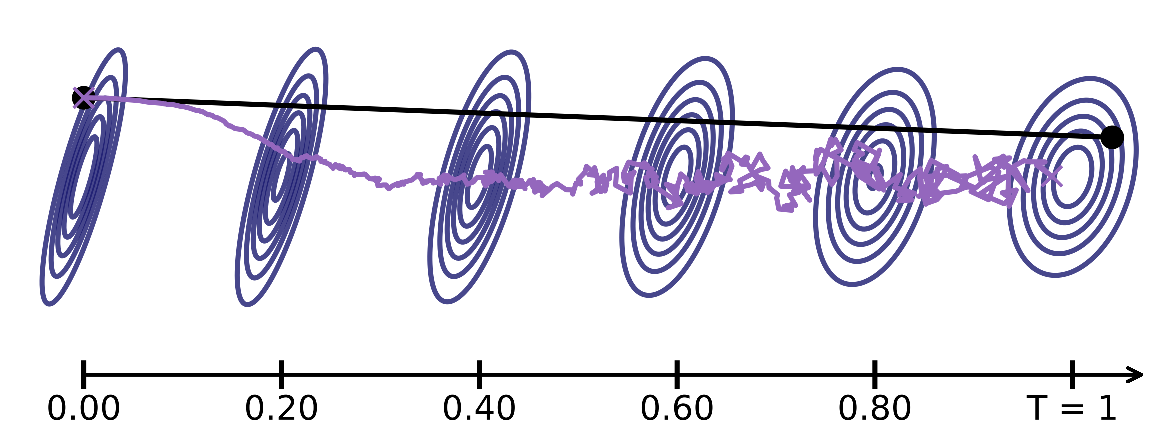
SDE ($\alpha = 1$):

$$dy_t = \left[-\beta_{T-t} y_t + 2\beta_t \nabla_y \log p_{T-t}(y_t)\right] dt + \sqrt{2\beta_{T-t}} dw_t.$$

ODE ($\alpha = 0$):

$$dy_t = \left[-\beta_{T-t} y_t + \beta_t \nabla_y \log p_{T-t}(y_t)\right] dt.$$

SDE ($\alpha = 1$), ODE ($\alpha = 0$)



Study of the convergence of diffusion models

Illustration of the different error types

Theoretical setting

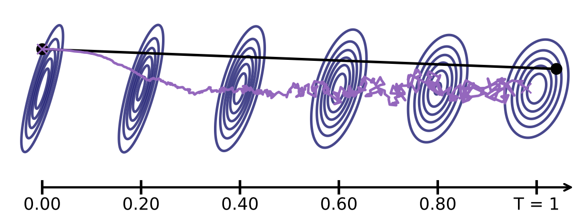
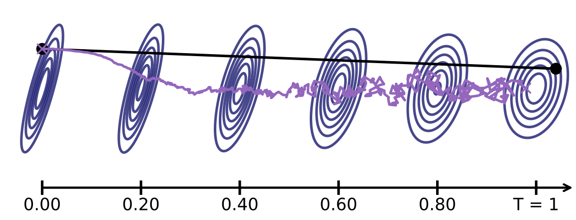


Illustration of the different error types

Theoretical setting



Initialization error

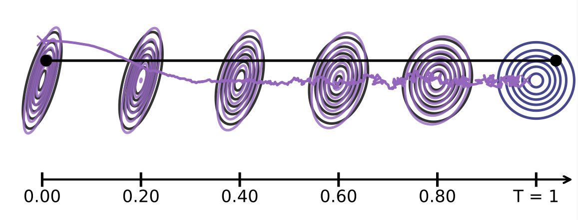
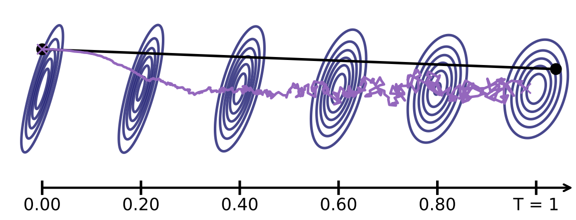
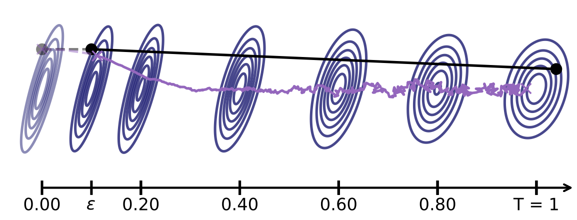


Illustration of the different error types

Theoretical setting



Truncation error



Initialization error

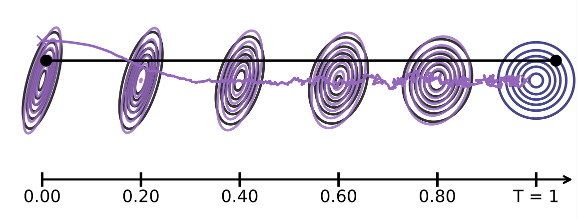
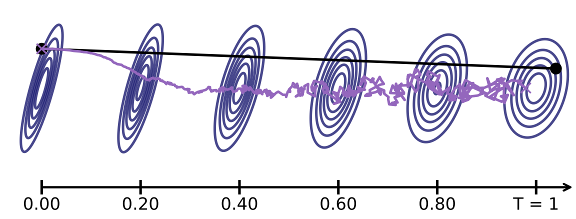
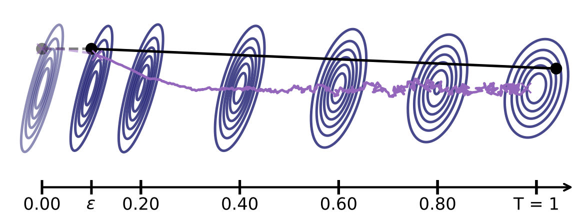


Illustration of the different error types

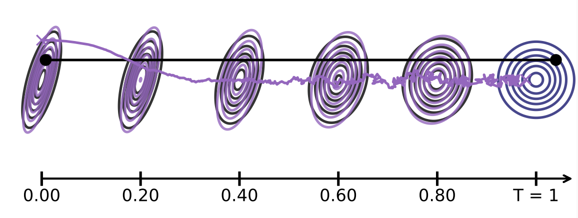
Theoretical setting



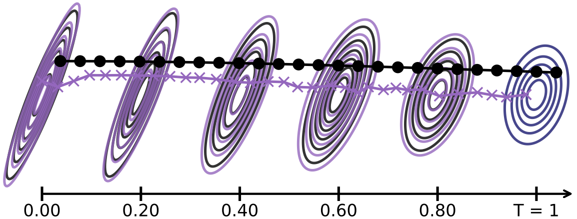
Truncation error



Initialization error



Discretization error



$$dy_t = \left[\beta_t y_t + (1+\alpha)\beta_t \nabla_y \log p_t(y_t)\right] dt + \sqrt{2\alpha\beta_t} dw_t, \quad 0 \leqslant t \leqslant T, \quad y_T \sim p_T.$$

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

Sampling through diffusion models

$$dy_t = [-\beta_t y_t + (1+\alpha)\beta_t \nabla_y \log p_t(y_t)]\, dt + \sqrt{2\alpha\beta_t} dw_t, \quad 0 \leqslant t \leqslant T, \quad y_T \sim \underbrace{\cancel{y_T \sim p_T}}_{y_T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})} .$$

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

$$dy_t = \left[-\beta_t y_t + (1+\alpha)\beta_t \nabla_y \log p_t(y_t)\right] dt + \sqrt{2\alpha\beta_t} dw_t, \quad \underset{\varepsilon}{\not{0}} \leqslant t \leqslant T, \quad y_T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}).$$

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**
- In fact, another time ε to consider them on $[\varepsilon, T]$ $\rightarrow$ **truncation error**

Sampling through diffusion models

$$dy_t = \left[-\beta_t y_t + (1+\alpha)\beta_t \nabla_y \log p_t(y_t)\right] dt + \sqrt{2\alpha\beta_t} dw_t, \quad \varepsilon \leqslant t \leqslant T, \quad y_T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}).$$

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

$$dy_t = \left[-\beta_t y_t + (1+\alpha)\beta_t \underbrace{\nabla_y \log p_t(y_t)}_{s_\theta(t,y_t)} \right] dt + \sqrt{2\alpha\beta_t} dw_t, \quad \varepsilon \leqslant t \leqslant T, \quad y_T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}).$$

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

$$dy_t = [-\beta_t y_t + (1+\alpha)\beta_t s_\theta(t, y_t)]\, dt + \sqrt{2\alpha\beta_t} dw_t, \quad \varepsilon \leqslant t \leqslant T, \quad y_T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}).$$

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**
- In fact, another time ε to consider them on $[\varepsilon, T]$ $\rightarrow$ **truncation error**
- A scheme to discretize the equations $\rightarrow$ **discretization error**
- A model/neural network s_θ to learn the score $\rightarrow$ **score approximation error**

$\Longrightarrow$ Let us focus on the **initialization error** and the **discretization error**.

Restriction to the Gaussian case

Gaussian assumption

Gaussian assumption: p_{data} is a centered Gaussian distribution $\mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$. ($\boldsymbol{\Sigma}$ 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,

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

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

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,

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

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

$$dy_t = \left[-\beta_t y_t + (1+\alpha)\beta_t \nabla \log p_t(y_t)\right] dt + \sqrt{2\alpha\beta_t} dw_t, \quad \varepsilon \leqslant t \leqslant T, \quad y_T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}). \qquad (1_\alpha)$$

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

Explicit solution of the backward equations

$$dy_t = \left[-\beta_t y_t + (1+\alpha)\beta_t \nabla \log p_t(y_t)\right] dt + \sqrt{2\alpha\beta_t} dw_t, \quad \varepsilon \leqslant t \leqslant T, \quad y_T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}). \tag{1_α}$$

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

Proposition 4: Solution to the backward equations under Gaussian assumption

For $\alpha \in \mathbb{R}^+$, the strong solution of Equation (1_α) is

$$y_t = e^{-\alpha(B_T - B_{T-t})} \boldsymbol{\Sigma}_{T-t}^{\frac{1+\alpha}{2}} \boldsymbol{\Sigma}_T^{-\frac{1+\alpha}{2}} y_0 + \boldsymbol{\xi}_t \text{ with } \operatorname{Cov}(\boldsymbol{\xi}_t) = \boldsymbol{\Sigma}_{T-t} - e^{-2\alpha(B_T - B_{T-t})} \boldsymbol{\Sigma}_{T-t}^{1+\alpha} \boldsymbol{\Sigma}_T^{-\alpha}$$

Explicit solution of the backward equations

$$dy_t = [-\beta_t y_t + (1+\alpha)\beta_t \nabla \log p_t(y_t)]\, dt + \sqrt{2\alpha\beta_t} dw_t, \quad \varepsilon \leqslant t \leqslant T, \quad y_T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}). \qquad (1_\alpha)$$

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

$$\begin{aligned}
\mathbf{\Sigma}_t^{\mathsf{SDE}} &= \text{Cov}(y_t^{\mathsf{SDE}}) = \mathbf{\Sigma}_t + e^{-2(B_T - B_t)} \mathbf{\Sigma}_t^2 \mathbf{\Sigma}_T^{-2} \left[\text{Cov}(y_T) - \mathbf{\Sigma}_T\right], \\
\mathbf{\Sigma}_t^{\mathsf{ODE}} &= \text{Cov}(y_t^{\mathsf{ODE}}) = \mathbf{\Sigma}_t \mathbf{\Sigma}_T^{-1}\,\text{Cov}(y_T),
\end{aligned}$$

Explicit solution of the backward equations

$$dy_t = [-\beta_t y_t + (1+\alpha)\beta_t \nabla \log p_t(y_t)]\, dt + \sqrt{2\alpha\beta_t} dw_t, \quad \varepsilon \leqslant t \leqslant T, \quad y_T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}). \qquad (1_\alpha)$$

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 $0 \leqslant t \leqslant T$,

$$\boldsymbol{\Sigma}_t^{\mathsf{SDE}} = \operatorname{Cov}(y_t^{\mathsf{SDE}}) = \boldsymbol{\Sigma}_t + e^{-2(B_T - B_t)} \boldsymbol{\Sigma}_t^2 \boldsymbol{\Sigma}_T^{-2} \left[\operatorname{Cov}(y_T) - \boldsymbol{\Sigma}_T\right],$$
$$\boldsymbol{\Sigma}_t^{\mathsf{ODE}} = \operatorname{Cov}(y_t^{\mathsf{ODE}}) = \boldsymbol{\Sigma}_t \boldsymbol{\Sigma}_T^{-1} \operatorname{Cov}(y_T),$$

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

$$dy_t = [-\beta_t y_t + (1+\alpha)\beta_t \nabla \log p_t(y_t)]\, dt + \sqrt{2\alpha\beta_t} dw_t, \quad \varepsilon \leqslant t \leqslant T, \quad y_T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}). \tag{1_α}$$

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

Proposition 7: Solution to the equations under Gaussian assumption

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

$$\begin{aligned}
\boldsymbol{\Sigma}_t^{\text{SDE}} &= \operatorname{Cov}(y_t^{\text{SDE}}) = \boldsymbol{\Sigma}_t + e^{-2(B_T - B_t)} \boldsymbol{\Sigma}_t^2 \boldsymbol{\Sigma}_T^{-2} \left[\operatorname{Cov}(y_T) - \boldsymbol{\Sigma}_T\right], \\
\boldsymbol{\Sigma}_t^{\text{ODE}} &= \operatorname{Cov}(y_t^{\text{ODE}}) = \boldsymbol{\Sigma}_t \boldsymbol{\Sigma}_T^{-1} \operatorname{Cov}(y_T),
\end{aligned}$$

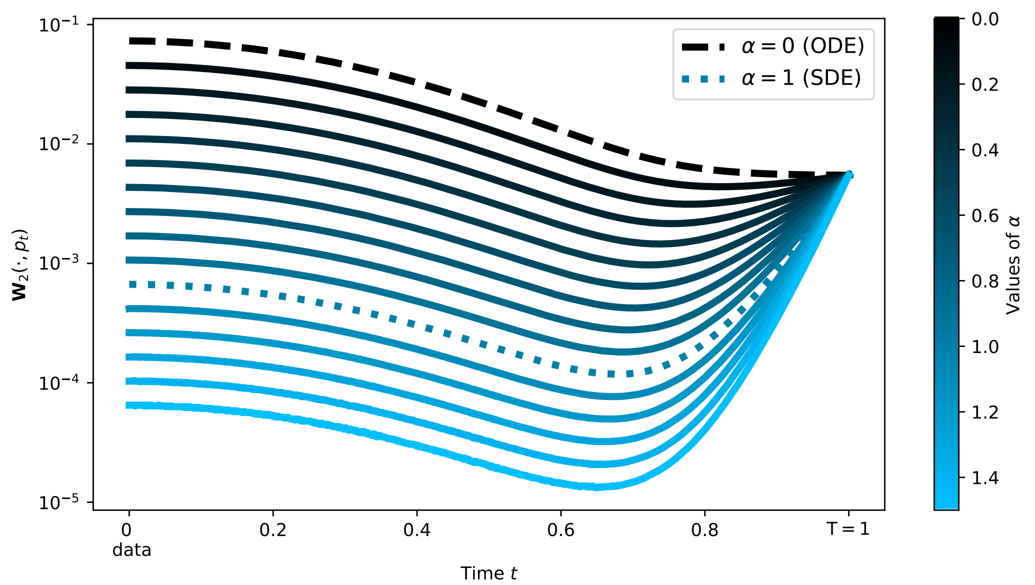
If $y_T \sim p_T = \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_T)$,

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

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

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

Initialization error: exact Wasserstein error



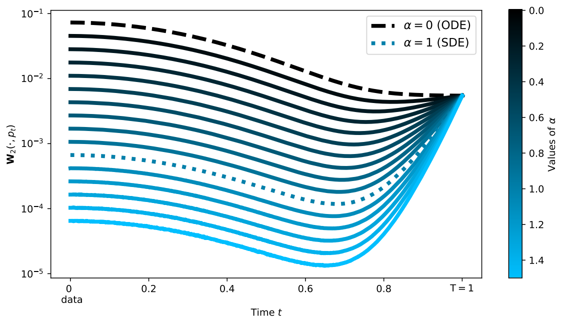
By considering p_t^α the marginals of the backward Equation (1_α),

$$dy_t = \left[-\beta_t y_t + (1+\alpha)\beta_t \nabla \log p_t(y_t)\right] dt + \sqrt{2\alpha\beta_t} dw_t, \quad \varepsilon \leqslant t \leqslant T, \quad y_T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}). \tag{1_α}$$

Proposition 8: Marginals of the generative processes under Gaussian assumption

Under Gaussian assumption, for all $0 \leqslant t \leqslant T$,

$$\mathcal{W}_2(p_t^{\alpha'}, p_t^{\text{Forward}}) \leqslant \mathcal{W}_2(p_t^{\alpha}, p_t^{\text{Forward}}).$$



Exponential forgetting of the initial condition

Under Gaussian assumption, the strong solution to backward SDEs can be written as:

$$y_t = e^{-\alpha(B_T - B_{T-t})} \boldsymbol{\Sigma}_{T-t}^{\frac{1+\alpha}{2}} \boldsymbol{\Sigma}_T^{-\frac{1+\alpha}{2}} y_T + \boldsymbol{\xi}_t, 0 \leqslant t \leqslant T.$$

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

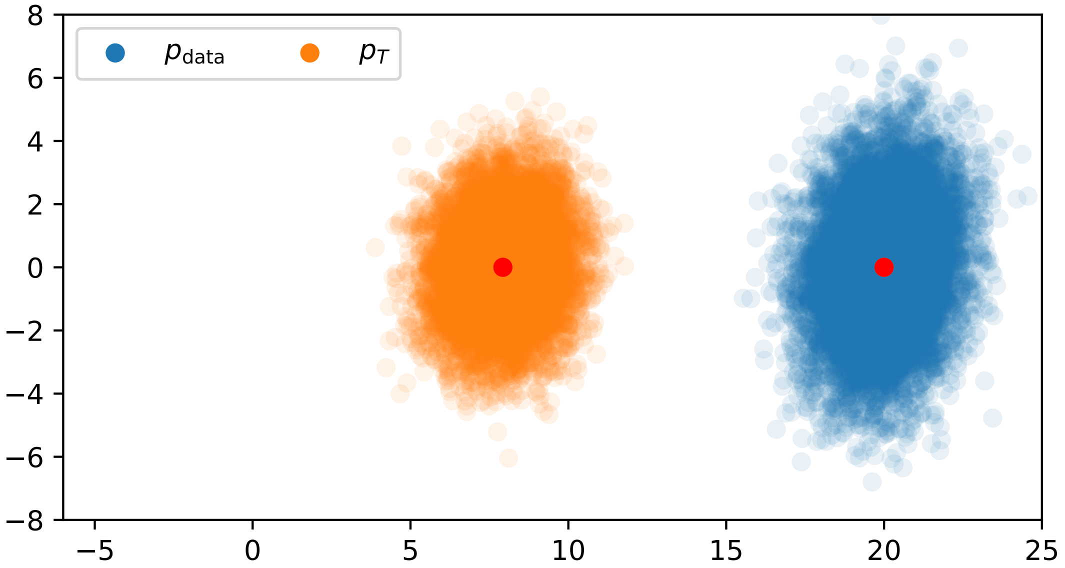
$$y_t^{\text{ODE}} = \boldsymbol{\Sigma}_T^{-1/2} \boldsymbol{\Sigma}_t^{1/2} y_T, \quad 0 \leqslant t \leqslant T,$$

- $y \mapsto \boldsymbol{\Sigma}_T^{-1/2} \boldsymbol{\Sigma}_t^{1/2} y$ is the transport map between p_T and p_t.
- False in general:, see [Lavenant and Santambrogio 2022][2]
- However, used in [Khrulkov et al. 2023][3]

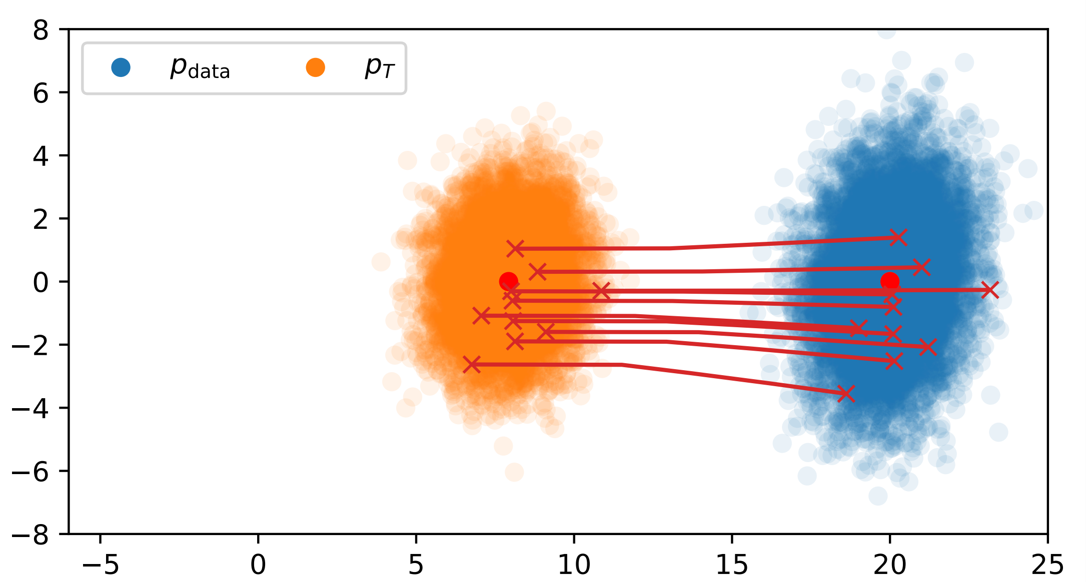
[2] Hugo Lavenant and Filippo Santambrogio (2022). "The flow map of the Fokker–Planck equation does not provide optimal transport". In: *Applied Mathematics Letters* 133, p. 108225. ISSN: 0893-9659. DOI: https://doi.org/10.1016/j.aml.2022.108225. URL: https://www.sciencedirect.com/science/article/pii/S089396592200180X

[3] 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

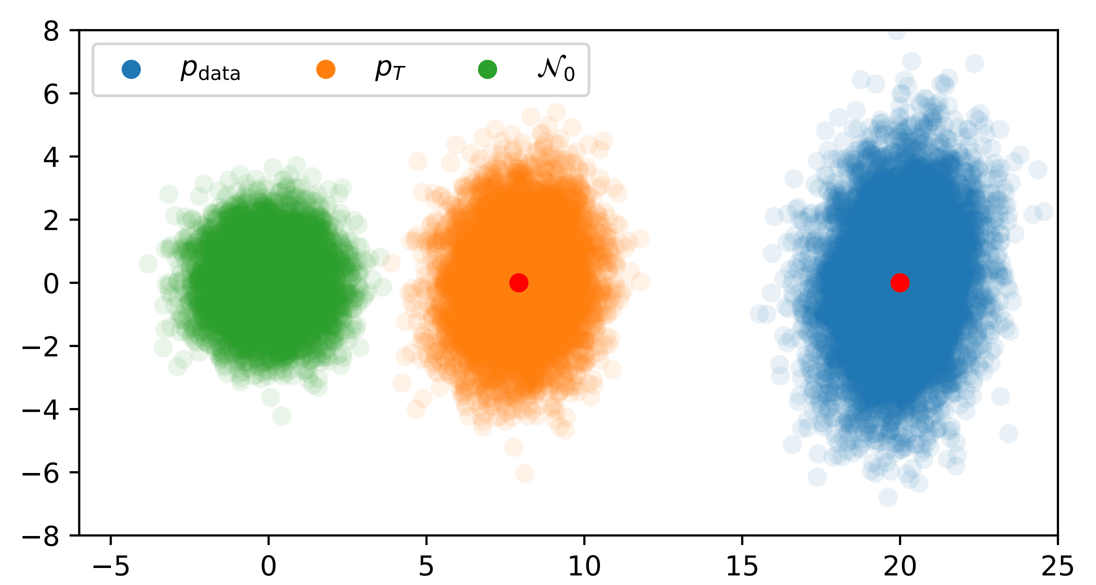
Initialization error: Focus on the ODE



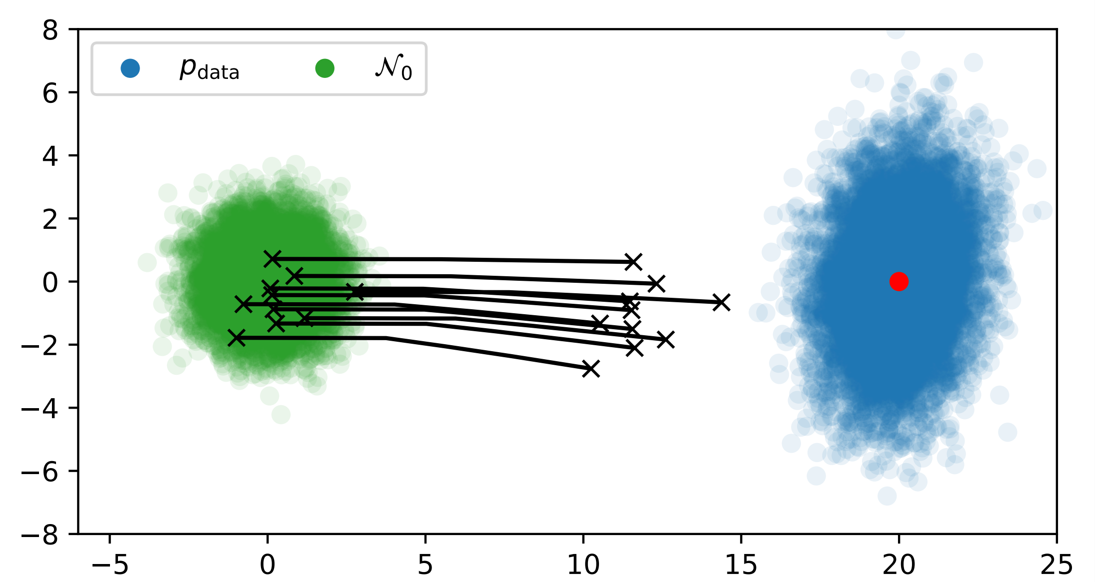
Initialization error: Focus on the ODE



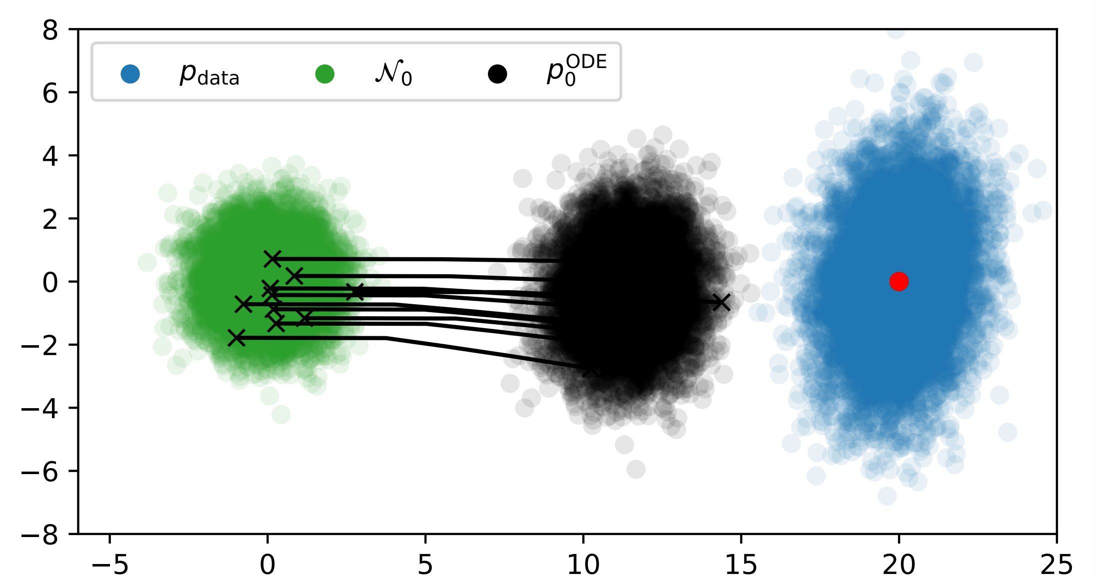
Initialization error: Focus on the ODE



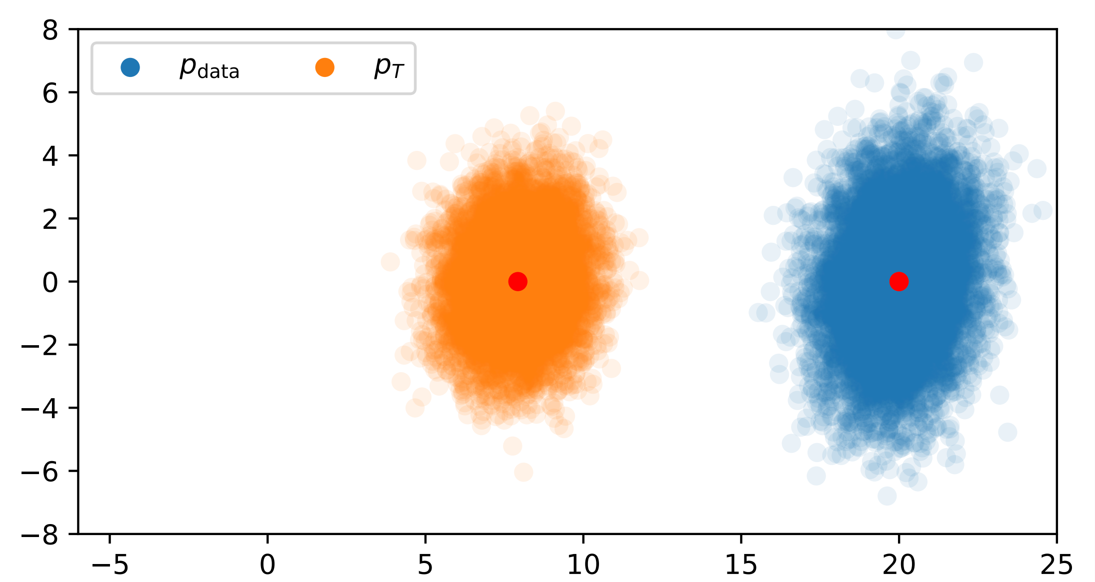
Initialization error: Focus on the ODE



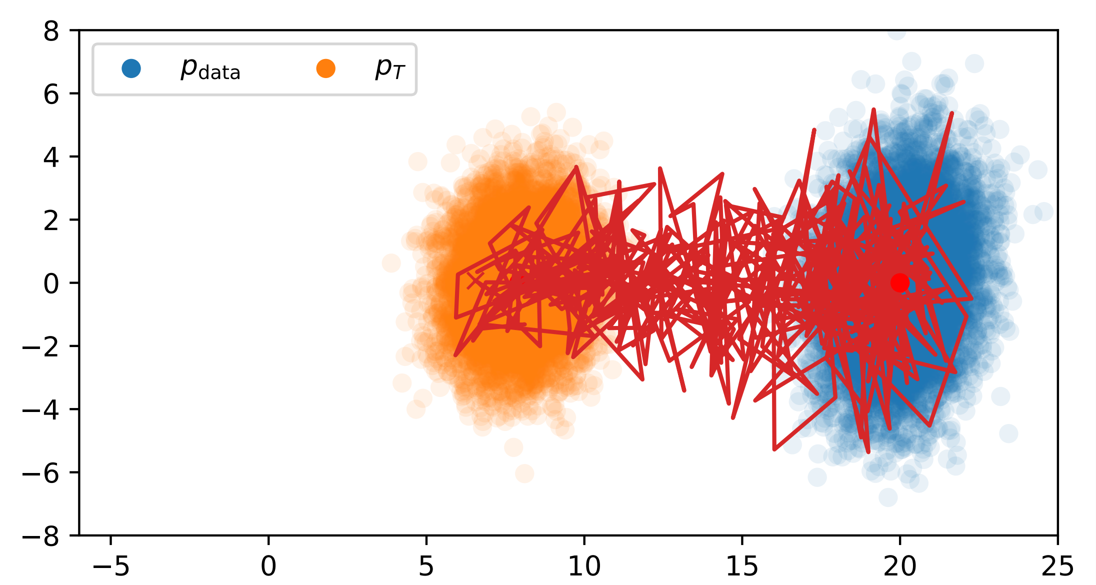
Initialization error: Focus on the ODE



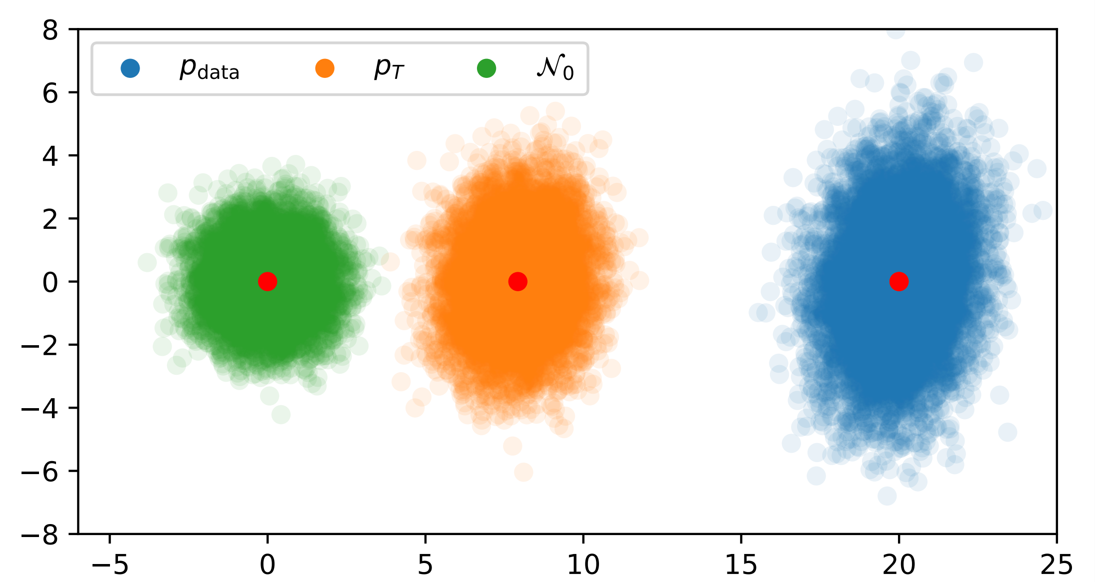
Initialization error: Focus on the SDE



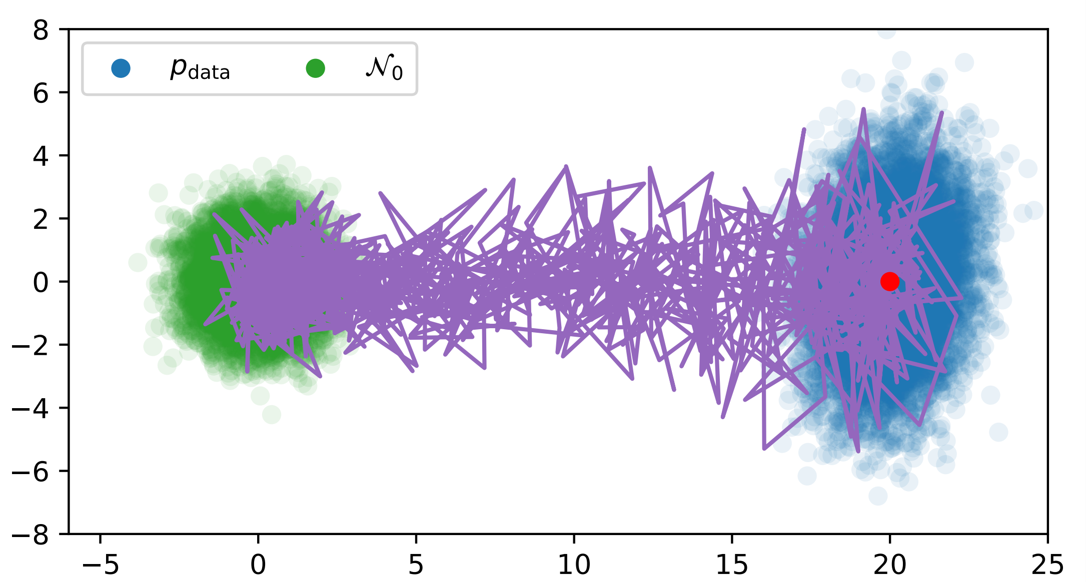
Initialization error: Focus on the SDE



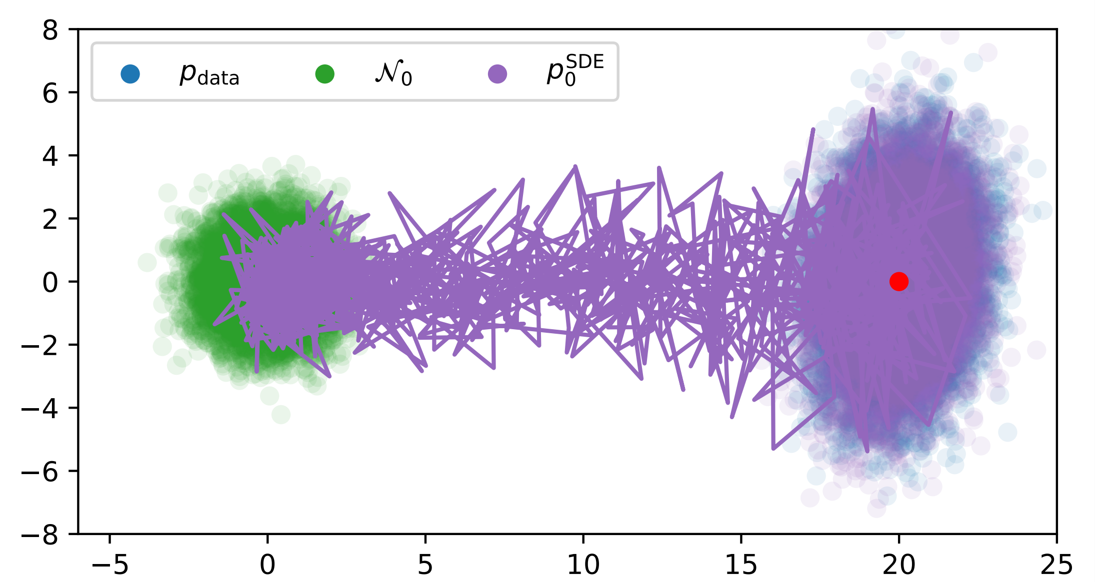
Initialization error: Focus on the SDE



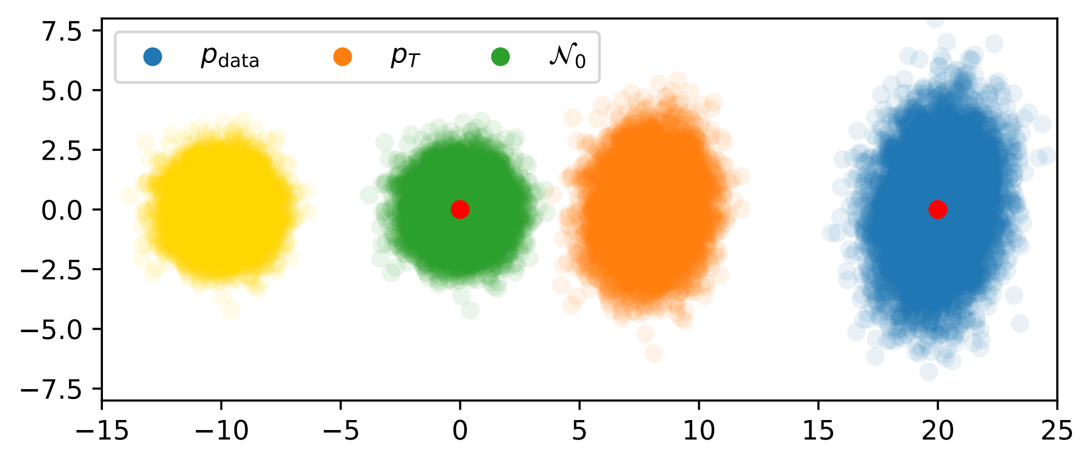
Initialization error: Focus on the SDE



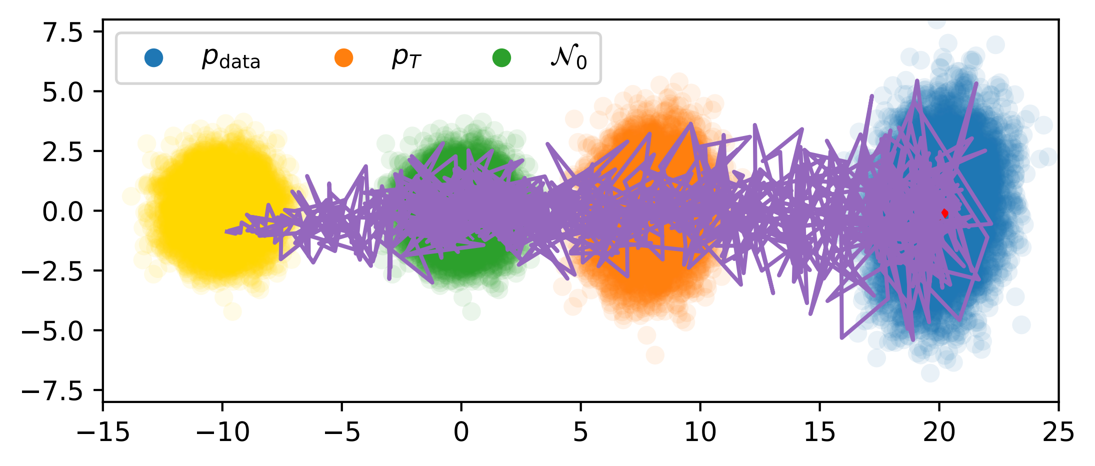
Initialization error: Focus on the SDE



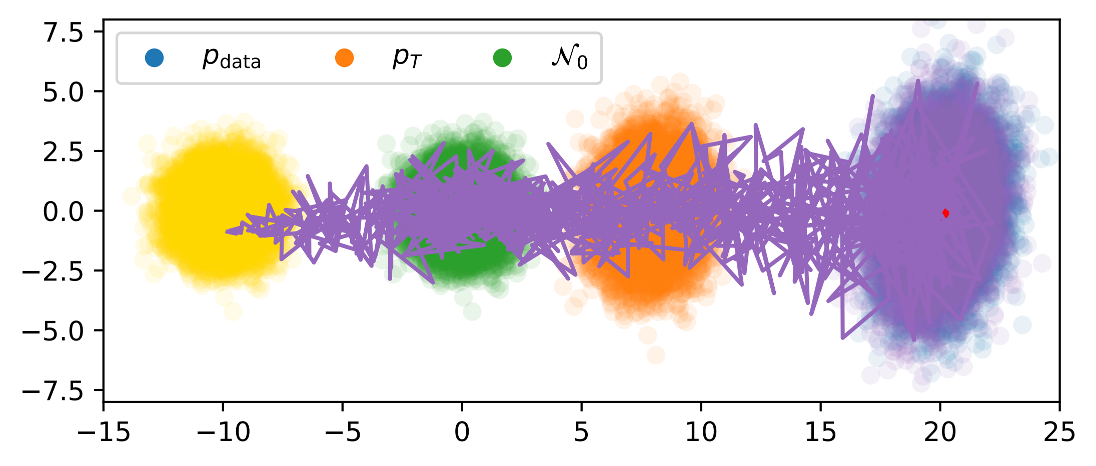
Initialization error: Focus on the SDE



Initialization error: Focus on the SDE



Initialization error: Focus on the SDE



Study of the discretization error

The Euler-Maruyama scheme

We propose to discretize the backward equations with the Euler-Maruyama scheme.

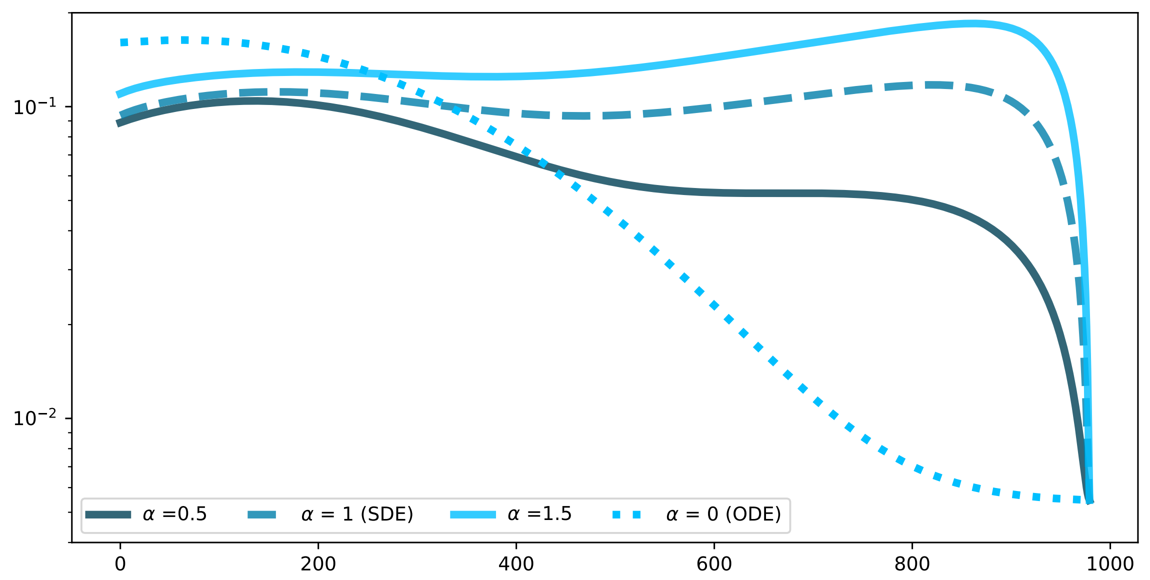
$$dy_t = [-\beta_t y_t + (1+\alpha)\beta_t s_\theta(t, y_t)]\, dt + \sqrt{2\alpha\beta_t} dw_t, \quad \varepsilon \leqslant t \leqslant T, \quad y_T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}). \tag{1_α}$$

becomes

$$\begin{cases} y_N & \sim \mathcal{N}(0, \boldsymbol{I}) \text{ (initialization error)} \\ y_{k-1} & = [-\beta_{t_k} y_k + 1 + \alpha)\beta_{t_k} \nabla \log p_{t_k}(y_k)]\, \Delta_t + 2\alpha\beta_{t_{k-1}} z_k, z_k \sim \mathcal{N}(0, I) \end{cases}$$

We observe the behavior with respect to α with the adding of initialization error and discretization error.

Discretization of the SDEs: exact Wasserstein error



Conclusion

There is a trade-off between initialization error and discretization error to modelize

$$y_t = e^{-\alpha(B_T - B_{T-t})} \boldsymbol{\Sigma}_{T-t}^{\frac{1+\alpha}{2}} \boldsymbol{\Sigma}_T^{-\frac{1+\alpha}{2}} y_0 + \boldsymbol{\xi}_t, 0 \leqslant t \leqslant T.$$

- The exponential decrease is beneficial for the initialization error.
- However, it is difficult to capture it with a discretization scheme.
- Consequently, optimal α seems to be in $]0, 1[$ wrt these two error types.

A bridge to Flow matching

Another direction to avoid initialization error

As a generative model, Flow Matching aims to learn a velocity field v_t such that by using

$$dx_t = v_t(x_t)dt, x_0 \sim \mathcal{N}(0, I),$$

we have

$$x_1 \sim p_{\text{data}}.$$

There is a direct link with the "ODE of diffusion models. But how is it possible to avoid initialization error ?

Another direction to avoid initialization error

As a generative model, Flow Matching aims to learn a velocity field v_t such that by using

$$dx_t = v_t(x_t)dt, x_0 \sim \mathcal{N}(0, I),$$

we have

$$x_1 \sim p_{\text{data}}.$$

There is a direct link with the "ODE of diffusion models. But how is it possible to avoid initialization error ?
The corresponding forward process to Flow Mathing is

$$dx_t = -\frac{1}{1-t} x_t dt + \sqrt{\frac{2t}{1-t}} dw_t \tag{1}$$

with $\frac{1}{1-t} \underset{t \longrightarrow 1}{\longrightarrow} +\infty$.

Link with memoryless stochastic process? (reading group of tuesday).

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.

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.

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.
- Add theoretical score approximation ?
  → Samuel Hurault et al. (2025). *From Denoising Score Matching to Langevin Sampling: A Fine-Grained Error Analysis in the Gaussian Setting*. arXiv: 2503.11615 [cs.LG]. URL: https://arxiv.org/abs/2503.11615

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)*.

Hurault, Samuel et al. (2025). *From Denoising Score Matching to Langevin Sampling: A Fine-Grained Error Analysis in the Gaussian Setting*. arXiv: 2503.11615 [cs.LG]. URL: https://arxiv.org/abs/2503.11615.

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.

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.

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 $\boldsymbol{v} = \boldsymbol{A}x + \sigma\boldsymbol{\varepsilon}$?

A large literature [Song et al. 2021[4],Lugmayr et al. 2022[5],Chung et al. 2022[6],Choi et al. 2021[7]] uses the Bayes formula

$$\nabla_x \log p_t(x_t \mid \boldsymbol{v}) = \nabla_x \log p_t(\boldsymbol{v} \mid x_t) + \nabla_x \log p_t(x_t). \quad (2)$$

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

[4] 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

[5] 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

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

[7] 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