# Diffusion models for Gaussian distributions: Exact solutions and Wasserstein errors

Émile Pierret[a], supervised by Bruno Galerne[a,b]
Mathematical Imaging and Random Geometry Workshop, Nice
November $25^t h$ 2024

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

# Introduction

# 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 $\beta_t$ 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 $\beta_t$ 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 $\beta_t$ 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 $t \to +\infty$, $x_\infty \sim \mathcal{N}_0$

# Probability-flow ODE

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

$$dy_t = -\beta_t \left[ y_t + 2\nabla_y \log p_t(y_t) \right] dt + \sqrt{2\beta_t} d\overline{w}_t, \quad 0 \leqslant t \leqslant T, \quad y_T \sim p_T \tag{3}$$

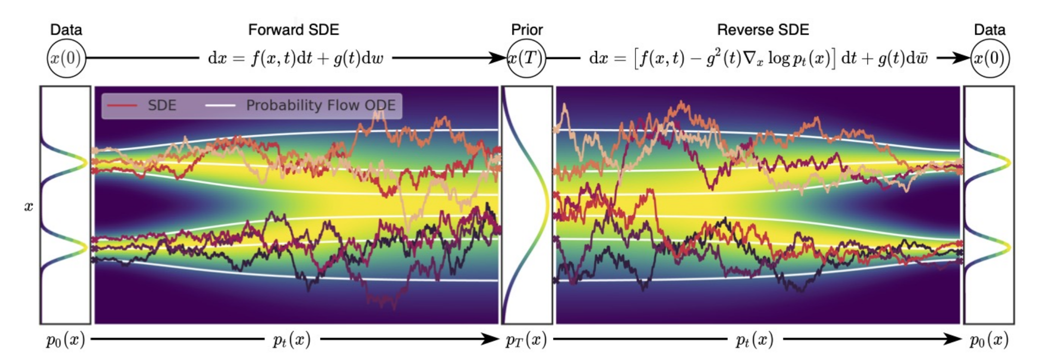
# Probability-flow ODE

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

$$dy_t = -\beta_t\left[y_t + 2\nabla_y \log p_t(y_t)\right]dt + \sqrt{2\beta_t}d\overline{w}_t, \quad 0\leqslant t\leqslant T, \quad y_T\sim p_T \tag{3}$$

are the same as those of this ODE

$$dy_t = -\beta_t\left[y_t + \nabla_y \log p_t(y_t)\right]dt, \qquad 0\leqslant t\leqslant T, \quad y_T\sim p_T. \tag{4}$$



# Study of the convergence

# Sampling through diffusion models

$$
\left.\begin{array}{l}
dy_t = -\beta_t \left[y_t + 2\nabla_y \log p_t(y_t)\right] dt + \sqrt{2\beta_t} d\overline{w}_t, \\
\text{or} \\
dy_t = -\beta_t \left[y_t + \nabla_y \log p_t(y_t)\right] dt,
\end{array}\right. \quad \text{where} \quad 0 \leqslant t \leqslant T, \quad y_T \sim p_T. \tag{5}
$$

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

## Sampling through diffusion models

$$dy_t = -\beta_t \left[ y_t + 2\nabla_y \log p_t(y_t) \right] dt + \sqrt{2\beta_t} d\overline{w}_t,$$

or

$$dy_t = -\beta_t \left[ y_t + \nabla_y \log p_t(y_t) \right] dt,$$

where $0 \leqslant t \leqslant T,$ ~~$y_T \sim p_T$~~ $y_T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ . (5)

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**

# Sampling through diffusion models

$$dy_t = -\beta_t \left[y_t + 2\nabla_y \log p_t(y_t)\right] dt + \sqrt{2\beta_t} d\overline{w}_t,$$

or

$$dy_t = -\beta_t \left[y_t + \nabla_y \log p_t(y_t)\right] dt,$$

where $\cancel{0}_{\varepsilon} \leqslant t \leqslant T, \quad y_T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}).$ (5)

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 $\varepsilon$ to consider them on $[\varepsilon, T]$ → **truncation error**

# Sampling through diffusion models

$$dy_t = -\beta_t \left[y_t + 2\nabla_y \log p_t(y_t)\right] dt + \sqrt{2\beta_t} d\overline{w}_t,$$

or where $\varepsilon \leqslant t \leqslant T, \quad y_T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}).$ (5)

$$dy_t = -\beta_t \left[y_t + \nabla_y \log p_t(y_t)\right] dt,$$

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 $\varepsilon$ to consider them on $[\varepsilon, T]$ → **truncation error**
- A scheme to discretize the equations → **discretization error**

# Sampling through diffusion models

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

or where $\varepsilon \leqslant t \leqslant T, \quad y_T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}).$ (5)

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

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 $\varepsilon$ to consider them on $[\varepsilon, T]$ → **truncation error**
- A scheme to discretize the equations → **discretization error**
- A model/neural network $s_\theta$ to learn the score → **score approximation error**

# Sampling through diffusion models

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

or where $\varepsilon \leqslant t \leqslant T, \quad y_T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}).$ (5)

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

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 $\varepsilon$ to consider them on $[\varepsilon, T]$ → **truncation error**
- A scheme to discretize the equations → **discretization error**
- A model/neural network $s_\theta$ to learn the score → **score approximation error**

# Restriction to the Gaussian case

# Claims

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

# Claims

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

$$\nabla \log p_t(x) = -\mathbf{\Sigma}_t^{-1} x, \quad 0 < t \leqslant T \tag{6}$$

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

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

$$\nabla \log p_t(x) = -\mathbf{\Sigma}_t^{-1} x, \quad 0 < t \leqslant T \tag{6}$$

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

**Proposition 3: LLineartiy 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 SDE

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

# Explicit solution of the backward SDE

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

**Proposition 4: Solution to the equations under Gaussian assumption**

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

$$y_t^{\text{SDE}} = e^{-(B_T - B_t)} \boldsymbol{\Sigma}_t \boldsymbol{\Sigma}_T^{-1} y_T + \boldsymbol{\xi}_t, \quad 0 \leqslant t \leqslant T \tag{7}$$

Under Gaussian assumption, the solution to ODE (4) can be written as:

$$y_t^{\text{ODE}} = \boldsymbol{\Sigma}_T^{-1/2} \boldsymbol{\Sigma}_t^{1/2} y_T, \quad 0 \leqslant t \leqslant T, \tag{8}$$

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

# Explicit solution of the backward SDE

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

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

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

# Explicit solution of the backward SDE

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

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

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

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

$$\boldsymbol{\Sigma}_t^{\mathsf{SDE}} = \boldsymbol{\Sigma}_t + e^{-2(B_T - B_t)} \boldsymbol{\Sigma}_t^2 \boldsymbol{\Sigma}_T^{-2} (\boldsymbol{I} - \boldsymbol{\Sigma}_T), \quad 0 \leqslant t \leqslant T.$$
$$\boldsymbol{\Sigma}_t^{\mathsf{ODE}} = \boldsymbol{\Sigma}_T^{-1} \boldsymbol{\Sigma}_t$$

# Explicit solution of the backward SDE

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

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

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

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

$$\mathbf{\Sigma}_t^{\mathsf{SDE}} = \mathbf{\Sigma}_t + e^{-2(B_T - B_t)} \mathbf{\Sigma}_t^2 \mathbf{\Sigma}_T^{-2} (\boldsymbol{I} - \mathbf{\Sigma}_T), \quad 0 \leqslant t \leqslant T.$$
$$\mathbf{\Sigma}_t^{\mathsf{ODE}} = \mathbf{\Sigma}_T^{-1} \mathbf{\Sigma}_t$$

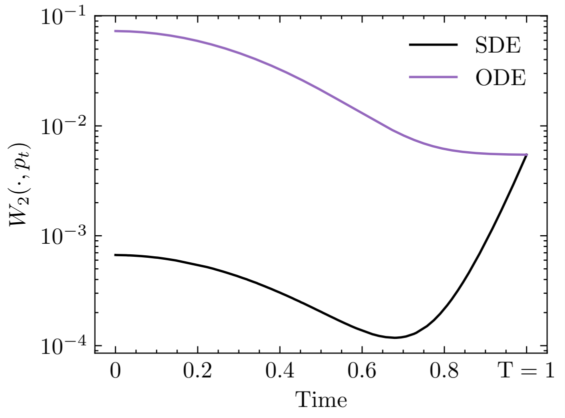
Under initialization error, the SDE and the ODE does not have the same marginals !

**Proposition 5: Marginals of the generative processes under Gaussian assumption**

Under Gaussian assumption,

$$\mathbf{W}_2(p_t^{\mathsf{SDE}}, p_t) \leqslant \mathbf{W}_2(p_t^{\mathsf{ODE}}, p_t) \tag{7}$$

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

$$\begin{aligned} dy_t &= -\beta_t[y_t + 2s_\theta(t, y_t)]dt + \sqrt{2\beta_t}d\overline{w}_t, \\ &\text{or} \\ dy_t &= -\beta_t[y_t + s_\theta(t, y_t)]dt, \end{aligned} \quad \text{where} \quad \varepsilon \leqslant t \leqslant T, \quad y_T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}). \tag{8}$$

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 $\varepsilon$ to consider them on $[\varepsilon, T]$ → **truncation error**
- A scheme to discretize the equations → **discretization error**
- A model/neural network $s_\theta$ to learn the score → **score approximation error**

# Truncation error under Gaussian assumption

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

$$\nabla \log p_t(x) = -\mathbf{\Sigma}_t^{-1} x, \quad 0 < t \leqslant T \tag{9}$$

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

# Truncation error under Gaussian assumption

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

$$\nabla \log p_t(x) = -\mathbf{\Sigma}_t^{-1} x, \quad 0 < t \leqslant T \tag{9}$$

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

$$\nabla \log p_t(x) = -\mathbf{\Sigma}_t^{-1} x, \quad 0 < t \leqslant T \tag{9}$$

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

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

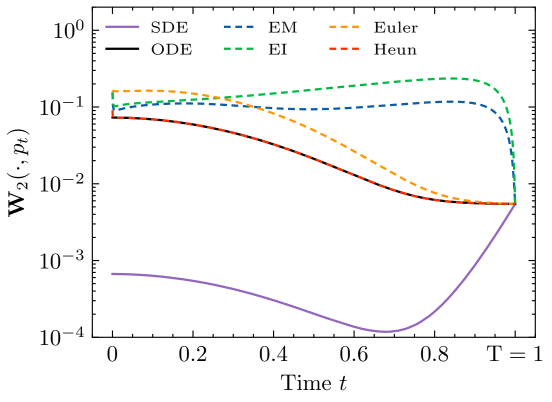
Consequently, $\nabla \log p_0(x)$ is not defined in general.

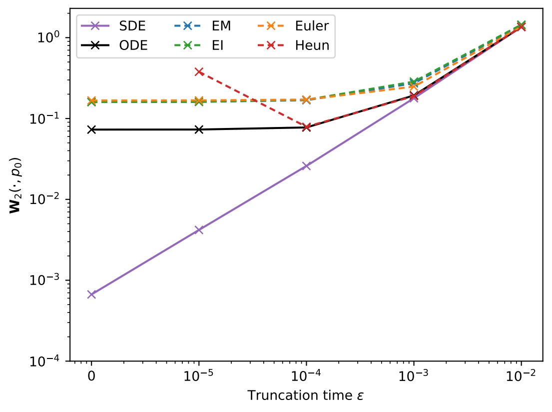
# Discretization error

# Discretization schemes

| | | | |
|---|---|---|---|
| SDE schemes | Euler-Maruyama (EM) | $\begin{cases} \tilde{\boldsymbol{y}}_0^{\Delta,\text{EM}} & \sim \mathcal{N}_0 \\ \tilde{\boldsymbol{y}}_{k+1}^{\Delta,\text{EM}} & = \tilde{\boldsymbol{y}}_k^{\Delta,\text{EM}} + \Delta_t \beta_{T-t_k} \left( \tilde{\boldsymbol{y}}_k^{\Delta,\text{EM}} - 2\boldsymbol{\Sigma}_{T-t_k}^{-1} \tilde{\boldsymbol{y}}_k^{\Delta,\text{EM}} \right) + \sqrt{2\Delta_t \beta_{T-t_k}} z_k, \ z_k \sim \mathcal{N}_0 \end{cases}$ | (10) |
| | 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 $\gamma_{2,k} = \frac{1}{2}(\exp(2B_{T-t_k} - 2B_{T-t_{k+1}}) - 1)$ | (11) |
| ODE schemes | Explicit Euler | $\begin{cases} \hat{\boldsymbol{y}}_0^{\Delta,\text{Euler}} & \sim \mathcal{N}_0 \\ \hat{\boldsymbol{y}}_{k+1}^{\Delta,\text{Euler}} & = \hat{\boldsymbol{y}}_k^{\Delta,\text{Euler}} + \Delta_t f(t_k, \hat{\boldsymbol{y}}_k^{\Delta,\text{Euler}}) \quad \text{with } f(t,y) = \beta_{T-t} y - \beta_{T-t} \boldsymbol{\Sigma}_{T-t}^{-1} y \end{cases}$ | (12) |
| | Heun's method | $\begin{cases} \hat{\boldsymbol{y}}_0^{\Delta,\text{Heun}} & \sim \mathcal{N}_0 \\ \hat{\boldsymbol{y}}_{k+1/2}^{\Delta,\text{Heun}} & = \hat{\boldsymbol{y}}_k^{\Delta,\text{Heun}} + \Delta_t f(t_k, \hat{\boldsymbol{y}}_k^{\Delta,\text{Heun}}) \quad \text{with } f(t,y) = \beta_{T-t} y - \beta_{T-t} \boldsymbol{\Sigma}_{T-t}^{-1} y \\ \hat{\boldsymbol{y}}_{k+1}^{\Delta,\text{Heun}} & = \hat{\boldsymbol{y}}_k^{\Delta,\text{Heun}} + \frac{\Delta_t}{2} \left( f(t_k, \hat{\boldsymbol{y}}_k^{\Delta,\text{Heun}}) + f(t_{k+1}, \hat{\boldsymbol{y}}_{k+1/2}^{\Delta,\text{Heun}}) \right) \end{cases}$ | (13) |

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



# Conclusion

# Conclusion

- The simple Gaussian setting gives good insights on the error types.
- We find results already observed empirically for more general data distributions [Karras et al. 2022][1].
- The computation of exact 2-Wasserstein error is fast and a low amount of storage.
- The score approximation error remains the highest error type.
- We consider our work as lower bound of diffusion models convergence.
- **Pending question:** Link between Gaussian distributions results and more general distributions ?

[1] Tero Karras et al. (2022). "Elucidating the Design Space of Diffusion-Based Generative Models". In: *Proc. NeurIPS*

# Conclusion

- The simple Gaussian setting gives good insights on the error types.
- We find results already observed empirically for more general data distributions [Karras et al. 2022][1].
- The computation of exact 2-Wasserstein error is fast and a low amount of storage.
- The score approximation error remains the highest error type.
- We consider our work as lower bound of diffusion models convergence.
- **Pending question:** Link between Gaussian distributions results and more general distributions ?

Thank you for your attention !

[1] Tero Karras et al. (2022). "Elucidating the Design Space of Diffusion-Based Generative Models". In: *Proc. NeurIPS*

# 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)*.

Karras, Tero et al. (2022). "Elucidating the Design Space of Diffusion-Based Generative Models". In: *Proc. NeurIPS*.

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

A large literature [Song et al. 2021[2],Lugmayr et al. 2022[3],Chung et al. 2022[4],Choi et al. 2021[5]] uses the Bayes formula

$$\nabla_x \log p_t(x_t \mid \boldsymbol{v}) = \nabla_x \log p_t(\boldsymbol{v} \mid x_t) + \nabla_x \log p_t(x_t). \tag{14}$$

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

---

[2] 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

[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

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

[5] 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 |